aws

User Guide

AWS Transform

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS Transform: User Guide

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

Table of Contents

What is AWS Transform? 1
Terminology 2
Getting started 5
Setting up 5
Before you begin 5
Sign up for an AWS account 5
Getting started with IAM-only access 5
Using IAM credentials 6
Getting started with AWS Organizations 7
Getting started with AWS IAM Identity Center 8
Using third-party identity providers 9
Onboarding 12
Using your own S3 bucket 13
Enable AWS Transform 22
Quick start: Trying AWS Transform 24
Managing users 24
Adding users in IAM Identity Center 24
Adding users to AWS Transform 25
Managing users with IAM-only access 25
Understanding collaborator permissions 26
AWS Transform environment 28
Language settings 30
Start your project 31
AWS Transform chat 31
Connectors 33
User traceability for agentic operations 34
Managing connectors 35
Developer tools 37
Kiro Power 37
Agent plugin 37
IDE plugin 38
MCP server 38
Authentication 39
Build custom agents for AWS Transform 39

Custom 41
What is AWS Transform custom? 41
Key Capabilities 41
Transformation Patterns 41
How AWS Transform custom Works 44
Understanding Key Concepts 44
Transformation Definitions 44
Transformation definition categories 45
Transformation Registry 49
Draft vs Published Transformations 50
References vs. Lessons 50
Build and Validation Commands 51
Continual Learning 51
Client-Side Skills 52
Getting started 52
Choose where to run AWS Transform custom 53
AWS Transform CLI prerequisites 54
Installing the AWS Transform CLI 55
Configuring Authentication 56
Configuring AWS Region 58
Running Your First Transformation 61
Understanding Transformation Results 62
Creating a Custom Transformation 62
AWS Transform Web Application (Optional) 63
Introduction to custom transformation commands 63
Workflows 64
Executing Transformations 64
Continual Learning 70
Advanced Configuration 71
Create Custom Transformations 85
Command Reference 89
Interactive Commands 89
Transformation Definition Commands 90
Lesson Commands 95
Tag Commands 95
Update Commands 95

MCP Commands ... 96
AWS-Managed Transformations ... 96
Overview ... 96
Available AWS-Managed Transformations ... 96
Customizing AWS-Managed Transformations ... 121
Common Use Cases ... 122
Lambda Runtime Upgrades ... 122
VPC endpoints (AWS PrivateLink) ... 123
Considerations for AWS Transform custom VPC endpoints ... 124
Prerequisites ... 124
Creating an interface VPC endpoint for AWS Transform custom ... 125
Creating a VPC endpoint policy for AWS Transform custom ... 125
Using an on-premises computer to connect to a AWS Transform custom endpoint ... 126
Troubleshooting ... 126
Log Locations ... 126
Common Issues ... 127
Getting Support ... 129
S3 access issues ... 129
CLI version history ... 130
Continuous modernization ... 158
What is AWS Transform continuous modernization? ... 158
Key capabilities ... 158
Analysis types ... 159
Understanding key concepts ... 160
Sources ... 160
Repositories ... 160
Analyses ... 160
Findings ... 161
Remediations ... 161
Transformation definitions ... 161
How AWS Transform continuous modernization works ... 161
Compute options ... 162
Security agent setup ... 165
Getting started ... 166
Prerequisites ... 166
Quick start ... 166

Working with continuous modernization ... 167
Source management ... 167
Repository discovery and management ... 169
Running analysis ... 169
Managing findings ... 171
Creating remediations ... 172
Remote execution ... 172
Scheduling recurring analysis ... 181
Tagging and access control ... 183
AWS Transform web application ... 183
Developer tools ... 185
Agent skills ... 185
CLI reference ... 186
Troubleshooting ... 190
Migration assessments ... 193
Why perform a migration assessment ... 193
Prerequisites ... 194
How migration assessments work ... 194
Uploading inventory data ... 194
Reviewing discovery results ... 195
Managing inventory scope ... 196
Configuring assessment scenarios ... 196
Assessment capabilities ... 197
Using chat-based assessments ... 198
Rough estimation with limited data ... 198
Adding inventory through chat ... 198
Modifying costs and adding services ... 198
Comparing scenarios ... 199
Creating scenarios ... 199
Running comparisons ... 199
What-if analysis ... 199
Generating deliverables ... 200
Quick start ... 200
Getting data for a migration assessment ... 201
Running an assessment ... 203
Reviewing and refining the results ... 205

Example prompts ... 206
Related resources ... 207
Compute ... 207
Amazon EC2 right-sizing ... 207
Processor architecture and AWS Graviton ... 208
Tenancy and Dedicated Hosts ... 208
Pricing models ... 209
Example prompts ... 209
Databases ... 210
Amazon RDS for SQL Server ... 210
SQL Server on Amazon EC2 ... 211
Storage ... 212
Amazon EBS ... 212
Amazon S3 ... 212
Amazon FSx for NetApp ONTAP ... 212
Example prompts ... 213
Analytics ... 213
Amazon MSK Express ... 213
Business value ... 216
Value pillars ... 216
Inputs ... 217
Sustainability ... 217
Example prompts ... 217
Additional cost components ... 218
On-premises pricing ... 218
Network costs ... 218
Support costs ... 218
End user computing ... 218
Adding other AWS services ... 219
Related topics ... 219
Discovery tool ... 220
Workflow ... 220
Set up the discovery tool ... 221
Prerequisites ... 221
Disk sizing and management ... 224
Deploy on VMware ... 226

Deploy on Hyper-V ... 228
Deploy on Linux ... 228
Configure the discovery tool ... 231
Required permissions ... 256
VMware vCenter ... 256
Hyper-V hosts (Windows) ... 256
Linux servers (SSH) ... 257
Windows servers (WinRM) — OS metrics ... 258
Windows servers (WinRM) — SQL Server collection ... 259
Oracle Database (SQL) ... 261
Network collection ... 261
Quick reference: Minimum permissions by use case ... 262
Data collection ... 263
Collection schedule ... 263
How average, p95, and peak statistics are calculated ... 264
Discovered inventory ... 265
OS-related data ... 269
Managing storage ... 289
Understanding disk usage states ... 289
Resuming collection ... 290
Configuring data retention ... 290
Reviewing usage and pruning data ... 291
Excluding VMs from collection ... 291
Security considerations ... 294
Securing the discovery tool VM ... 294
Securing Credentials ... 294
Credential storage ... 296
Using Auto-Connect Feature With Caution ... 296
CSV Import Security ... 298
Revoking Access Considerations ... 298
Troubleshoot the discovery tool ... 299
Incomplete or failed export ... 299
Verifying discovery tool connectivity to vCenter ... 299
WinRM Troubleshooting ... 301
Kerberos troubleshooting ... 301
Oracle Database troubleshooting ... 306

SNMP Troubleshooting ... 307
Network collection errors ... 308
OS metrics collection errors ... 309
Access issues in Discovered inventory ... 310
SSH key authentication troubleshooting ... 310
Linux installer troubleshooting ... 311
Common error messages ... 312
Release notes ... 312
Migrations (including VMware) ... 322
Capabilities and key features ... 322
Architecture ... 323
Migration jobs ... 323
Getting a workspace ... 323
Job types ... 324
Creating and starting a job ... 325
Downloading a workspace summary report ... 326
Use expert mode ... 326
Limitations ... 327
Discover source data ... 328
Build migration plan ... 329
Migration planning terminology ... 330
Workflow ... 330
7Rs recommendations ... 332
Diagrams and reports ... 334
Connect target AWS accounts and regions ... 335
Step 1: Migration type selection ... 335
Step 2: MAP agreement ... 335
Step 3: Connector configuration ... 336
Supported target Regions ... 342
Build landing zone ... 343
Connector setup ... 344
Foundation setup ... 347
Workload account design ... 352
IaC formats ... 354
Deployment approvals process ... 355
Tag landing zone resources ... 355

Reversing changes ... 356
Related resources ... 356
Generate landing zone IaC without deployment ... 357
Migrate network ... 359
Map source network to new VPCs ... 360
Apply security posture to existing VPCs ... 378
Generated artifacts ... 384
Migrate servers ... 388
Prerequisites and Configure Migration Defaults ... 389
MCP server ... 409
Setting up the AWS Transform MCP server ... 409
Using AWS Transform for migrations via MCP ... 412
Best practices ... 413
Example prompts ... 414
Release notes ... 416
October 2026 ... 416
September 2026 ... 417
August 2026 ... 417
July 2026 ... 418
June 2026 ... 419
May 2026 ... 419
April 2026 ... 420
March 2026 ... 420
February 2026 ... 420
January 2026 ... 421
December 2025 ... 421
October 2025 ... 421
September 2025 ... 421
August 2025 ... 421
May 2025 ... 422
Source code containerization ... 423
Capabilities and key features ... 423
How containerization works ... 424
Prerequisites ... 424
Containerization workflow ... 424
Starting a containerization job ... 425

Permissions ... 426
Base policy ... 426
Networking policy ... 426
Storage policy ... 427
AWS KMS policy ... 427
Amazon ECS policy ... 427
Amazon EKS policy ... 427
Step 1: Security disclaimer ... 428
What the disclaimer covers ... 428
What you need to do ... 428
Step 2: Clone source code ... 429
Option 1: Git repositories ... 429
Option 2: Zip file upload ... 430
Optional: Configure private dependencies ... 430
Confirming containerization preferences ... 431
Step 3: Containerize ... 431
What happens during containerization ... 431
Supported application types ... 432
What you need to do ... 432
Step 4: Review artifacts ... 432
Reviewing changes from Git repositories ... 433
Reviewing changes from zip uploads ... 433
Requesting changes ... 433
Step 5: Publish images ... 433
What happens during publishing ... 433
What you need to do ... 434
Step 6: Generate IaC ... 434
Amazon Elastic Kubernetes Service (Kubernetes) ... 435
Amazon Elastic Container Service (Terraform) ... 435
Automated validation and security scanning ... 436
Generating IaC without published images ... 436
What you need to do ... 436
Step 7: Deploy test infrastructure ... 437
What happens during test deployment ... 437
What you need to do ... 437
Iterating on the infrastructure ... 437

Session recovery ... 438
Step 8: Clean up test infrastructure ... 438
What you need to do ... 438
Step 9: Deploy cutover infrastructure ... 439
What you need to do ... 439
Session recovery ... 439
Mainframe modernization ... 440
Capabilities and key features ... 440
High-level walkthrough ... 441
Human in the loop (HITL) ... 441
Supported file types for transformation of mainframe applications ... 442
Supported Regions and quotas for AWS Transform mainframe ... 442
Modernization journey ... 443
Phase 1: Assess ... 443
Phase 2: Mobilize ... 444
Phase 3: Migrate and modernize ... 445
Phase 4: Operate, optimize, and innovate ... 446
Transformation of mainframe applications workflow ... 447
Prerequisite: Prepare project inputs in S3 ... 448
Mainframe source artifacts collection ... 449
Sign-in and create a job ... 457
Tracking transformation progress ... 458
Set up a connector ... 458
Assess and reimagine job plan ... 463
Reimagine job plan ... 467
Custom job plan ... 468
Build and deploy modernized application ... 497
Prerequisites ... 498
Step 1: Retrieve the modernized code ... 498
Step 2: Build the modernized application ... 499
Step 3: Configure the test environment ... 499
Step 4: Deploy the modernized application ... 500
Step 5: Test the modernized application ... 500
Additional example ... 501
Full-stack Windows modernization ... 502
.NET ... 502

Multiple experiences ... 502
Capabilities and key features ... 502
Supported versions and project types ... 503
Limitations ... 503
Human intervention ... 504
More information ... 504
How to work with the .NET agent ... 504
.NET web app ... 508
.NET IDE ... 524
Porting UI layer code ... 533
Convert cloud-native code to AWS services ... 535
Transformation Report ... 536
Hand-off to AI code companion ... 537
Best practices ... 538
SQL Server modernization ... 543
Supported regions ... 544
Capabilities and key features ... 544
Supported versions and project types ... 545
Getting started with a modernization job ... 549
Connect to database (online workflow) ... 549
Upload DDL files (offline workflow) ... 601
Best practices and troubleshooting ... 605
Security ... 611
Data protection ... 611
Cross-region processing ... 613
Data encryption ... 622
Service improvement ... 632
Identity and access management ... 633
Audience ... 633
Authenticating with identities ... 633
Managing access using policies ... 635
How AWS Transform works with IAM ... 636
Identity-based policy examples ... 642
Troubleshooting ... 651
Using service-linked roles ... 653
AWS managed policies ... 659

AWS Transform permissions reference ... 682
Compliance validation ... 684
Resilience ... 685
VPC endpoints (AWS PrivateLink) ... 685
Considerations for AWS Transform VPC endpoints ... 685
Prerequisites ... 686
Creating an interface VPC endpoint for AWS Transform ... 686
Using an on-premises computer to connect to a AWS Transform endpoint ... 687
Web application VPC access ... 687
How it works ... 687
Architecture ... 688
Prerequisites ... 689
Configuring the VPC endpoint policy ... 689
Setting up controlled internet egress ... 690
Verification ... 697
Troubleshooting ... 697
Cost considerations ... 698
Cleanup ... 699
Monitoring ... 701
CloudTrail logs ... 701
AWS Transform information in CloudTrail ... 702
Understanding AWS Transform log file entries ... 703
Notifications ... 703
Email notifications ... 703
Quotas ... 705
Supported Regions ... 710
Supported AWS Regions (enabled by default) ... 710
Support ... 712
Prerequisites ... 712
Enabling Support Case Management ... 712
Creating a Support Case ... 712
Viewing Support Cases ... 713
Adding Communications to a Case ... 714
Resolving a Case ... 714
Disabling Support Case Management ... 714
Usage Telemetry ... 715

Operational Data 716
Change log 717
Document history 762

What is AWS Transform?

AWS Transform is a service that helps you accelerate and simplify the transformation of infrastructure, applications, and code with an agentic AI experience. It provides specialized tools to streamline modernization and migration efforts across various workloads.

With AWS Transform, you can:

- Modernize and migrate IBM z/OS and Fujitsu GS21 to AWS
- Migrate virtual and bare metal server environments to Amazon EC2, including VMware, Hyper-V, and other platforms:
  - **Discovery**. Collect and analyze your on-premises server inventory from multiple sources to understand your environment before planning.
  - **Migration planning**. Group applications, identify dependencies, and build prioritized migration waves.
  - **Landing zone creation**. Design and deploy a multi-account AWS foundation with organizational structure, governance controls, and account setup.
  - **Network migration**. Translate your source network configuration into production-ready AWS VPC architecture and deploy it.
  - **Server rehost**. Rehost your servers to run natively on Amazon EC2 with guided replication, testing, and cutover workflows.
  - **Source code containerization**. Containerize applications from source code and deploy to Amazon Elastic Container Service or Amazon Elastic Kubernetes Service.
- Modernize .NET applications to Linux-ready cross-platform .NET
- Assess workloads for migration readiness

AWS Transform helps you offload labor-intensive, complex transformation tasks across the discovery, planning, and execution phases to AI agents with expertise in languages, frameworks, and infrastructure. This approach allows your teams to focus on innovation while it efficiently transforms complex, large-scale projects through a unified web experience.

There are no additional charge to use AWS Transform.

Terminology

Within this section, italics indicate an official term within the definition of a different term.

Account connection

Account in this context is a generalized reference to a customer-owned container or security boundary for resources in AWS or remote service, for example, an AWS account or GitHub account.

Artifact

An output deliverable produced by AWS Transform.

Administrator

Administrators can read and mutate everything in the workspace. They can begin chats with AWS Transform, start and stop jobs, and upload/download artifacts. Administrators can interact with running jobs for human-in-the-loop (HITL) actions, and can approve critical HITL actions such as merging to main, performing graph decomposition, or deploying code to production environments. Administrators can mutate workspaces, connectors, and users.

Agent

A task-specific service that executes a specific transformation type. For example, VMware migrations.

Approver

Approvers have permissions similar to Administrators except that they do not have permissions to mutate workspaces, connectors, or users. Approvers cannot mutate workspaces, connectors, or users.

Asset

Input for a transformation job. For example, customer's source code, server, database, network. Assets are accessed via a connector.

Collaborator request

A *task* in which AWS Transform is asking a human to do something.

Connector

Connectors are asset providers that allow access to customer-owned resources in a system external to AWS Transform.

When you set up a connector, the administrator of the account to which you are connecting must accept the connection. In order to accept the connection, they must have permissions given in the connector acceptor policy.

The following two accounts must either be identical, or in the same AWS Organizations organization:

- The account from which the AWS Transform administrator enables the service.
- The account that will be on the receiving end of your transformation. This account must be assigned an IAM role that allows it to use a connector.

Contributor

Contributors can read everything in the workspace. They can begin chats with AWS Transform, start and stop jobs, upload or download artifacts, and interact with running jobs for HITL actions. However, they cannot perform critical HITL actions such as merging to main, performing graph decomposition, or deploying code to production environments. Contributors also cannot mutate workspaces, connectors, or users.

Objective

A user-defined end state that AWS Transform works to reach. Users write this, and then AWS Transform converts the objective into a series of tasks that it perform in concert with users when required.

Job

A long-running process (weeks/months+) that AWS Transform is working on in order to fulfill an objective defined by a user. Made up of multiple tasks and collaborator requests.

Plan

A list of tasks that AWS Transform undertakes (with help from human users) in pursuit of an objective.

Reader

Readers can view the status and outcomes of AWS Transform jobs, but cannot make any changes. They can read everything in the workspace, download artifacts, view jobs, and view human-in-the-loop (HITL) actions. However, readers cannot perform mutating actions in AWS Transform.

Task

An individual unit of work that is part of a job.

Worklog

A log of what actions AWS Transform and users have performed as part of a job.

Workspace

A AWS Transform resource that contains other resources like connectors and jobs. A workspace serves as a permissions boundary.

Getting started with AWS Transform

Topics

- Setting up AWS Transform
- Enable AWS Transform
- Quick start: Trying AWS Transform
- Managing users
- AWS Transform environment
- AWS Transform Connectors

Setting up AWS Transform

Before you begin

Before you set up AWS Transform make sure you have an AWS account with administrator access

> **Note**
>
> If you want to try out AWS Transform as a proof-of-concept or for test environments see Quick start: Trying AWS Transform.

Sign up for an AWS account

To get started with AWS, you need an AWS account. For information about creating an AWS account, see Getting started with an AWS account in the *AWS Account Management Reference Guide*.

Getting started with IAM-only access

With IAM-only access, users authenticate with their existing AWS credentials.

To set up IAM-only access:

1. In the AWS console, navigate to AWS Transform and choose **Get started**.

2. Complete the account settings configuration.
3. Under **User access**, select **IAM-only access**.
4. Choose **Enable** to create your profile.

Users log in to the AWS Transform web application through AWS Sign-In. Their user identifier in the web application is based on their IAM principal.

> **Important**
>
> You cannot add an identity provider after setup is complete. To manage users with IAM Identity Center or an existing identity provider, choose one of those options during initial setup.

With IAM-only access, any IAM principal with the `transform:AccessTransformProfile` permission on the profile resource can access AWS Transform. You manage user access through IAM policies rather than through identity provider user assignments.

The `transform:AccessTransformProfile` action only grants access to AWS Transform. Once inside AWS Transform, workspace roles control what actions a user can perform. These roles include Admin, Contributor, Approver, and Read-only. Workspace roles apply regardless of how the user authenticated.

Using IAM credentials

User information handling

When IAM principals access AWS Transform, the service handles user identity differently than with identity provider-based access.

User identity

: IAM users are identified by their IAM principal.

Dual identity

: A user who has both an identity provider identity (such as IAM Identity Center) and an IAM session appears as two separate users in AWS Transform. Their work is not linked between the two identities.

Email notifications

Email notifications are only enabled for SSO users. IAM principals can still be added to workspaces but they will not receive email notifications.

Audit

IAM API calls are logged in AWS CloudTrail with the caller's IAM identity. Actions performed by IAM users in a job are also traceable in AWS Transform worklogs.

Enabling IAM access alongside an identity provider

If you set up AWS Transform with IAM Identity Center or a third-party identity provider, you can also enable IAM access as an additional way to authenticate. This allows IAM principals to access AWS Transform alongside your existing identity provider users.

IAM access is enabled by default when you create a profile with IAM Identity Center or a third-party identity provider. To enable or disable IAM access at any time:

1. In the AWS Transform console, navigate to **Settings**.
2. Under **Access AWS Transform with IAM credentials**, toggle **Enable IAM access** on or off.

When IAM access is enabled, both identity provider users and IAM principals can use AWS Transform. IAM users and identity provider users can collaborate in the same workspace.

Getting started with AWS Organizations

Follow these steps to set up AWS Transform:

1. Sign in to your AWS Organizations management account.
2. Navigate to the AWS Transform service.
3. Choose **Enable service** for your organization to use AWS Transform.
4. Configure the necessary permissions for organizational member accounts.
5. Access the AWS Transform web experience from your member accounts.

> **Note**
>
> To use the Landing Zone Accelerator (LZA) on AWS solution to build your landing zone together with AWS Transform for migration capabilities, your AWS Transform account and LZA installation must be in the same AWS Organization. Using separate Organizations IDs for LZA and AWS Transform deployments is not supported because this can cause inconsistencies in organizational management and resource deployments. To learn how to set up your LZA installation using Organizations see Deploy a cloud foundation to support highly-regulated workloads and complex compliance requirements in the *Landing Zone Accelerator on AWS Implementation Guide user guide*.

Getting started with AWS IAM Identity Center

Follow these steps to use IAM Identity Center for AWS Transform and to add users and groups.

By default, no users have access to AWS Transform when you first enable it.

> **Note**
>
> IAM Identity Center is not limited to the region in which it is set up. If you already set up IAM Identity Center in a region that is not supported by AWS Transform, you can use it for AWS Transform.

1. Set up IAM Identity Center following the instructions in To enable an instance of IAM Identity Center.

   Configure IAM Identity Center to use an external enterprise identity provider, and replicate its user and group info into IAM Identity Center.
2. In the AWS console, select AWS Transform and choose **Get started**.
3. Choose **Enable service** for your organization to use AWS Transform.
4. Select an encryption key. The default selection is an AWS managed key. To use a custom key:
   a. Under **Encryption key**, choose **Customize encryption settings**.
   b. Select **Use an AWS KMS key**.
   c. Choose an existing key or create a new one.

d. Choose **Submit** to apply your changes, and then choose **Enable AWS Transform**.

Choose **View profile** to view the configuration. The Web application URL is used by your users to access the AWS Transform unified web experience.

5. Select **Users** in the navigation pane and select **Assign users or groups**.
6. Search for the name of the user or groups you want to authorize to use AWS Transform. The search references users and groups propagated from your identity provider.
7. Select a group or user, select **Done**, and then, **Assign**. These users are authorized to use the AWS Transform unified web interface.

Using third-party identity providers

AWS Transform supports integration with third-party identity providers (IdPs) such as Azure Active Directory (Entra ID) and Okta Workforce Identity. This allows you to use your existing identity management system for user authentication.

Prerequisites

Before configuring third-party identity provider integration, ensure that users in your identity provider have name, email, and username attributes configured

Stored Information

When you use AWS Transform with IdPs, AWS stores minimal user information that is encrypted and secured:

Stored User Information

AWS Transform stores basic user profile information upon first login, including display name, email address, username (preferred_username), and a unique user identifier. This information is encrypted using either a customer-owned KMS key or a service-owned key, depending on the customer's AWS Transform profile configuration. The data is stored in AWS Transform's authentication database and is only collected during the initial login session. This populates the search results when inviting other users to a workspace.

Data Lifecycle

User information is stored only for users who have logged in to the AWS Transform web app at least once, and may become stale if users update their information in their identity provider

without logging back into AWS Transform. All stored user information is deleted when the AWS Transform profile is deleted.

Client Secret Storage

The client secret provided during setup is stored using AWS Secrets Manager via a Service Linked Secret (SLS) in your account.

User Identifier Handling

Entra

Uses the "oid" (object identifier) claim as the unique user identifier, which is immutable and uniquely identifies users across the Microsoft tenant. This value is visible to customers in the Entra console and appears in CloudTrail logs.

Okta Workforce Identity

Uses different claims for user identification depending on token type - the "sub" claim in ID tokens and the "uid" claim in Access tokens. AWS Transform validates that both claims contain the same value during authentication. This value is visible to customers in the Okta console and appears in CloudTrail logs.

Setting up Azure Active Directory (Entra ID)

To configure Azure Active Directory integration with AWS Transform:

1. Navigate to the Azure portal and select **Azure Active Directory**.
2. In the left navigation pane, choose **Manage** > **App registrations**.
3. Choose **+ New registration**.
4. Enter an application name, choose your supported account type, leave the redirect URI blank, and choose **Register**.
5. In the left navigation, choose **Manage** > **Manifest**.
6. Update `requestedAccessTokenVersion` from `null` to `2` and choose **Save**.
7. Choose **Manage** > **Expose an API** and choose **Add a scope**.
8. Create an Application ID URI using the default structure `api://<client-id>`.
9. Add the scope `transform:read_write`.

10. Choose **Add a certificate or secret** and create a new client secret. Save this value as it's needed for profile creation.
11. Find the Issuer URL by choosing **Endpoints** and selecting the OpenID Connect metadata document. The "issuer" field in the metadata is your Issuer URL.
12. Create a profile in the AWS Transform console using the Client ID, Client Secret, and Issuer URL.
13. After profile creation, add a redirect URI by choosing **Add a platform**, selecting **Web**, and entering `<web-application-url>/login/callback`.

Setting up Okta Workforce Identity

To configure Okta Workforce Identity integration with AWS Transform:

1. Navigate to your Okta Workforce Identity console.
2. Choose **Applications** > **Applications** and select **Create App Integration**.
3. Select **OIDC - OpenID Connect** and **Web Application**, then choose **Next**.
4. Name your application, leave the Grant Type as *Authorization Code*, leave redirect URIs blank, configure user assignments, and choose **Save**.
5. Navigate to the **Sign On** tab and set the Issuer to **Okta URL** instead of Dynamic.
6. Copy the Client ID and configure it as the Audience for your Authorization Server by going to **Security** > **API** and adding an Authorization Server.
7. In the Authorization Server, add the scope `transform:read_write` under the **Scopes** tab.
8. Add an Access Policy that allows the OIDC Application to use this Authorization Server and configure a rule for the policy.
9. On the Authorization Server Settings page, note the Issuer URL for profile creation in AWS Transform.
10. Create a profile in AWS Transform using the Issuer URL, Client ID, and Client Secret from the application settings.
11. After profile creation, add `<web-application-url>/login/callback` as a redirect URL in the application's General tab.

> **Note**
>
> If you would like to be redirected back to the AWS Transform web application after logout, you'll need to configure your web application URL as a trusted origin under **Security** > **API**.

User onboarding

This section describes the experience for users who have been granted access to AWS Transform.

Accepting the invitation

When a user is added to AWS Transform, they receive an email invitation containing:

- A greeting and information about the invitation
- The AWS Transform web application URL
- Their username
- A link to accept the invitation and set up their password

To set up their account:

1. The user clicks the "Accept invitation" link in the email.
2. On the "New user sign up" page, they enter and confirm a password.
3. The password must meet security requirements, including:
    - At least 8 characters
    - At least one uppercase letter
    - At least one lowercase letter
    - At least one number
    - At least one special character
4. After creating a password, they see a confirmation that their account was successfully created.

Signing in to AWS Transform

To sign in to AWS Transform:

1. Navigate to the AWS Transform web application URL provided in the invitation email.
2. Enter the username.
3. Choose **Next**.
4. Enter the password.
5. Choose **Sign in**.

Welcome experience

Upon first login, users see the AWS Transform welcome page with:

- A personalized greeting
- Available transformation capabilities
- Option to create a workspace

The welcome page provides information about the transformation capabilities available in AWS Transform, including:

- Modernize IBM z/OS migrations to AWS
- Migrate VMware workloads to Amazon EC2
- Modernize .NET applications to Linux-ready cross-platform .NET
- Assess workloads for migration readiness

Users can start by creating a workspace or asking their team to add them to an existing workspace.

Using your own Amazon S3 bucket

By default, AWS Transform uses a service-managed Amazon S3 bucket to store transformation artifacts. You can choose to use your own Amazon S3 bucket instead for greater control over data storage, encryption, and access policies.

Note

If you have an Amazon S3 bucket for mainframe transformations, you can continue to use that S3 connector and you don't need to use this functionality.

Note

Your Amazon S3 bucket stores only the uploads and transformation artifacts that you interact with. Internal system-generated artifacts are not stored in your bucket. Additionally, AWS Transform indexes the artifacts in your bucket. It stores the indexed data in a service-managed knowledge base that is used to provide you an enriched chat experience.

Prerequisites

Before you configure your own Amazon S3 bucket, make sure that the following requirements are met:

- The bucket must be in the same AWS Region where AWS Transform is enabled.
- The required bucket policy must be applied to your bucket. For more information, see the section called “Required bucket policy”.
- If you use the AWS Transform web application, CORS must be configured on your bucket. For more information, see the section called “Required CORS configuration”.
- If you use a custom AWS KMS key, the required key policy must be applied. For more information, see the section called “KMS key policy (optional)”.

Supported encryption configurations

Your bucket must use one of the following encryption configurations:

- SSE-S3 (AES256) — Leave the KMS key field empty. AWS Transform writes objects without specifying an encryption header, and Amazon S3 applies the bucket's default encryption with S3-managed keys. For more information, see Using server-side encryption with Amazon S3 managed keys (SSE-S3) in the *Amazon S3 User Guide*.

- SSE-KMS with a customer-managed KMS key — Provide the KMS key ARN. AWS Transform writes objects with `x-amz-server-side-encryption: aws:kms` using the provided key. Follow the section called "KMS key policy (optional)".

The following are not supported:

- SSE-KMS with an AWS-managed KMS key (for example, `aws/s3`)
- SSE-C
- DSSE-KMS
- Asymmetric KMS keys or KMS keys with a key spec other than SYMMETRIC_DEFAULT

> **Warning**
>
> AWS-managed KMS keys are not supported. If your bucket is encrypted with `aws/s3`, migrate to a customer-managed key or SSE-S3 before using the bucket with AWS Transform.

Configuring your Amazon S3 bucket

You can configure AWS Transform to use your own Amazon S3 bucket from the AWS Transform console.

To use your own Amazon S3 bucket

1. In the AWS Transform console, choose **Settings**.
2. Under **Artifact storage**, choose **Use my own S3 bucket**. You can use a bucket in your current AWS account or in another account.
3. For **Bucket**, enter the Amazon S3 URI.
4. (Optional) For **KMS key**, enter an AWS KMS key ARN to encrypt objects in the bucket. If you leave this field empty, AWS Transform writes objects without an encryption header, and Amazon S3 applies the bucket's default encryption settings. For more information, see Setting default server-side encryption behavior for Amazon S3 buckets in the *Amazon S3 User Guide*.

If your bucket's default encryption is SSE-KMS, enter the customer-managed KMS key ARN in this field and update your key policy. For more information, see the section called “KMS key policy (optional)”.

5. Choose **Save**. AWS Transform validates the bucket configuration and permissions before applying the configuration.

After you save the configuration, AWS Transform uses your bucket to store transformation artifacts.

> **Note**
>
> AWS Transform validates your configuration only when you save. If you later change the bucket policy, CORS configuration, or KMS key policy, re-save your profile settings to re-validate.

Required bucket policy

You must configure the bucket policy to grant AWS Transform access. Add the following bucket policy to your Amazon S3 bucket to allow the AWS Transform service principal to read, write, delete, and list transformation artifacts.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "transform.amazonaws.com"
                ]
            },
            "Action": [
                "s3:GetObject",
                "s3:PutObject",
                "s3:DeleteObject",
                "s3:AbortMultipartUpload",
                "s3:ListMultipartUploadParts",
                "s3:PutObjectTagging"
```

```
            ],
            "Resource": "arn:aws:s3:::bucket-name/AWSTransform/*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "profile-account-id",
                    "aws:SourceArn": "profile-arn"
                }
            }
        },
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "transform.amazonaws.com"
                ]
            },
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::bucket-name",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "profile-account-id",
                    "aws:SourceArn": "profile-arn"
                }
            }
        }
    ]
}
```

Replace the following values:

- *bucket-name* – The name of your Amazon S3 bucket.
- *profile-account-id* – The AWS account ID associated with your AWS Transform profile.
- *profile-arn* – The ARN of your AWS Transform profile.

Required CORS configuration

If you use the AWS Transform web application, you must configure Cross-Origin Resource Sharing (CORS) on your Amazon S3 bucket. You can find the web application domain after enabling AWS Transform. For more information, see *Getting started*.

```
[
```

```
    {
        "AllowedHeaders": [
            "host",
            "content-type",
            "if-none-match",
            "x-amz-checksum-sha256",
            "x-amz-expected-bucket-owner",
            "x-amz-server-side-encryption",
            "x-amz-server-side-encryption-aws-kms-key-id",
            "x-amz-server-side-encryption-context",
            "x-amz-source-account",
            "x-amz-source-arn"
        ],
        "AllowedMethods": [
            "GET",
            "PUT",
            "HEAD"
        ],
        "AllowedOrigins": [
            "webapp-domain"
        ],
        "ExposeHeaders": [
            "ETag",
            "x-amz-checksum-sha256",
            "x-amz-request-id",
            "x-amz-id-2"
        ],
        "MaxAgeSeconds": 3600
    }
]
```

Replace the following values:

- *webapp-domain* – Your web application origin URL (for example, `https://1a2b3c4d5e6f7a8b9.transform.us-east-1.on.aws`), which can be found on the AWS Transform settings page. **Do not include the trailing slash.**

KMS key policy (optional)

This section applies only when your bucket is encrypted with a customer-managed KMS key. If your bucket uses SSE-S3, skip this section.

If you specify your own AWS KMS key for bucket encryption, add the following statement to your key policy. AWS Transform relies on Forward Access Sessions (FAS) to create grants for its service principal. These grants are used to validate your key and access artifacts in your Amazon S3 bucket.

```
{
    "Sid": "AllowAWSTransformServiceAccess",
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::key-owner-account-id:root"
    },
    "Action": [
        "kms:CreateGrant",
        "kms:DescribeKey"
    ],
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "kms:ViaService": "transform.region.amazonaws.com"
        }
    }
}
```

Replace the following values:

- *key-owner-account-id* – The AWS account ID that owns the KMS key.
- *region* – The AWS Region where AWS Transform is enabled (for example, `us-east-1`).

Switching back to the default bucket

You can switch back to the service-managed bucket at any time.

To switch to AWS Transform managed storage

1. In the AWS Transform console, choose **Settings**.
2. Under **Artifact storage**, choose **Edit**.
3. Choose **AWS Transform managed storage**.
4. Choose **Save**.

Warning

If you switch storage configurations while transformation jobs are in progress, those jobs fail. Any artifacts that were already generated by those in-progress jobs would also not be accessible.

Important

When you switch back to the service-managed bucket:

- Artifacts in your bucket are not automatically migrated. To retain any artifacts, download them from your Amazon S3 bucket and re-upload them through the AWS Transform web application to the artifact store for the corresponding workspace and job.
- AWS Transform retires its grant on your KMS key. Objects written to your bucket remain in your bucket; delete them manually if they are no longer needed.

Uploading files directly to your bucket

You can upload files directly to your Amazon S3 bucket without using the AWS Transform web application. To make uploaded files available to transformation agents, upload them to the `User Uploads` folder for the job. The directory path for this folder uses the following format:

```
AWSTransform/Workspaces/workspace-id/Jobs/job-id/User Uploads/
```

Replace the following values:

- *`workspace-id`* – The ID of your AWS Transform workspace.
- *`job-id`* – The ID of the transformation job.

Each job in AWS Transform has its own `User Uploads` folder. Files that you upload to this path appear in the web application and are available to transformation agents while they complete the job.

File paths must not contain `..`, `//`, or leading or trailing spaces — files that violate these constraints are not visible to agents.

Deleting bucket contents can cause job failures

Deleting `job_objective` will cause job failures for active jobs or if the session is restored. For all other files, deleting them may result in unexpected agent behavior or job failures.

Important

If your bucket uses SSE-KMS, include the following headers on every direct upload:

```
x-amz-server-side-encryption: aws:kms
x-amz-server-side-encryption-aws-kms-key-id: your-kms-key-arn
```

Objects must be encrypted with the KMS key registered in your AWS Transform profile. Objects encrypted with any other key are not accessible to AWS Transform.

Compatibility Matrix

The following table summarizes the compatibility of AWS Transform with common Amazon S3 bucket features.

Feature	Compatibility	Notes
Bucket versioning	Supported	Manage noncurrent versions with a lifecycle rule.
Object Ownership	Required: Bucket owner enforced	Bucket owner preferred and Object writer are not supported.
ACL	Not supported	ACLs must be disabled on the bucket.
Object Lock (governance mode)	Partially supported	Deletes might be blocked by retention settings.
Object Lock (compliance mode)	Not supported	—
Block Public Access	Supported	—

Feature	Compatibility	Notes
Requester Pays	Not supported	—
S3 Access Points	Not supported	Use the bucket name.
S3 Object Lambda Access Points	Not supported	Use the bucket name.
S3 on Outposts	Not supported	—
S3 Express One Zone (directory buckets)	Not supported	—
Cross-Region buckets	Not supported	—
Cross-account buckets	Supported	—
Cross-account KMS keys	Not supported	—
Lifecycle rules	Supported	—
CloudTrail data events	Supported	—

Enable AWS Transform

To enable AWS Transform:

1. Sign in to the AWS Management Console.

2. In the search bar at the top of the console, search for *AWS Transform*.
3. Select **AWS Transform** from the search results.
4. Choose **Get started** to enable the service in your current Region.
5. Optional: configure IAM Identity Center. You will also be able to choose to use a third-party identity provider (IdP) in a later step.

6. Select an **Encryption key**: **default AWS managed key** or **Customize encryption settings.**
7. Optional: Use your own Amazon S3 bucket. By default, AWS Transform uses a service-managed Amazon S3 bucket to store transformation artifacts. You can choose to use your own bucket instead. For more information, see Using your own Amazon S3 bucket.
8. Choose which AWS Transform capabilities you want to enable:
   - **Command line interface** (CLI), needed to create and run custom transformations. To enable the CLI, view and follow the download instructions.
   - **Web application**, the agentic user interface for modernization. Choose **Enable web application** to use it.
9. Choose **Enable AWS Transform**.
10. Optional: choose **Enable View profile** to access the AWS Transform **Users**, **Settings**, and **Connectors** tabs, or **Manage users** to manage users.

    You can access the **Users**, **Settings**, and **Connectors** tabs at any time by choosing the menu icon in the top left corner of the console.
11. Configure User access by choosing one of the following options:
    - IAM Identity Center
    - A third-party identity provider (IdP)
    - IAM-only access

    > **Note**
    >
    > This choice is finalized and cannot be changed when you enable AWS Transform.
12. Choose **Enable web application**.
13. The system displays "Enabling AWS Transform" while it creates the necessary resources.

After AWS Transform is enabled, the **Settings** tab displays the following information:

- **Web application URL** - The URL for accessing the AWS Transform web application
- **Start URL for IDE** - The URL for accessing AWS Transform in integrated development environments
- **Region** - The AWS Region where AWS Transform is enabled

Quick start: Trying AWS Transform

The easiest way to try out AWS Transform is with a standalone AWS account. You may want to do this as a proof-of-concept or for test environments. Use this procedure:

1. Sign in to the AWS Management Console.
2. Navigate to the AWS Transform service.
3. Choose **Get started** to enable the service.
4. Select and set up your identity provider.
5. Assign users to AWS Transform service.
6. After the service is enabled, you'll see the AWS Transform web application URL.
7. Open that URL in a new browser window to access the AWS Transform web experience.

Now you're ready to set up your workspace.

Managing users

How you manage user access to AWS Transform depends on the access model you chose during setup. If you configured IAM Identity Center, you add users through IAM Identity Center. If you configured a third-party identity provider, you manage users in that provider. If you chose IAM-only access, user access is managed through IAM policies.

Adding users in IAM Identity Center

To add users in IAM Identity Center:

1. Navigate to the IAM Identity Center console.
2. In the navigation pane, choose **Users**.
3. Choose **Add user**.
4. Enter the required information:

   - **Username** - A unique identifier for the user (cannot be changed later)
   - **Email address** - The user's email address
   - **First name** and **Last name** - The user's name
   - **Display name** - The name that appears in the user list

5. For **Password**, choose how the user receives their password:
   - **Send an email** - Send setup instructions via email
   - **Generate a one-time password** - Create a password to share manually
6. Choose **Next** to review the user information.
7. Review the details and choose **Add user**.

After the user is added, they'll receive an email invitation to set up their IAM Identity Center account. The invitation link is valid for 7 days.

You can also learn about working with IAM Identity Center and AWS Transform in this video:

VideoTitle

Adding users to AWS Transform

After adding users to IAM Identity Center, you can grant them access to AWS Transform:

1. Return to the AWS Transform console.
2. In the navigation pane, choose **Users and groups**.
3. Select the **Users** tab or the **Groups** tab.
4. Search for and select the users or groups that you want to add from IAM Identity Center.
5. Choose **Assign users and groups** to grant the selected users or groups access to AWS Transform.

After adding users, they appear in the **Users** list with a status of "Pending" until they accept the invitation and sign in.

Managing users with IAM-only access

If you configured AWS Transform with IAM-only access, user access is managed through IAM policies. Any IAM principal with the `transform:AccessTransformProfile` permission on the profile resource can access AWS Transform.

To grant a user or role access to AWS Transform:

1. Navigate to the IAM console.

2. Attach a policy that includes the `transform:AccessTransformProfile` action to the IAM user or role. For an example policy, see Allow users to access AWS Transform with IAM credentials.

To revoke access, remove the policy from the IAM user or role.

Understanding collaborator permissions

AWS Transform uses a workspace-based permission model to control access to resources and actions. Each user is assigned a specific role within a workspace, which determines what actions they can perform. A user can have different roles in different workspaces.

User roles

AWS Transform supports five user roles within each workspace. These roles apply within the context of a workspace, and a user will be assigned roles in each workspace they are a member of. The access permissions defined for each role are workspace agnostic, so user A with the Administrator role in workspace A has the same permissions as user B with the Administrator role in workspace B.

Role permissions

Detailed permissions for each role:

Action	ResourceType	Admin	Approver	Contributor	ReadOnly
Create	Workspace	✓	✓	✓	✓
List	Workspace	✓	✓	✓	✓
Get	Workspace	✓	✓	✓	✓
Update	Workspace	✓	✗	✗	✗
Delete	Workspace	✓	✗	✗	✗
Create	ChatMessage	✓	✓	✓	✗
Read	ChatMessage	✓	✓	✓	✓

Action	ResourceType	Admin	Approver	Contributor	ReadOnly
Create	RoleAssociation	✓	✗	✗	✗
Read	RoleAssociation	✓	✓	✓	✓
Update	RoleAssociation	✓	✗	✗	✗
Delete	RoleAssociation	✓	✗	✗	✗
Read	CriticalHITLTask	✓	✓	✓	✓
Update	CriticalHITLTask	✓	✓	✗	✗
Delete	CriticalHITLTask	✓	✓	✗	✗
Read	HITLTask	✓	✓	✓	✓
Update	HITLTask	✓	✓	✓	✗
Delete	HITLTask	✓	✓	✓	✗
Create	Job	✓	✓	✓	✗
Read	Job	✓	✓	✓	✓
Update	Job	✓	✓	✓	✗
Delete	Job	✓	✓	✓	✗
Read	Worklog	✓	✓	✓	✓
Create	Artifact	✓	✓	✓	✗
Read	Artifact	✓	✓	✓	✓
Update	Artifact	✓	✓	✓	✗
Delete	Artifact	✓	✓	✓	✗
Create	Connector	✓	✓	✓	✗

Action	ResourceType	Admin	Approver	Contributor	ReadOnly
Read	Connector	✓	✓	✓	✓
Update	Connector	✓	✓	✓	✗
Delete	Connector	✓	✓	✓	✗

Human-in-the-loop (HITL) actions

AWS Transform provides two types of HITL actions - standard and critical:

Standard HITL actions

These are routine actions that can be performed by users with Contributor, Approver, or Administrator roles.

Critical HITL actions

These are actions with significant impact, and thus require higher permission levels. Examples include:

- Merging code to main branches
- Performing graph decomposition
- Deploying code to production environments

Critical HITL actions can only be performed by users with Approver or Administrator roles.

To ensure there's a differentiation between Standard HITL and Critical HITL actions in AuthZ policies, AWS Transform provides two separate HITL APIs, one for completing a standard HITL action, and one for completing a critical HITL action.

AWS Transform environment

AWS Transform is an agentic service that uses natural language processing to help you plan and execute your workload transformations. You interact with AWS Transform primarily through a conversational interface, where the service adapts and responds based on the context of your

discussion. For example, you can start by describing your current architecture and transformation goals using everyday language, such as "I need to migrate my on-premises VMs to EC2."

As you converse with AWS Transform, the service builds a customized job plan that aligns with your specific requirements. The conversation flow is dynamic and driven by you, allowing you to refine your transformation plan iteratively. You can modify the job plan during the conversation by making requests like "add a testing phase before production cutover" or "remove the backup step since we already have a solution." You can also go back to a previous task and perform it again. AWS Transform continuously updates the plan based on your input, ensuring the final transformation strategy meets your technical and business needs.

Here's what you see when you open AWS Transform.

- View control pane: This is the narrow pane on the left of AWS Transform. You can choose one of the icons to choose what view is shown to the right of the view control pane. Hover over each icon for a tool tip explaining the view. The give standard views, from top to bottom, are:
  - **Job Plan**
  - **Dashboard**
  - **Approvals**
  - **Artifacts**
  - **Worklog**

  Some workflows provide additional views.

  When you are working in a job plan, the views are:
  - **Chat**
  - **Approvals**
  - **Artifact store**
  - **Worklog**

  When you are in a workspace, the views are at the top:
  - **Jobs**
  - **Artifacts**
  - **Collaborators (Users)**
  - **Connectors**
  - **Settings**

- The **View** pane is next to the view controls.
- The **Chat** pane is in the center. This is where you conduct your conversation with AWS Transform.

> **Note**
>
> Users with read-only permissions are unable to send messages in the chat.

- To the right is the **Collaboration** pane. This appears when *human in the loop* (HITL) activities are performed, such as:
  - Uploading data files
  - Reviewing information and plans provided by AWS Transform

Language settings

AWS Transform supports localization for select workflows. You can change the display language of the web application by choosing the **Settings** icon in the top right corner and selecting your preferred language from the **Language** menu.

The following languages are supported:

- English (US)
- English (UK)
- Deutsch (German)
- Español (Spanish)
- Français (French)
- 日本語 (Japanese)
- Bahasa Indonesia (Indonesian)
- Italiano (Italian)
- Português (Portuguese)
- 한국어 (Korean)
- 中文 (简体) (Chinese Simplified)
- 中文 (繁體) (Chinese Traditional)
- Türkçe (Turkish)

The following workflows support localization:

- Migrations (including VMware)

> **Note**
>
> Other AWS Transform workflows are currently available in English only. Localization support for additional workflows may be added in future updates.

Start your project

AWS Transform guides you through your modernization and migration projects. To get started:

1. Create a workspace to host your project. On the **Workspaces** page, choose **Create workspace**. Follow the instructions and then open the workspace.
2. In the **Chat** pane, type "create a job"
3. Choose a job, and follow the instructions provide by AWS Transform. If your input is required, the **Collaboration** pane appears and explains what is required.

Workflow flexibility

The AWS Transform workflow environment provides flexibility in the progression of your modernization projects. Using natural language in the chat pane you can:

- Retry a task that AWS Transform has already completed, providing different human-in-the-loop input along the way.
- Rerun a job, for example, if there have been changes in your modernization sources.

Chatting with AWS Transform

AWS Transform chat is available at every stage of your project. To open the chat click the purple hexagonal icon in the lower right corner of the web console.

The chat is there for you to ask anything. For example, you can ask the chat to explain concepts, guide you through a process, explain a AWS Transform request or response, or explain a AWS Transform report.

> **Note**
>
> Users with read-only permissions are unable to send messages in the chat.

AWS Transform chat integrations

AWS Transform chat is integrated with:

Experience-Based Acceleration

Experience-Based Acceleration (EBA) is offered through AWS Transform chat. It enables you to perform EBA assessment for Windows workloads and generate a plan. You can start by importing assessment results from CAST. It helps you discover your application portfolio, after which you can use AWS Transform chat to select applications that meet your business needs (for example, filtering applications based on complexity or lines of code). You can then perform deeper assessment of the selected application and generate a modernization plan.

AWS Countdown Premium

AWS Countdown Premium (CDP) integrated into AWS Transform provides sustained, expert guidance with designated AWS engineering support. Your designated CDP Engineer dives deep into your tech stack, providing personalized, context-aware support when issues arise. Users can leverage AWS Transform to log a support ticket directly from its chat interface. The support ticket embeds worklog and job plan details to help support understand the customer's context. If a customer or partner is part of the CDP Program, the support ticket automatically gets routed to the designated CDP Engineer who can help debug issues, interpret transformation outputs, and drive progress for expedited issue resolution. For example:

- For .NET workloads, CDP can assist with repository connector issues, dependency resolution, and post-transformation deployment.
- In mainframe scenarios, CDP can help troubleshoot refactored Java code, configure database migration tools, and build CI/CD pipelines.
- In VMware migrations, CDP can accelerate network configuration, MGN setup, and agent installation.

AWS Skill Builder

AWS Skill Builder provides relevant learning modules through the chat. You can ask AWS Transform chat about your learning needs, and it presents you relevant course catalog. Skill Builder delivers contextual micro-learning experiences as you work through transformation stages.

AWS Transform Connectors

AWS Transform connectors enable you to securely access resources across account boundaries for migration and modernization workflows. This enables you to access and manage resources across account boundaries while maintaining security controls and permissions. A connector represents a connection between your AWS Transform-enabled source account and external resources in AWS target accounts or in third-party systems.

Connectors require requests for access from the *source connector*, and approval from the owner of the target account. You make this request when you are setting up a source connector. Similarly, you may receive requests to approve *destination connectors* to allow other accounts access to your account.

- **Connector creation:** Create connectors in your AWS Transform-enabled source account, specifying the target account and required permissions
- **Permission Setup:** AWS Transform requires specific IAM roles and permissions in the destination account to perform migration actions, including:
  - Managing migration data with AWS KMS
  - Making cross-region API calls
  - Installing replication agents on VMware servers
  - Creating network infrastructure (VPCs, subnets, routing)
  - Launching Amazon EC2 instances and deploying CloudFormation stacks
  - Executing migration workflows through Migration Hub
- **Approval Process:** You must approve destination connector requests before AWS Transform can access your resources.
- **Active Connection:** Once you approve a connector, it enables AWS Transform jobs to securely access and manage resources in the target account.

User traceability for agentic operations

By default, AWS CloudTrail records agent actions on your AWS resources under the AWS Transform service identity. User traceability attributes those actions to the specific AWS IAM Identity Center user who initiated or interacted with the job, enabling security auditing and incident investigation at the individual-user level.

> **Note**
>
> User traceability is supported for IAM Identity Center (IDC) users only.

To enable user traceability, enable **Background session** in your AWS Transform profile settings. When enabled, AWS Transform automatically creates an IAM Identity Center background session when you interact with a job and uses it to produce identity-enhanced IAM role sessions for connector operations. These sessions embed your user identity in downstream AWS service calls and AWS CloudTrail events. When background sessions are disabled, AWS Transform falls back to standard assumed-role sessions without user identity context.

New connectors are automatically configured to support user traceability. For existing connectors, you must manually add `sts:SetContext` to the connector role's trust policy:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": { "Service": ["transform.amazonaws.com"] },
      "Action": ["sts:AssumeRole", "sts:SetContext"],
      "Condition": {
        "StringEquals": { "aws:SourceAccount": "your-account-id" }
      }
    }
  ]
}
```

You can view and revoke your active background sessions from the **Background sessions** tab on the job page. Revoking a session immediately invalidates it and any connector sessions derived from it.

For more information, see Trusted identity propagation overview in the *AWS IAM Identity Center User Guide*.

Managing connectors

On the Connectors page you can see the:

- **Source Connectors Tab:** Lists all connectors you've created to connect to other accounts, showing target accounts and connector status
- **Destination Connectors Tab:** Shows incoming connector requests from other accounts wanting to access your resources, and requiring your approval or rejection

AWS Transform requires confirmation for critical actions such as approving, rejecting, or deleting connectors, to avoid accidental changes that could disrupt your active migration workflows.

To create a connector

1. Navigate to the **Source Connectors** tab.
2. Specify the target account and resource type.
3. Configure required permissions and access scope.

To set up permissions

1. Choose to create a new IAM role with required permissions, or
2. Select an existing role you previously created for connectors.
3. Ensure the role has all necessary permissions for AWS Transform operations.

To manage incoming requests

1. Review connector requests in the **Destination Connectors tab**.
2. Approve legitimate requests to enable cross-account access.
3. Reject unauthorized or unnecessary connection attempts.

To manage your connectors

1. Monitor your active connectors for security and compliance.

2. Delete connectors when you no longer need them. Note that this will cause running jobs using the connector to fail.
3. Update permissions as your requirements change.

Developer tools

You can access AWS Transform agents from agentic IDEs, agent plugins, and the MCP server. Choose the surface that fits your workflow.

> **Note**
>
> The Kiro Power and agent plugins automatically install the MCP server. You only need to install the MCP server manually if you want to use it without a Kiro Power or agent plugin.

Kiro Power

The AWS Transform Kiro Power provides full access to AWS Transform agents from the Kiro IDE.

To install the AWS Transform Kiro Power

1. Open Kiro IDE and navigate to the Powers panel.
2. Find **AWS Transform** in the list and install it.
3. Open Kiro Chat, click on the Power, and select **Try power**.

Alternatively, in the Powers panel choose **Add Custom Power** > **Import power from GitHub** and paste: `https://github.com/kirodotdev/powers/tree/main/aws-transform`

Agent plugin

The AWS Transform agent plugin is available at awslabs/agent-plugins on GitHub, in the `plugins/aws-transform` directory. The plugin supports Claude Code, Codex, and Cursor.

The plugin includes `.claude-plugin`, `.codex-plugin`, and `.mcp.json` configurations. It provides access to all AWS Transform agents, such as .NET, mainframe, VMware, SQL Server, and custom transformations.

To install in Claude Code

Run the following commands:

```
/plugin marketplace add awslabs/agent-plugins
/plugin install aws-transform@agent-plugins-for-aws
```

To install in Codex or Cursor

Follow the instructions in the agent-plugins repository README.

IDE plugin

The AWS Transform IDE plugin is available on VS Code and Open VSX compatible editors. It exposes AWS Transform custom features, including creating and running transformation definitions. It also exposes mainframe features, including traceability between modernization requirements, business rules, and source code, as well as the ability to generate technical documentation on demand. To get started, install using one of the following methods:

- **VS Code:** Install from the VS Code Marketplace.
- **Open VSX:** Install from the Open VSX registry.

MCP server

Use the MCP server to integrate AWS Transform programmatically with any MCP-compatible client. You can install the MCP server from PyPI as `awslabs.aws-transform-mcp-server`. It provides 19 tools for workspace management, job management, human-in-the-loop (HITL) tasks, artifact handling, connector management, chat, and resource browsing.

Prerequisites

- Python 3.10 or later

Installation

Claude Code

```
claude mcp add awslabs.aws-transform-mcp-server -- uvx awslabs.aws-transform-mcp-server@latest
```

Kiro

Add to `~/.kiro/settings/mcp.json`:

```
{
  "mcpServers": {
    "awslabs.aws-transform-mcp-server": {
      "command": "uvx",
      "args": ["awslabs.aws-transform-mcp-server@latest"]
    }
  }
}
```

For configuration details for your specific MCP client, see the aws-transform-mcp-server README on GitHub.

Authentication

AWS Transform supports SSO (AWS Identity and Access Management Identity Center) and IAM role-based authentication.

SSO (IAM Identity Center)

Ask your AI assistant to "Configure AWS Transform with SSO." The assistant provides a start URL and opens your browser for login.

IAM role

Your environment provides AWS credentials automatically. Set `AWS_PROFILE` in your MCP client configuration `env` block to select a specific profile.

For information about setting up authentication for AWS Transform, see Getting started with AWS Transform.

Build custom agents for AWS Transform

The agent builder toolkit enables AWS Partners and customers to build agents tailored to their specific modernization needs that work within AWS Transform.

To install the agent builder toolkit

1. Open Kiro IDE and navigate to the Powers panel.

2. Find **AWS Transform Agent Toolkit** in the list and install it.

Alternatively, in the Powers panel choose **Add Custom Power** > **Import power from GitHub** and paste: `https://github.com/kirodotdev/powers/tree/main/aws-transform-agent-toolkit`

The agent builder toolkit enables Migration and Modernization Competency Partners, ISVs, or customers to create transformation solutions for specific use cases. You can integrate specialized agents, tools, knowledge bases, and workflows with AWS Transform agentic AI capabilities.

Agent lifecycle

Build

Use the Kiro Power to combine the base agent and SDK to write your own customized agent.

Share

Distribute agents with teams or across partner networks.

Register

Register agents with AWS Transform for discovery by other users.

Custom

What is AWS Transform custom?

AWS Transform custom uses agentic AI to perform large-scale modernization of software, code, libraries, and frameworks to reduce technical debt. It handles diverse scenarios including language version upgrades, API and service migrations, framework upgrades and migrations, code refactoring, and organization-specific transformations.

Through continual learning, the agent improves from every execution and developer feedback, delivering high-quality, repeatable transformations without requiring specialized automation expertise.

Key Capabilities

AWS Transform custom provides the following capabilities:

- **Natural language-driven transformation definition** - Create custom transformations using natural language, documentation, and code samples
- **Transformation execution** - Apply transformations consistently and reliably across multiple codebases
- **Continual learning** - Automatically improve transformation quality from every execution
- **AWS-managed transformations** - Use ready-to-use, AWS-vetted transformations for common scenarios

Transformation Patterns

AWS Transform custom supports diverse transformation patterns to address your modernization needs. Each pattern has different complexity characteristics based on the scope and nature of the changes required.

Pattern	Description	Complexity	Examples
API and Service Migrations	Migrating between API versions or equivalent services	Medium	AWS SDK v1→v2 (Java, Python, JavaScript),

Pattern	Description	Complexity	Examples
	while maintaining functionality		Boto2→Boto3, JUnit 4→5, javax→jakarta
Language Version Upgrades	Upgrading to newer versions of the same programming language, adopting new features and replacing deprecated functionality	Low-Medium	Java 8→17, Python 3.9→3.13, Node.js 12→22, TypeScript version upgrades
Framework Upgrades	Upgrading to newer versions of the same framework, addressin g breaking changes	Medium	Spring Boot 2.x→3.x, React 17→18, Angular upgrades, Django upgrades
Framework Migrations	Migrating to entirely different framework s that serve similar purposes	High	Angular→React, Redux→Zustand, Vue.js→React
Library and Dependency Upgrades	Upgrading third-party libraries to newer versions while maintaining the same language and framework	Low-Medium	Pandas 1.x→2.x, NumPy upgrades, Hadoop/HBase/Hive library upgrades, Lodash upgrades
Code Refactori ng and Pattern Modernization	Modernizing code patterns and adopting best practices without changing external functionality	Low-Medium	Print→Logging frameworks, string concatenation→f-strings, type hints adoption, observabi lity instrumentation

Pattern	Description	Complexity	Examples
Script and File-by-File Translations	Translating independent scripts or configuration files where files are mostly self-contained	Low-Medium	AWS CDK→Terraform, Terraform→CloudFormation, Excel→Python notebooks, Bash→PowerShell
Architecture Migrations	Migrating between hardware architectures or runtime environments with minimal code changes	Medium-High	x86→AWS Graviton (ARM), on-premises→Lambda, traditional server→containers
Language-to-Language Migrations	Translating codebases from one programming language to another	Very High	Java→Python, JavaScript→TypeScript, C→Rust, Python→Go
Custom and Organization-Specific Transformations	Unique organizational requirements and specialized modernization needs	Varies	Custom internal library migrations, organization-specific coding standards, proprietary framework migrations

Note

For COBOL/mainframe languages, use AWS Transform for Mainframe. For .NET Framework upgrades to .NET Core, consider AWS Transform for Windows. For VMware migrations to AWS, consider AWS Transform for VMware.

How AWS Transform custom Works

AWS Transform custom is usually used in large-scale projects where multiple codebases or modules are transformed. Teams typically follow a four-phase workflow:

Define Transformation - Provide natural language prompts, documentation, and code samples to the agent, which generates an initial transformation definition. This definition can be iteratively refined through chat or direct edits. This phase can be skipped when using AWS-managed transformations.

Pilot or Proof-of-Concept - Test the transformation on sample codebases and refine based on results. This validation phase helps estimate the cost and effort of the full transformation. Continual learning improves quality during this phase.

Scaled Execution - Set up automated bulk execution using the CLI, with developers reviewing and validating results. Monitor progress using the web application and track transformations across multiple repositories.

Monitor and Review - Continual learning automatically improves the transformation quality. Review the lessons that have been extracted from previous runs to ensure they meet quality standards, and archive any that are not useful.

Understanding Key Concepts

This section explains key concepts for working with AWS Transform custom.

Transformation Definitions

A **transformation definition** contains the instructions and knowledge needed to perform a specific code transformation. It is represented as a skill and consists of:

- `SKILL.md` (required) - Instructions in Markdown with YAML frontmatter (`name` and `description` fields) that contains the core transformation logic and execution instructions
- `references/` folder (optional) - Documentation loaded as needed during transformation execution
- `scripts/` folder (optional) - Scripts that the transformation downloads and runs during execution

AWS Transform CLI automatically downloads transformation definitions to the current directory when needed for execution, inspection, or modification.

> **Important**
>
> When publishing a transformation, the directory must contain only `SKILL.md` and optionally the `references/` folder, the `scripts/` folder, or both. No other files or subdirectories are allowed.

> **Note**
>
> CLI version 2.0 and later uses the skills format. Existing transformation definitions are automatically handled by the CLI.

Transformation definition categories

A transformation definition can have a **category** that says how it is used:

- `analysis` – Reads code and reports findings. Continuous modernization runs these transformation definitions as a custom analysis.
- `remediation` – Changes code. Continuous modernization uses these transformation definitions to remediate findings.

Set the category with the `--category` option of `atx custom def publish` or `atx custom def save-draft`:

```
atx custom def publish -n <transformation-name> --description "Description" --sd
 <directory> --category analysis
atx custom def publish -n <transformation-name> --description "Description" --sd
 <directory> --category remediation
atx custom def list    # The Category column shows each transformation definition's
 category
```

AWS Transform uses the category to decide where a transformation definition can be used.

Use	Allowed transformation definitions
Custom analysis (`--type custom`)	Your own transformation definitions with the `analysis` category
Remediation with `--transformation-name`	Your own transformation definitions with the `remediation` category, or AWS managed transformations

Before submitting a custom analysis, schedule, or remediation, AWS Transform checks the transformation definition's category.

To change a transformation definition's category, publish it again with a different `--category` value. If you publish a new version without `--category`, the new version keeps the category of the previous version.

For how to run an `analysis` transformation definition, see Custom analysis.

> **Note**
>
> A category is different from a resource tag. To manage resource tags, see Tag Commands.

Write a transformation definition for analysis

To get useful findings, write the transformation definition so that it describes what to look for, how to classify each issue, and what not to do. We recommend the following sections in `SKILL.md`.

Section	What to include
Introduction	One or two sentences that state what the analysis finds. State that it is read-only and must not modify any files.
Entry criteria	When the analysis applies, for example "The repository contains a `pom.xml` or `build.gradle` file."

Section	What to include
Scope	The files and directories to inspect, and the ones to skip, such as build output and vendored dependencies.
What to report	The specific patterns or conditions that count as an issue. Ask for one finding per issue, with the file path and line number.
Categories	A small, fixed set of category values, and which issues belong in each.
Severity criteria	What makes an issue `high`, `medium`, or `low`.
Exit criteria	How the agent knows it is done: every in-scope file was inspected, every finding has a file, category, and severity, and no file was changed. Also state what to report when nothing is found.

The following example finds Java projects that still target a Java version earlier than 17.

```
---
name: java-version-detector
description: Report where a repository still targets a Java version earlier than 17, in
 build files, Dockerfiles, toolchain files, and CI workflows. Analysis only - no code
 is modified.
---

# Java Version Detector

Find where the repository targets a Java version earlier than Java 17. This is a
read-only analysis: report findings, and do not modify any files.

## Entry Criteria

The repository contains a Java build file (`pom.xml`, `build.gradle`,
`build.gradle.kts`, or `gradle.properties`), or a file that selects a JDK
(`.java-version`, `.tool-versions`, `.sdkmanrc`, a `Dockerfile`, or a file under
```

```
`.github/workflows/`).

## Scope

- Inspect build files, toolchain files, Dockerfiles, and CI workflow files.
- Skip build output (`build/`, `target/`) and vendored dependencies.

## What to Report

Report one finding for each place that selects a Java version earlier than 17:

1. A compile target, such as `maven.compiler.source`, `maven.compiler.release`,
   `java.version`, `sourceCompatibility`, or a Gradle toolchain `languageVersion`.
2. A Dockerfile base image that ships a JDK or JRE earlier than 17, such as
   `openjdk:8` or `amazoncorretto:11`.
3. A CI job or toolchain file that selects a JDK earlier than 17.

For each finding, include the file path, the line number, and the version you found.

## Categories

- `outdated-compile-target` - a build file compiles for an old Java version
- `outdated-runtime-image` - a Dockerfile uses an old JDK or JRE image
- `outdated-ci-jdk` - a CI workflow or toolchain file selects an old JDK

## Severity Criteria

- **high** - the project compiles for Java 8 or earlier
- **medium** - the project compiles for Java 9 through 16
- **low** - only the runtime image or CI uses an old JDK, and the compile target is 17
 or later

## Exit Criteria

1. Every in-scope file has been inspected.
2. Every finding has a file path, line number, category, and severity from the lists
 above.
3. No file in the repository has been modified.
4. If nothing is found, state that the repository targets Java 17 or later.
```

When this transformation definition runs against a project whose `pom.xml` sets `<source>1.8</source>`, it produces a `high` severity finding in the `outdated-compile-target` category, with `AWS/java-version-upgrade` as its fix.

Tips for writing analysis transformation definitions

- **Keep it read-only.** State explicitly that the analysis must not modify files. Fixing issues is the job of a remediation.
- **Be specific about what counts as an issue.** Name the files, settings, and patterns to look for. Vague instructions produce inconsistent findings.
- **Ask for one finding per issue.** Each finding should point to a specific file and line so that it can be reviewed and remediated individually.
- **Define severity and categories in the transformation definition.** The agent uses your criteria to classify each finding.
- **Keep the scope focused.** A custom analysis reports only what its transformation definition describes. To find many kinds of issues, write separate transformation definitions or use the built-in analysis types.
- **Say what to do when nothing is found.** A repository that has no issues correctly produces zero findings.
- **Include the finding instructions in the transformation definition.** If you pass configuration with `-g` or `--configuration` when you run the analysis, AWS Transform uses your configuration instead of its default instructions. Keeping the "What to Report" section in `SKILL.md` makes the transformation definition produce findings either way.

Test the transformation definition on a repository that you know contains the issue before you publish it. Save it as a draft with `atx custom def save-draft`, or run it locally with `atx ct analysis run` (see Custom analysis).

Transformation Registry

The **transformation registry** is your AWS account's centralized repository for storing and managing transformation definitions. Transformations in the registry can be:

- Listed using `atx custom def list`
- Executed across multiple codebases
- Shared with other users in your AWS account
- Version-controlled

> **Important**
>
> Transformation definitions are account-specific. If you want to use a transformation in a different AWS account, you must publish it separately in that account.

Draft vs Published Transformations

AWS Transform custom supports two states for transformations in the registry:

Draft transformations are in-progress or untested transformation definitions. They are saved as specific versions and can be retrieved, updated, and executed by users referring to that specific version. Drafts are useful for iterative development, testing, and refinement before the transformation is ready to be shared with your team. Drafts are also associated to specific conversation. If you restart the CLI, you can use `atx --conversation-id {id}` to restore a previous one.

Published transformations are available in your account's transformation registry for execution by other users with the required IAM permissions. Published transformations can be discovered using `atx custom def list`.

The typical workflow is:

1. Create transformation locally
2. Save as draft for testing (`atx custom def save-draft`)
3. Refine and validate
4. Publish to share with your team (`atx custom def publish`)

You can also publish a transformation directly without saving it as a draft.

References vs. Lessons

AWS Transform custom uses two types of knowledge to improve transformation quality:

References are user-provided documentation stored in the `references/` folder of a transformation definition. References support text files only (maximum 10MB total for all files) and usually contain documentation, API specifications, migration guides, and code samples. References are loaded as needed during transformation execution. You add references when creating or updating a transformation definition in interactive mode.

Lessons are automatically extracted from previous runs of a transformation. The continual learning system generates them from execution trajectories, developer feedback, and code fixes it encounters during transformations. It groups related lessons into categories so you can review them together. Unlike references, which you provide upfront, lessons accumulate over time as you run the transformation across different codebases, and the system applies them automatically to improve future runs. You review and manage lessons interactively with `atx custom def learnings`.

Build and Validation Commands

The **build or validation command** is an optional parameter that specifies how to validate your code during the transformation process. This command is executed at various points during the transformation to ensure code integrity.

Providing a command that validates the results and returns issues if validation fails is very important to improve transformation quality through continual learning. If no build or validation is needed, omit from your input.

For examples and detailed guidance, see Build and Validation Commands in the Workflows section.

Continual Learning

Continual learning is the system that automatically captures feedback from every transformation execution and improves transformation quality over time. The system gathers information through:

- Explicit feedback - Comments and code fixes provided in interactive mode
- Implicit observations - Issues the agent encounters while transforming and debugging code

The continual learning system processes this information to create **lessons** that it adds to the transformation definition to improve future transformations. The system runs the learning process automatically after transformations complete, requiring no additional input from you.

> **Important**
>
> Lessons are transformation-specific and are not shared across different transformations or different customer accounts.

Managing Lessons

The system generates lessons automatically, but you stay in control of which ones it applies to future runs. The `atx custom def learnings` command opens an interactive terminal session for reviewing and curating the lessons a transformation has accumulated.

In this session, you can:

- Browse lessons grouped into categories, and see how many active lessons each category contains.
- Open a lesson to read its full detail, including the lesson body, its impact, and how many previous runs consulted it.
- Archive a lesson to prevent the system from applying it to future runs, and restore it later if needed.
- Permanently delete an archived lesson that is no longer useful.

For step-by-step instructions, see the section called “Continual Learning”.

Client-Side Skills

Client-side skills are additional capabilities that extend the agent during transformation executions. Unlike transformation definitions which define what transformation to perform, client-side skills provide supplementary tools, scripts, and instructions that the agent can use alongside its built-in capabilities during any transformation.

Client-side skills are discovered from project-level and user-level directories. Place skills at the project level when they enforce standards specific to a repository, and at the user level when they apply across all your projects. Project-level skills are only available when a code repository path is provided. For detailed configuration and usage, see the section called “Client-Side Skills”.

Getting started with AWS Transform custom

This section describes how to choose an interface, set up AWS Transform custom, and run your first transformation.

Choose where to run AWS Transform custom

Run AWS Transform custom from the command line, Kiro Power, an agent plugin, or an IDE plugin. Each option provides access to the same AWS-managed and custom transformation definitions.

> **Important**
>
> **Recommended for guided setup:** Use Kiro Power or the AWS Transform agent plugin. During setup, the agent checks prerequisites and installs or updates the AWS Transform CLI when needed. It then guides you through your first transformation. You still need appropriate AWS permissions and access to the repository that you want to transform.

AWS Transform CLI

Use the AWS Transform CLI for terminal workflows, scripts, and automation. With this option, you complete setup manually. Review the the section called “AWS Transform CLI prerequisites”, then the section called “Installing the AWS Transform CLI”.

Kiro Power

Use the AWS Transform Kiro Power for a guided, conversational workflow in Kiro IDE.

1. Open Kiro IDE and navigate to the Powers panel.
2. Find **AWS Transform** and install it.
3. Open Kiro Chat, choose the Power, and then choose **Try power**.

After installation, ask Kiro to set up AWS Transform custom and show you the transformations available for your repository.

For the GitHub import option and authentication information, see the section called “Kiro Power”.

Agent plugin

Use the AWS Transform agent plugin with Claude Code, Codex, or Cursor in the repository that you want to transform.

To install in Claude Code

Add the AWS agent plugin marketplace and install the AWS Transform plugin:

```
/plugin marketplace add awslabs/agent-plugins
/plugin install aws-transform@agent-plugins-for-aws
```

To install in Codex or Cursor

Follow the installation instructions in the agent-plugins repository README on the GitHub website. For more information, see the section called “Agent plugin”.

After installation, ask your coding agent to set up AWS Transform custom and show you the transformations available for your repository.

IDE plugin

Use the AWS Transform IDE plugin to create and run transformation definitions from VS Code or an Open VSX compatible editor.

- **VS Code:** Install the plugin from the VS Code Marketplace.
- **Open VSX:** Install the plugin from the Open VSX registry.

For more information, see the section called “IDE plugin”.

If you chose the AWS Transform CLI, continue with the CLI prerequisites that follow. If you chose another option, follow the setup instructions in the selected tab.

For centralized monitoring of transformation campaigns across multiple repositories, see the section called “AWS Transform Web Application (Optional)”.

AWS Transform CLI prerequisites

Before installing AWS Transform custom, ensure you have the following:

Supported Platforms

AWS Transform custom supports the following operating systems:

- **Linux** - Full support for all Linux distributions
- **macOS** - Full support for macOS systems
- **Windows** - Full support for native Windows using Windows Terminal and PowerShell (The `atx ct` commands are not yet supported on Windows)

- **Windows Subsystem for Linux (WSL)** - Supported when running under WSL

Required Software

- **Node.js 22 or later** - Download from https://nodejs.org/en/download
- **Git** - Must be installed and the working directory needs to be a valid Git repo to execute a transformation. Run `git init; git add .; git commit -m "Initial commit"` to initialize a Git repo in your working directory.

Network Requirements

An internet connection with access to the following endpoints is required:

- `transform-cli.awsstatic.com`
- `transform-custom.<region>.api.aws`
- `*.s3.amazonaws.com`

If you are working in an internet-restricted environment, update firewall rules to allowlist these URLs.

Installing the AWS Transform CLI

The recommended installation method is using the installation script. Choose the tab for your operating system.

To install the AWS Transform CLI

1. Run the installation script:

   Linux and macOS

   ```
   curl -fsSL https://transform-cli.awsstatic.com/install.sh | bash
   ```

   Windows (PowerShell)

   On native Windows, run the installation script from Windows Terminal using PowerShell.

   ```
   irm https://transform-cli.awsstatic.com/install.ps1 | iex
   ```

2. Verify the installation:

```
atx --version
```

   The command should display the installed version of the AWS Transform CLI.

Configuring Authentication

AWS Transform custom requires AWS credentials to authenticate with the service. Configure authentication using one of the following methods.

Environment Variables

Set the following environment variables.

Linux and macOS

```
export AWS_ACCESS_KEY_ID=your_access_key
export AWS_SECRET_ACCESS_KEY=your_secret_key
export AWS_SESSION_TOKEN=your_session_token
```

Windows (PowerShell)

```
$env:AWS_ACCESS_KEY_ID="your_access_key"
$env:AWS_SECRET_ACCESS_KEY="your_secret_key"
$env:AWS_SESSION_TOKEN="your_session_token"
```

You can also specify a profile using the `AWS_PROFILE` environment variable.

Linux and macOS

```
export AWS_PROFILE=your_profile_name
```

Windows (PowerShell)

```
$env:AWS_PROFILE="your_profile_name"
```

> **Note**
>
> Environment variables set with `$env:` in PowerShell apply to the current session only. To persist them across sessions, set them through **System Properties** > **Environment Variables**, or use `[System.Environment]::SetEnvironmentVariable()`.

AWS Credentials File

Configure credentials in the AWS credentials file, located at `~/.aws/credentials` on Linux and macOS, or `%USERPROFILE%\.aws\credentials` on Windows:

```
[default]
aws_access_key_id = your_access_key
aws_secret_access_key = your_secret_key
```

IAM Permissions

Your AWS credentials must have permissions to call the AWS Transform custom service. We recommend attaching the `AWSTransformCustomFullAccess` AWS managed policy to your IAM user or role. This policy provides full access to AWS Transform custom, including the permission to create the service-linked role required for CloudWatch metrics emission to your account.

For more granular control, AWS Transform custom also provides the following AWS managed policies:

- `AWSTransformCustomExecuteTransformations` – Provides access to execute transformations.
- `AWSTransformCustomManageTransformations` – Provides access to create, update, read, and delete transformation resources, as well as execute transformations.

For more information about these policies, see AWS managed policies for AWS Transform. For custom IAM policies with resource-level permissions, refer to the AWS Transform Custom IAM Service Authorization Reference Guide.

Note

- Transformation definitions are AWS resources with ARNs (Amazon Resource Names). You can use IAM policies to control access to specific transformations or groups of transformations by specifying the transformation ARN in your IAM policy resource statements. Tags can be used for grouped access control.
- AWS IAM Identity Center is required to access the AWS Transform, but is not required to use the CLI.

Configuring AWS Region

AWS Transform custom is available in specific AWS regions and automatically detects your region configuration using standard AWS CLI precedence.

Supported Regions

AWS Transform custom is available in the following AWS regions:

- `us-east-1` (US East - N. Virginia)
- `eu-central-1` (Europe - Frankfurt)
- `eu-west-2` (Europe - London)
- `ca-central-1` (Canada - Central)
- `ap-northeast-1` (Asia Pacific - Tokyo)
- `ap-northeast-2` (Asia Pacific - Seoul)
- `ap-southeast-2` (Asia Pacific - Sydney)
- `ap-south-1` (Asia Pacific - Mumbai)

How Region is Determined

The CLI checks these sources in priority order to determine which region to use:

1. `AWS_REGION` environment variable (highest priority)
2. `AWS_DEFAULT_REGION` environment variable
3. Selected profile in the AWS config file (via `AWS_PROFILE` environment variable)

4. Default profile in the AWS config file
5. Default fallback to `us-east-1` if no configuration is found

> **Note**
>
> The AWS config file is located at `~/.aws/config` on Linux and macOS, or `%USERPROFILE%\.aws\config` on Windows.

> **Note**
>
> If `ATX_CUSTOM_ENDPOINT` is set, the region is extracted from the endpoint URL and overrides all other settings.

Setting Your Region

Choose one of these methods to configure your region:

Option 1: Environment variable (recommended for temporary use):

Linux and macOS

```
export AWS_REGION=<your-region>
```

Windows (PowerShell)

```
$env:AWS_REGION="<your-region>"
```

Option 2: AWS CLI configuration (recommended for permanent setup):

```
aws configure set region <your-region>
```

Option 3: Profile-specific configuration:

Linux and macOS

```
aws configure set region <your-region> --profile your_profile_name
```

```
export AWS_PROFILE=your_profile_name
```

Windows (PowerShell)

```
aws configure set region <your-region> --profile your_profile_name
$env:AWS_PROFILE="your_profile_name"
```

Option 4: Direct file editing:

Edit the AWS config file directly (`~/.aws/config` on Linux and macOS, or `%USERPROFILE%\.aws\config` on Windows):

```
[default]
region = <your-region>

[profile your_profile_name]
region = <your-region>
```

Verifying Your Region Configuration

To check which region AWS Transform custom will use:

Check current region setting:

```
aws configure get region
```

Check environment variables:

Linux and macOS

```
echo "AWS_REGION: $AWS_REGION"
echo "AWS_DEFAULT_REGION: $AWS_DEFAULT_REGION"
echo "AWS_PROFILE: $AWS_PROFILE"
```

Windows (PowerShell)

```
echo "AWS_REGION: $env:AWS_REGION"
echo "AWS_DEFAULT_REGION: $env:AWS_DEFAULT_REGION"
echo "AWS_PROFILE: $env:AWS_PROFILE"
```

View detailed region resolution in debug logs:

Linux and macOS

```
tail -f ~/.aws/atx/logs/debug.log
```

Windows (PowerShell)

```
Get-Content "$env:USERPROFILE\.aws\atx\logs\debug.log" -Wait -Tail 10
```

Example log output showing region source:

```
2026-01-07 22:34:28 [DEBUG]: Initializing FrontendServiceClient with config:
{
  "endpoint": "https://transform-custom.us-east-1.api.aws",
  "region": "us-east-1",
  "regionSource": "AWS_REGION"
}
```

The `regionSource` field shows where the region was derived from (e.g., "aws-config (profile: default)", "AWS_REGION", "default").

> **Important**
>
> If your region configuration points to an unsupported region, the CLI will display a clear error message with instructions to update your configuration to a supported region.

Running Your First Transformation

The fastest way to get started is to use an AWS-managed transformation. AWS-managed transformations are pre-built, AWS-vetted transformations that are ready to use without any setup.

To run an AWS-managed transformation interactively:

```
atx
```

Then ask the agent to execute a transformation:

```
Execute the AWS/java-aws-sdk-v1-to-v2 transformation on the codebase at ./my-java-
project
```

To run a transformation non-interactively:

```
atx custom def exec -n AWS/java-aws-sdk-v1-to-v2 -p ./my-java-project -c "mvn clean
 install" -x -t
```

For detailed information about execution modes, configuration options, and command flags, see the section called “Command Reference”.

Understanding Transformation Results

When a transformation completes, AWS Transform custom provides a report describing the results. If the transformation was not validated (e.g. successful build for a Java applications), the report provides recommendations for further actions.

Most transformations are performed in multiple steps, using Git to commit intermediate ones. You can review the changes made by the transformation using standard Git commands:

```
git status
git log
git diff <original commit-id>
```

Creating a Custom Transformation

When AWS-managed transformations don't meet your needs, you can create custom transformations tailored to your specific requirements.

To create a custom transformation

1. Start the AWS Transform CLI in interactive mode:

   ```
   atx
   ```

2. Tell the agent you want to create a new transformation and describe your requirements.
3. Provide reference materials such as documentation and code samples to improve quality.
4. Test and refine the transformation iteratively.

5. Publish it to your organization's transformation registry.

For detailed instructions on creating custom transformations, see the section called “Workflows”.

AWS Transform Web Application (Optional)

The AWS Transform web application is an optional interface for monitoring large-scale transformation campaigns across multiple repositories. Before using AWS Transform web application, you need to have a user identity enabled to access AWS Transform in your organization. For information about enabling AWS Transform, see Setting up AWS Transform.

To use the AWS Transform web application:

1. Visit https://aws.amazon.com/transform/ and sign in using AWS IAM Identity Center credentials.
2. Select the "Custom / code" job type in the AWS Transform web application.
3. Tell the agent the name of the transformation definition you would like to use.
4. The web application provides an AWS Transform CLI command that can be shared with your team. This command invokes the transformation definition and logs execution results to the web application.
5. After transformation results are captured, view statistics, time savings measurements, and ask questions about the progress of your job in the Dashboard.

The web application is designed for enterprise-scale operations where you need centralized monitoring of transformations across multiple codebases.

> **Note**
>
> AWS IAM Identity Center is required to access the AWS Transform web application. The CLI does not require IAM Identity Center - only standard AWS credentials are needed.

Introduction to custom transformation commands

Here are a few of the commands you can use with custom transformations. The complete list of commands is in the AWS Transform custom transformations command reference.

- `atx custom`
  - **Executes custom transformation interactive experience, allowing creation, discovery, execution, and refinement of transformations.**
  - This is the default command for `atx`.
  - `--trust-all-tools` (`-t`) is optional and implicitly allows all requested tool requests by the agent. Use with caution especially in production environments. You can configure tool trust for specific tools and commands using the trust settings file.
- `atx custom --help` | `atx custom -h`
  - **Shows help menu.**
  - Each command also includes a help menu, for example, `atx custom def exec --help`.
- `atx --version` | `atx -v`
  - **Shows version.**
  - The version number changes with each release.
- `atx custom def list`
  - **Prints list of transformations available in the transformation registry.**
- `atx custom def exec`
  - **Executes a transformation**
- `atx custom def learnings`
  - **Opens an interactive session to view and manage the lessons a transformation has learned from previous runs**
- `atx mcp`
  - **Used to manage MCP server configurations**

Workflows

Executing Transformations

This section describes the different ways to execute transformations and options for controlling execution behavior.

Execution Modes

AWS Transform custom supports three execution modes to accommodate different workflows.

Interactive Conversational Mode

Start the CLI with `atx` and ask the agent to execute a transformation through natural language. This mode allows you to have a full conversation with the agent, interrupt execution at any point, and provide feedback during the transformation process.

Use this mode when you want maximum control and the ability to guide the agent through complex scenarios.

Direct Interactive Execution

Use `atx custom def exec -n <transformation-name> -p <path>` to start a specific transformation interactively. This mode allows you to review and interact with the agent at the beginning, during, or at the end of execution. The agent will pause at key decision points and ask for your input.

This is ideal for testing and refining transformations before running them autonomously.

You can run transformations in non-interactive mode or headless mode. Non-interactive mode suppresses prompts during a named transformation. Headless mode lets you run the agent with a plain-text prompt, bypassing the interactive interface entirely.

Non-interactive mode

Use `atx custom def exec -n <transformation-name> -p <path> -x -t` for full automation. Add `-x` to run in non-interactive mode, and `-t` to trust all tools automatically without prompting.

This mode is designed for CI/CD pipeline integration and bulk execution where no human intervention is available or desired.

Headless mode

To complete tasks without interacting with the agent, run `atx -x "<prompt>" -t` and provide your instruction in plain text.

Headless transformation execution

Use this mode to apply an existing transformation definition to your codebase. The transformation runs each step automatically without requiring your approval.

```
atx -x "apply transformation definition <transformation_definition_name> to
 <codebase_path>" -t
```

Headless transformation development

Create or modify transformation definitions.

To convert a legacy transformation definition to the new skill format (SKILL.md + references/), run the following command:

```
atx -x "convert <legacy_transformation_definition_name> transformation definition to
 skill and save as draft" -t
```

To create a new transformation definition, run the following command:

```
atx -x "create a transformation definition to <description> with references docs
 <reference_docs_path>" -t
```

Common Command Flags

When executing transformations with `atx custom def exec`, the following flags are commonly used:

- `-n` or `--transformation-name` - Specifies the name of the transformation to execute
- `-p` or `--code-repository-path` - Specifies the path to your codebase (use "." for current directory)
- `-c` or `--build-command` - Specifies the build or validation command to run
- `-x` or `--non-interactive` - Enables non-interactive mode (no user prompts)
- `-t` or `--trust-all-tools` - Automatically trusts all tools without prompting
- `-d` or `--do-not-learn` - Prevents lesson extraction from this execution
- `--tv` or `--transformation-version` - Specifies a specific version of the transformation
- `-g` or `--configuration` - Provides a configuration file or inline configuration

> **Important**
>
> The `-t` or `--trust-all-tools` flag automatically approves all tool executions without prompting and bypasses most security guardrails, (commands matching your

`alwaysPromptCommands` list still require explicit permission unless overridden by `trustedShellCommands`). Passing in `--non-interactive` and `--trust-all-tools` is required for a fully autonomous experience but not required to execute the transformation. Use with caution in production environments.

Using Configuration Files

AWS Transform custom supports optional configuration files in YAML or JSON format. Configuration files allow you to specify execution parameters and provide additional context to the agent.

To use a configuration file:

```
atx custom def exec --configuration file://config.yaml
```

You can also provide configuration as inline key-value pairs:

```
atx custom def exec --configuration "key=value,key2=value2"
```

Example configuration file (config.yaml):

```
codeRepositoryPath: ./my-project
transformationName: my-transformation
buildCommand: mvn clean install
additionalPlanContext: |
  The target Java version to upgrade to is Java 17.
  Ensure compatibility with our internal logging framework version 2.3.
validationCommands: |
  mvn test
  mvn verify
```

The `additionalPlanContext` parameter provides extra context for the agent's execution plan. This is especially useful with AWS-managed transformations to customize their behavior for your specific needs.

Build and Validation Commands

The build or validation command is an optional parameter that specifies how to validate your code during the transformation process. AWS Transform custom will attempt to infer the best build

command based on the transformation if not specified, though it is recommended to be specific for quality.

Examples of build and validation commands:

- Java: `mvn clean install` or `gradle build`
- Python: `pytest` or `python -m py_compile`
- Node.js: `npm run build` or `npm test`
- Linters: `eslint .` or `pylint .`

Even for languages or transformations that don't require building, providing a command that validates the results and returns issues if validation fails is very important to improve transformation quality.

If no build or validation is needed, omit from your input.

Controlling Learning Behavior

By default, AWS Transform custom extracts lessons from every transformation execution. You can prevent learning for specific executions.

To prevent learning from an execution:

```
atx custom def exec -n my-transformation -p ./my-project -d
```

The `-d` or `--do-not-learn` flag opts out of allowing lesson extraction from the current execution.

Resuming Conversations

AWS Transform custom allows you to resume previous conversations within 30 days of creation.

To resume the most recent conversation:

```
atx --resume
```

To resume a specific conversation:

```
atx --conversation-id <conversation-id>
```

> **Important**
>
> Conversations can only be resumed within 30 days of creation. After 30 days, the conversation can no longer be resumed.

Tracking Agent Minutes

AWS Transform custom tracks the agent minutes consumed during a transformation session. Agent minutes accumulate throughout the conversation lifecycle and are displayed when the conversation ends:

```
Agent minutes used: 12.50
```

Agent minutes persist across interruptions. If you interrupt a session with Ctrl+C and resume it later, the previously accumulated minutes carry over and continue accumulating in the resumed session.

To check Agent Minutes during an interactive session:

Type `/usage` at the input prompt to display the current accumulated Agent Minutes without ending the conversation.

To set an Agent Minutes budget limit:

```
atx custom def exec -n my-transformation -p ./my-project --limit 30
```

The `--limit` option sets a maximum Agent Minutes budget for the session. Agent Minutes reflect active agent work time, not wall clock time. When the limit is reached, the CLI displays a message and exits with instructions to resume:

```
## Budget limit reached: 30.00 / 30.00 Agent Minutes. Exiting.
```

You can resume the conversation later with an increased limit:

```
atx --conversation-id <conversation_id> -t --limit <increased_limit>
```

Continual Learning

This section describes how to review and manage the lessons created by continual learning.

Understanding Lessons

The continual learning system automatically extracts lessons from previous runs of a transformation. The system creates them asynchronously based on:

- Developer feedback provided in interactive mode
- Code issues encountered during transformations

Lessons accumulate over time as you run the transformation across different codebases. The system applies them automatically to improve future runs. Each lesson belongs to a category containing all lessons of a similar domain so that you can review related lessons together. For lessons you would not like used, you can archive or delete the lesson entirely.

Viewing and Managing Lessons

Use the `learnings` command to open an interactive session for browsing and managing the lessons of a transformation definition.

To open the lessons viewer:

```
atx custom def learnings -n my-transformation
```

The viewer opens on a list of lesson categories, each showing how many active lessons it contains. Select a category to see its lessons, then select a lesson to view its full detail, including the lesson body, its impact, and how many previous runs consulted it.

Archiving and Restoring Lessons

The system applies lessons automatically. If you do not want the system to apply a lesson, you can archive it. The system retains archived lessons but does not apply them to future runs. All archived lessons are grouped together so you can review them and restore any to active use.

Deleting Lessons

Permanently remove a lesson that is not useful. Deletion cannot be undone, and the system might relearn a deleted lesson from future runs.

A lesson must be archived before it can be deleted.

Advanced Configuration

This section describes advanced features and configuration options for AWS Transform custom.

Environment Variables

You can customize CLI behavior using environment variables.

> **Note**
>
> The following examples show Linux and macOS syntax (`export`). On Windows, set environment variables in PowerShell using `$env:NAME="value"`. See the **Windows (PowerShell)** tabs for the equivalent commands.

ATX_SHELL_TIMEOUT

Override the default timeout for shell commands (900 seconds/15 minutes).

Linux and macOS

```
export ATX_SHELL_TIMEOUT=1800  # 30 minutes
```

Windows (PowerShell)

```
$env:ATX_SHELL_TIMEOUT=1800  # 30 minutes
```

This is useful for large codebases or long-running build processes.

ATX_DISABLE_UPDATE_CHECK

Disable automatic version checks and update notifications during command execution.

Linux and macOS

```
export ATX_DISABLE_UPDATE_CHECK=true
```

Windows (PowerShell)

```
$env:ATX_DISABLE_UPDATE_CHECK="true"
```

ATX_GIT_COMMITTER_NAME and ATX_GIT_COMMITTER_EMAIL

Configure the author identity used for the checkpoint commits that AWS Transform custom creates in your repository as it applies changes during a transformation. When these variables are not set, checkpoint commits are attributed to a default identity (`ATX Bot <checkpoint@atx.bot>`). Set both variables to attribute checkpoints to a specific author.

Linux and macOS

```
export ATX_GIT_COMMITTER_NAME="Jane Developer"
export ATX_GIT_COMMITTER_EMAIL="jane@example.com"
```

Windows (PowerShell)

```
$env:ATX_GIT_COMMITTER_NAME="Jane Developer"
$env:ATX_GIT_COMMITTER_EMAIL="jane@example.com"
```

Trust Settings

Trust settings allow you to pre-approve specific tools and commands to execute without prompts. You can also require explicit permission for specific shell commands regardless of trust level. These settings are configured in the `~/.aws/atx/trust-settings.yaml` file.

The file contains three lists:

- `trustedTools` - Tools that can execute without prompting
- `trustedShellCommands` - Shell commands that can execute without prompting
- `alwaysPromptCommands` - Shell command patterns that require explicit permission unless overridden by `trustedShellCommands`, regardless of the `-t` flag or session trust. These patterns are not enforced in non-interactive mode (`-x`).

Default trusted tools:

- `file_read`
- `get_transformation_from_registry`
- `list_available_transformations_from_registry`

Editing trust settings:

You can manually edit the trust-settings.yaml file to add or remove trusted tools and commands. Both `trustedShellCommands` and `alwaysPromptCommands` support glob wildcard patterns using `*`.

> **Note**
>
> If a command matches both lists, `trustedShellCommands` takes priority.

The following describes each command list and provides examples:

- `trustedShellCommands` - Commands matching these patterns execute without prompting, bypassing all other guardrails. Patterns are matched against the full command string.

  Examples:

  - `cd *` - Matches compound commands starting with cd
  - `*&&*` - Trusts all commands with && operators
- `alwaysPromptCommands` - Commands matching these patterns require explicit permission unless overridden by `trustedShellCommands`, regardless of the `-t` flag or session trust. These patterns are not enforced in non-interactive mode (`-x`). Patterns are matched against each sub-command in compound expressions (&&, `||`, command substitutions).

  Examples:

  - `rm -rf *` - Always prompts for recursive force-delete commands
  - `sudo *` - Always prompts for commands run with sudo
  - `find * -exec *` - Always prompts for find commands with -exec

Session-level trust:

During interactive prompts, you can choose:

- `(y)es` - Execute once
- `(n)o` - Deny
- `(t)rust` - Trust for the current session only

Session-level trust settings are temporary and reset when the CLI restarts, providing temporary approval without permanently modifying trust-settings.yaml.

> **Note**
>
> Session trust is not available for commands matching your `alwaysPromptCommands` list.

Model Context Protocol (MCP) Servers

The AWS Transform CLI supports Model Context Protocol (MCP) servers, which extend its functionality with additional tools.

Configuration:

Configure MCP servers in the `~/.aws/atx/mcp.json` file. The AWS Transform CLI supports two types of MCP servers: local command-based servers and remote HTTP servers.

Local command-based servers:

Local servers run as child processes on your machine. Configure them with the `command` property:

```
{
  "mcpServers": {
    "my-local-server": {
      "command": "npx",
      "args": ["-y", "@example/mcp-server"]
    }
  }
}
```

Remote HTTP servers:

Remote servers connect to MCP servers hosted at an HTTP or HTTPS URL. Configure them with the `url` property:

```
{
```

```
  "mcpServers": {
    "my-remote-server": {
      "url": "https://api.example.com/mcp",
      "headers": {
        "Authorization": "Bearer ${MCP_API_TOKEN}"
      }
    }
  }
}
```

The `headers` property is optional and supports environment variable expansion using `${VAR_NAME}` syntax. This allows you to store sensitive values like API tokens in environment variables rather than in the configuration file.

Configuration properties:

Local command-based servers support the following properties:

- `command` (required) - The command to run the server
- `args` (optional) - Array of command-line arguments
- `env` (optional) - Environment variables to pass to the server process

Remote HTTP servers support the following properties:

- `url` (required) - The HTTP or HTTPS URL of the remote MCP server
- `headers` (optional) - HTTP headers to include in requests, with support for `${VAR_NAME}` environment variable expansion

Managing MCP servers:

View list of configured MCP servers:

```
atx mcp tools
```

List available tools offered by a specific MCP server:

```
atx mcp tools --server <server-name>
```

Usage tracking:

The CLI automatically tracks MCP tool usage during transformation executions. Usage statistics are persisted as `mcp_usage.json` in the conversation directory alongside `metadata.json`. The file records per-tool metrics for each execution, including:

- Number of invocations per tool
- Number of errors per tool
- Total execution time per tool
- Last error details (if any)

Client-Side Skills

Client-side skills are additional capabilities that extend the agent during transformation executions. They allow you to provide custom tools, scripts, and instructions that the agent can use alongside its built-in capabilities.

Skill discovery directories:

Skills are discovered from four directories in precedence order. If a skill with the same name exists in multiple directories, the first directory in the list takes priority:

1. `<project>/.aws/atx/skills/` - Project-level, AWS Transform CLI-specific
2. `<project>/.agents/skills/` - Project-level, cross-client (available to any compatible agent tooling)
3. `~/.aws/atx/skills/` - User-level, AWS Transform CLI-specific
4. `~/.agents/skills/` - User-level, cross-client (available to any compatible agent tooling)

The `.aws/atx/skills/` directories are specific to the AWS Transform CLI. The `.agents/skills/` directories are cross-client, meaning skills placed there are available to any compatible agent tooling beyond the AWS Transform CLI.

Skill directory structure:

Each skill is a directory containing a `SKILL.md` file with YAML frontmatter:

```
~/.aws/atx/skills/
### my-skill/
    ### SKILL.md          # Required: frontmatter + instructions
    ### references/       # Optional: reference docs the agent can read
```

```
    #   ### guide.md
    ### scripts/          # Optional: scripts the agent can execute
        ### validate.py
```

SKILL.md format:

```
---
name: my-skill
description: When to use this skill
---
# Skill Title

Instructions for the agent...
```

The `name` field must match the parent directory name.

Disabling a skill:

To prevent a skill from being loaded without removing its files, add `disable-model-invocation: true` to the frontmatter:

```
---
name: my-skill
description: When to use this skill
disable-model-invocation: true
---
```

When this property is set, the CLI skips the skill during discovery. The agent cannot see or use the skill unless a transformation definition explicitly instructs it to read the skill file. Use this to temporarily disable a skill, mark it as work-in-progress, or keep reference material intended for human readers only.

> **Note**
>
> A disabled skill's files remain on disk. If a transformation definition instructs the agent to read a specific file path, the agent can still access the content. The `disable-model-invocation` property prevents automatic discovery and context injection, not file-system access.

Skill availability by execution mode:

- **Exec mode** (`atx custom def exec` with `--code-repository-path`) - Discovers skills from both user-level and project-level directories.
- **Interactive mode** (`atx`) - Only user-level skills are discovered initially. When you provide a code repository path during the session, project-level skills are also loaded.

Verifying skill discovery:

Check the CLI's debug log after a run to verify which skills were discovered:

Linux and macOS

```
grep -i "skill" ~/.aws/atx/logs/debug.log | tail -20
```

Windows (PowerShell)

```
Select-String -Pattern "skill" "$env:USERPROFILE\.aws\atx\logs\debug.log" | Select-
Object -Last 20
```

Skills that fail validation are skipped with a warning in the debug logs.

> **Note**
>
> Client-side skills require CLI version 2.0 or later.

Choosing Between Project-Level and User-Level Skills

Where you place a skill determines who benefits from it and when it activates.

Project-level skills (`<project>/.aws/atx/skills/`):

Commit these to version control so every team member running transformations against the repository automatically discovers them. Use project-level skills for:

- Repository-specific compliance checks (Dockerfile rules, Terraform policies, migration safety validators)
- Organization coding standards that apply to this codebase (observability patterns, error handling, naming conventions)

- Build or test scripts unique to the project (custom linters, architecture fitness functions)
- API migration guides for internal libraries used in this repository

User-level skills (`~/.aws/atx/skills/`):

These remain on your machine and activate during all transformations regardless of which repository you target. Use user-level skills for:

- Personal workflow tools (changelog generators, commit message formatters)
- Cross-project preferences (preferred test patterns, documentation style reminders)
- License compliance checks that your organization requires across all repositories
- Coverage thresholds or quality gates you enforce on every codebase you work with

Tips for effective skills:

- Write clear `description` fields in your `SKILL.md` frontmatter. The agent uses this field to decide when a skill is relevant.
- Exit validation scripts with code 0 on success and non-zero on failure. The agent interprets exit codes to determine compliance.
- Print clear, actionable error messages in scripts. The agent reads output to understand what to fix.
- Place skills in the cross-client directory (`.agents/skills/`) at either level to share them with other AI development tools beyond the AWS Transform CLI.

Client-Side Skill Examples

These examples show two common patterns: a script-based validation skill and a reference-only skill.

Example: Dockerfile Compliance Checker (Script-Based)

This skill validates Dockerfiles against security and operational best practices. It uses a validation script that the agent runs before and after making changes.

Directory structure:

```
.aws/atx/skills/
```

```
### dockerfile-compliance/
    ### SKILL.md
    ### scripts/
    #   ### lint_dockerfile.sh
    ### references/
        ### dockerfile-best-practices.md
```

SKILL.md:

```
---
name: dockerfile-compliance
description: Validates Dockerfiles against security and operational best practices
---
# Dockerfile Compliance Checker

When a transformation creates or modifies Dockerfiles, run the compliance checker.

## When to use

- After creating a new Dockerfile
- After modifying FROM, RUN, USER, or EXPOSE directives
- When containerizing an application as part of a transformation

## How to use

Run: `bash scripts/lint_dockerfile.sh <path-to-Dockerfile>`

If violations are found, consult `references/dockerfile-best-practices.md`
for compliant patterns.
```

The validation script checks for unpinned base image tags, running as root, hardcoded secrets in ENV directives, and missing HEALTHCHECK definitions. The agent runs the script, fixes violations using patterns from the reference file, and re-runs the script to confirm compliance.

Example: API Deprecation Helper (Reference-Only)

This skill guides the agent through replacing deprecated API calls during upgrade transformations. It uses only reference files with no scripts.

Directory structure:

```
.aws/atx/skills/
```

```
### api-deprecation-helper/
    ### SKILL.md
    ### references/
        ### aws-sdk-v2-to-v3.md
        ### react-class-to-hooks.md
```

SKILL.md:

```
---
name: api-deprecation-helper
description: Guides the agent through replacing deprecated API calls with modern
 equivalents
---
# API Deprecation Helper

When performing upgrade transformations, use this skill to identify and replace
deprecated API calls with their modern equivalents.

## When to use

- During any version upgrade transformation
- When build warnings mention deprecated APIs
- When transforming code that uses legacy patterns

## Process

1. Identify deprecated API calls in the codebase
2. For each deprecated call, find the replacement in `references/`
3. Apply the replacement, preserving the original behavior
4. Verify the replacement compiles and tests pass
```

The reference files contain before-and-after code examples. For instance, `aws-sdk-v2-to-v3.md` maps patterns like `s3.putObject(params).promise()` to the modular v3 equivalent using `S3Client` and `PutObjectCommand`.

Tags and Organization

You can organize customer-owned transformation definitions with tags for access control, categorization, and cost allocation.

You can add, update, or remove tags only on customer-owned transformation definitions. You cannot manage tags on AWS-managed transformations.

> **Note**
>
> Some of these commands require specifying the Amazon Resource Name (ARN) for a Transformation Definition. The ARN structure is: `arn:aws:transform-custom:<region>:<account-id>:package/<td-name>`

To list tags for a transformation:

```
atx custom def list-tags --arn <transformation-arn>
```

To add tags to a transformation:

```
atx custom def tag --arn <transformation-arn> --tags '{"env":"prod","team":"backend"}'
```

To remove tags from a transformation:

```
atx custom def untag --arn <transformation-arn> --tag-keys "env,team"
```

Tags can be used for grouped access control in IAM policies. You can create policies that grant permissions to all transformations with specific tags (e.g., all transformations tagged with `team:frontend` or `environment:production`).

Use tags for cost allocation

You can use tags on customer-owned transformation definitions to organize your AWS bill. AWS Transform custom records the transformation definition as the billing resource for the Agent Minutes an execution consumes. Each execution inherits the tags of the transformation definition it runs. If several teams or projects share one AWS account, tag each customer-owned definition with its cost owner. You can then separate the transformation costs for each team or project.

After you tag your customer-owned transformation definitions, activate those tag keys in the AWS Billing and Cost Management console. Tag values do not appear in your cost data until the tag key is activated. Activation applies to usage going forward. To attribute usage from earlier months, request a cost allocation tag backfill. For more information, see Cost allocation and tagging in the *AWS Billing and Cost Management User Guide*.

After a tag key is active, you can group AWS Transform charges by that key in Cost Explorer. You can also include resource IDs in AWS Cost and Usage Reports to see the transformation definition

ARN and its tag values on each line item. For the columns that carry those values, see Resource tags columns in the *AWS Data Exports User Guide*. The following example returns AWS Transform cost and Agent Minutes for one day, grouped by a `CostCenter` tag.

```
aws ce get-cost-and-usage \
    --time-period Start=2026-09-01,End=2026-09-02 \
    --granularity DAILY \
    --metrics UnblendedCost UsageQuantity \
    --filter '{"Dimensions":{"Key":"SERVICE","Values":["AWS Transform"]}}' \
    --group-by Type=TAG,Key=CostCenter
```

Cost Explorer returns charges with no value for the key as a single group with an empty value. This shows you how much spend remains unattributed.

What you cannot attribute by tag

Tag-based attribution operates on the transformation definition. Confirm the following against your cost model before you rely on tags for chargeback.

Usage	Why it is not attributed, and what to do instead
AWS-managed transformations	You cannot add, update, or remove tags on AWS-managed transformation definitions. Their Agent Minutes charges do not inherit customer-defined cost allocation tags.
Sessions that run no transformation definition	Agent Minutes consumed in an `atx` session that does not run a transformation definition are recorded without a resource ID, so there is no resource to inherit tags. Run the work through a tagged transformation definition when the spend must be attributed.
Individual executions of one definition	Tags are a property of the transformation definition, not of an execution, and `atx custom def exec` does not accept tags. All executions of a definition share its tag values. To separate two cost owners, give each its own transformation definition.
Individual users or IAM principals	Charges are attributed to the transformation definition and the executing account, not to the identity that started the

Usage	Why it is not attributed, and what to do instead
	execution. Per-user attribution is not available through cost allocation tags.

Logs

AWS Transform CLI maintains three types of logs for troubleshooting and debugging.

Conversation logs:

Linux and macOS

```
~/.aws/atx/custom/<conversation_id>/logs/<timestamp>-conversation.log
```

Windows

```
%USERPROFILE%\.aws\atx\custom\<conversation_id>\logs\<timestamp>-conversation.log
```

These logs contain the full conversation history for a specific session.

Subagent logs:

Linux and macOS

```
~/.aws/atx/custom/<conversation_id>/logs/subagents/<name>.log
```

Windows

```
%USERPROFILE%\.aws\atx\custom\<conversation_id>\logs\subagents\<name>.log
```

These logs contain output from subagents that the main agent spawns during transformations. You do not need to manage subagents directly.

Developer debug logs:

Linux and macOS

```
~/.aws/atx/logs/debug*.log
```

```
~/.aws/atx/logs/error.log
```

Windows

```
%USERPROFILE%\.aws\atx\logs\debug*.log
%USERPROFILE%\.aws\atx\logs\error.log
```

These logs provide advanced troubleshooting information for the CLI itself.

> **Note**
>
> There may be multiple debug log files in the logs directory (i.e. debug1.log, debug2.log). Review and provide all relevant logs for example, ~/.aws/atx/custom/<conversation-id>/* and ~/.aws/atx/logs/*, when opening support tickets for faster resolution.

CLI Updates

Keep your CLI up to date to access new features and improvements.

To check for updates:

```
atx update --check
```

To update to the latest version:

```
atx update
```

To update to a specific version:

```
atx update --target-version <version>
```

Create Custom Transformations

This section describes how to create, modify, and manage custom transformation definitions.

Creating a New Transformation

Use the interactive CLI to create a new transformation definition.

To create a transformation definition

1. Start the AWS Transform CLI:

```
atx
```

2. Tell the agent you want to create a new transformation.
3. Provide a clear, detailed description of the transformation objective. Include:
   - The source and target state (e.g., "upgrade from version X to version Y")
   - Specific changes required (e.g., "update import statements, replace deprecated methods")
   - Any special considerations or constraints
4. When the agent requests clarification or additional information, provide specific examples and reference materials.
5. Review the initial transformation definition created by the agent.
6. Test the transformation on a sample codebase.
7. Iterate by providing feedback, code fixes, or additional examples.
8. Save the transformation locally or publish it to the registry.

Best practices for creating transformations:

- Start with simple, well-defined transformations before attempting complex ones
- Provide comprehensive reference materials including migration guides and code samples
- Test on multiple sample codebases before publishing
- Use deterministic build or validation commands to enable continual learning
- Consider breaking complex transformations into multiple smaller steps
- Mark crucial information with "CRITICAL:" or "IMPORTANT:" in your transformation definitions to ensure the agent prioritizes these requirements
- When you need exact requirements followed (like using a specific command or string value), explicitly specify the complete string in your transformation definitions. You can wrap these in bash quotes to clearly indicate they're terminal commands or literal strings, which reduces variability and ensures consistent execution

Providing Reference Materials

You can provide reference files to AWS Transform custom by specifying file paths during the conversation. These files are stored in the `references/` folder of the transformation definition.

Recommended types of reference files:

- Before/after example code
- Documentation for APIs, libraries, or features involved
- Human-readable migration guides

To provide a reference file:

```
Take a look at the documentation here: /path/to/migration-guide.md
```

You can also provide a directory containing multiple reference files:

```
Take a look at the docs we have here: /path/to/docs/
```

> **Note**
>
> Only text-based files (.md, .html, .txt, code files) are supported. Binary files, images, and rich text files (e.g., .pdf, .png, .docx) are not currently supported. It is often possible to extract the text content and use that as reference. If you have many small text files, consider concatenating them into few descriptively-named files. There is a limit of 10MB total for all files.

Modifying an Existing Transformation

You can modify custom transformations both before and after saving them as drafts or publishing them. You cannot modify AWS-managed transformations. If you need to customize them, you can provide additional context using the config file.

To modify an existing transformation

1. Start the AWS Transform CLI:

```
atx
```

2. Tell the agent you want to modify an existing transformation.
3. Choose whether to:
   - Provide a file path to a locally stored transformation (i.e. not a saved draft or published)
   - Request the list of transformations from the registry
4. If choosing from the registry, select the transformation you want to modify.
5. Work with the agent to describe the changes you want to make.
6. Test the updated transformation on a sample codebase.
7. Publish your updates to the registry if desired.

Publishing and Managing Transformations

You can publish and manage your transformations using the interactive experience or with the following commands.

To save a transformation as a draft:

```
atx custom def save-draft -n my-transformation --description "Description of the
 transformation" --sd ./transformation-directory
```

To publish a transformation:

```
atx custom def publish -n my-transformation --description "Description of the
 transformation" --sd ./transformation-directory
```

To list available transformations:

```
atx custom def list
```

To download a transformation definition:

```
atx custom def get -n my-transformation
```

This downloads the transformation definition to your current working directory. You can specify a target directory with the `--td` flag and a version with the `--tv` flag.

To delete a transformation definition:

```
atx custom def delete -n my-transformation
```

> **Important**
>
> This permanently deletes the specified transformation definition from your account.

Managing Transformation Versions

AWS Transform custom maintains versions of your transformation definitions. You can specify a version when executing or downloading a transformation.

To execute a specific version:

```
atx custom def exec -n my-transformation --tv v1 -p ./my-project
```

To download a specific version:

```
atx custom def get -n my-transformation --tv v1
```

If no version is specified, the latest version is used.

Command Reference

This section provides a comprehensive reference of all AWS Transform custom CLI commands.

Interactive Commands

atx

Start an interactive conversation with the AWS Transform CLI.

```
atx                          # Start new conversation
```

```
atx --resume                    # Resume most recent conversation
atx --conversation-id <id>      # Resume specific conversation
atx -t                          # Start with all tools trusted
```

Interactive session commands

The following commands can be typed at the input prompt during an interactive session:

- `/usage` - Display the accumulated agent minutes for the current session

Transformation Definition Commands

atx custom def exec

Execute a transformation definition on a code repository.

Option	Long Form	Parameter	Description
`-p`	`--code-repository-path`	`<path>`	Path to the code repository to transform. For current directory, use "."
`-c`	`--build-command`	`<command>`	Command to run when building repository
`-n`	`--transformation-name`	`<name>`	Name of the transformation definition in the registry
`-x`	`--non-interactive`	-	Runs the transformation with no user assistance
`-t`	`--trust-all-tools`	-	Trusts all tools (no tool prompts)

Option	Long Form	Parameter	Description
-d	--do-not-learn	-	Opt out of allowing lesson extraction from this execution
-g	--configuration	<config>	Path to config file (JSON or YAML) or key=value pairs
--tv	--transformation-version	<version>	Version of the transformation definition to use
--limit	--limit	<limit>	Set Agent Minutes budget limit. Transformation exits when limit is reached and can be resumed with an increased limit
-h	--help	-	Display help for command

atx custom def list

List available transformation definitions.

```
atx custom def list
atx custom def list --json    # Output in JSON format
```

The **Category** column shows whether each transformation definition is an `analysis` or a `remediation` transformation. A transformation definition without a category shows —. In `--json` output, the category appears in the `category` field. This field is omitted when no category is set.

atx custom def get

Get details of a specific transformation definition.

Option	Long Form	Parameter	Description
-n	--transformation-name	<name>	The name of the transformation definition to retrieve
--tv	--transformation-version	<version>	The version of the transformation definition to retrieve
--td	--target-directory	<directory>	The target directory at which to save the transformation definition (default: ".")
--json	--json	-	Output response in JSON format
-h	--help	-	Display help for command

atx custom def delete

Delete a transformation definition permanently.

```
atx custom def delete -n <transformation-name>
```

atx custom def publish

Publish a transformation definition to the registry.

Option	Long Form	Parameter	Description
-n	--transformation-name	<name>	The name of the transformation definition to publish

Option	Long Form	Parameter	Description
`--tv`	`--transformation-version`	`<version>`	The version of the transformation definition to use (not applicable with --description or --source-directory)
`--sd`	`--source-directory`	`<directory>`	The source directory from which to read the local transformation definition files (not applicable with --transformation-version)
`--description`	`--description`	`<description>`	A description for the transformation definition (not applicable with --transformation-version)

Option	Long Form	Parameter	Description
`--category`	`--category`	`<category>`	The category of the transformation definition. Use `analysis` for transformations that read code and report findings, or `remediation` for transformations that change code. If you don't specify a category when you publish a new version, the previous version's category is kept. Not applicable with --transformation-version.
`--json`	`--json`	-	Output response in JSON format
`-h`	`--help`	-	Display help for command

atx custom def save-draft

Save a transformation definition as a draft in the registry. Drafts expire after 30 days and do not show up in the registry. This command returns the version number of the draft. You must execute and retrieve drafts explicitly using the transformation name and version number.

```
atx custom def save-draft -n <transformation-name> --description "Description" --sd
 <directory>
```

To set the category of the transformation definition, add `--category analysis` or `--category remediation`.

Lesson Commands

atx custom def learnings

Open an interactive session to view and manage the lessons a transformation definition has learned from previous runs. Browse lessons by category, view lesson detail, and archive, restore, or delete lessons.

```
atx custom def learnings -n <transformation-name>
```

Tag Commands

atx custom def list-tags

List tags for a transformation definition.

```
atx custom def list-tags --arn <transformation-arn>
```

atx custom def tag

Add tags to a transformation definition.

```
atx custom def tag --arn <arn> --tags '{"env":"prod","team":"backend"}'
```

atx custom def untag

Remove tags from a transformation definition.

```
atx custom def untag --arn <arn> --tag-keys "env,team"
```

Update Commands

atx update

Update the AWS Transform CLI.

```
atx update                    # Update to latest version
atx update --check            # Check for updates only
```

```
atx update --target-version <version>  # Update to specific version
```

MCP Commands

atx mcp tools

Allows clients to view/confirm their MCP tool configurations. Updates to the configurations must be managed directly via the file ~/.aws/atx/mcp.json

```
atx mcp tools     # View list of MCP servers
atx mcp tools -s    # List tools for the specified server
```

AWS-Managed Transformations

AWS-managed transformations are pre-built, AWS-vetted transformations for common use cases that are ready to use without any additional setup.

Overview

AWS-managed transformations have the following characteristics:

- **Validated by AWS** - These transformations are vetted by AWS to be high quality
- **Ready to use** - No additional setup required
- **Continuously growing** - Additional transformations are continually being added
- **Customizable** - Pre-built transformations can be customized by providing additional guidance or requirements specific to your organization's needs using the `additionalPlanContext` configuration parameter
- **Early access support** - Some transformations may be marked as early access as they undergo further testing and refinement

Available AWS-Managed Transformations

The following catalog lists the AWS-managed transformations that are currently available, grouped by use case. Each table shows the transformation name, the language or stack it applies to, and its status. To find a transformation quickly, use the search box at the top of this page to search by name, language, or use case.

This catalog is the canonical list of AWS-managed transformations. The Kiro Power, the agent plugin, the VS Code plugin, and the CLI all run the same transformations. To list the transformations available in your registry, run `atx custom def list`.

> **Note**
>
> Transformations marked **Early access** are functional but might be updated frequently based on customer feedback.

Runtime upgrades

Upgrade a language or runtime to a newer version, or change the runtime distribution.

Transformation	Language / stack	Status	Description
`AWS/java-version-upgrade`	Java	Generally available	Upgrade Java applications using any build system from any source JDK version to any target JDK version, with dependency modernization including Jakarta EE migration, database drivers, ORM frameworks, and Spring ecosystem updates. Specify the target JDK version in chat or with the `additionalPlanContext` parameter.
`AWS/python-version-upgrade`	Python	Generally available	Migrate Python projects from Python

Transformation	Language / stack	Status	Description
			3.8, 3.9, or 3.10 to Python 3.11, 3.12, or 3.13 while maintaining functionality and performance. Specify the target Python version in chat or with the `additionalPlanContext` parameter.
`AWS/nodejs-version-upgrade`	Node.js	Generally available	Upgrade Node.js applications from any source Node.js version to any target Node.js version. Specify the target version in chat or with the `additionalPlanContext` parameter.
`AWS/lambda-nodejs-runtime-upgrade`	Node.js (Lambda)	Early access	Upgrade AWS Lambda functions from older Node.js runtimes (nodejs4.3 through nodejs22.x) to nodejs24.x, addressing breaking changes in the Lambda Runtime Interface Client (RIC) and the Node.js 24 language runtime.

Transformation	Language / stack	Status	Description
`AWS/dotnet-lambda-upgrade`	.NET (Lambda)	Generally available	Upgrades AWS Lambda .NET functions from older runtimes (.NET Core 3.1, .NET 6, .NET 8) to .NET 10. Covers project files (.csproj/.fsproj), NuGet packages (Amazon.Lambda.*, AWSSDK.*, Microsoft.Extensions.*), Lambda configuration (SAM, Serverless Framework, CDK, Terraform, CloudFormation), Dockerfiles, code modernization, test projects, and artifact paths. Handles managed runtime (dotnet10), Native AOT (provided.al2023), and container-image deployment models. Trigger keywords: Lambda .NET upgrade, net10.0, TFM migration, dotnet10, Amazon.Lambda, AWSSDK, csproj, fsproj, TargetFramework.

Transformation	Language / stack	Status	Description
`AWS/oracle-java-to-corretto`	Java (JDK distribution)	Early access	Migrate Java projects from Oracle JDK to Amazon Corretto. Replaces Oracle-specific internal APIs with standard Java equivalents, updates build configurations (Maven and Gradle) and container base images, removes commercial JVM flags, and generates a licensing report.

Transformation	Language / stack	Status	Description
`AWS/ruby-upgrade`	Ruby	Early access	Upgrade Ruby applications from Ruby 2.x to Ruby 4.0, and their frameworks to Rails 8.0 or Sinatra 4.1. You can use this transformation with Rails and ActiveRecord apps, Sinatra apps, and standalone gems. The transformation pairs each Ruby version bump with the matching framework upgrade and runs the test suite. It then isolates failures before advancing to the next version.

Transformation	Language / stack	Status	Description
`AWS/dotnet-modernization`	.NET	Generally available	Migrate .NET Framework solutions to modern .NET (net10.0 default; net8.0 or later supported) with controlled user interaction. Assesses complexity and dependencies, presents a migration plan with parallel execution, transforms projects with system-enforced exit-crit eria validation, and generates a summary with next steps.

SDK migrations

Migrate between major versions of an AWS SDK.

Transformation	Language / stack	Status	Description
`AWS/java-aws-sdk-v1-to-v2`	Java	Generally available	Upgrade the AWS SDK from v1 to v2 for Java projects using Maven or Gradle.
`AWS/python-boto2-to-boto3`	Python	Generally available	Migrate Python applications from boto2 to boto3, based on the official

Transformation	Language / stack	Status	Description
			AWS migration documentation.
`AWS/nodejs-aws-sdk-v2-to-v3`	Node.js	Generally available	Upgrade Node.js applications from the AWS SDK for JavaScript v2 to v3 for modular architecture, first-class TypeScrip t support, and improved performan ce, without changing the Node.js version.

Framework upgrades and migrations

Upgrade a framework to a newer version, or migrate to a different framework.

Transformation	Language / stack	Status	Description
`AWS/spring-boot-version-upgrade`	Java / Spring Boot	Generally available	Migrate Spring Boot 3.x projects to Spring Boot 4.x (targetin g 4.1.0 GA). Covers version upgrades, starter renames, Jackson 3 migration , Spring Security 7 DSL, @MockBean removal, property renames, package relocations, observabi lity consolidation, Spring AI 2.0, Spring

Transformation	Language / stack	Status	Description
			Cloud 2025.1, Spring Batch, Kafka, AMQP, gRPC, Native/AOT, Jersey, JSpecify, embedded server changes, third-party compatibility, and verification.
`AWS/JBoss-to-Spring-Boot`	Java (JBoss or WildFly to Spring Boot)	Early access	Migrate Java EE or Jakarta EE enterprise applications running on JBoss EAP or WildFly to Spring Boot, eliminating application server dependencies for a cloud-native, containerized deployment model.

Transformation	Language / stack	Status	Description
`AWS/java-ee-to-quarkus`	Java (Java EE or Jakarta EE to Quarkus)	Early access	This is an early-access transformation that migrates Java EE and Jakarta EE applications from an application server (JBoss EAP, WildFly, Payara, GlassFish, or WebLogic) to a standalone Quarkus 3.x JAR, converting only the features your application uses. It then builds the application and runs its tests to confirm the migration succeeded. JSF conversion might require manual review.
`AWS/early-access-angular-to-react-migration`	Angular to React	Early access	Transform an Angular application to React with comprehensive validation, including Playwright E2E domain-model tests and a migration report containing visual regression screenshots and business logic coverage.

Transformation	Language / stack	Status	Description
`AWS/angular-version-upgrade`	Angular	Early access	Upgrade an older Angular application to a target Angular version, including components, services, templates, and routing.
`AWS/vue.js-version-upgrade`	Vue.js	Early access	Perform a major version upgrade from Vue.js 2 to Vue.js 3, modernizing components, state management, routing, and global APIs. Minor and patch updates are outside the scope.

GenAI and model migrations

Assess generative AI and agent workloads for migration to Amazon Bedrock and AWS runtimes.

Transformation	Language / stack	Status	Description
`AWS/GenAI-to-Bedrock-Migration-Assessment`	Generative AI workloads	Early access	Assess generative AI workloads for migration from third-party providers (OpenAI, Google Gemini, Anthropic direct, and open-source models) to . Discovers SDK usage and models,

Transformation	Language / stack	Status	Description
			clarifies requirements, designs a model mapping, estimates cost and risk, and generates assessment artifacts. Assessment-only; does not modify source code. Also covers agentic frameworks such as CrewAI, LangGraph, and Strands.
`AWS/agent-runtime-advisor`	AI agent workloads	Early access	Early-access advisory transformation: reads a repository containing LLM/agent workloads and recommends an AWS runtime (AgentCore, ECS, EKS, Lambda, Lambda MicroVMs) plus a Bedrock model and API path. Produces assessment documents under .agent-advisor/ only - never modifies application source.

Language-to-language migrations

Translate a codebase from one programming language to another.

Transformation	Language / stack	Status	Description
`AWS/vba-to-python-migration`	VBA to Python	Early access	Migrate Excel VBA macros, modules, UserForms, and embedded logic to equivalent Python scripts and modules, using openpyxl, pandas, tkinter or PyQt6, and standard libraries. Target language is Python 3.8 or later. Use when convertin g .bas, .cls, .frm, or .xlsm-embedded VBA code.

Database migrations

No database-specific AWS-managed transformations are currently available. To modernize Microsoft SQL Server databases and their associated .NET applications to Amazon Aurora PostgreSQL, see the section called “SQL Server modernization”.

Observability

Modernize logging and monitoring to AWS-native services.

Transformation	Language / stack	Status	Description
`AWS/early-access-log4j-to-slf4j-migration`	Java (logging)	Early access	Migrate Java applicati ons from Log4j (1.x or 2.x) to SLF4J with a Logback backend. Handles source code, dependency

Transformation	Language / stack	Status	Description
			management (Maven and Gradle), and logging configuration files, and validates by compile, test, and residual import scan.
`AWS/datadog-monitors-to-cloudwatch-alarms`	Infrastructure as code / monitoring	Early access	Migrate DataDog metric monitors that track AWS service metrics and custom metrics to native CloudWatch alarms as infrastructure as code. Generates CloudFormation YAML, CDK TypeScript, and Terraform HCL.

Architecture

Re-architect, re-platform, or optimize applications for AWS.

Transformation	Language / stack	Status	Description
`AWS/java-performance-optimization`	Java (performance)	Generally available	Optimize Java application performance by analyzing JFR profiling data to detect CPU and memory hotspots and anti-patterns, then applying targeted code fixes.

Transformation	Language / stack	Status	Description
			For instructions on collecting JFR data, see the JFR runtime guide.
`AWS/early-access-java-x86-to-graviton`	Java (Arm64 / Graviton)	Early access	Validate Java application compatibility with the Arm64 architecture for AWS Graviton processors, and resolve incompatibilities by updating dependencies, detecting architecture-specific code patterns, and recompiling native libraries when source code is available. Many modern Java applications are already Arm64-compatible.

Transformation	Language / stack	Status	Description
`AWS/oracle-service-bus-to-aws`	Java / integration to serverless	Early access	Migrate Oracle Service Bus (OSB) and BPEL process configurations to an AWS-native serverless architecture, generating a deployable CDK TypeScript project with Amazon API Gateway, AWS Lambda, and AWS Step Functions.

Transformation	Language / stack	Status	Description
`AWS/mulesoft-to-aws-native-java`	Java / MuleSoft to serverless	Early access	Transform MuleSoft Mule 3.x and 4.x applications into AWS serverless architectures (Amazon API Gateway, AWS Lambda, and AWS Step Functions) targeting Java 17 on AWS Lambda with SnapStart. Maps flow triggers to AWS event sources, converts DataWeave and MEL to tiered AWS equivalents (VTL, Step Functions, Lambda), replaces connectors with AWS SDK for Java v2 calls, and orchestrates flows with Step Functions. Produces SAM templates, Java Lambda handler code, Maven pom.xml, architecture docs, and JUnit 5 tests.

Transformation	Language / stack	Status	Description
`AWS/mulesoft-to-aws-native-python`	Python / MuleSoft to serverless	Generally available	Transforms MuleSoft Mule 3.x/4.x applications into AWS serverless architectures (API Gateway + Lambda + Step Functions) targeting Python 3.12 on Lambda. Maps flow triggers to AWS event sources, converts DataWeave /MEL to tiered AWS equivalents (VTL, Step Functions, Lambda), replaces connectors with boto3 AWS SDK calls, and orchestrates flows with Step Functions. Produces SAM templates, Python Lambda handler code, pinned requirements.txt with a dependency lockfile, architecture docs, a deployment runbook with samconfig.toml, and pytest tests. Triggers: MuleSoft migration, Mule to AWS, DataWeave

Transformation	Language / stack	Status	Description
			conversion, connector replacement, SAM template generatio n, Lambda Powertool s Python, pytest, serverless migration.
`AWS/payshield-hsm-to-aws-payment-cry ptography`	Java (HSM to AWS Payment Cryptogra phy)	Generally available	Migrate Java applicati ons from the Thales PayShield HSM socket-based command protocol to the AWS Payment Cryptography Service (APC) SDK v2. Covers command mapping, key migration (TR-31 and TR-34 import, KCV verification), SDK dependency setup (Maven, Gradle, and Ant), infrastru cture removal (socket pools, response parsers), test infrastru cture update, and parity testing.

Transformation	Language / stack	Status	Description
`AWS/websphere-jaxrs-to-bedrock-agentcore`	Java (WebSphere to AgentCore)	Generally available	Generate AgentCore tool definitions from the REST endpoints in a WebSphere or Liberty application, so existing business logic can be called by AI agents. The transformation reads your source code and writes new files to an output directory without changing anything in place. It finds endpoints that use JAX-RS in both the javax.ws.rs and jakarta.ws.rs namespaces, including EJB exposed through a JAX-RS facade. For each endpoint it maps Java parameters and types to a JSON schema, resolves the HTTP method, and rates its confidence so you know what to review. Output includes a discovery report, a JSON schema per endpoint, tool implementations

Transformation	Language / stack	Status	Description
			in Python, and a manifest for registering them with an AgentCore gateway.
`AWS/f5-scf-to-aws-native`	Infrastructure as code / load balancing	Generally available	This transformation converts F5 BIG-IP load balancer configurations (SCF or TMSH exports, or tmsh scripts) into AWS load balancing and edge services (ALB, AWS WAF, CloudFront) as Terraform infrastructure-as-code. Some advanced F5 features are flagged as gaps, not converted. Triggers: F5 migration, BIG-IP, SCF, TMSH, load balancer migration, Terraform.

Codebase analysis

Analyze codebases and portfolios to plan modernization. These transformations produce reports and do not modify your code.

Transformation	Language / stack	Status	Description
`AWS/comprehensive-`	Multiple languages	Generally available	This transformation performs deep static

Transformation	Language / stack	Status	Description
`codebase-analysis`			analysis of codebases to generate hierarchical, cross-referenced documentation covering all aspects of the system. It combines architectural documentation, structural reference material, and dependency analysis to create a comprehensive knowledge base organized for maximum usability and navigation. The transformation places special emphasis on technical debt analysis, providing prominent, actionable insights on outdated components and maintenance concerns at the root level.

Transformation	Language / stack	Status	Description
`AWS/agentic-readiness-analysis`	Multiple languages	Early access	Evaluate whether systems are ready to be safely called by AI agents, covering APIs, identity, state management, human-in-the-loop, and observability.
`AWS/modernization-readiness-analysis`	Multiple languages	Early access	Scan portfolios for cloud-native maturity gaps and map findings to AWS modernization pathways.
`AWS/portfolio-agentic-readiness-analysis`	Portfolio	Early access	Aggregate individual agentic readiness analysis reports across a portfolio of applications to identify cross-cutting blockers, shared remediation patterns, and prioritized recommendations.

Transformation	Language / stack	Status	Description
`AWS/portfolio-modernization-readiness-analysis`	Portfolio	Early access	Aggregate individual modernization analysis reports across a portfolio into a consolidated roadmap with prioritized migration waves and recommended modernization pathways.

Transformation	Language / stack	Status	Description
`AWS/business-rules-extraction`	Multiple languages	Early access	This transformation statically analyzes a monolithic codebase to produce rewrite-ready documentation without requiring a build, a run, or source code modifications. It decomposes the system into bounded domains and extracts per-domain business rules, workflows, cross-cutting concerns, and database structure. The output includes a machine-readable manifest, an interactive dashboard, and language-neutral implementation specifications per domain to guide modernization or full rewrites.

Security

Fix security vulnerabilities in source code based on scanner findings.

Transformation	Language / stack	Status	Description
`AWS/security-issue-fixer`	Multiple languages	Generally available	Fix security vulnerabilities in source code based on third-party scanner results (SARIF, JSON, CSV, or plain-text findings with CWE IDs, file paths, and line numbers). Does not perform scanning or discovery; it only remediates supplied findings while preserving functional behavior. Covers injection, broken authentication, cryptographic failures, authorization, CSRF, insecure deserialization, SSRF, path traversal, XXE, and more. Supports Python, JavaScript, TypeScript, Java, Go, C#, .NET, C, C++, and Ruby.

Customizing AWS-Managed Transformations

You can customize AWS-managed transformations to meet your organization's specific needs by providing additional context through the `additionalPlanContext` configuration parameter.

Example: Customizing Java version upgrade

```
codeRepositoryPath: ./my-project
transformationName: AWS/java-version-upgrade
buildCommand: mvn clean install
additionalPlanContext: |
  The target Java version to upgrade to is Java 17.
  Update all internal library dependencies to versions compatible with Java 17.
  Ensure compatibility with our custom authentication framework.
```

Example: Customizing AWS SDK migration

```
codeRepositoryPath: ./my-project
transformationName: AWS/java-aws-sdk-v1-to-v2
buildCommand: gradle build
additionalPlanContext: |
  Maintain our existing error handling patterns.
  Use our organization's standard credential provider chain.
  Update logging to use our internal logging framework.
```

Common Use Cases

This section provides step-by-step guidance for common transformation scenarios using the AWS Transform CLI.

Lambda Runtime Upgrades

Lambda periodically deprecates older language runtimes, requiring teams to upgrade their Lambda functions to supported versions. AWS Transform custom can automate these upgrades using the AWS-managed language version upgrade transformations. For example, to upgrade a Lambda function written in Python 3.9 to Python 3.13, you would use the `AWS/python-version-upgrade` transformation. Similar transformations are available for Java (`AWS/java-version-upgrade`) and Node.js (`AWS/nodejs-version-upgrade`), covering the most common Lambda runtime languages.

To upgrade a single Lambda function interactively, navigate to the root of your function's source repository and start the AWS Transform CLI with `atx`. In the interactive session, describe the upgrade you need—for example, "Upgrade this Python 3.9 Lambda function to Python 3.13." The agent will select the appropriate AWS-managed transformation, analyze your code, update language syntax, adjust dependencies, and modernize deprecated API calls. You can provide a

build or validation command such as `pytest` or `python -m py_compile *.py` so the agent can verify the transformed code compiles and passes tests. Throughout the interactive session, you can review the agent's proposed changes, provide feedback, and request adjustments before accepting the final result.

When you need to upgrade Lambda runtimes across many repositories, use non-interactive mode to run transformations at scale. For each repository, invoke the CLI with the appropriate flags to run without user input. For example, to upgrade a Node.js 16 Lambda function to Node.js 22:

```
atx custom def exec \
  -n AWS/nodejs-version-upgrade \
  -p ./my-lambda-repo \
  -c "npm test" \
  -g "additionalPlanContext='Upgrade from Node.js 16 to Node.js 22'" \
  -x -t
```

The `-x` flag runs the transformation non-interactively, and `-t` trusts all tool executions so no prompts are required. You can script this command across dozens or hundreds of repositories using a simple shell loop. After each transformation completes, review the Git diff to verify the changes and run your test suite to confirm the upgraded function works correctly. You can also integrate AWS Transform custom into your CI/CD pipeline to identify deprecated Lambda functions on an ongoing basis. This approach not only addresses the immediate problem of deprecated runtimes, but also establishes a continuous process to detect and upgrade functions before they reach end-of-support status.

Central platform teams can create campaigns through the AWS Transform web application to define the transformation, specify target Lambda repositories, and track progress across the organization. For a detailed walkthrough of building a production-grade scaled deployment solution using AWS Batch, see Building a scalable code modernization solution with AWS Transform custom.

AWS Transform custom and interface endpoints (AWS PrivateLink)

You can establish a private connection between your VPC and AWS Transform custom by creating an *interface VPC endpoint*. Interface endpoints are powered by AWS PrivateLink, a technology that enables you to privately access AWS Transform custom services without an internet gateway,

NAT device, VPN connection, or Direct Connect connection. Traffic between your VPC and AWS Transform custom does not leave the Amazon network.

Each interface endpoint is represented by one or more Elastic Network Interfaces in your subnets.

For more information, see Interface VPC endpoints (AWS PrivateLink) in the *Amazon VPC User Guide*.

Note

- You must enable AWS PrivateLink integration for Amazon S3 since AWS Transform custom makes S3 API calls. For detailed instructions, see the AWS PrivateLink for Amazon S3 documentation. If you encounter S3 access issues while using AWS Transform custom, refer to our troubleshooting guide.
- If you are not using AWS PrivateLink Private DNS feature (see Private DNS), you must:
  - Configure routing to VPC interface endpoints (see the Routing to VPC interface endpoints documentation)
  - Set the `ATX_CUSTOM_ENDPOINT` environment variable to specify your custom domain, for example:

```
ATX_CUSTOM_ENDPOINT=https://transform-custom.<region>.api.aws atx
```

Considerations for AWS Transform custom VPC endpoints

Before you set up an interface VPC endpoint for AWS Transform custom, ensure that you review Interface endpoint properties and limitations in the *Amazon VPC User Guide*.

AWS Transform custom supports making calls to all of its API actions through the interface endpoint.

Prerequisites

Before you begin any of the procedures below, ensure that you have the following:

- An AWS account with appropriate permissions to create and configure resources.
- A VPC already created in your AWS account.

- Familiarity with AWS services, especially Amazon VPC and AWS Transform custom.

Creating an interface VPC endpoint for AWS Transform custom

You can create a VPC endpoint for the AWS Transform custom service using either the Amazon VPC console or the AWS Command Line Interface (AWS CLI). For more information, see Creating an interface endpoint in the *Amazon VPC User Guide*.

Create the following VPC endpoints for AWS Transform custom using this service name:

- com.amazonaws.*region*.transform-custom

Replace *region* with AWS Region where you desire to use AWS Transform custom CLI, for example, *com.amazonaws.us-east-1.transform-custom*.

For more information, see Accessing a service through an interface endpoint in the *Amazon VPC User Guide*.

Creating a VPC endpoint policy for AWS Transform custom

You can attach an endpoint policy to your VPC endpoint that controls access to AWS Transform custom. The policy specifies the following information:

- The principal that can perform actions.
- The actions that can be performed.
- The resources on which actions can be performed.

For more information, see Controlling access to services with VPC endpoints in the *Amazon VPC User Guide*.

Example: VPC endpoint policy for AWS Transform custom actions

The following is an example of an endpoint policy for AWS Transform custom. When attached to an endpoint, this policy grants access to the listed AWS Transform custom actions for all principals on all resources.

```
{
   "Statement":[
```

```
      {
         "Principal":"*",
         "Effect":"Allow",
         "Action":[
            "transform-custom:*"
         ],
         "Resource":"*"
      }
   ]
}
```

Using an on-premises computer to connect to a AWS Transform custom endpoint

This section describes the process of using an on-premises computer to connect to AWS Transform custom through a AWS PrivateLink endpoint in your AWS VPC.

1. Create a VPN connection between your on-premises device and your VPC.
2. Create an interface VPC endpoint for AWS Transform custom.
3. Set up an inbound Amazon Route 53 endpoint. This will enable you to use the DNS name of your AWS Transform custom endpoint from your on-premises device.

Troubleshooting

This section provides guidance for troubleshooting common issues with AWS Transform custom.

Log Locations

AWS Transform CLI maintains logs in the following locations:

Conversation logs:

Linux and macOS

```
~/.aws/atx/custom/<conversation_id>/logs/<timestamp>-conversation.log
```

Windows

```
%USERPROFILE%\.aws\atx\custom\<conversation_id>\logs\<timestamp>-conversation.log
```

These logs contain the full conversation history for debugging specific transformation executions.

Subagent logs:

Linux and macOS

```
~/.aws/atx/custom/<conversation_id>/logs/subagents/<name>.log
```

Windows

```
%USERPROFILE%\.aws\atx\custom\<conversation_id>\logs\subagents\<name>.log
```

These logs contain output from subagents that the main agent spawns during transformations. You do not need to manage subagents directly.

Developer debug logs:

Linux and macOS

```
~/.aws/atx/logs/debug*.log
~/.aws/atx/logs/error.log
```

Windows

```
%USERPROFILE%\.aws\atx\logs\debug*.log
%USERPROFILE%\.aws\atx\logs\error.log
```

These logs provide detailed information about CLI operations and errors. Each log file has a 5MB limit before it rolls over. There may be multiple debug logs in this directory, such as debug1.log and debug2.log.

Common Issues

Installation issues:

If installation fails, ensure you have Node.js 22 or later installed:

```
node --version
```

Download Node.js from https://nodejs.org/en/download if needed.

Authentication issues:

Verify your AWS credentials are configured correctly:

```
aws sts get-caller-identity
```

Ensure your IAM user or role has the required `transform-custom:*` permissions.

Network connectivity issues:

If you encounter connection errors, verify network access to required endpoints:

- `transform-cli.awsstatic.com`
- `transform-custom.<region>.api.aws`
- `*.s3.amazonaws.com`

If working in an internet-restricted environment, update firewall rules to allowlist these URLs.

Region configuration issues:

If you encounter region-related errors:

- Verify your region is supported
- Check for environment variables that may override your configuration: `echo $AWS_REGION $AWS_DEFAULT_REGION`
- Check your region configuration: `aws configure get region`
- Update your region if needed: `aws configure set region <your-region>`
- Check debug logs for region resolution details

Git issues:

Ensure Git is installed and your repository is under git source control:

```
git --version
git status
```

AWS Transform custom requires repositories to be under git source control.

Transformation execution issues:

If a transformation fails:

1. Review the conversation logs at `~/.aws/atx/custom/<conversation_id>/logs/`
2. Check for build or test failures in the transformation output
3. Verify the build command is correct for your project
4. Try running the transformation in interactive mode to provide feedback

Conversation resumption issues:

If you cannot resume a conversation:

- Verify the conversation is less than 30 days old
- Check the conversation ID is correct
- Ensure you have network connectivity

Getting Support

For additional assistance, visit AWS Support through the AWS Console.

When opening a support ticket, include:

- Conversation logs from `~/.aws/atx/custom/<conversation_id>/logs/`
- Subagent logs from `~/.aws/atx/custom/<conversation_id>/logs/subagents/`
- Debug logs from `~/.aws/atx/logs/`
- Steps to reproduce the issue
- AWS Transform CLI version (`atx --version`)

S3 access issues

The AWS Transform custom service vends S3 pre-signed URLs to facilitate uploading and downloading the potentially large transformation files to/from client machines. This means that not only does the CLI interact with the AWS Transform service endpoints but it also interacts with the S3 service endpoints. These interactions use pre-signed URLs so your client's IAM credentials **do**

not require any S3 permissions but sometimes your local machine's proxy server configurations and your network's S3 VPC Endpoint Policies can restrict this traffic.

Please examine the the section called “Log Locations” for more detailed error messages to help root cause any tool failures or S3 access denied relating to downloading and uploading transformation files.

If you are using a VPN/Proxy server that leverages the `https_proxy` and `no_proxy` environment variables, consider adding the following values to your `no_proxy` environment variable value to bypass the proxy for the S3 and AWS Transform custom service endpoints, for example:

Linux and macOS

```
export no_proxy=.s3.amazonaws.com,transform-custom.<region>.api.aws
```

Windows (PowerShell)

```
$env:no_proxy=".s3.amazonaws.com,transform-custom.<region>.api.aws"
```

If you are using an AWS PrivateLink for Amazon S3 in your VPC, this may have a policy defined to restrict S3 traffic. Please ensure your S3 VPC endpoint policy allows `GetObject` and `PutObject` operations for the AWS Transform custom service-owned buckets. This can be accomplished by adding the following statement to your VPC endpoint policy:

```
{
    "Effect": "Allow",
    "Principal": "*",
    "Action": ["s3:PutObject","s3:GetObject"],
    "Resource": "arn:aws:s3:::aws-transform-custom-*/*"
}
```

Explicit `Deny` statements in policies take precedence over `Allow` statements. Review S3 VPC Endpoint policies for `Deny` statements that may be restricting access.

AWS Transform CLI version history

Review the following table for details about current and past releases of the AWS Transform CLI. The table includes the version, release date, and release notes for each version.

Version	Release date	Release notes
3.19.0 (latest)	October 13, 2026	• Changed `atx ct findings list` to list about 1,000 findings per run by default. Use `--limit` (up to 10,000) to list more. When more findings match, the command shows the total, prints a `--next-token` to continue, and exits with status 2. • Changed the MCP `findings_list` tool to return about 100 findings per call by default. • Changed `atx ct remote analysis` and `atx ct analysis run` to tell you to wait a minute and check `atx ct analysis list` before you submit again, when a server error or timeout leaves it unclear whether an analysis was created. • Fixed `atx ct analysis get` and `atx ct analysis run --json` to count findings past 10,000. • Fixed `atx ct status` to show the findings count for accounts of any size. • Fixed `atx ct findings list` failing when its filters match more findings than one read covers. • Fixed `atx ct analysis delete --cascade-findings` deleting an analysis even when it couldn't list the analysis's findings, for example because a request was throttled. The findings were left behind, and the analysis could no longer be found. The command now deletes large analyses in batches and deletes the analysis last. If a step fails, the command keeps what it hasn't deleted yet, tells you why and how many it deleted, and finishes the job when you run it again.
3.18.0	October 8, 2026	• Added categories for transformation definitions: `analysis` and `remediation` . • Added the `--category` option to `atx custom def publish` and `atx custom def save-draft` . • Added a Category column to `atx custom def list`.

Version	Release date	Release notes
		• Changed the agent to set the category when it saves a transformation definition. • Changed custom analyses (`--type custom`) to require one of your own transformation definitions in the `analysis` category. • Changed remediations to require one of your own transformation definitions in the `remediation` category, or an AWS managed transformation definition. To keep using a transformation definition with no category, publish it again with `--category .` • Changed custom analyses to report findings based on their transformation definition. • Changed custom analysis findings to recommend an AWS managed transformation as the fix when one applies. • Changed `atx ct findings list` to show the transform ation definition behind each custom finding. • Changed `atx ct remote analysis` and `atx ct remote status` to show clearer errors. • Fixed DNS lookup failures being reported as an unsupport ed Region. They're now reported as a network or DNS issue. • Fixed `/usage` and `--limit` undercounting Agent Minutes when you resume a session that didn't exit cleanly. • Fixed accented characters being replaced when the agent edits a file that isn't UTF-8, such as ISO-8859-1 or Windows-1252. • Fixed `atx ct` commands to exit with a non-zero code when they fail. • Fixed transformation commits to leave out build tool caches, such as `.gradle/` or `.nuget/`, and symlinks to paths outside the repository. Your `.gitignore` file isn't changed.

Version	Release date	Release notes
		• Fixed source-provider tokens to be masked in analysis and remediation error messages, including older ones. • Fixed rapid tech-debt analysis to mark a GitHub repository that it can't read as failed. • Fixed discovery scans of large GitHub organizations failing at the GitHub API rate limit. • Improved the reliability of remote analyses and remediati ons of large sources. • Fixed `atx ct remote status --group` to show every job's result in large Amazon EC2 groups. • Fixed agentic-readiness findings to match the severity in the assessment report. • Fixed agentic-readiness and modernization-readiness analyses to keep the HTML report in the run directory.
3.17.0	October 1, 2026	• Fixed the terminal UI to print the `atx --convers ation-id <id>` command for resuming the conversat ion. This command now appears when you quit or when a run stops after exhausting its retries. • Fixed `atx` on Node.js 16 or older to print which Node.js version is required and exit, instead of crashing with a dump of minified code. • Changed `atx ct remediation status` , `atx ct remote status`, and `atx ct analysis get` to show a plain-language explanation and a suggested next step when a repository or job fails. The raw error is still shown as detail. • Fixed agentic-readiness analyses run without `--agent-s cope` to determine the agent scope from each repository instead of defaulting to `read-only` .

Version	Release date	Release notes
3.16.0	September 25, 2026	• Added `--commit-message-template` to `atx ct source add` and `atx ct source update` to record the commit message format a source's repositories require; remediation commits are written to match instead of being rejected at push. Pass an empty string to remove a template. • Added `--branch-name-template` to `atx ct source add` and `atx ct source update` to record the branch name shape a source's repositories require; result branches are created in that namespace instead of being rejected by branch-name rules. Pass an empty string to remove a template. • Added `--git-committer-name` and `--git-committer-email` to `atx ct source add` and `atx ct source update` to set the git identity placed on remediation commits. Both are required together; pass both as empty strings to clear. • Added `atx ct remote analysis --display-name` to set a display name for a remote analysis, shown in `atx ct analysis list` and `atx ct analysis get`. • Increased the number of repositories a single remote analysis in customer-managed mode can cover to 1,000, up from 100. • Changed `atx ct source update` to no longer require `--token` — pass any combination of `--token` and the new template and committer options. Source settings can now be updated from any machine, not just the host that ran `atx ct source add`. • Changed `atx ct` to bound the number of simultaneous connections it opens to the analysis service. A very wide submission may log an informational `socket usage at capacity` message; tune with `ATX_MAX_SOCKETS` or raise your shell's open-file limit.

Version	Release date	Release notes
		• Added a `change_ready` finding status — a finding that has been remediated is reported as `change_ready` instead of staying open, so a finding whose change is waiting for review is distinguishable from one not yet worked on. Filter with `atx ct findings list --status change_ready` or count with `atx ct findings count --status change_ready`. The status is set automatically when a remediation succeeds for the finding's repository and can't be set by hand; dismissing a `change_ready` finding still works. • Changed failure reporting — when a repository or job fails, `atx ct remediation status`, `atx ct remote status`, and `atx ct analysis get` now show a plain-language explanation and a suggested next step, keeping the raw error as detail. • Fixed remote custom analyses (`--mode batch` and `--mode ec2`) failing to start — `--transformation-name` and `--configuration` are now forwarded to the container command. • Fixed `atx ct remote remediation --repos` and `--sources` refusing the whole submission when a repository has open findings with no automated fix; fixable findings are remediated and the rest are reported on stderr. • Fixed `atx ct remote status` to name repositories whose finding counts could not be retrieved instead of leaving them blank. • Fixed an `atx ct remote analysis --mode ec2` submission that fails while starting to clean up the analysis records it created, instead of leaving them listed as running.

Version	Release date	Release notes
3.15.0	September 16, 2026	• Made the terminal UI the default experience for interactive sessions and `exec` runs. To keep the previous line-oriented output, pass `--classic` or set `ATX_CLASSIC=1` . • Updated the `--tui` terminal UI to honor `-q, --quiet`, hiding tool activity from auto-approved tools while still rendering tools that need your approval and any failures. • Fixed an issue where the agent could stall for a step when creating or editing a file. • Changed `atx ct` remediations run with `--mode ec2` to respect the per-account concurrency limit, matching analyses and Batch-mode remediations. Work that exceeds your runner's capacity now queues and is released as slots free up; `atx ct remote status --group` reports the waiting jobs as `queued`. • Fixed `atx ct remediation create` to report a clear input error instead of a generic server error when given an unknown or malformed finding id. • Fixed `atx ct` analyses and remediations on a Bitbucket Cloud source that failed with "401 Unauthorized" when the source was added with only an API token. An account email is no longer required for Bitbucket Cloud, and sources already connected that way work without being re-added.
3.14.1	September 11, 2026	• Fixed `atx ct remote provision --mode batch` to use a shorter region identifier when the generated infrastructure bucket name would exceed the Amazon S3 63-character limit, while preserving cross-region bucket isolation.

Version	Release date	Release notes
3.14.0	September 10, 2026	• Added the `--tui` option, which runs the CLI in a new terminal UI. • When a shell command started with `background=true` finishes, its result is now delivered to the agent automatic ally on its next step. Completions are picked up without the agent having to check on them. • Fixed an issue where Agent Minutes double-counted a turn's partial usage when a sub-agent was billed. • Fixed an issue where a turn whose stream closed without completing hung or reported success. It now fails with an error. • Fixed `atx ct` per-repository analysis progress reporting. A scoped container no longer overwrites the progress of the repositories running alongside it. • Fixed `atx ct remote` Amazon EC2 jobs to report why an SSM launch failed, instead of surfacing only the probe's exit code.

Version	Release date	Release notes
3.13.0	September 2, 2026	• Added `atx ct remediation run` as an alias for `atx ct remediation create`, for consistency with `atx ct analysis run`. • `atx ct remote network discover` now prints a next-step hint to run `atx ct remote network create` when it finds no private subnets. • `atx ct remote status` now reports jobs waiting for account concurrency as `queued` instead of `pending`. This distinguishes a job that is waiting for capacity from one whose state could not be read. Applies to both Amazon EC2 and AWS Batch. • `atx ct remote status` now reports where a failed job's container log is. For Amazon EC2 this is the log group and stream, and for AWS Batch this is the CloudWatch stream. Previously it showed only an exit code. • Changed resuming with `--conversation-id` to apply `-g`, `--configuration` on top of the session's stored configuration one setting at a time, instead of replacing all of it. Settings you do not pass again are kept rather than dropped. • Fixed an issue where cancelling a remote analysis terminated the container running it. The job now completes cleanly instead of being reported as failed and retried by AWS Batch. • Fixed an issue where cancelling a remote analysis left its containers polling until the wait timeout. Containers now end promptly, and a cancelled job is no longer reported to AWS Batch as a failure and retried. • Fixed an issue where cancelling a `security` analysis stopped only local polling while the code review jobs ran to completion. The jobs are now stopped service-side. • Fixed an issue where `atx ct analysis cancel` marked a running `rapid-techdebt-analysis` as cancelled

Version	Release date	Release notes
		while the scan continued over the remaining repositories. Applies to both local and remote runs. • Fixed an issue where `atx ct analysis cancel` marked a running agentic-readiness or modernization-readiness analysis as cancelled while every remaining repository was still assessed. Assessments that already finished keep their results. Applies to both local and remote runs. • Fixed `atx ct remote status --group` to report a finished Amazon EC2 job as `complete`, so `--wait` returns instead of polling to its timeout. • Fixed `atx ct remote status` to still report what it observed when the Amazon EC2 instance is gone, instead of failing outright. • Fixed `atx ct remote status` to include waiting jobs in its summary line, so the counts add up to the number of jobs submitted. • Fixed `atx ct remote status --batch` to show why a job failed to start, instead of leaving it as `pending`. • Fixed `atx ct remote status --batch` to no longer report `not found` for a batch that was just submitted and is waiting for capacity. • Fixed `atx ct analysis get` and `atx ct remediation status` to settle a run whose repositories have all finished. • Fixed `atx ct remote analysis --resume-incomplete` to no longer resubmit a repository that is still queued, which could run the same repository twice. • Fixed analysis and remediation for self-hosted GitLab and Bitbucket Data Center instances that are served under a URL subpath, for example `https://host/gitlab`, or over plain HTTP. Previously repository discovery succeeded but every clone and push failed.

Version	Release date	Release notes
3.12.0	August 26, 2026	• The CLI now supports `atx ct` commands on Windows. • Changed `atx ct` remote Amazon EC2 analyses to default to 5 parallel workers. • Added a `-q`, `--quiet` flag to `atx exec` and `atx custom def exec`. The flag mutes tool-call output for auto-approved tools, while tools that require approval still render in full. Combine it with `--trust-all-tools` to silence all tool activity. • Fixed `file://` configuration paths on Windows, including paths with spaces, so they now resolve correctly. • Fixed reading of files that have a UTF-8 byte-order mark, so first-line edits and searches no longer fail. • Fixed file overwrites to preserve the original line endings, avoiding spurious whole-file changes on Windows. • Fixed analysis report parsing and patch application on Windows to handle CRLF line endings. • Fixed status symbols to fall back to plain text on older Windows terminals that cannot display them. • Fixed an issue where a kill-family command issued by the agent through shell expansion could terminate the CLI's own process. For example, commands such as `kill $PPID` or `kill $$` could affect the CLI. The `shell` tool now checks these against the CLI's own process and its ancestor chain before running them. • Fixed clone operations to retry on transient network failures. • Fixed custom analyses to retry repository resolution when it first returns no repositories. • Fixed comprehensive and custom analyses to keep the local clone needed for artifact upload.

Version	Release date	Release notes
		• The CLI now rejects invalid `--severity` filters on remote analyses with a clear error. Previously, it ignored them. • Fixed an issue where a completed recurring scheduled analysis reported 0 findings even when findings existed. `atx ct analysis get` and `atx ct analysis run --wait` now report the correct finding count. • Fixed an issue where AWS Batch remote analyses could fail with `InsufficientFreeAddressesInSubnet` when a subnet was low on IP addresses. The dispatcher now waits for available IP capacity before submitting each job. • Fixed a dispatcher stall where AWS Batch and Amazon EC2 jobs could get stuck pending under high deferral volume. This could happen with IP-constrained subnets or busy Amazon EC2 workers. • Fixed remote provisioning to advertise the correct region-qualified dispatcher IAM role name. This ensures the PassRole grant an administrator sets up matches the role the dispatcher uses. • To apply the dispatcher fixes to an already-deployed stack, run the following command: `atx ct remote update --mode ec2|batch --stack-name <stack-name> [--execute] [--ack]`

Version	Release date	Release notes
3.11.0	August 19, 2026	• Added `--repos` to `atx ct remediation create` for multi-repo remediation in a single command. • Added `--transformation-name` to `atx ct remote remediation` to run a remediation directly from a transformation definition without requiring existing findings. • Added `--json` output to `atx ct analysis run` for machine-readable results. • Added an Agent Minutes cost summary to `atx ct analysis get` and `atx ct remediation status` . • Added a local file export for `atx ct remote provision` in dry-run mode: the generated AWS CloudFormation templates are now written to a local file (override the location with `--out-dir`) for review before deployment. • Fixed an issue where kill-family commands issued by the agent could terminate the CLI's own process. Affected commands include `pkill -f node`, `xargs kill`, and `find -exec kill`. The `shell` tool now checks these against the CLI's own process and its ancestor chain before running them. • Fixed an issue where GitLab source discovery used repository topics as the language field. This could cause repositories with special characters in their topics to fail registration. Language is no longer inferred from topics. • Fixed an issue where `atx ct` commands that failed could exit silently with empty output. The CLI now surfaces actionable error messages. • Fixed an issue where finalizing a local analysis run could error when the service reported the run was already terminal. • Fixed an issue where `atx ct remote update` did not update the shared dispatcher infrastructure. As a

Version	Release date	Release notes
		result, upgrading the CLI and running `remote update` continued to use the previously deployed dispatcher. `remote update` now reconciles the dispatcher. • Fixed an issue where remote analysis runs (Amazon EC2 or AWS Batch) could report a false failure. This occurred when all repositories in a scoped run had already completed on the service. • Fixed an issue where `atx ct` repository discovery could silently drop repositories the service rejected. The CLI now counts and reports skipped repositories ("N repositor ies were skipped"), with an overflow summary for large organizations. • Changed user-facing error and warning messages to refer to AWS Transform and state a clear next step (refresh credentials, retry, or contact support). Messages no longer reference internal service names.

Version	Release date	Release notes
3.10.0	August 10, 2026	• Added support for `background=true` in the `shell` tool for long-running or non-terminating commands (dev servers, build watchers, `tail -f`). It returns immediately with a process ID instead of waiting for the command to exit. Up to 5 background processes can be tracked at once. • Added new `check_background_shell` and `stop_background_shell` tools to monitor and terminate background processes. • Added `llmConfig` to the CLI configuration allowlist, enabling users to customize LLM settings via config files. • Added `--mode aws-managed` to `atx ct schedule create` and `atx ct remote analysis` to run analyses using fully managed AWS infrastructure without provision ing your own Amazon EC2 or AWS Batch resources. • Added `--schedule-id` and `--display-name` options to `atx ct analysis` to filter results by schedule and identify analyses by a human-readable label. • Added `--region` support for `--mode aws-managed` operations, allowing you to target a specific AWS Region for managed analysis runs. • Improved reliability of `atx ct schedule get`, `enable`, `disable`, and `delete` commands. • Improved accuracy and consistency of `atx ct schedule list` results. • Added `--stack-name` to `atx ct schedule create` to pin a schedule to a specific target stack. • Added new `atx ct analysis list-artifacts` and `atx ct analysis get-artifact` commands for retrieving output artifacts from AWS-managed infrastru cture analysis runs.

Version	Release date	Release notes
		• Fixed an issue where background processes tracked by the CLI were not properly terminated on CLI exit, which could leave orphaned processes running. • Fixed an issue where `atx ct schedule create` did not reject the `--workers` and `--batch-name` flags when used with `--mode aws-managed` . • Fixed inconsistent wording in schedule not-found error messages. • Fixed an issue where endpoint overrides were not resolved consistently across all CLI commands. • Improved accuracy of duplicate schedule name detection. • Fixed an issue where `atx ct remote analysis --resume` did not resume a partially-completed batch run. The flag is now `--resume-incomplete` , which re-runs only the repositories that did not finish (failed, interrupted, or not yet started) and skips completed ones. • Updated `atx ct schedule --help` to include AWS Managed Analysis run offerings.

Version	Release date	Release notes
3.9.0	July 31, 2026	• Improved the performance of `atx custom def export-ki-markdown` on transformation definitions that contain many knowledge items. The command no longer makes a redundant service call for each knowledge item, so it runs significantly faster and no longer times out on large definitions. Also fixed cases where an export could omit knowledge items or report an inconsistent total. • Fixed an issue where agentic-readiness and moderniza tion-readiness analyses run with `atx ct analysis run` could report zero findings for certain report formats. Findings are now recovered reliably. • Fixed an issue where scheduled analyses created with `atx ct schedule create` against GitLab, Bitbucket , or local sources could fail on every run while trying to retrieve credentials. The source provider is now determine d automatically from your registered source, and the `--provider` option is no longer required. • Fixed an issue where security analyses run in remote (AWS Batch or EC2) environments could incorrectly report that the Security Agent was not configured. The CLI now detects a configured Security Agent correctly and reports transient credential or throttling errors more gracefully.

Version	Release date	Release notes
3.8.0	July 29, 2026	• Added the `--recurrence` option to `atx ct schedule create`, so you can set a schedule's cadence in plain terms (`daily`, `weekly:<MONDAY..SUNDAY>`, or `monthly:<1..28>`) instead of writing a raw cron expression. • Added the `--existing-scheduler-role-arn` option to `atx ct remote provision`, so you can reuse an IAM role that your administrator has already created for the scheduler instead of having the CLI create one during provisioning. • Added the `--existing-instance-profile-name` option to `atx ct remote provision --mode ec2`, so you can attach an IAM instance profile that your administrator has already created instead of having the CLI create the transform role and instance profile. • Improved transformation selection guidance so that it now uses detection metadata, such as severity and matching criteria, when available. This helps surface the transformations that best fit your project and their relative priority. • Fixed an issue where canceling a scheduled remote analysis stopped the running containers but could leave the analysis stuck in a `running` state. Canceling now reliably marks the analysis as canceled. • Fixed an issue where `atx ct schedule delete` could leave a lingering analysis record behind. Deleting a schedule now also removes the underlying scheduled analysis.

Version	Release date	Release notes
3.7.0	July 22, 2026	• Added the `atx ct remote` command group for running analyses and remediations remotely on EC2 or AWS Batch. You can provision, update, and tear down the remote execution infrastructure, check submission status, detect deployed stacks, and manage source credentials in AWS Secrets Manager. • Added the `atx ct schedule` command group for recurring scheduled analysis and remediation jobs. You can create, list, enable, disable, and delete schedules (job options mirror `remote analysis` and `remote remediation`), plus `setup` and `teardown` for the scheduler infrastructure. Schedules capture local source bundles at create time. • Improved `atx ct analysis run` to consume tech-debt -quick matching rules from the transformation package registry. • Improved long-running `atx ct` commands to show progress output, including per-repo progress during analysis runs (including security analyses) and per-repo failure reasons in remediation status. • Fixed several remote execution reliability issues. EC2 status now shows findings and pull request results, matching Batch; EC2 updates no longer fail on stacks missing new parameters; and additional fixes address exit codes, transform handling, tag-based detection, and JSON output.

Version	Release date	Release notes
3.6.0	July 17, 2026	• Added the `atx custom def learnings` subcommand, which opens an interactive TUI session to view and manage the learnings automatically generated for a transformation definition based on previous runs. • Added usage telemetry collection during transformation execution. To disable telemetry, set the `ATX_DISABLE_TELEMETRY=true` environment variable. For more information, see Usage Telemetry. • Fixed an issue on Windows where file operations using ~ (home directory) paths resolved to an unexpected directory. This caused file read/write failures during a session. • Fixed an issue where MCP server and tool names containing special characters could cause connection failures. The CLI now automatically sanitizes names to use only valid characters.

Version	Release date	Release notes
3.5.0	July 8, 2026	• Added native Windows support. You can now run the AWS Transform Custom commands directly on Windows. (The `atx ct` commands are not yet supported on Windows) • Added steering inputs to `atx ct analysis run` so agentic-readiness and modernization-readiness analyses reflect your preferences. Use `--context` for strategic framing, `--agent-scope` to set agent access for agentic-readiness, and `--prefer`/`--avoid` to steer modernization recommendations toward or away from specific technologies. • Transformation definitions in the skill format can now include a `scripts/` folder, letting you bundle helper scripts alongside your `SKILL.md` and references. • Added `--wait` to `atx ct remediation create`, which waits until the remediation completes. • Added headless mode for one-shot execution. Run `atx -x "prompt" -t` to pass a prompt, get a response, and exit with no interactive prompts. Headless mode requires trusting all tools.
3.4.1	July 2, 2026	• Fixed an interactive-session issue that version 3.4.0 introduced. After the agent finishes responding, the CLI might stay on the "Thinking" prompt (shown with an ellipsis in the UI) instead of returning to the input prompt. The CLI now returns you to the input prompt after each response.

Version	Release date	Release notes
3.4.0 (not recommended)	June 29, 2026	**⚠ Known issue** In interactive sessions, the CLI might stay on the "Thinking" prompt (shown with an ellipsis in the UI) after the agent finishes responding. This prevents you from entering your next message. To resolve this issue, upgrade to version 3.4.1 or later by running `atx update`. • Added `atx ct schema`, a machine-readable manifest of the full `atx ct` command surface for scripting and automation. • Added token scope validation when you add or update a source, surfacing credential or permission issues immediately instead of failing later during analysis. • Improved auto-remediation pull requests to include finding context and support draft mode for more reviewable PRs. • Improved CLI progress feedback with real-time context about the current activity, estimated agent minutes, and sub-agent activity. • Added configurable checkpoint commit author identity through the `ATX_GIT_COMMITTER_NAME` and `ATX_GIT_COMMITTER_EMAIL` environment variables. • Various small bug fixes and improvements.
3.3.0	June 18, 2026	• Added consistent `--json` and `--help` support across `atx ct` commands. • Fixed security agent setup so the agent space is created during setup and removed on teardown. • Various small bug fixes and improvements.

Version	Release date	Release notes
3.2.0	June 15, 2026	• Added continuous modernization commands (`atx ct`) to the standard CLI distribution for all users. • Improved sub-agent execution to show a "Thinking..." spinner while waiting for server responses, providing clear visual feedback that work is in progress. • Improved spinner to automatically restart with "Working on it..." after 3 seconds of silence following agent output, so the terminal never appears frozen between processing steps. • Improved thinking spinner with a subtle shimmer animation for a more polished visual appearance.
3.1.0	June 3, 2026	• Added `ATX_MAX_VALIDATION_LOOPS` environment variable to configure maximum validation loops. • Fixed update check no longer shows false "update available" warnings when the installed version is already newer than the registry version. • Fixed `delete --transformation-name ""` no longer shows a nonsensical confirmation prompt; empty values are now rejected with a clear validation error. • Fixed unrecognized subcommands now show "unknown command" instead of a confusing internal error message. • Fixed version comparison to use numeric comparison instead of string inequality for update checks.

Version	Release date	Release notes
3.0.0	May 21, 2026	• Added pre-flight transformation name validation that suggests similar names from your registry when a name is not found, helping catch typos without needing a separate list command. • Added optional explanations for shell commands, showing what they do and why. • Improved trust settings banner to include more informati on about tool permissions. • Added learning-disabled banner when using the do-not-le arn flag, giving visibility to the learning feature and how to enable it. • **Breaking change:** Node.js 22 or later is now required. Upgrade from https://nodejs.org/ if you are using an older version.

Version	Release date	Release notes
2.0.0	May 14, 2026	• Added remote MCP server support — connect to HTTP and HTTPS MCP servers with automatic SSE fallback, Bearer token authentication with environment variable expansion in headers, and proxy support (HTTP_PROXY/HTTPS_PROXY). • Added `atx resume` subcommand as alias for `atx --resume`. • Added `grep` tool for multi-file text search with output control — supports regex patterns, file glob filtering, case-insensitive search, configurable depth/result limits, and output truncation. • Improved MCP implementation to use the standard MCP SDK for more consistent behavior and easier support for future MCP features. • Improved shell output handling — output is now gracefully truncated at 30K characters per stream instead of causing a hard error, preserving the beginning and end of output for context. • Fixed an issue where Ctrl+C during an interactive session could prematurely exit the CLI on Linux/macOS instead of aborting the current operation and returning to the prompt. • Fixed trust rejection message to show a dedicated reason for why trust is unavailable, separate from why the command needed authorization. • Fixed dangerous command warning no longer mentions trusting when the command is in `alwaysPromptCommands`.

Version	Release date	Release notes
1.10.0	April 29, 2026	• Improved tool descriptions with cross-tool guidance, helping the agent choose the best tool for each task and reducing unnecessary shell usage. • Added `alwaysPromptCommands` in `trust-settings.yaml` — configure shell command patterns that always require explicit permission, regardless of trust settings. Supports glob wildcards and matches against each sub-command in compound expressions. `trustedShellCommands` can override these patterns. Not enforced in non-interactive mode to avoid breaking automated pipelines. • Fixed relative paths not being resolved into absolute paths.
1.9.0	April 23, 2026	• Improved tool approval prompts — each option now shows what it does inline, a “Why” message explains why approval is needed, and trust confirmation includes a hint about `trust-settings.yaml` for permanent trust.
1.8.0	April 16, 2026	• Changed agent output prefix from `>` to `ATX:` for clearer visual distinction between user input and agent output. • Fixed display of resume command when CLI retries are exhausted. • Fixed resume from a different working directory. In interactive mode, prompts for corrected path; in non-inter active mode, fails fast with a descriptive error.
1.7.1	April 9, 2026	• Fixed SSO login failure ("Session token not found or invalid") when the credential provider region differs from the profile's `sso_region` . • Fixed credential caching so long-running sessions no longer fail when credentials are rotated on disk.

Version	Release date	Release notes
1.7.0	April 2, 2026	• Fixed `exec` command to exit with code 1 on input validation errors, making the CLI reliable in scripts and CI/CD pipelines. • Added display of warning messages from the service. • Added MFA prompt support for AWS credential profiles. • Fixed STS credential calls to respect HTTP(S) proxy settings.
1.6.0	March 18, 2026	• Added `--limit <minutes>` option to set an Agent Minutes budget for `exec` and `interactive` commands. When the limit is reached, the CLI displays usage and exits with resume instructions. • Fixed logging of the initial user message in conversation logs.
1.5.0	March 11, 2026	• Added support for concurrent task execution, enabling faster transformations. • Added Agent Minutes usage display at the end of each conversation, and `/usage` command in interactive mode to check usage at any time.
1.4.0	March 2, 2026	• Improved error messages when saving transformation definitions with unsupported files. • Added timeout to service calls to prevent the CLI from hanging when the connection is interrupted. • Fixed CLI process hang after successful completion. • Fixed region validation when resuming conversations.
1.3.0	February 19, 2026	• Removed restriction for MCP server paths — users may now specify any path for the `command` property in MCP configuration. • Improved stability of MCP server connections.

Version	Release date	Release notes
1.2.0	February 9, 2026	• Added support for proxy configuration via environment variables (`https_proxy` , `no_proxy`, etc.). • General stability improvements and bug fixes.
1.1.1	January 16, 2026	Bug fixes and stability improvements.
1.1.0	January 13, 2026	Added AWS CLI region configuration support, early credentia l validation checks, and fixed campaign resume functionality.
1.0.1	December 18, 2025	Graceful shutdowns, cleaner error logs, and general improvements.
1.0.0	December 1, 2025	Initial release of the AWS Transform CLI.

AWS Transform continuous modernization

What is AWS Transform continuous modernization?

AWS Transform continuous modernization provides analysis and remediation for your source code repositories. You can connect GitHub organizations, GitLab groups, Bitbucket workspaces, and local repositories, then run automated analyses to identify technical debt, security vulnerabilities, modernization opportunities, and agent-readiness across your entire codebase.

All analysis and remediation run in your AWS account using your credentials. Your source code stays under your control.

> **Note**
>
> We recommend starting with the Kiro Power or agent plugin. The agent skill orchestrates setup, onboarding, and ongoing use of continuous modernization including infrastructure provisioning, source configuration, analysis execution, findings triage, and remediation. For installation instructions, see Developer tools.

Key capabilities

AWS Transform continuous modernization provides the following capabilities:

- **Tech Debt Analysis** — Scan repositories for outdated dependencies, security vulnerabilities, code quality issues, and modernization opportunities. Run quick agent-driven scans across package manifests or comprehensive code-level analysis. Define custom analysis criteria using transformation definitions tailored to your environment.
- **Autonomous Remediation** — Generate validated pull requests at scale. Each finding can be auto-remediated using its associated transformation definition. Remediation creates branches and opens PRs/MRs automatically across GitHub, GitLab, and Bitbucket.
- **Reporting** — Generate HTML reports showing findings by severity, repository, and analysis type. Track remediation progress and finding resolution over time.
- **Continuous Monitoring** — Schedule recurring analyses to continuously monitor your portfolio for new issues. Run them on AWS Transform-managed infrastructure with no setup, or on a

provisioned Amazon EC2 or Batch stack using Amazon EventBridge Scheduler. Configure daily, weekly, or monthly cadences.

Analysis types

AWS Transform continuous modernization supports the following analysis types to address your modernization needs.

Type	Description
`rapid-techdebt-analysis`	Fast scan of package manifests (`pom.xml`, `package.json` , `requirements.txt`) using the AWS Transform agent to identify stale versions and outdated dependencies. Does not analyze source code.
`tech-debt-comprehensive`	Deep code-level technical debt analysis using the AWS Transform agent. Examines source code to identify debt patterns, code quality issues, architecture concerns, and improveme nt opportunities.
`security`	Static application security testing (SAST) using the AWS Security Agent. Scans source code for insecure coding patterns such as SQL injection, hardcoded secrets, and unsafe deserialization. Requires one-time infrastructure setup.
`agentic-readiness`	AI and agent integration readiness assessmen t. Scores 56 criteria across five categorie s: Infrastructure & Platform, Application Architecture, Data Foundations, Identity/ Security/Governance, and Operations & Observability.
`modernization-readiness`	Cloud modernization opportunity assessmen t. Evaluates readiness across infrastructure,

Type	Description
	application, data, security, and operations dimensions. Identifies candidates for container ization, serverless migration, and platform upgrades.
`custom`	Run one of your own transformation definitio ns as an analysis where each issue described is reported as a finding, which can link to an AWS managed transformation that fixes it.

Understanding key concepts

Sources

A source tells continuous modernization where your repositories are. Supported source types:

- **GitHub** — Organizations with personal access tokens (classic)
- **GitLab** — Groups or users, including self-hosted instances
- **Bitbucket** — Workspaces (Cloud) or projects (Data Center)
- **Local** — Parent directories containing git repositories

Repositories

Repositories are discovered by scanning sources. After discovery, you can filter by source, labels, or other criteria; apply labels for organization (team, priority, migration wave); and target specific repositories for analysis or remediation.

Analyses

An analysis is a scan of one or more repositories using a specific analysis type. Each analysis produces findings that identify issues in your code. You can run analyses on demand or schedule them to run automatically. Analysis types include rapid-techdebt-analysis, tech-debt-comprehensive, security, agentic-readiness, modernization-readiness, and custom. Each analysis tracks its status (pending, running, complete, cancelled, or failed) and the repositories it scanned.

Findings

Findings are the results of analysis. Each finding includes severity (high, medium, or low), status (open, dismissed, or obsolete), and the fix transform that can remediate it (if auto-fixable). New findings start as open, and users can dismiss them with a reason. Re-analysis marks resolved findings as obsolete automatically.

Remediations

Remediations apply transformation definitions to fix findings. Three modes: findings-based (each finding uses its own fix transform), TD override (override the transformation definition for specified findings), and direct TD (run a transformation against a repository without findings). Output depends on the source provider (GitHub PR, GitLab MR, Bitbucket PR, or local branch).

Transformation definitions

A transformation definition contains the instructions and knowledge needed to perform a specific code transformation. There are two kinds:

- **AWS managed transformation definitions** are pre-built and vetted by AWS, and are ready to use without any setup. Their names start with `AWS/`, for example `AWS/java-version-upgrade`. Findings link to them as the fix.
- **Custom transformation definitions** are ones you create and publish to the transformation registry in your AWS account. Use them to look for your own issues with a custom analysis, or to make your own changes with a remediation.

A transformation definition's **category** determines where it can be used: custom analysis runs transformation definitions with the `analysis` category, and remediation runs transformation definitions with the `remediation` category or AWS managed transformations. For more information, see Transformation definition categories.

How AWS Transform continuous modernization works

AWS Transform continuous modernization is typically used in large-scale projects where multiple codebases need continuous analysis and remediation. Teams typically follow this workflow:

1. **Connect Sources** — Add GitHub organizations, GitLab groups, Bitbucket workspaces, or local directories as sources. Provide authentication tokens with appropriate scopes.

2. **Discover Repositories** — Run discovery scans to enumerate all repositories in your sources. Use labels to organize repositories by team, priority, or migration wave.
3. **Run Analysis** — Execute analyses across your portfolio. Choose quick scans for fast results or comprehensive analysis for detailed findings. Security analysis requires one-time infrastructure setup.
4. **Triage Findings** — Review findings by severity, repository, or analysis type. Dismiss false positives with documented reasons. Re-run analysis to mark resolved issues as obsolete.
5. **Remediate** — Create remediations to fix findings. Continuous modernization automatically creates branches and opens pull/merge requests with the fixes. Track remediation status and retry failures.
6. **Set Up Continuous Analysis** — Schedule recurring analyses with the `atx ct schedule` commands, which use Amazon EventBridge Scheduler. Configure analysis cadence (daily, weekly, or monthly) to continuously monitor your portfolio for new issues. Combine with automated remediation to maintain code health over time.

Compute options

Continuous modernization supports several compute options for running analyses and remediation. Analyses can run locally, on infrastructure you provision and manage in your AWS account (Amazon Amazon EC2 or AWS Batch), or on AWS Transform-managed infrastructure that requires no setup. The agent skills in the Kiro Power and agent plugin help you set up each option.

> **Note**
>
> Regardless of compute option, you create and own all resources in your AWS account using your credentials, and your source code remains under your control. The infrastructure that runs analyses and remediation is either customer-managed—a stack you provision in your account—or AWS Transform-managed, where AWS operates the compute for you and there is nothing to provision.

Local (default)

By default, analyses run on your local machine. This option requires no additional infrastructure. The server runs locally and executes analyses using local compute resources. Good for trying out the tool, small repositories, or individual use.

AWS Transform-managed (no infrastructure to provision)

Run analyses on AWS Transform-managed infrastructure without provisioning or managing a stack. With `--mode aws-managed`, you submit the analysis to AWS Transform, which runs it on AWS-operated infrastructure. There is no Amazon EC2 instance, no AWS Batch stack, and no VPC or subnets to configure. There is nothing to provision or tear down—the submission is the run. This is the fastest way to run remotely when you don't want to manage any infrastructure.

This option requires a source hosted on a supported SCM provider (GitHub, GitLab, or Bitbucket). AWS Transform-managed infrastructure does not support local sources because it cannot reach repositories on your machine. You can choose the AWS Region the workload runs in with the `--region` option.

AWS Transform-managed infrastructure runs analyses only. For remediation, or to run a custom transformation definition (`--type custom`), use AWS Batch or Amazon EC2.

To run on AWS Transform-managed infrastructure, use the `atx ct remote analysis --mode aws-managed` command (see Remote execution), or install the Kiro Power or agent plugin (see Developer tools) and ask the agent: *"Run my analysis on AWS with no infrastructure"*. A recurring analysis on AWS Transform-managed infrastructure requires an execution role (see Scheduling recurring analysis).

You can also start an analysis on AWS Transform-managed infrastructure from the continuous modernization page in the AWS Transform web application (see AWS Transform web application).

AWS Batch (Fargate) (recommended)

For most portfolios, you can run analyses on AWS Batch with Fargate, which is the recommended compute option. Each analysis runs as an isolated job in its own container with serverless compute, so there is no persistent infrastructure to manage, and jobs scale out for parallel analysis of multiple sources or analysis types.

Provisioning deploys the required infrastructure including:

- AWS Batch job queue and compute environment
- Job definition with the continuous modernization container image
- IAM roles for batch job execution
- A Lambda function for job submission

Provisioning infrastructure requires administrator permissions. To run analyses and remediations on an already-provisioned Batch stack with least privilege, attach the AWS managed policy `AWSTransformInfrastructureExecutorAccessBatch`.

To set up Batch execution, use the `atx ct remote` commands directly (see Remote execution), or install the Kiro Power or agent plugin (see Developer tools) and ask the agent: *"Run my analysis on Fargate"* or *"Set up Batch execution for continuous modernization"*. The agent deploys the stack, stores your source credentials in Secrets Manager, and submits analysis jobs to AWS Batch.

Amazon Amazon EC2

Run analyses on a persistent Amazon EC2 instance in your AWS account. This option offloads compute from your local machine, supports larger analyses, and enables recurring scheduled analysis. The instance stays running between submissions.

Provisioning deploys a AWS CloudFormation stack that includes:

- An Amazon EC2 instance (Amazon Linux 2023) with Docker and the continuous modernization container
- An IAM role with permissions for AWS Transform, Amazon S3, AWS KMS, Secrets Manager, and the security agent
- `AmazonSSMManagedInstanceCore` for shell access via SSM (no SSH key pair or inbound ports required)
- A security group with no inbound rules

Provisioning infrastructure requires administrator permissions for Amazon EC2 lifecycle management, AWS CloudFormation stack operations, IAM role creation, Amazon S3 bucket operations, Secrets Manager secret management, and SSM commands. To run analyses and remediations on an already-provisioned Amazon EC2 stack with least privilege, attach the AWS managed policy `AWSTransformInfrastructureExecutorAccessEC2`.

To set up Amazon EC2 execution, use the `atx ct remote` commands directly (see Remote execution), or install the Kiro Power or agent plugin (see Developer tools) and ask the agent: *"Set up an EC2 instance for continuous modernization analysis"*. The agent provisions the infrastructure, verifies the container is healthy, and submits your analyses via SSM — no SSH required.

Security agent setup

The `security` analysis type uses the AWS Security Agent service for static application security testing (SAST) of your source code. Unlike other analysis types, it requires one-time infrastructure setup in your AWS account.

> **Note**
>
> The `security` analysis type is not available in the following Regions: Canada (Central) (`ca-central-1`), Europe (London) (`eu-west-2`), and Asia Pacific (Seoul) (`ap-northeast-2`). The other analysis types are unaffected.

The setup command provisions a AWS CloudFormation stack that includes:

- An Amazon S3 bucket for source code upload
- An IAM role assumed by `securityagent.amazonaws.com`
- A managed policy for the security agent role

To run security analyses with least privilege after setup, attach the AWS managed policy `AWSTransformSecurityAgentExecutorAccess` to your IAM user or role.

Your IAM user or role must have the following additional permissions to run the setup:

- `cloudformation:CreateStack`, `cloudformation:UpdateStack`, `cloudformation:DescribeStacks`
- `iam:CreateRole`, `iam:PutRolePolicy`, `iam:AttachRolePolicy`, `iam:CreatePolicy`
- `s3:CreateBucket`, `s3:PutBucketEncryption`, `s3:PutBucketPublicAccessBlock`

Use the following commands to manage the security agent infrastructure:

```
# Provision the security agent infrastructure
atx ct setup security-agent

# Check the status
atx ct setup security-agent --status

# Delete the infrastructure
```

```
atx ct setup security-agent --delete
```

After setup completes, you can run security analyses the same way as other analysis types.

Getting started

Prerequisites

To use continuous modernization effectively, you should have a basic understanding of Git version control, familiarity with your source code platform (GitHub, GitLab, or Bitbucket), working knowledge of AWS IAM permissions, and experience with command-line tools.

Before you use continuous modernization, verify that you have the following:

- AWS Transform CLI installed:

```
curl -fsSL https://transform-cli.awsstatic.com/install.sh | bash
```

- Node.js 22 or later
- Valid AWS credentials (`AWS_PROFILE` or environment variables)
- The AWS managed policy `AWSTransformCustomFullAccess` attached to your IAM user or role

> **Note**
>
> Running analyses and remediations on remote infrastructure or with the security agent requires additional managed policies. See the compute options and security agent setup in How AWS Transform continuous modernization works.

Quick start

> **Note**
>
> You no longer need to start a local server. The `atx ct server` command is deprecated.

1. Add a source:

```
atx ct source add --name name --provider provider --org org-or-group --token token
```

Where *name* is a descriptive identifier for your source, *provider* is the platform (`github`, `gitlab`, `bitbucket`, or `local`), *org-or-group* is your organization or group name, and *token* is your authentication token.

2. Discover repositories:

```
atx ct discovery scan --source name
```

3. Run an analysis:

```
atx ct analysis run --type type --source name --wait
```

4. Explore findings:

```
atx ct findings list --json
```

5. Remediate findings:

```
atx ct remediation create --ids id1,id2 --name "remediation-name"
```

Working with continuous modernization

Source management

Use `atx ct source` commands to connect repositories. Supported providers: GitHub, GitLab, Bitbucket, local.

GitHub organizations

Token: personal access token (classic) with `repo` scope. Read-only for analysis, full repo for remediation.

```
atx ct source add --name name --provider github --org org --token pat
```

GitLab groups and users

Token: personal access token with `api` scope.

```
atx ct source add --name name --provider gitlab --org group-or-user --token pat
# Self-hosted:
atx ct source add --name name --provider gitlab --org group-or-user --token pat --url
 https://gitlab.example.com
```

Bitbucket workspaces and projects

Bitbucket Cloud — scopes: `read:repository:bitbucket`, `write:repository:bitbucket`, `read:pullrequest:bitbucket`, `write:pullrequest:bitbucket`. Also needs `--email` and `--username`.

```
atx ct source add --name name --provider bitbucket --org workspace --token api-token --
email email --username username
```

Bitbucket Data Center:

```
atx ct source add --name name --provider bitbucket --org project-key --token http-
access-token --url https://bitbucket.example.com
```

Local repositories

```
atx ct source add --name name --provider local --path parent-directory
```

> **Important**
>
> `--path` must point to a parent directory containing git repos as subdirectories, not to a single repo.

Repository conventions

Repositories can enforce rules that reject a push at the end of a remediation — a required commit message format (GitLab push rules, Bitbucket commit hooks, GitHub rulesets), a restricted branch namespace, or a known commit author. Record those conventions on the source so remediation commits and result branches are written in the shape the repositories require. Each option is available on both `atx ct source add` and `atx ct source update`.

```
# Commit message format ({PLACEHOLDER} is replaced with a description of each change)
atx ct source update --name name --commit-message-template "fix: {PLACEHOLDER}"
```

```
# Branch namespace ({PLACEHOLDER} is replaced with the branch name AWS Transform
 generates)
atx ct source update --name name --branch-name-template "customized/{PLACEHOLDER}"

# Git committer identity (both options are required together)
atx ct source update --name name --git-committer-name "ATX Automation" --git-committer-
email atx-automation@example.com
```

Pass an empty string to remove a template or to clear the committer pair. The options apply to security and transformation remediations, and to local sources whose repositories you push by hand.

Managing sources

```
atx ct source list
atx ct source update --name name --token new-token
atx ct source remove --name name
```

`atx ct source update` accepts any combination of `--token` and the repository convention options above; passing no options is an error. You can update a source from any machine, not just the host that ran `atx ct source add`.

Repository discovery and management

```
atx ct discovery scan --source name
atx ct discovery status --source name
atx ct discovery scan --source name --path new-directory
```

After discovery:

```
atx ct repository list
atx ct repository list --source name
atx ct repository list --labels "team:frontend,priority:high"
atx ct repository update --source name --repo "source::repo" --labels
 "team:frontend,priority:high"
atx ct repository update --source name --labels "migration:wave-1"
```

Running analysis

The `--type` flag specifies the kind of analysis to run:

- `rapid-techdebt-analysis` – Outdated dependencies and easy wins.
- `tech-debt-comprehensive` – Deeper AI-powered analysis covering dependency, security, pattern, performance, maintainability, architecture, code-quality, and infrastructure findings.
- `security` – Static application security testing (SAST) for insecure coding patterns in source code.
- `agentic-readiness` – Readiness of your repositories for AI agents (frameworks, APIs, documentation).
- `modernization-readiness` – Modernization opportunities across your infrastructure, application, data, security, and operations dimensions.
- `custom` – Issues described by one of your own transformation definitions. Requires `--transformation-name`.

```
atx ct analysis run --type type --source name [--repo source::repo] [--wait]
atx ct analysis get --id id --json
atx ct analysis list --json
atx ct analysis list --status pending|running|complete|cancelled|failed --json
atx ct analysis list --type type --json
atx ct analysis cancel --id id
atx ct analysis delete --id id [--cascade-findings]
```

If you omit `--repo`, the command analyzes every repository under `--source`. To scope the run to specific repositories, pass `--repo` (comma-separated) — each a fully-qualified *`source`*::*`repo`*, or a bare name used with `--source`.

To label a run so it's easy to find later, pass `--display-name "`*`name`*`"` on `atx ct analysis run` or `atx ct remote analysis`. The name is shown in `atx ct analysis list` and `atx ct analysis get`.

Custom analysis

A **custom analysis** runs one of your own transformation definitions against your repositories to look for the issues you care about. To run one:

1. Write a transformation definition that describes what to look for and how to report it. See Write a transformation definition for analysis.
2. Publish it with the `analysis` category. See Transformation definition categories.

3. Run it with `--type custom` and `--transformation-name`, locally, remotely, or on a schedule.

```
atx ct analysis run --type custom --transformation-name name --source source --
repo source::repo --wait
```

Configuration with `-g` flag: key-value, JSON, or file path.

If the command fails because the transformation definition was not found, check the name with `atx custom def list`. Transformation definitions are specific to an AWS account and Region. If it fails because of the transformation definition's category, publish it again with `--category analysis`.

Managing findings

```
atx ct findings list --json
atx ct findings list --repo source::repo --source name --severity high|medium|low --
type analysis-type --status open|dismissed|obsolete|change_ready --analysis-id id --
fix-transform transform-name --json
```

Finding statuses

- `open` — Active
- `dismissed` — Manually dismissed (requires reason)
- `obsolete` — System-set when re-analysis no longer produces the finding
- `change_ready` — System-set when a remediation succeeds for the finding's repository: a code change is ready to review or was already applied. You can't set this status by hand, but you can still dismiss a `change_ready` finding if you decide not to take the change.

```
atx ct findings update --id id --status dismissed --reason "reason"
atx ct findings update --id id --status open
atx ct findings batch-update --ids id1,id2 --status dismissed --reason "reason"
atx ct findings get --id id
atx ct findings delete --id id
```

Finding obsolescence

Re-analysis marks resolved findings as obsolete. Cannot be re-opened. Retained for audit.

Creating remediations

Three modes: findings-based, TD override, direct TD. With `--transformation-name`, use one of your own transformation definitions with the `remediation` category, or an AWS managed transformation (see Transformation definition categories).

```
atx ct remediation create --ids id1,id2 --name "name"
atx ct remediation create --ids id1,id2 --transformation-name TD
atx ct remediation create --transformation-name TD --repo source::repo
```

Output by provider: GitHub PR, GitLab MR, Bitbucket PR, Local branch.

> **Note**
>
> Token must have write access for PR/MR creation.

Local execution with `--local` flag.

```
atx ct remediation create --transformation-name TD --repo source::repo -g
 "additionalPlanContext=Upgrade to Node.js 22"
atx ct remediation list
atx ct remediation status --id id
atx ct remediation retry --id id
atx ct remediation cancel --id id
atx ct remediation delete --id id
```

Remote execution

By default, analyses and remediations run on your local machine. For larger portfolios, you can offload work to remote infrastructure. You can run on AWS Transform-managed infrastructure with nothing to provision (analyses only), or on infrastructure you provision and manage in your AWS account—a persistent Amazon EC2 instance or AWS Batch (Fargate) jobs. The `atx ct remote` commands provision, run, monitor, and tear down customer-managed infrastructure. Regardless of

where execution happens, you create all resources in your AWS account and your source code stays under your control.

With the `atx ct remote` analysis and remediation commands, you choose which repositories to process using `--sources` and `--repos` (both comma-separated). `--sources` selects every repository in the named sources. `--repos` selects specific repositories, each written as *`source`*::*`repo`*. If you provide both, the command processes the union of the two sets: every repository in the named sources, plus the named repositories. To scope a run to specific repositories, pass only `--repos`.

> **Note**
>
> Provisioning, updating, and tearing down infrastructure creates and modifies AWS CloudFormation stacks and IAM roles, and requires administrator permissions. Pass `--ack` to acknowledge this and skip the interactive prompt. Running analyses and remediations on already-provisioned infrastructure uses least-privilege executor policies — see Tagging and access control and the compute options in How AWS Transform continuous modernization works for the managed policies involved.

Running on AWS Transform-managed infrastructure (no provisioning)

To run an analysis remotely without provisioning anything, use `--mode aws-managed`. The submission goes to AWS Transform, which runs the analysis on AWS Transform-managed infrastructure. There is no stack to provision, no networking to configure, and no credentials to store in AWS Secrets Manager. The submission is the run. Choose the AWS Region the workload runs in with `--region`.

```
# Run an analysis on AWS Transform-managed infrastructure
atx ct remote analysis --type type --mode aws-managed --sources name [--
repos repo1,repo2] [--region region]

# Poll the submission (there is no remote status command in this mode)
atx ct analysis get --id id --json
```

This mode runs analyses only. It does not support remediation, the `custom` analysis type, or local sources. Because there is no stack, the `--stack-name`, `--tags`, `--existing-instance`, and `--batch-name` options don't apply. A single submission covers up to 100 repositories. To cover larger

scopes, split them across multiple submissions with `--repos`. Unlike Amazon EC2 and Batch runs, you monitor progress with `atx ct analysis get` rather than `atx ct remote status`.

Networking

Remote compute must run in private subnets. Discover existing networking or create a new VPC before you provision:

```
# List VPCs, private subnets, and security groups in the current account and Region
atx ct remote network discover
atx ct remote network discover --vpc vpc-id --json

# Create a new VPC with private subnets, a NAT gateway, and a security group
atx ct remote network create --cidr 10.1.0.0/16 --ack
```

`--cidr` is optional and defaults to `10.1.0.0/16`.

Whether you bring your own network or use `network create`, it must meet these requirements:

- **Private subnets in two Availability Zones** — provisioning rejects public subnets (a subnet whose route table has a default route to an internet gateway). Provide at least two private subnets in different Availability Zones. AWS Batch spreads jobs across all of them. Amazon EC2 uses the first subnet.
- **Outbound internet through a NAT gateway** — jobs run in private subnets. They need outbound access to pull the container image from public Amazon ECR and to reach AWS Transform and other AWS APIs. Each private subnet's route table must send `0.0.0.0/0` to a NAT gateway in a public subnet. You can instead reach AWS APIs over interface endpoints — see AWS Transform custom and interface endpoints (AWS PrivateLink).
- **Security group with outbound access** — the security group needs only outbound (egress) rules. No inbound rules are required. AWS Batch requires you to pass `--securityGroup` at provision time. For Amazon EC2, if you omit `--securityGroup` the stack creates an egress-only security group with no inbound rules, and you reach the instance through SSM. A group that allows all outbound traffic (the default for a new security group) is sufficient.
- **Enough free IP addresses** — each running AWS Batch (Fargate) job uses one private IP address from a subnet. An Amazon EC2 instance uses one private IP address in total, no matter how many workers it runs. Size the subnets for the number of jobs you run in parallel. If a subnet runs low on addresses, jobs can fail with `InsufficientFreeAddressesInSubnet`. If you use `network create`, subnet size scales with the VPC CIDR. The VPC must be `/26` or larger. The

default `10.1.0.0/16` produces `/24` subnets. For the arithmetic, the concurrency limit for each analysis type, and minimum subnet sizes, see Sizing subnets and concurrency.

Sizing subnets and concurrency

Before you provision AWS Batch infrastructure for a large repository fleet, size your private subnets for the number of jobs that run in parallel. A subnet that is too small is the most common cause of failed jobs in a large run.

Each AWS Batch job runs as one Fargate task, which needs one elastic network interface and therefore **one private IPv4 address**. Public IP address assignment is disabled, so a private address is required — a job cannot start without one. AWS Batch does not queue a job it cannot place; it fails the job.

> **Important**
>
> Free addresses do not add up across subnets. AWS Transform treats your available capacity as the **smallest** number of free addresses in any one of the subnets you pass to `--subnets`, not the total across them. Two `/28` subnets give you 11 concurrent jobs, not 22. Adding a subnet never raises capacity, and adding a small subnet lowers it. Give every subnet in the stack enough free addresses on its own.

Checking free addresses before you provision

Provisioning does not check subnet capacity for you. Check it yourself with `--json`, which reports `availableIpCount` for each subnet. The default table output does not include this value. Compare the **smallest** count against the concurrency limit for the analysis type you plan to run.

```
atx ct remote network discover --vpc vpc-id --json
```

Concurrency limits for each analysis type

AWS Transform limits how many remote jobs run at the same time in an AWS account and Region. The limit depends on the analysis type. These limits are shared across all users in the account and across both compute modes, and they cap the number of addresses a run can consume. To request a change, contact AWS Support.

Analysis type	Concurrent jobs
`security`	5
`tech-debt-quick`	128
`tech-debt-comprehensive` , `agentic-readiness` , `modernization-readiness` , and `custom` analysis, and remediation other than security remediation	30, shared across all of these
Security remediation	128

> **Note**
>
> The types in the third row share a single pool of 30 jobs. Running a comprehensive tech-debt analysis and a custom remediation at the same time gives you 30 jobs in total, not 60.

Minimum subnet size

Amazon VPC reserves five addresses in every subnet, so the usable count is five fewer than the subnet size — a `/28` has 11 usable addresses and a `/24` has 251. For details, see Subnet CIDR blocks in the *Amazon VPC User Guide*.

The addresses each subnet needs are the peak concurrent jobs, plus one or more for each interface endpoint in that subnet, plus any other network interfaces that share the subnet. Peak concurrent jobs is the smaller of the number of repositories in the submission and the concurrency limit for the analysis type. Because the limit caps the total, a 1,000-repository fleet needs no more addresses than a 128-repository one.

Analysis type	Peak concurrent jobs	Minimum subnet size
`security`	5	`/27`

Analysis type	Peak concurrent jobs	Minimum subnet size
`tech-debt-comprehensive ,agentic-readiness ,modernization-readiness , custom`	30	/26
`tech-debt-quick`	Up to 128	/24
Every type running at its limit at once	291	/23

The minimums above include headroom for interface endpoints and for the addresses AWS reserves. Size up if other workloads share the subnet. A /25 is not enough for 128 concurrent jobs: it provides 128 addresses but only 123 usable ones.

We recommend dedicating subnets to remote execution. AWS Transform admits jobs down to the last free address and cannot account for other workloads that claim addresses in the same subnet.

Running large fleets

On Amazon EC2 and Batch infrastructure that you provision, a single submission covers up to 1,000 repositories (AWS Transform-managed runs are limited to 100). Submissions above the limit are rejected before any job starts. To cover larger fleets, split them with `--repos`, and list the repositories in a source with `atx ct repository list --source` *`name`* `--json`.

Splitting a fleet does not reduce the addresses you need. Submissions that run at the same time draw from the same subnets, up to the concurrency limit for the analysis type.

There is no option that limits how many repositories are scanned at once. The CLI submits every job in a submission, and AWS Transform releases jobs up to the concurrency limit for the analysis type and holds the rest. To pace a large fleet, size the subnets for that limit or submit in smaller groups.

The CLI bounds the number of simultaneous connections it opens to the analysis service. On a very wide submission you may see `socket usage at capacity` on stderr — that message is informational, not an error. If your shell's open-file limit is at or below the bound, the CLI warns once at startup; raise the limit (for example `ulimit -n 4096`) or lower the bound with the ATX_MAX_SOCKETS environment variable.

Choosing Amazon EC2 for constrained networks

Amazon EC2 uses one private IP address in total, in the first subnet you pass, no matter how many workers the instance runs. If free addresses are scarce, Amazon EC2 avoids the problem entirely.

The tradeoff is concurrency: `--workers` accepts 1–5, so an Amazon EC2 instance runs at most 5 repositories in parallel. For fleets that need more, use AWS Batch. For `security` analysis the account limit is also 5, so AWS Batch offers no additional concurrency for that type.

Size `--workers`, `--instance-type`, and `--volume-size` when you provision. Instance memory must be at least 2 GB for each worker plus 4 GB for the operating system. `atx ct remote update` keeps these values; changing them requires tearing down and provisioning again, which replaces the instance and deletes its volume.

Provisioning infrastructure

Deploy the Amazon EC2 or AWS Batch stack. Omit `--execute` to preview the template or changeset; add `--execute` to apply.

Provisioning creates the compute stack for the mode you choose, plus dispatcher and scheduler stacks:

- **AWS Batch** — a job queue and compute environment, a job definition with the continuous modernization container image, IAM roles for job execution, and a Lambda function for job submission. AWS Batch requires a security group.
- **Amazon EC2** — a persistent Amazon EC2 instance with an IAM instance profile and a security group. If you omit `--securityGroup`, the stack creates a security group with no inbound rules; access is via SSM.
- **Dispatcher** — an `AtxDispatcherStack` stack that queues jobs and dispatches them to your compute stack. It is always required. Provisioning creates it, and `update` reconciles it to the latest template. Pass `--skip-dispatcher` on `update` to skip that reconcile. The dispatcher is not removed. For information about how `update` affects a custom image, see Container image.
- **Scheduler** — an `atx-scheduler` stack (an Amazon EventBridge Scheduler schedule group and invocation role) used by recurring analyses. Pass `--skip-scheduler` to opt out.

By default, Amazon EC2 stacks are named `atx-runner` and AWS Batch stacks are named `AtxInfrastructureStack`. To run more than one stack of the same mode, give each a distinct

name when you provision: use `--stack-name` (Amazon EC2 only; the name must start with `atx-runner`) or `--suffix` to append a suffix to all resource names. For `update`, `detect`, and remote analyses or remediations, pass `--stack-name` to target a custom-named stack. If you omit it, they use the default name (`atx-runner` for Amazon EC2, `AtxInfrastructureStack` for AWS Batch). `teardown` always requires `--stack-name`. On Amazon EC2, `--workers` sets the number of parallel worker containers (1-5, default 5) and sizes the instance.

```
# Preview, then deploy an EC2 stack
atx ct remote provision --mode ec2 --vpc vpc-id --subnets subnet-a,subnet-b
atx ct remote provision --mode ec2 --vpc vpc-id --subnets subnet-a,subnet-b --execute
 --ack

# Deploy a Batch stack
atx ct remote provision --mode batch --vpc vpc-id --subnets subnet-a,subnet-b --
securityGroup sg-id --execute --ack

# Update to the latest template (--stack-name optional; defaults to the standard stack
 name)
atx ct remote update --mode ec2|batch [--stack-name stack-name] --execute --ack

# Tear down (--stack-name required)
atx ct remote teardown --mode ec2|batch --stack-name stack-name --execute --ack
```

Container image

When you run remote analyses and remediations, they execute inside a container image. By default, when you provision a remote environment, it uses the public AWS Transform image, `public.ecr.aws/d9h8z6l7/aws-transform:latest`. AWS Batch sets it as the job definition image. Amazon EC2 uses it as the runner image.

To run a different image—for example, a private Amazon ECR image that bundles additional languages or tools—pass `--image-uri` when you provision:

```
# Batch: provision with a custom image
atx ct remote provision --mode batch --vpc vpc-id --subnets subnet-a,subnet-b --
securityGroup sg-id --image-uri account-id.dkr.ecr.region.amazonaws.com/repo:tag --
execute --ack

# EC2: provision with a custom image
atx ct remote provision --mode ec2 --vpc vpc-id --subnets subnet-a,subnet-b --image-
uri account-id.dkr.ecr.region.amazonaws.com/repo:tag --execute --ack
```

> **Note**
>
> A later `atx ct remote update` (see Provisioning infrastructure) resets the image to the template default. To keep a custom image, re-run `provision` with `--image-uri` after you update.

Storing source credentials

Remote containers clone your repositories using tokens stored in AWS Secrets Manager. Register a token for each SCM source before running remote analysis or remediation:

```
atx ct remote credentials --source name --token token
atx ct remote credentials --source name --remove
```

Running remotely

Remote analysis runs one container for each repository; remote remediation runs one container for each finding. Use `--sources`, `--repos`, and `--labels` to control fan-out, and `--stack-name` or `--tags` to select which provisioned stack to use.

```
# Run analysis across a source on Batch
atx ct remote analysis --type type --mode batch --sources name [--repos repo1,repo2]
 [--labels "team:frontend"]

# Run remediation for specific findings on EC2
atx ct remote remediation --mode ec2 --ids id1,id2
atx ct remote remediation --mode ec2 --sources name --min-severity high
```

Monitoring and managing runs

```
# Check whether infrastructure is deployed
atx ct remote detect --mode ec2|batch

# Track a submission (Batch by batch ID, EC2 by group ID)
atx ct remote status --batch batch-id --stack-name name
atx ct remote status --group ec2-group-id --wait

# Resume a partially-failed Batch run (re-submits only incomplete repos).
```

```
# On resume, --batch-name takes the existing batch ID reported by "remote status --
batch".
atx ct remote analysis --type type --mode batch --sources name --resume-incomplete --
batch-name batch-id

# Cancel a running submission
atx ct remote cancel --mode batch --batch batch-id --stack-name name
atx ct remote cancel --mode ec2 --group ec2-group-id
```

Scheduling recurring analysis

Use `atx ct schedule` to run analyses automatically on a recurring cadence. You can schedule analyses but not remediations. Job options mirror `atx ct remote analysis`. Schedules run remotely, in one of two ways:

- **AWS Transform-managed** (`--mode aws-managed`)—a server-side schedule that fires analyses on AWS Transform-managed infrastructure. There is no Amazon EventBridge schedule and nothing to provision. It requires an execution role (`--execution-role`) that AWS Transform assumes at each run (see Execution role for AWS Transform-managed schedules).
- **Customer-managed** (`--mode ec2|batch`)—an Amazon EventBridge Scheduler schedule in your account dispatches each run to a persistent Amazon EC2 instance or AWS Batch stack that you provision first (see Remote execution).

The `--recurrence` value accepts `daily`, `weekly:`*DAY* (for example, `weekly:MONDAY`), or `monthly:`*N* where *N* is a day from 1 to 28. Schedules on AWS Transform-managed infrastructure run in UTC.

```
# AWS Transform-managed schedule (no infrastructure; requires an execution role)
atx ct schedule create --name name --mode aws-managed --execution-role role-arn --
recurrence daily --type type --sources name [--repos repo1,repo2]

# Customer-managed schedule (EventBridge Scheduler dispatching to your EC2 or Batch
 stack)
atx ct schedule create --name name --mode ec2|batch --recurrence weekly:MONDAY --
type type --sources name [--repos repo1,repo2]

# Manage schedules of either type by their schedule ID (from schedule list)
atx ct schedule list
atx ct schedule get schedule-id
atx ct schedule disable schedule-id
```

```
atx ct schedule enable schedule-id
atx ct schedule delete schedule-id
```

To view the analyses a schedule has run, use `atx ct analysis list --schedule-id schedule-id`, which returns the schedule's fired runs, newest first.

To remove the scheduler role and schedule group used by customer-managed schedules, run `atx ct schedule teardown --execute`.

Execution role for AWS Transform-managed schedules

A schedule created with `--mode aws-managed` requires an `--execution-role` ARN that AWS Transform assumes each time the schedule runs. Configure the role as follows:

- The identity creating the schedule must have `iam:PassRole` permission on the execution role.
- The role's trust policy must allow the `transform-custom.amazonaws.com` service principal to assume it.
- At a minimum, the role must have the AWS managed policy `AWSTransformCustomFullAccess` attached, plus `secretsmanager:GetSecretValue` and `secretsmanager:DescribeSecret` permissions on secrets under the `atx/*` prefix so that scheduled runs can retrieve the source clone credentials.

The following inline policy grants the AWS Secrets Manager access that scheduled runs need to retrieve source clone credentials. Attach it to the execution role alongside the `AWSTransformCustomFullAccess` managed policy, replacing *region* and *account-id* with the AWS Region and account the schedule runs in.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AtxSourceCredentials",
      "Effect": "Allow",
      "Action": [
        "secretsmanager:GetSecretValue",
        "secretsmanager:DescribeSecret"
      ],
      "Resource": "arn:aws:secretsmanager:region:account-id:secret:atx/*"
    }
```

```
  ]
}
```

Tagging and access control

You can apply tags (comma-separated `key=value` pairs) to sources, analyses, and remediations with the `--tags` option. Tags are also supported on remote infrastructure, stored credentials, and networking resources. With tags, you can organize resources. Combined with IAM tag conditions, tags implement attribute-based access control (ABAC) so that teams access only the resources that carry their tags.

```
atx ct source add --name name --provider github --org org --token pat --
tags team=platform,env=prod
atx ct analysis run --type type --source name --tags team=platform
atx ct remediation create --ids id1,id2 --tags team=platform
```

By default, resources are tagged with the tags you define in `~/.aws/atx/settings.json`. Add the tags you want applied to every resource under `applyTags`, and they become your default tags.

```
{
  "applyTags": [
    { "team": "alpha" }
  ]
}
```

> **Note**
>
> Tags passed with `--tags` are merged over any configured default tags, and `--tags` wins for any key set in both places.

AWS Transform web application

Use the AWS Transform web application to create and run analyses, review findings, create remediations, and track generated pull requests across your code sources.

Before you use the web application, your organization must enable your user identity to access AWS Transform. For more information about setting up AWS Transform, see Setting up AWS Transform.

Sign in

To access the AWS Transform web application, complete the following steps.

1. Open `https://aws.amazon.com/transform/` and sign in with AWS IAM Identity Center credentials.
2. If continuous modernization does not appear, sign in with IAM credentials instead:
   a. In the AWS Management Console, open AWS Transform and choose **Settings**.
   b. Turn on **Access AWS Transform with IAM credentials**.
   c. Copy the **Web application URL (with IAM)** and paste it into the same browser window where the console is open.
3. Open the left navigation menu and choose **continuous modernization**.

Infrastructure modes

When you create an analysis, choose one of the following infrastructure modes:

- **AWS managed** – Run on infrastructure managed by AWS Transform. You don't need to provision any infrastructure.
- **Customer owned** – Run on a deployed stack in your own AWS account. Use this mode when you need control over compute, networking, or security configuration.

> **Note**
>
> To run security analysis, use customer-owned infrastructure. Security analysis runs on the Security Agent deployed in your account.

To use customer-owned infrastructure, open the **Settings** tab. Use the AWS CloudFormation quick-create links to deploy the following stacks in order:

1. `AtxDispatcherStack` – Message dispatcher (always required).
2. Compute stack – `AtxInfrastructureStack` (AWS Batch) or `atx-runner` (Amazon EC2).
3. `atx-scheduler` – Required for recurring scheduled analyses.
4. `AtxSecurityAgentStack-<region>` – Required only for security analysis.

For CLI-based provisioning and networking configuration, see Remote execution.

Getting started workflow

1. **Connect sources** – Open the **Sources** tab and add repositories from GitHub, GitLab, or Bitbucket.
2. **Run or schedule an analysis** – Open the **Analyses** tab, select repositories, choose an analysis type, select an infrastructure mode, and choose **Run**. To run on a recurring cadence (daily, weekly, or monthly), choose **Schedule** instead.
3. **Review findings** – Open the **Findings** tab to view results by severity.
4. **Create a remediation** – Select findings and choose **Create remediation**.
5. **Review pull requests** – Open the **Remediations** tab to view generated PR links per repository.

Chat with AWS Transform directly from the web application to ask questions about your analyses, findings, or remediations.

Developer tools

You can access continuous modernization from agentic IDEs using the AWS Transform Power and agent plugin. The agent orchestrates setup, onboarding, and ongoing use of continuous modernization through conversational interaction.

- **Kiro IDE** — Install the AWS Transform Kiro Power. Open Kiro Chat and ask the agent to help with continuous modernization workflows.
- **Claude Code, Codex, Cursor** — Install the awslabs/agent-plugins (`plugins/aws-transform` directory). The agent plugin provides the same capabilities as the Kiro Power.

Agent skills

The Kiro Power and agent plugin include skills that orchestrate continuous modernization workflows through conversational interaction:

Skill	Description
Guide	Interactive onboarding that walks you through the full workflow step by step

Skill	Description
Source	Add, list, and remove source connections
Discovery	Scan sources to discover repositories
Analysis	Run and manage analyses across all types
Findings	Query, filter, and manage analysis findings
Remediation	Create and manage remediations to fix findings
EC2 Execution	Provision and manage an Amazon EC2 instance for remote analysis
Batch Execution	Submit analysis jobs to AWS Batch with Fargate
Schedule	Set up recurring analysis on a cron schedule using Amazon EventBridge Scheduler
Reporting	Generate self-contained HTML reports with charts showing findings, severity distribution, and remediation progress

CLI reference

The following table summarizes all `atx ct` subcommands.

Subcommand	Description
`atx ct status`	View system status including sources, repositories, analyses, findings, and remediati ons
`atx ct source add`	Add a source (GitHub, GitLab, Bitbucket, or local)

Subcommand	Description
`atx ct source list`	List configured sources
`atx ct source get`	Show details for a single source
`atx ct source update`	Update a source's token, commit message or branch name templates, or git committer identity
`atx ct source remove`	Remove a source
`atx ct discovery scan`	Discover repositories from a source
`atx ct discovery status`	Check discovery scan status
`atx ct repository list`	List repositories with optional filters
`atx ct repository get`	Get a single repository
`atx ct repository update`	Update repository labels
`atx ct repository delete`	Delete a repository
`atx ct analysis run`	Run an analysis on repositories
`atx ct analysis get`	Get analysis details
`atx ct analysis list`	List analyses with optional status and type filters
`atx ct analysis cancel`	Cancel a running analysis
`atx ct analysis delete`	Delete an analysis
`atx ct findings list`	List findings with filters for repo, source, severity, type, status, and analysis
`atx ct findings count`	Count findings aggregated by severity, repo, or analysis type

Subcommand	Description
`atx ct findings get`	Get a single finding
`atx ct findings update`	Update finding status (open or dismissed)
`atx ct findings batch-update`	Batch update multiple findings
`atx ct findings delete`	Delete a dismissed or obsolete finding
`atx ct remediation create`	Create a remediation from findings or a transformation definition
`atx ct remediation list`	List all remediations
`atx ct remediation status`	Check remediation status and view PR/MR links
`atx ct remediation retry`	Retry a failed remediation
`atx ct remediation cancel`	Cancel a running remediation
`atx ct remediation delete`	Delete a remediation
`atx ct remote provision`	Deploy remote execution infrastructure (Amazon EC2 or Batch AWS CloudFormation stack)
`atx ct remote analysis`	Run analysis remotely on Amazon EC2 or Batch
`atx ct remote remediation`	Run remediation remotely on Amazon EC2 or Batch
`atx ct remote status`	Show status of a remote submission
`atx ct remote detect`	Check whether remote infrastructure is deployed
`atx ct remote update`	Update remote execution infrastructure to the latest template

Subcommand	Description
`atx ct remote credentials`	Manage source tokens in Secrets Manager for remote execution
`atx ct remote cancel`	Cancel a running remote run
`atx ct remote teardown`	Destroy remote execution infrastructure
`atx ct remote network`	Discover or create VPC, subnets, and security groups for remote provisioning
`atx ct schedule create`	Create a recurring analysis schedule (Amazon EventBridge Scheduler)
`atx ct schedule list`	List schedules
`atx ct schedule get`	Show a single schedule
`atx ct schedule enable`	Enable a schedule
`atx ct schedule disable`	Disable a schedule
`atx ct schedule delete`	Delete a schedule
`atx ct schedule teardown`	Delete the scheduler role and schedule group
`atx ct setup security-agent`	Provision or manage security agent infrastru cture
`atx ct mcp`	Start the MCP server exposing continuous modernization tools
`atx ct schema`	Print a machine-readable JSON manifest of the full `atx ct` command surface — every command, option, and argument — so agents and automation can discover the CLI without parsing help text

Troubleshooting

Checking system health

You no longer need to start a local server; the `atx ct server` command is deprecated. Verify the system is reachable with `atx ct status --health`.

Discovery scan returns zero repositories

For local sources, verify that `--path` points to a parent directory that contains repositories as subdirectories (for example, `/home/user/repos`), not to a repository directly (for example, `/home/user/repos/my-app`). The scanner looks for child directories that contain a `.git` folder.

SETUP_REQUIRED error

The source exists in your account but credentials are not configured on this machine. Run `atx ct source add --name` *`name`* to configure credentials locally.

AUTH_REQUIRED error

No valid token found for the source. Verify your token is correct and has the required scopes. For GitHub, ensure the token has `repo` scope. For GitLab, ensure `api` scope.

INVALID_INPUT error

Verify you are using the correct analysis type name. Valid values are: `rapid-techdebt-analysis`, `tech-debt-comprehensive`, `security`, `agentic-readiness`, `modernization-readiness`, `custom`.

Permission errors during remediation

Remediation creates branches and pull/merge requests, which requires write access to the repository. Update your token with write permissions. For GitHub, ensure full `repo` scope. For GitLab, ensure `api` scope. For Bitbucket, ensure `write:repository:bitbucket` and `write:pullrequest:bitbucket` scopes.

Security agent setup fails

Verify your IAM user or role has the required AWS CloudFormation, IAM, and Amazon S3 permissions listed in the Security agent setup section of How AWS Transform continuous modernization works. Check the status with `atx ct setup security-agent --status`.

Security analysis or security agent setup unavailable in a Region

The `security` analysis type is not available in Canada (Central) (`ca-central-1`), Europe (London) (`eu-west-2`), or Asia Pacific (Seoul) (`ap-northeast-2`). Run `security` analyses and `atx ct setup security-agent` from a supported Region. The other analysis types are unaffected. See the Security agent setup section of How AWS Transform continuous modernization works.

Remote provisioning fails with a permissions error

Provisioning, updating, and tearing down remote infrastructure (`atx ct remote provision`, `update`, `teardown`) creates and modifies AWS CloudFormation stacks and IAM roles, and requires administrator permissions. To run analyses and remediations on already-provisioned infrastructure with least privilege, attach `AWSTransformInfrastructureExecutorAccessEC2` or `AWSTransformInfrastructureExecutorAccessBatch`. For details, see the compute options in How AWS Transform continuous modernization works.

Remote network discovery returns no subnets

Remote compute must run in private subnets, and `atx ct remote network discover` excludes public subnets. If no private subnets exist, create networking with `atx ct remote network create`.

AWS Batch jobs fail with InsufficientFreeAddressesInSubnet

Each AWS Batch job needs one free private IPv4 address, and AWS Batch fails a job it cannot place rather than queueing it. Run `atx ct remote network discover --vpc` *`vpc-id`* `--json` and read `availableIpCount` for every subnet in the stack, then compare the smallest count against the concurrency limit for the analysis type you are running. Free addresses do not add up across subnets — capacity is the smallest count, not the total. To resolve, use larger subnets, remove the smallest subnet from the stack, submit fewer repositories at a time, or run with `--mode ec2`, which uses one address in total. For the arithmetic and minimum subnet sizes, see Sizing subnets and concurrency.

Repositories stay pending for a long time with no error

The run is being paced. AWS Transform limits how many remote jobs run at the same time in an AWS account and Region — as few as 5 for `security` analysis — and holds the rest until capacity frees up. A run can also wait for free IP addresses in your subnets. Both appear as `pending` in `atx ct remote status`, which does not distinguish waiting from running. Confirm the limit for your analysis type in Sizing subnets and concurrency and expect the run to

progress at that rate. Do not submit the run again — resubmitting adds jobs that wait behind the same limit. If free addresses are plentiful but jobs still wait, the run is at the concurrency limit for its type and larger subnets will not help.

Remote status reports no jobs for a submission that was accepted

Expected while every job in the submission is still waiting for capacity. AWS Transform records job details when the first job starts, so `atx ct remote status --batch` can report nothing until then. Wait and run the command again.

Repositories fail with no reason after several hours

A job waits for capacity for up to about 33 hours before AWS Transform abandons it. Jobs are also abandoned if AWS Transform cannot evaluate subnet capacity, which happens on stacks provisioned before capacity checking was added. Update the stack with `atx ct remote update --mode batch --execute --ack`, then submit the affected repositories again in smaller groups. If the run was large, size the subnets for the concurrency limit of the analysis type first — see Sizing subnets and concurrency.

Remote run cannot clone repositories

Remote containers use tokens stored in AWS Secrets Manager. Register a token for each source with `atx ct remote credentials --source` *`name`* `--token` *`token`* before running remote analysis or remediation.

Schedule fails to run

Schedules run on remote infrastructure. Provision an Amazon EC2 or Batch stack with `atx ct remote provision` before creating a schedule, and confirm the schedule is enabled with `atx ct schedule get` *`name`*.

Continuous modernization not visible in the web application

If continuous modernization does not appear after you sign in to the AWS Transform web application, sign in with the IAM credentials of the AWS account where AWS Transform is enabled instead of AWS IAM Identity Center. For steps, see AWS Transform web application.

Migration assessments

AWS Transform assessments help you evaluate the cost, feasibility, and business value of migrating on-premises infrastructure to AWS. Using agentic AI, AWS Transform generates a data-driven total cost of ownership (TCO) business case in minutes from a server inventory, finding the best-fit AWS services for your workloads and producing pricing options, licensing analysis, and actionable next steps. From there, you refine the business case through natural language chat—adding missing inventory, adjusting on-premises costs, modeling alternative AWS services, changing assumptions, and exploring what-if scenarios—without re-running the full assessment.

If you are new to migration assessments, see the section called “Quick start” for guidance on collecting inventory data, running an assessment, and which prompts to use.

Why perform a migration assessment

A migration assessment turns a server inventory into numbers you can act on. It shows what your estate costs to run today, what it would cost on AWS, and what to do next. Migrations can stall when these questions go unanswered. An assessment is usually the first step in a migration.

A migration assessment helps you do the following:

- **Understand what you have on premises** – you get one inventory of your servers, operating systems, databases, and storage. It often includes workloads that no single team tracks. The assessment also flags gaps in your data, which shows you where your records are thin.
- **Define migration scope** – decide which servers and applications to migrate, which to retire, and which to defer, and see the cost of each choice.
- **Uncover what your estate really costs** – hardware, licenses, data center space, support, and network. Licenses cover both your hypervisor and operating systems and your databases. Teams often spread these costs across separate budgets.
- **Find savings** – you can right-size over-provisioned servers, move to AWS Graviton, cut license costs, and compare On-Demand pricing against Reserved Instance terms.
- **Align stakeholders on next steps** – you get one total cost of ownership (TCO) business case, plus a business value analysis and reports. Your finance, infrastructure, and application teams can all work from them. The results also show whether you qualify for AWS programs, such as the AWS Migration Acceleration Program (MAP).

You refine an assessment through chat. This means you can start with the data you have today, get a directional business case, and improve it as better data arrives.

Prerequisites

Before you create a migration assessment, confirm that you have the following:

- An AWS Transform workspace
- Inventory data in a supported format, or you can use rough estimation through chat
- Appropriate permissions to create and run assessments

How migration assessments work

Migration assessments follow a structured workflow that guides you from initial data upload through final deliverable generation.

To complete a migration assessment

1. Create a migration assessment job in your AWS Transform workspace.
2. Upload your on-premises inventory data.
3. Review the discovery results that AWS Transform generates from your data.
4. Manage the inventory scope by excluding servers or adjusting groupings.
5. Configure your assessment scenario with pricing and infrastructure assumptions.
6. Run the assessment.
7. Review and refine the results through chat.

After you review results, you can create additional scenarios, compare scenarios, and generate deliverables such as presentations and reports.

Uploading inventory data

AWS Transform can work with the data you already have, rather than requiring a new collection. It recognizes the common discovery and inventory formats, and falls back to a flexible parser for files that it does not recognize. If you have a file that contains inventory data, upload it and review what AWS Transform finds, because a flexible parse is best-effort.

The following table gives examples of commonly used sources. It is not an exhaustive list.

Data source	Description
AWS Transform Discovery Tool export	Automated server inventory discovered by the AWS Transform discovery tool
RVTools	Exports from VMware environments in ZIP/CSV or Excel format. Both full exports and vInfo-only exports are supported.
CMDB data	Configuration management database exports
Migration Evaluator	Quick Insights file from the AWS Migration Evaluator console
MPA format	AWS Migration Portfolio Assessment (MPA) import file
NetApp Data Infrastructure Insights (DII)	Detailed storage discovery data from NetApp Data Infrastru cture Insights, covering capacity, IOPS, and throughput for both host-attached and array-level storage. Use it for storage assessments, where it gives AWS Transform more to work with than the storage fields in a server inventory.
Partner discovery tools	Exports from AWS partner and third-party discovery tools, including ModelizeIT, Cloudamize, Matilda Cloud, and Device42
AWS Transform data template	Microsoft Excel file created from the AWS Transform Assessment Data template

AWS Transform identifies the format during ingestion. The ingestion process validates your data and accepts partial data when some fields are missing. It also supports incremental uploads, so you can upload additional files at any time to supplement your inventory. You can add or correct details through chat.

Reviewing discovery results

After AWS Transform processes your inventory data, it generates a discovery summary. The summary includes the following information:

- Server count by operating system

- Physical and virtual server breakdown
- SQL Server detection
- Apache Kafka cluster detection
- Storage summaries
- Data quality warnings

Review the discovery results to confirm that AWS Transform correctly identified your infrastructure before you proceed with the assessment.

Managing inventory scope

After you review the discovery results, you can refine the scope of your assessment. Use the following actions to manage your inventory scope:

- Query inventory for specific servers or workloads
- Exclude servers from the assessment scope
- Review server groupings and application dependencies
- Identify SQL Server and Apache Kafka workloads for specialized assessment

Configuring assessment scenarios

Before you run an assessment, configure the assumptions that AWS Transform uses to generate recommendations. You can modify these assumptions at any time and run new scenarios with different configurations.

The following table lists the available assumption categories.

Assumption	Description
Pricing model	On-Demand, Reserved Instances, or Savings Plans; Database Savings Plans for Amazon RDS for SQL Server
Target AWS Region	The AWS Region where you plan to host migrated workloads
Utilization and right-sizing	The performance tier and utilization assumptions used to right-size Amazon EC2 instances

Assumption	Description
Processor architecture	Whether recommendations use x86, AWS Graviton, or either architecture
Tenancy	Shared, dedicated, or mixed tenancy for Amazon EC2 instances
Instance type exclusions	Amazon EC2 instance families or types to exclude from recommendations
Amazon EBS configuration	Volume type and performance settings for storage
SQL Server licensing	Bring Your Own Media (BYOM) or License Included (LI) for Amazon RDS for SQL Server; License Included (LI) or Bring Your Own License (BYOL) for SQL Server on EC2

Assessment capabilities

AWS Transform assessments cover the workload categories that matter most in a typical enterprise migration. The following topics describe each assessment capability in detail.

- the section called “Compute”—best-fit, lowest-cost Amazon EC2 instance recommendations, including AWS Graviton, Dedicated Hosts, and multiple pricing models.
- the section called “Databases”—Microsoft SQL Server on Amazon RDS for SQL Server and on Amazon EC2, with licensing analysis and edition recommendations.
- the section called “Storage”—block, object, and file storage across Amazon EBS, Amazon S3, and Amazon FSx for NetApp ONTAP.
- the section called “Analytics”—self-managed Apache Kafka clusters assessed for Amazon MSK Express.
- the section called “Business value”—quantified business value across the AWS Cloud Value Framework pillars.
- the section called “Additional cost components”—on-premises pricing, network, support, and end user computing cost components.

Using chat-based assessments

You can interact with AWS Transform through chat to create and refine assessments. Chat supports three modes of interaction.

You can iteratively refine your assessment by continuing the conversation. AWS Transform maintains context across messages and updates the assessment results based on your input.

Rough estimation with limited data

You can get a rough cost estimate by describing your environment in chat without uploading detailed inventory files. Rough estimation is a standalone feature and cannot be combined with uploaded inventory data or data enrichment through chat.

You can use prompts like these:

- "I have 200 Windows servers and 150 Linux servers, estimate my AWS costs"
- "Give me a rough estimate for migrating 500 VMs to AWS"
- "How much would running 2000 large Linux servers with roughly 500 TB of SAN storage cost? About 300 run MySQL and about 100 run Oracle."

Adding inventory through chat

You can provide inventory details directly through chat to supplement or replace uploaded files. When a server is missing from an export, you can add it in chat rather than regenerating and re-uploading the file.

Use prompts like the following:

- "There is a server missing from my inventory. It is called prod-server5, it runs SQL Server Standard Edition on Windows, and has 128 CPU cores and 256 GB of RAM."
- "Add 20 web servers running Linux with 4 CPUs and 16 GB RAM"

Modifying costs and adding services

You can adjust on-premises costs and add AWS services to your assessment through chat.

Example prompts for on-premises adjustments:

- "Increase on-premises costs by 15% to account for upcoming hardware refresh"
- "Add $100,000 annual maintenance costs to the on-premises baseline"

You can also add rough cost estimates for AWS services that are not fully supported by AWS Transform assessments. This provides a more complete analysis, but these estimates are less accurate than the automated recommendations.

- "Add AWS Backup costs for all migrated servers"
- "Include Amazon CloudWatch monitoring costs in the estimate"
- "Migrate servers svr1, svr2, and svr3 to Amazon Connect. Remove them from EC2 and add $10,000 a month in Amazon Connect costs."

Comparing scenarios

You can create multiple assessment scenarios with different assumptions and compare them to identify the optimal migration strategy.

Creating scenarios

Try prompts such as:

- "Create a BYOM scenario for migrating SQL Server databases to RDS for SQL Server"
- "Create a scenario with Database Savings Plans for RDS for SQL Server"
- "Create a scenario with AWS Graviton instances where supported"
- "Create a scenario with all workloads in us-west-2"

Running comparisons

Here are some example prompts:

- "Show me a cost comparison across all scenarios"
- "Which scenario has the lowest total cost of ownership?"

What-if analysis

Model the impact of specific changes without creating a full new scenario.

Example prompts:

- "What if I move my SQL Server workloads to RDS for SQL Server instead of EC2?"
- "What is the cost difference between BYOM and License Included for RDS for SQL Server?"
- "What if I exclude the 50 smallest servers from the migration?"
- "What if I only use storage optimized instances?"
- "What is the impact of moving from gp3 to io2 volumes?"

Generating deliverables

After you complete your assessment, you can generate deliverables in multiple formats. The following table describes the available output formats.

Format	Description	Customization
PPTX (PowerPoint)	Executive presentation with summary findings and recommendations	Fixed structure
XLSX (Excel)	Detailed data export with server-level recommendations and cost breakdowns	Fixed structure
PDF	Report document with sections for compute, storage, licensing, and cost comparisons	Customizable through chat

Quick start guide for migration assessments

This guide explains how to get the inventory data that an assessment needs, how to run an assessment, how to review the results, and which prompts to use. Start here if you want to produce a first business case quickly. Before you start, confirm that you have the items in the section called “Prerequisites”.

This guide covers the following topics, which take you from an inventory file to a finished business case:

- Get data for a migration assessment
- Run an assessment

- Review and refine the results
- Use the example prompts

Getting data for a migration assessment

Every assessment starts with a server inventory, and AWS Transform accepts many formats. Three routes cover most cases: export your inventory from your hypervisor, run the AWS Transform discovery tool, or use data you already have. For more information about supported formats, see the section called “Uploading inventory data”.

Exporting your inventory from vCenter

If your workloads run on VMware, an export from vCenter is the fastest way to get data. Both of the following tools run in minutes, and neither needs agents on your servers.

- **Export for vCenter** – an open-source Python script from AWS. It collects only the vCenter data that AWS Transform needs, and writes files that match the names and column headers of RVTools CSV files. You need Python 3.10 or later and read-only vCenter credentials. The script creates a `vcexport.zip` file that you upload to your assessment. Because the output matches the RVTools format, AWS Transform reads it as an RVTools export. For instructions on installing and running the script, see export-for-vcenter on the GitHub website.
- **RVTools** – a widely used third-party Windows tool from Dell that reports on VMware vSphere environments. Many teams already run it, so your infrastructure team may have a recent export. It exports to an Excel file or a ZIP file of CSV files, both of which AWS Transform accepts. Download RVTools only from RVTools on the Dell website.

Both tools report how your virtual machines are configured. Export for vCenter also reads the performance statistics that vCenter already collects, and RVTools does not. For each powered-on VM, it adds maximum and average CPU and memory use, and the largest virtual disk read and write request sizes.

That data matters, because AWS Transform sizes instances from the peak utilization in your inventory when it is there, and falls back to default assumptions when it is not. To get a first business case, an RVTools export is still a good option, and you can add usage data later.

Data from a hypervisor has limits. It covers only the virtual machines that vCenter manages. Add physical servers, other hypervisors, and cloud workloads separately. It also cannot see inside the

guest, so you get no database engines or editions, and no connections between servers. Without SQL Server editions and versions, AWS Transform cannot analyze your SQL Server licensing.

Export for vCenter has two limits of its own. It skips powered-off VMs, VMs with no IP address, and VMs in a `notRunning` guest state. Its usage data also defaults to a 60-minute window, which can miss month-end or seasonal peaks.

Collecting richer data with the discovery tool

The AWS Transform discovery tool is a virtual appliance that you deploy in your own environment to discover your inventory. It supports VMware vCenter and Microsoft Hyper-V, and you can import more servers from a comma-separated values (CSV) file. The tool collects data without agents. It reads from your hypervisors, and it connects to your operating systems over Secure Shell (SSH) and Windows Remote Management (WinRM). You supply the credentials. For instructions on setting up the appliance, see Discovery tool.

The tool takes longer to set up than an export, because you deploy the appliance, configure your sources, and supply credentials. In return, it collects data that an export cannot give you:

- **Usage data over time** – operating system and storage metrics, collected as the tool runs. AWS Transform can then size servers against real demand instead of assumptions. This has the largest effect on the accuracy of your estimate.
- **Database discovery** – the SQL Server and Oracle databases on your servers. SQL Server editions and versions are what make license and edition analysis possible. Your inventory records Oracle databases, and AWS Transform assesses their host servers as general compute.
- **Network connections** – connections between your servers, and the process behind each one. AWS Transform assessments do not map application dependencies, so use this data for your own wave planning.
- **Wider coverage** – one inventory across vCenter, Hyper-V, and imported servers, so you avoid merging separate exports.

Many teams run both. Start with an export to get a directional business case and agree on scope. Then run the assessment again with the discovery tool data to firm up the numbers.

Using data you already have

You can often use data you already have. AWS Transform recognizes the common discovery and inventory formats, and falls back to a flexible parser for files that it does not recognize. A

spreadsheet, a configuration management database (CMDB) export, a storage report, or an export from a third-party discovery tool are all reasonable starting points, as are the tools in Migration discovery in AWS Marketplace. AWS Transform detects the format when you upload, and accepts partial data when fields are missing. Check what AWS Transform reports back for an unrecognized file, because a flexible parse is best-effort. For a spreadsheet that you build yourself, the AWS Transform Assessment Data template gives you the columns that AWS Transform expects. For more information about commonly used sources, see the section called “Uploading inventory data”.

Running an assessment

After you upload your inventory, an assessment takes three steps. Define the scope, define the assumptions, and run it. You can work through the steps one at a time, which lets you check each one before you move on. You can also combine all three in a single prompt when you already know what you want.

AWS Transform uses default values for anything you leave out, so you get a result either way. Being explicit gives you a better result. The defaults cannot know which servers you plan to migrate, which AWS Region you want, or what commitments your finance team will make. A business case built on unstated assumptions is also hard to defend when stakeholders question the numbers. When you state your inputs, you know which inputs produced which result.

The assumptions that you set in the second step shape the result more than any other input, and they are where your organization's constraints belong. The pricing model and term is often the largest cost variable in an assessment.

To run an assessment

1. **Define the scope**

   Decide which servers and workloads belong in the assessment. An export usually holds more than you plan to migrate, and servers that you plan to retire inflate both your on-premises baseline and your AWS estimate. For more information about scope, see the section called “Managing inventory scope”.

   Take the following scoping actions:

   - Exclude the servers that you plan to decommission rather than migrate, and the servers you want to keep out of this business case.

- Move your non-production servers into a separate scenario, so that the main business case covers production.
- Add servers that the export missed, such as physical servers that vCenter does not manage.
- Check that AWS Transform found your SQL Server workloads, because it assesses them differently from other servers.

AWS Transform also excludes some resources on its own, including powered-off and suspended servers, virtual machine templates, and the appliances that discovery tools and collectors run on. Check the in-scope count against what you expect, because these exclusions change your server count.

2. **Define the assumptions**

   Set the pricing and infrastructure assumptions that AWS Transform uses.

   The following are basic examples. You can configure many more assumptions than these, including processor architecture, instance type exclusions, and your on-premises costs. For more information about assumptions, see the section called “Configuring assessment scenarios”.

   - **Target AWS Region** – for example, the US East (N. Virginia) AWS Region, or the AWS Region closest to your users. Prices vary by AWS Region.
   - **Pricing model and term** – for example, three-year Reserved Instances if your finance team will commit to a term, or On-Demand if it will not.
   - **Right-sizing** – the CPU and memory utilization that AWS Transform assumes for servers with no performance data.
   - **Tenancy and licensing** – for example, mixed tenancy, so that Bring Your Own License Windows workloads go on Dedicated Hosts. Choose Bring Your Own Media for SQL Server on Amazon Relational Database Service (Amazon RDS) if you own the licenses.

   Lower assumed utilization leaves more headroom and recommends larger instances. Higher assumed utilization trades headroom for cost.

3. **Run the assessment**

   Run the assessment. A run can take several minutes, and the time depends on the size of your inventory.

After the run finishes, check the results before you share them, as described in the following section.

Reviewing and refining the results

After the assessment runs, review the output. AWS Transform answers in chat, so you can ask about the results, drill into a group of servers, and question a recommendation without generating a file. When you need something to share, AWS Transform produces the following deliverables:

Microsoft PowerPoint presentation (PPTX)

An executive presentation with summary findings and recommendations.

Microsoft Excel workbook (XLSX)

A detailed export with recommendations and cost breakdowns for each server.

Portable Document Format (PDF)

A report that covers compute, storage, licensing, and cost comparisons.

Of the three deliverables, the PDF is the one that you can customize through chat. AWS Transform delivers all three files together in a ZIP file. The sheets and slides that you get depend on the workloads that your assessment covered.

Treat the first run as a draft. Verify the following in the results:

- Confirm that the server count in the total cost of ownership (TCO) summary matches the scope you set.
- Confirm that the recommendations that AWS Transform generates for your highest-cost servers look reasonable. The XLSX file lists the estimated cost for each server.
- Investigate anything that seems wrong, such as a recommended instance type that is larger than the source server.

When the numbers look right, take the following actions:

- Create a scenario for each option that you are weighing, such as Amazon RDS compared with SQL Server on Amazon EC2, and then compare the scenarios side by side. An assessment holds up to five scenarios.
- Generate the reports and presentations that you need to share.

You can change your assumptions and run the scenario again, create new scenarios with different assumptions, and compare scenarios side by side to find the option that fits. Re-uploading your inventory isn't required, but a changed scenario does run again in full. Start with your best understanding, and refine it as stakeholders weigh in.

Example prompts

You do each of these steps through chat. The following examples show one prompt for each step. Each prompt asks for several things at once, and that is deliberate. AWS Transform keeps context across messages, so a prompt that names your scope, your assumptions, and the output you want works better than a series of short questions. Change the details to match your environment.

Defining scope

This prompt checks how AWS Transform read your inventory, then narrows the scope:

> "How many servers are in my inventory, and which of them run SQL Server? List any data quality warnings you found. Then take srv-101, srv-102, and srv-103 out of scope, because you plan to decommission them. Confirm the final server count in scope, with its CPU, memory, and storage."

Defining assumptions

This prompt states your pricing, infrastructure, and licensing assumptions instead of leaving them on the defaults:

> "Assess the servers in scope in the US East (N. Virginia) AWS Region. Price them with three-year Reserved Instances, No Upfront. Allow AWS Graviton where my workloads support Arm, and keep the rest on x86. Use mixed tenancy, so that my Bring Your Own License Windows servers go on Dedicated Hosts. Assume Bring Your Own Media for SQL Server on Amazon RDS. Use gp3 volumes for block storage. Compare the TCO over five years."

Running the assessment

This step is simple once the scope and the assumptions are set. Run the assessment, and ask for detail so that you can check the result:

> "Run the assessment and give me a detailed breakdown of the results."

From there, stay in the same conversation to compare scenarios, run what-if analysis, and generate reports.

Combining the steps in one prompt

When you already know your scope and your assumptions, state them together and run the assessment in one prompt:

> "Assess my inventory, but take srv-101, srv-102, and srv-103 out of scope, because you plan to decommission them. Price the rest in us-east-1 with three-year Reserved Instances, No Upfront. Use mixed tenancy, so that my Bring Your Own License Windows servers go on Dedicated Hosts. Assume Bring Your Own Media for SQL Server on Amazon RDS. Then run the assessment and give me a detailed breakdown of the results."

Related resources

For more information about migration assessments, see the following topics:

- For more information about inventory formats, see the section called “Uploading inventory data”.
- For more information about scope, see the section called “Managing inventory scope”.
- For more information about assumptions, see the section called “Configuring assessment scenarios”.
- For instructions on collecting inventory automatically, see Discovery tool.

Compute assessments

AWS Transform analyzes your on-premises servers and recommends the best-fit, lowest-cost Amazon EC2 instance for each one. For every server, AWS Transform produces multiple pricing options, licensing analysis, and right-sizing recommendations so you can build a data-driven total cost of ownership business case in minutes.

Amazon EC2 right-sizing

AWS Transform matches each server to an Amazon EC2 instance based on its CPU, memory, and performance requirements. When your inventory includes utilization data, AWS Transform sizes

against observed utilization—typically P95 CPU utilization and peak memory—rather than the provisioned capacity, so over-provisioned on-premises servers are right-sized to what the workload actually uses. When utilization data is not available, AWS Transform applies configurable default utilization assumptions.

You can control how aggressively AWS Transform right-sizes by choosing a performance tier:

- **Conservative**—plans for more headroom and recommends larger instances.
- **Balanced**—the default, which balances headroom against cost.
- **Aggressive**—plans closer to observed utilization and recommends smaller instances.

AWS Transform considers current-generation instance families that match your workload's operating system and excludes older-generation families by default. You can further include or exclude specific instance families or types through assessment configuration or chat.

Processor architecture and AWS Graviton

By default, AWS Transform recommends x86 instances (Intel and AMD). You can change the processor architecture preference to include AWS Graviton-based instances, which are AWS-designed Arm processors that deliver better price performance for many workloads. The architecture preference supports three settings:

- **x86 only**—the default; recommends Intel and AMD instances.
- **AWS Graviton only**—restricts recommendations to AWS Graviton-based instance families.
- **Any**—considers both x86 and AWS Graviton instances and selects the best fit.

> **Note**
>
> Moving to AWS Graviton may require recompiling or repackaging applications for the Arm architecture. SQL Server workloads on Amazon EC2 are recommended on x86 instances only.

Tenancy and Dedicated Hosts

AWS Transform supports shared tenancy, dedicated tenancy, and mixed tenancy. The tenancy you choose affects both pricing and how AWS Transform accounts for Microsoft licensing.

- **Shared**—instances are priced using the standard Amazon EC2 instance pricing.
- **Dedicated**—instances are placed on Amazon EC2 Dedicated Hosts and priced using Dedicated Host pricing. AWS Transform packs your instances onto the fewest hosts needed, which is relevant when your licensing terms require you to license physical cores or use dedicated hardware.
- **Mixed**—AWS Transform places Bring Your Own License (BYOL)-eligible Windows workloads on Dedicated Hosts to reuse existing licenses, and places other workloads on shared tenancy with License Included pricing.

Bring Your Own License for Windows Server requires Dedicated Hosts and applies to Windows Server 2019 and earlier. Windows Server 2022 and later use License Included pricing on shared tenancy.

Pricing models

AWS Transform computes and reports multiple pricing models side by side and recommends the option with the lowest total annual cost. Because a three-year commitment is usually the least expensive, it is often the recommended option.

- **On-Demand**—pay by the hour with no commitment.
- **Reserved Instances**—one-year and three-year commitments, with No Upfront, Partial Upfront, and All Upfront payment options.
- **Savings Plans**—one-year and three-year commitments. You can choose Compute Savings Plans, which apply flexibly across instance families, sizes, Regions, and operating systems, or Amazon EC2 Instance Savings Plans, which offer a deeper discount in exchange for committing to a specific instance family in a Region.

Example prompts

- "Find the best-fit EC2 instance for each of my servers"
- "Show me the cost difference between On-Demand, Savings Plans, and Reserved Instances"
- "Recommend AWS Graviton instances where my workloads support Arm"
- "Right-size my servers using aggressive utilization assumptions"
- "What if I only use storage optimized instances?"

Database assessments

AWS Transform assesses your Microsoft SQL Server workloads and helps you choose between a fully managed database on Amazon RDS for SQL Server and running SQL Server yourself on Amazon EC2. For each option, AWS Transform detects the SQL Server edition, right-sizes compute and storage to your workload, analyzes licensing, and produces a cost comparison so you can see the trade-offs side by side.

Amazon RDS for SQL Server

AWS Transform assesses the cost of migrating on-premises SQL Server databases to Amazon RDS for SQL Server. Using AI-powered agents, AWS Transform analyzes your on-premises SQL Server environment and delivers a complete migration business case in minutes, with compute and memory recommendations matched to your workload requirements so you avoid over-provisioning and only pay for what you need.

The RDS for SQL Server assessment includes the following capabilities:

- Bring Your Own Media (BYOM) licensing, allowing you to use your existing SQL Server licenses through Microsoft License Mobility with Software Assurance. With BYOM there is no SQL Server license fee from AWS; you pay only for AWS infrastructure and Windows operating system fees. BYOM supports SQL Server Standard and Enterprise editions.
- License Included (LI) licensing, where the SQL Server license is included in your Amazon RDS hourly rate. License Included supports the Web, Express, Standard, Developer, and Enterprise editions.
- Cost optimization using Database Savings Plans, which offer up to 20 percent savings compared to On-Demand pricing.
- Eligibility assessment for the AWS Migration Acceleration Program (MAP), which provides credits and support to offset migration costs.

Right-sizing and edition recommendations

On-premises SQL Server environments are frequently over-provisioned. AWS Transform right-sizes each database to its actual workload, taking advantage of the fact that scaling an RDS instance up or down later is a fast, low-downtime operation. Because the number of vCPUs is the largest driver of ongoing cost—instance classes with more vCPUs are more expensive, and SQL Server is licensed

per core— AWS Transform focuses right-sizing on matching compute and memory to demand, and favors memory-optimized instance classes for memory-bound workloads.

AWS Transform also reviews whether a workload needs SQL Server Enterprise edition or can run on the lower-cost Standard edition. Because more capabilities move into Standard edition with each SQL Server release, many workloads that historically required Enterprise edition can migrate to Standard, reducing licensing cost.

Getting started

You can start your RDS for SQL Server assessment with any supported data format, including RVTools exports, configuration management database (CMDB) data, exports from the AWS Transform discovery tool, and other third-party discovery tools. Create what-if scenarios to compare multiple cost models with customized assumptions, including AWS Region, resource utilization, and pricing terms.

Example prompts:

- "Estimate the cost of migrating my SQL Server databases to RDS for SQL Server"
- "Compare BYOM vs License Included pricing for RDS for SQL Server"
- "Show me Database Savings Plans options for my RDS workloads"
- "Which of my databases can run on Standard edition instead of Enterprise?"

SQL Server on Amazon EC2

AWS Transform assesses SQL Server workloads that you intend to run on Amazon EC2 and provides right-sizing recommendations, licensing analysis, and dedicated host mappings. For Amazon EC2, you can bring your existing SQL Server licenses through License Mobility (Bring Your Own License, or BYOL) or use License Included (LI) pricing.

Running SQL Server on Amazon EC2 gives you full control over the operating system and the SQL Server instance, while Amazon RDS for SQL Server removes the undifferentiated management of the database engine. AWS Transform lets you compare the two so you can decide per workload. Consider these factors when comparing:

- Bring Your Own License and Bring Your Own Media require active Microsoft Software Assurance, which is an annual cost that AWS Transform can account for in the comparison.

- Dedicated host mappings are relevant when your licensing terms require you to license physical cores or use dedicated tenancy.

Example prompts:

- "What if I move my SQL Server workloads to RDS for SQL Server instead of EC2?"
- "Compare BYOL on EC2 with License Included on RDS for my SQL Server estate"
- "Create a License Included scenario for all SQL Server workloads on EC2"

Storage assessments

AWS Transform models your storage footprint and recommends AWS storage services and configurations based on your current storage usage and performance requirements. Storage attached to servers is right-sized alongside the compute recommendation, and AWS Transform also models detached storage across block, object, and file storage services.

Amazon EBS

AWS Transform recommends Amazon Elastic Block Store (Amazon EBS) volume types and configurations for the block storage attached to your migrated servers, based on your current capacity, IOPS, and throughput requirements. AWS Transform accounts for storage capacity, provisioned IOPS, and provisioned throughput when estimating cost, so you can compare volume types such as gp3 and io2.

Amazon S3

AWS Transform models object storage on Amazon Simple Storage Service (Amazon S3). Use Amazon S3 modeling for unstructured data, backups, and archives that do not need to remain on block storage.

Amazon FSx for NetApp ONTAP

AWS Transform models file storage on Amazon FSx for NetApp ONTAP, including storage capacity, provisioned IOPS, and provisioned throughput. This is the target for detached and shared file storage, including the storage attached to VMware workloads.

Example prompts

- "Estimate storage costs for my file servers on Amazon FSx for NetApp ONTAP"
- "What is the impact of moving from gp3 to io2 volumes?"
- "Model my backup and archive data on Amazon S3"

Analytics assessments

AWS Transform assesses analytics and streaming workloads and recommends managed AWS alternatives that reduce the operational overhead of running the same workload yourself. The analytics assessment estimates the total cost of ownership of the target managed service and reviews the source workload for anything that would behave differently after migrating.

Amazon MSK Express

AWS Transform assesses self-managed Apache Kafka clusters for migration to Amazon Managed Streaming for Apache Kafka (Amazon MSK) using Express brokers. AWS Transform takes the discovered cluster topology—brokers, coordination nodes, and engine version—together with cluster-wide throughput, recommends an MSK Express broker configuration, and calculates the total cost of ownership across broker hours, storage, data-in, and cross-Availability Zone transfer.

> **Note**
>
> The MSK Express assessment recommends Express brokers, which are a broker type for Amazon MSK Provisioned. It does not assess Amazon MSK Serverless or Standard brokers. For guidance on operating Express brokers after you migrate, see Best practices for Express brokers in the *Amazon MSK Developer Guide*.

Because Amazon MSK is a managed service, migrating to it replaces your Kafka broker hosts and their directly attached storage. The MSK Express assessment therefore takes ownership of those broker hosts and volumes, so they are not additionally costed as Amazon EC2 instances or Amazon EBS volumes elsewhere in your business case. If you choose not to migrate a cluster to MSK Express, exclude it from scope and its broker hosts and volumes return to the compute and storage assessments as a lift-and-shift estimate. This is the intended way to compare the managed-service option against running the same workload on Amazon EC2.

How clusters are sized

AWS Transform sizes each cluster from its throughput, not from your current broker count. A cluster with many small brokers can be sized as a smaller number of larger Express brokers. AWS Transform evaluates three constraints independently for each broker size and chooses the one that requires the most brokers:

- **Ingress**—the brokers needed to absorb your peak producer throughput.
- **Egress**—the brokers needed to serve your peak consumer throughput.
- **Partitions**—the brokers needed to hold your partitions within the per-broker recommendation.

Each broker count is rounded up to a multiple of three, because MSK Express always deploys across three Availability Zones with a minimum of three brokers. AWS Transform recommends the least expensive broker size and count that fits within the Amazon MSK per-cluster broker quota. You can adjust the utilization that AWS Transform plans each broker against (the default is 0.75, which leaves headroom for spikes) and the retention window used to size storage (the default is 24 hours).

Throughput data you provide

The single most useful figure you can provide is peak cluster ingress throughput. AWS Transform can derive the remaining throughput figures from any one of them, so supplying peak ingress alone produces a sized cluster. If no throughput figure is available at all, AWS Transform reports the minimum deployable Express cluster (three brokers, no storage, and no network charge), which is a floor rather than an estimate. When a figure is missing, AWS Transform prompts you for it and you can answer in plain language.

Field	Why it matters
Peak data in (cluster-wide)	Drives Express instance selection, the storage estimate, and the data-in charge. Required for a sized estimate.
Average data in (cluster-wide)	Sizes retained storage. Assumed to be half of peak if absent.
Peak data out (cluster-wide)	Can be the binding constraint on instance choice for fan-out heavy workloads.

Field	Why it matters
Average data out (cluster-wide)	Affects the cross-Availability Zone transfer estimate when consumers do not use nearest-replica fetching.
Kafka engine version	Determines the compatibility findings and whether a version change is involved.

Cost dimensions

AWS Transform prices four dimensions from the published Amazon MSK rates for your target AWS Region, refreshed daily. There is no Reserved Instance or Savings Plan input for MSK Express; the estimate is an On-Demand, steady-state run rate.

- **Broker hours**—per broker, per hour, by instance size, for 730 hours a month.
- **Storage**—per GB-month of retained data. Storage is elastic and pay-as-you-go on Express, so it needs no sizing.
- **Data in**—per GB of data written to the cluster.
- **Cross-Availability Zone transfer**—per GB of client traffic that crosses an Availability Zone boundary. This is not billed by Amazon MSK, but a three-Availability-Zone cluster incurs it in practice, so AWS Transform includes it rather than leaving it for you to discover later.

Compatibility review

Alongside the cost estimate, AWS Transform reviews each cluster across five areas and reports findings you can act on. Sizing, cost, and compatibility are evaluated independently, so a cluster with a compatibility finding still gets a full cost estimate.

- **Topology**—Availability Zone count, broker count, and ZooKeeper compared with KRaft coordination.
- **Kafka version**—whether your source version is one that MSK Express supports (Apache Kafka 3.6, 3.8, 3.9, and 4.2).
- **Configurations**—broker- and topic-level settings that diverge from the Apache Kafka default.
- **Authentication**—the authentication mechanism and whether traffic is encrypted in transit. MSK Express supports unauthenticated access, TLS, SASL/SCRAM, and AWS IAM.
- **Quotas**—per-broker ingress, egress, partitions, and connections against Express limits.

Each finding carries a severity. Findings marked as requiring action indicate something in your current setup that cannot be carried over as-is, such as an unsupported authentication mechanism or a configuration value outside the range MSK Express accepts. Advisory findings indicate that the migration works but something will behave differently, most often because MSK Express manages a setting on your behalf.

Region availability

MSK Express is available in most, but not all, AWS commercial Regions. Because Express always deploys across three Availability Zones, it is not available in Regions with fewer than three usable Availability Zones, such as US West (N. California). Set your target AWS Region explicitly: it determines both whether the assessment can run and every cost line it produces. If you select a Region where MSK Express is not available, the MSK line of your business case does not produce a cost, but your other assessments still complete.

Example prompts

- "Estimate the cost of migrating my Kafka clusters to Amazon MSK Express"
- "My cluster's peak ingress is 30 MB/s—size it for MSK Express"
- "Why was this cluster sized with six brokers?"
- "Show me the compatibility findings for my Kafka clusters"

Business value assessments

AWS Transform evaluates the broader business value of migration beyond infrastructure cost savings. The business value assessment quantifies benefits across the pillars of the AWS Cloud Value Framework, so you can present a business case that reflects the full value of moving to AWS, not just the change in run-rate cost.

You can run the business value assessment alongside your compute, storage, database, and analytics assessments, or on its own from a small set of high-level inputs when you do not yet have a detailed inventory. The assessment is deterministic and benchmark-driven: the same inputs always produce the same result, and the figures are directional estimates rather than commitments.

Value pillars

AWS Transform quantifies value across the following pillars:

- **Staff productivity**—savings from administering fewer servers and less storage after migrating to managed AWS services, based on improvements in the number of virtual machines and terabytes each administrator can manage.
- **Operational resilience**—savings from reduced unplanned downtime and fewer critical incidents, based on industry benchmarks applied to your annual revenue.
- **Business agility**—value from faster developer throughput and time to market, based on benchmarks applied to your developer headcount.

AWS Transform reports the value of each pillar and the total quantified benefit as annual figures.

Inputs

The business value assessment uses a small set of organizational inputs to select the appropriate benchmarks. You can provide these through chat.

- Your industry, chosen from a set of supported industry categories
- Annual revenue
- Total number of employees
- Total server count and total storage capacity
- Your AWS Support level (Business or Enterprise)

You can also adjust optional assumptions, such as the percentage of your workload that is in scope for the migration and the assumed annual IT staff salary used in the staff productivity calculation.

Sustainability

As part of the business case, AWS Transform estimates the carbon footprint reduction that results from migrating to AWS, consistent with the sustainability pillar of the AWS Well-Architected Framework. You can explore sustainability metrics through chat.

Example prompts

- "Estimate the business value of migrating to AWS for my organization"
- "Estimate staff productivity gains from migrating to AWS"
- "What resilience improvements can I expect after migration?"
- "Show me the estimated carbon reduction for this migration"

Additional cost components

Beyond the per-workload service recommendations, AWS Transform models several cross-cutting cost components so your total cost of ownership comparison reflects the full picture on both sides of the migration.

On-premises pricing

AWS Transform estimates your current on-premises costs to provide a baseline for comparison with AWS pricing. Most organizations have on-premises costs that a server inventory does not capture —colocation rent, networking, custom vendor deals, hardware refresh cycles, and maintenance contracts. You can add these directly through chat, and AWS Transform folds them into the baseline. On-premises cost adjustments are reflected in the PDF report and chat responses, but not in the PPTX export.

Example prompts:

- "Update the on-premises server cost to $500 per server per month"
- "My data center is a colo and I pay $10M a year in rent, add this to my on-premises costs"
- "Add $100,000 annual maintenance costs to the on-premises baseline"

Network costs

AWS Transform estimates network-related costs for your migration, including data transfer and connectivity requirements, and includes them in the AWS side of the comparison.

Support costs

AWS Transform includes AWS Support plan costs in the assessment based on your selected support tier, such as Business Support or Enterprise Support.

End user computing

AWS Transform assesses end user computing workloads that are candidates for Amazon WorkSpaces rather than Amazon EC2, and models the cost of running them on Amazon WorkSpaces. This lets you separate virtual desktop workloads from server workloads in your business case.

Adding other AWS services

You can add rough cost estimates for AWS services that are not fully modeled by AWS Transform assessments, so your analysis is more complete. These estimates are less accurate than the automated recommendations. For example, you can ask AWS Transform to add AWS Backup, Amazon CloudWatch, or AWS Direct Connect costs, or to move specific servers to another service such as Amazon Connect and remove them from the compute model.

Related topics

- Getting started
- Custom jobs
- Discovery tool

Discovery tool

The AWS Transform discovery tool helps you automatically discover server inventory in your organization to prepare for migration. The tool supports VMware vCenter, Microsoft Hyper-V, and imported servers. To use the discovery tool, you configure one or more discovery sources, let the tool run, and then review the results in the **Discovered inventory** pane.

After you configure a discovery source, the discovery tool begins collecting information. The time that the discovery tool needs to completely analyze your network depends on the size of your environment. For a directional migration business case, you can use the Migration Portfolio Assessment (MPA) files that the tool generates after server collection completes.

Discovery tool workflow

The discovery tool workflow consists of two types of activities:

- Configuration activities
- Data review and use

The following steps describe how to install and use the discovery tool and work with the collected data:

1. Install the discovery tool.
2. Configure discovery sources. You can configure one or more of the following: VMware vCenter, Microsoft Hyper-V hosts, or servers through a CSV file import. Data discovery begins after you configure any source.
3. Set up OS access and then review the collection status of servers, databases, and network connections.
    - Adjust OS credentials as needed.
4. (Optional) Configure Oracle credentials to enable direct SQL collection, or use OS-level fallback detection through SSH or WinRM.
5. To generate a migration business case, upload the .zip file to Migration assessment, or unzip it and upload MPA files from the *mpa_exports* directory. The export includes data from all configured sources and contains MPA files for VMware, Hyper-V, and imported servers.

The discovery tool supports the following discovery paths:

- **VMware vCenter auto-discovery** – Automatically discover servers managed by VMware vCenter.
- **Hyper-V auto-discovery** – Automatically discover servers managed by Microsoft Hyper-V hosts.
- **Server import** – Manually import server inventory through a CSV file.

Setting up the discovery tool

Prerequisites

The following are the prerequisites for using the AWS Transform discovery tool:

General prerequisites

- The tool requires 4 vCPU and 16 GB of RAM. The OVA and VHD appliance images default to a 100 GB disk; larger inventories require more disk. For guidance, see the section called “Disk sizing and management”.
- The tool collects data by using a centralized approach. Servers in scope must allow inbound connectivity from the discovery tool VM (default ports, custom port configuration is supported):
  - Linux – SSH TCP/22
  - Windows – TCP/5985 for HTTP, TCP/5986 for HTTPS
  - SNMP – UDP/161 (used for network collection only, not OS metrics)
- For Linux, user accounts that can use SSH to connect to the server. The discovery tool runs commands over SSH for network collection and OS metrics. Most commands run as a regular user. A small number of commands attempt sudo and fall back automatically if sudo is unavailable: `dmidecode` (server UUID and manufacturer), `lvdisplay` (LVM detection), and `ss` or `netstat` (network connections with process information). Without sudo, the discovery tool still collects the majority of data but UUID, LVM, and process-level network details will be missing. We recommend configuring passwordless sudo for the SSH user to ensure complete data collection. For a full breakdown, see the section called “Required permissions”.
- For OS-level collection (OS metrics, network connections, and database discovery), target servers must run a supported operating system. For the supported target operating systems, see the section called “Requirements for OS-level collection”.

Appliance (OVA and VHD) prerequisites

- DHCP must be available on the network segment that the appliance connects to. The prebuilt Open Virtualization Appliance (OVA) and Virtual Hard Disk (VHD) images obtain an IP address by DHCP on the first boot, and you use that address to open the discovery tool console.
- The OVA and VHD appliances run Amazon Linux 2023. Your VMware or Hyper-V host must meet the Amazon Linux 2023 host requirements. See VMware host requirements for running AL2023 on VMware or Hyper-V host requirements for running AL2023 on Hyper-V in the *Amazon Linux 2023 User Guide*.

VMware prerequisites

- VMware vCenter Server version 6.5, 6.7, 7.0, or 8.0.
- Permissions to deploy an OVA into your VMware vCenter.
- For VMware vCenter Server setup, a vCenter user with the Read-Only role assigned at the root datacenter level. For details, see the section called “Required permissions”.
- VMware Tools must be installed and running in each guest virtual machine. The discovery tool reads the guest IP address, hostname, fully qualified domain name (FQDN), guest operating system, and per-partition disk usage from VMware Tools. For more information, see the section called “Guest agent requirements for OS-level collection”.

Hyper-V prerequisites

- Windows Server with the Hyper-V role enabled.
- WinRM enabled on Hyper-V hosts.
- A user account that is a member of the **Remote Management Users**, **Hyper-V Administrators**, and **Performance Monitor Users** groups on each Hyper-V host, with WMI read access to the `root\cimv2` namespace. For details, see the section called “Required permissions”.
- Supported authentication: NTLM (HTTPS only) and Kerberos (HTTP or HTTPS).
- Hyper-V Integration Services must be running in each guest virtual machine. The discovery tool reads the guest hostname and FQDN from Integration Services through Key-Value Pair (KVP) exchange. If Integration Services is not running, or the virtual machine is powered off, this data is empty. For more information, see the section called “Guest agent requirements for OS-level collection”.

- When you create the Hyper-V virtual machine, you must select **Generation 1**. Generation 2 virtual machines do not support the VHD format.

Server import prerequisites

- A CSV file with server hostnames or IP addresses and the credential names (optional) that map to the friendly names of the OS credentials configured or to be configured on the discovery tool. The CSV must use the following headers:

```
hostname_or_ip,os_credential_name,oracle_credential_name
```

- Servers must be reachable from the discovery tool VM. Default ports: SSH port 22 for Linux, WinRM port 5985/5986 for Windows. Custom ports are supported.

Linux installer prerequisites

- A supported Linux distribution: Amazon Linux 2, Amazon Linux 2023, RHEL 8–9, Rocky Linux 8–9, AlmaLinux 9, Ubuntu 20.04–24.04, Debian 10–12, or SLES 15 SP5.
- Minimum 4 vCPU and 16 GB RAM. Provide enough free disk space for the installation and your collected data; larger inventories require more disk. For guidance, see the section called “Disk sizing and management”.
- Port 5000 must be available (not in use by another service).
- systemd is required for service management.
- The discovery tool VM must have network access to your target infrastructure. Default ports: vCenter on port 443, Hyper-V hosts on port 5985/5986, servers on port 22. Custom ports are supported for SSH, WinRM, and SNMP.

Oracle Database prerequisites

- Network access from the discovery tool to Oracle hosts on port 1521 (or custom port).
- Supported Oracle versions: You can collect Oracle Database 12c Release 1 (12.1) and later through direct SQL connections. OS-level fallback detection works with all Oracle versions.
- A read-only Oracle service account with SELECT_CATALOG_ROLE grant. For details, see the section called “Oracle Database (SQL)”.

- For OS-level fallback detection: SSH or WinRM access to the Oracle host (uses existing OS credentials).

Guest agent requirements for OS-level collection

The discovery tool discovers virtual machines from vCenter or Hyper-V without any agent in the guest. However, OS-level collection — OS performance and storage metrics, network connection tracking, and SQL Server and Oracle database discovery — requires a network address to connect to the guest operating system. The discovery tool obtains that address, along with other guest details, from the guest agent. The guest agent is **VMware Tools** on VMware and **Hyper-V Integration Services** on Hyper-V.

What the discovery tool collects without the guest agent

Hypervisor-side inventory is still collected: CPU, RAM, and disk configuration; power state; virtual disk utilization; and hypervisor performance counters.

What the discovery tool cannot collect without the guest agent

The guest IP address, hostname, FQDN, guest operating system, and per-partition free space are unavailable. As a result, OS metrics, network connection, SQL Server, and Oracle collection have no address to connect to, and Kerberos authentication has no FQDN for service principal name (SPN) matching.

- **VMware** – The `guest.*` properties, including the guest FQDN, come from VMware Tools running in the guest. They are unavailable if VMware Tools is not installed or not running.
- **Hyper-V** – Guest OS detection uses Hyper-V KVP exchange and requires Integration Services running in the guest. It returns an empty value if Integration Services is not installed or the virtual machine is powered off.

If the guest agent is not available, you can supply each server's IP address directly by importing servers from a CSV file, which bypasses the guest-agent dependency for OS-level collection. For more information, see the section called “Import servers”.

Disk sizing and management

The discovery tool stores its data in an internal database on the root volume. The database grows with the number of servers in scope. Network collection increases disk usage further, especially

when private IP address collection is enabled. The discovery tool prunes data older than 30 days, so disk usage plateaus after the first month.

Recommended volume sizes

The following table shows the recommended disk size for each environment size, based on how many servers you monitor and whether network collection is enabled.

Environment	Servers	Network collection	Recommended volume
Any size	Any	Disabled	100 GB (the appliance default)
Small	Fewer than 500	Enabled	100 GB (the appliance default)
Medium	500–2,000	Enabled, no private addresses	100 GB
Medium	500–2,000	Enabled, private addresses on	150–200 GB
Large	2,000 or more	Enabled	200 GB

When in doubt, provision more disk than you expect to need. Resizing a running discovery tool requires downtime.

Increase the disk size

Choose the procedure that matches how you deployed the discovery tool. On the OVA and VHD appliances, cloud-init automatically expands the root partition on the first boot after you increase the disk. You do not need to partition the disk manually. On a Linux installer deployment, you manage the host yourself. You might also need to extend the partition and file system.

To increase the disk on a VMware OVA deployment

1. Deploy the Open Virtualization Format (OVF) template. Before you power on the virtual machine, open the vSphere Client, right-click the virtual machine, and choose **Edit Settings**.

2. Increase **Hard disk 1** to the size you want, and then save. The OVF deployment wizard does not expose the disk size, so you set it here.
3. Power on the virtual machine. If the virtual machine is already running, power it off first, increase **Hard disk 1** in **Edit Settings**, and then power it on.

To increase the disk on a Hyper-V VHD deployment

1. Before you start the virtual machine, in Hyper-V Manager, open the virtual machine **Settings**.
2. Select **Hard Drive**, choose **Edit**, choose **Expand**, and set the new size. The disk size cannot be set during import, so you set it here.
3. Start the virtual machine. If the virtual machine is already running, shut it down first, expand the hard drive in **Settings**, and then start it.

To size the disk for a Linux installer deployment

1. Provision a larger root disk on the host before you run the installer. The installer does not enforce a disk size.
2. If you increase the disk after you install, extend the partition and file system on the host by using the tools for your distribution. The discovery tool uses whatever space is available on the file system that holds its data directory.

Check disk usage

To check current disk usage on the discovery tool virtual machine, run `df -h /`.

If the discovery tool is already out of disk space, see Internal database disk space management in the AWS Transform discovery tool on the AWS re:Post website.

Deploy on VMware

VMware virtual machine specifications

- **Operating System** – Amazon Linux 2023
- **RAM** – 16 GB
- **CPU** – 4 cores

- **Disks** – 100 GB minimum (default, thin-provisioned so it still imports onto small datastores). For larger inventories, see the section called "Disk sizing and management".
- **VMware requirements** – See VMware host requirements for running AL2023 on VMware

Deploy the VMware OVA

1. Download the OVA file from this URL: https://s3.us-east-1.amazonaws.com/atx.discovery.collector.bundle/releases/latest/AWS-Transform-discovery-tool.ova
2. Sign in to vCenter as a VMware administrator.
3. Use one of these ways to install the OVA file:

   a. **Use the UI:** Choose **File**, choose **Deploy OVF Template**, select the discovery tool OVA file you downloaded in step 1, and then complete the wizard. Ensure the proxy settings in the server management dashboard are configured correctly.

   b. **Use the command line:** To install the discovery tool OVA file from the command line, download and use the VMware Open Virtualization Format Tool (ovftool). To download ovftool, select a release from the OVF Tool Documentation page. This is an example of using the ovftool command line tool to install the discovery tool OVA file.

   ```
   ovftool --acceptAllEulas --name='discovery tool' --datastore=datastore1 -
   dm=thin AWS-Transform-discovery-tool.ova 'vi://username:password@vcenterurl/
   Datacenter/host/esxi/'
   ```

   Descriptions of the replaceable values in the example:

   - The name is the name that you want to use for your discovery tool VM.
   - The datastore is the name of the datastore in your vCenter.
   - The OVA file name is the name of the downloaded discovery tool OVA file.
   - The username/password are your vCenter credentials.
   - The vcenterurl is the URL of your vCenter.
   - The vi path is the path to your VMware ESXi host.

4. Locate the deployed discovery tool in your vCenter. Right-click the VM, and then choose **Power**, **Power On**.
5. After a few minutes, the IP address of the discovery tool displays in vCenter. You use this IP address to connect to the discovery tool.

Deploy on Hyper-V

Hyper-V virtual machine specifications

- **Operating System** – Amazon Linux 2023
- **RAM** – We recommend allocating at least 16 GB
- **CPU** – We recommend allocating at least 4 cores
- **Disks** – 100 GB minimum (included in the VHD). For larger inventories, see the section called "Disk sizing and management".
- **Hyper-V requirements** – See Hyper-V host requirements for running AL2023 on Hyper-V

Deploy the Hyper-V VHD

1. Download the VHD file from this URL: https://s3.us-east-1.amazonaws.com/atx.discovery.collector.bundle/releases/latest/AWS-Transform-discovery-tool.vhd
2. Copy the VHD file to the Windows Server machine that has the Hyper-V role enabled.
3. Open Hyper-V Manager.
4. Choose **New**, and then choose **Virtual Machine**.
5. Complete the setup wizard. On the **Specify Generation** page, select **Generation 1**. Generation 2 virtual machines do not support the VHD format. On the **Assign Memory** page, allocate at least 16384 MB. On the **Connect Virtual Hard Disk** page, choose **Use an existing virtual hard disk** and select the VHD file that you copied.
6. Start the VM. After a few minutes, check the **Networking** tab of the VM in Hyper-V Manager to find the IP address, or connect to the VM console and run `ip addr`. You use this IP address to connect to the discovery tool.

Deploy on Linux

The discovery tool is available as a standalone Linux installer for deployment on supported Linux distributions. Use this option when you want to install the discovery tool directly on an existing Linux server instead of deploying a virtual machine.

Linux server specifications

- **RAM** – We recommend allocating at least 16 GB

- **CPU** – We recommend allocating at least 4 cores
- **Disks** – Provide enough free disk space for the installation and your collected data. For larger inventories, see the section called "Disk sizing and management".

Supported distributions for the installer host

- **Amazon Linux** – Amazon Linux 2, Amazon Linux 2023
- **Red Hat Enterprise Linux and derivatives** – RHEL 8–9, Rocky Linux 8–9, AlmaLinux 9
- **Ubuntu** – 20.04, 22.04, 24.04
- **Debian** – 10, 11, 12
- **SUSE** – SLES 15 SP5

Install the discovery tool

Download the installer script from this URL: https://s3.us-east-1.amazonaws.com/atx.discovery.collector.bundle/releases/latest/AWS-Transform-discovery-tool.sh

To install the discovery tool on Linux

1. Download the installer script to your Linux server.

```
curl -O https://s3.us-east-1.amazonaws.com/atx.discovery.collector.bundle/releases/
latest/AWS-Transform-discovery-tool.sh
chmod +x AWS-Transform-discovery-tool.sh
```

2. (Optional) Run the compatibility check to verify that your system meets the requirements.

```
sudo ./AWS-Transform-discovery-tool.sh check
```

3. Install the discovery tool. Choose one of the following options:

   Option 1: Install with automatic dependency installation (recommended)

   This option automatically installs any missing system dependencies before installing the service.

```
sudo ./AWS-Transform-discovery-tool.sh install --install-deps
```

Option 2: Install without automatic dependency installation

This option checks for missing dependencies and displays errors with manual install instructions if critical dependencies are missing. Use this option if your organization restricts automatic package installation and you prefer to manage system packages manually.

```
sudo ./AWS-Transform-discovery-tool.sh install
```

4. Start the discovery tool service.

```
sudo ./AWS-Transform-discovery-tool.sh start
```

5. Access the discovery tool at `https://ip_address:5000`.

To uninstall the discovery tool, run `sudo ./AWS-Transform-discovery-tool.sh uninstall`.

You can also use the following commands to manage the service:

- `sudo ./AWS-Transform-discovery-tool.sh status` – Check whether the service is running.
- `sudo ./AWS-Transform-discovery-tool.sh stop` – Stop the service.
- `sudo ./AWS-Transform-discovery-tool.sh logs` – View service logs.

System dependencies

The installer checks for the following system dependencies:

Dependency	Severity	Purpose
OpenSSL	Critical	Generates the database encryption key
libcrypt	Critical	Required by the Python runtime
Kerberos libraries	Warning	Required for Kerberos authentication to Windows servers. Without this, NTLM and basic authentication still work.

Critical dependencies are required for the discovery tool to start. If they are missing and you chose Option 2 (without `--install-deps`), the installer displays an error with instructions to install them manually. Warning-level dependencies are optional — the installer warns you but proceeds with installation.

> **Note**
>
> The installer checks system compatibility including glibc version and systemd availability. On systems with systemd 250 or later (Amazon Linux 2023, RHEL 9, Rocky 9, AlmaLinux 9, Ubuntu 24.04+, Debian 12+), the database encryption key is encrypted at rest using systemd-creds. On older systems, the encryption key is stored as a permission-protected file.

Configure the discovery tool

Access the discovery tool console

1. In a web browser access: `https://`*`ip_address`*`:5000`, where *`ip_address`* is the IP address of the discovery tool from Deploy Discovery Tool. The discovery tool uses a self-signed certificate for HTTPS connection which results in a security warning. Choose **Accept the risk and continue** to continue to the discovery tool console.
2. If you're accessing the discovery tool console for the first time, create a discovery tool login password. Create a password, which you will use for future logins.

   > **Important**
   >
   > Remember this password - there is no password recovery mechanism.

Accessing the discovery tool VM

- The discovery tool VM comes by default with a username and password ("discovery", "password"). For strong security, we recommend that you update the password by using `sudo passwd discovery` after logging into the VM through your hypervisor's console (for example, vSphere Client for VMware or Hyper-V Manager for Hyper-V).

- SSH access is disabled by default. Users can use preconfigured `enablessh` and `disablessh` aliases to enable/disable SSH access to the discovery tool VM. Users can SSH into the VM via `ssh discovery@<VM-IP>` after enabling SSH access. Users are encouraged to keep SSH access disabled most of the times and enable it only while actively required. Password change is enforced when running `enablessh`.
- The discovery tool data directory is at `/home/discovery/.local/share/DiscoveryTool` on the standard OVA and VHD appliance, where the default VM user is `discovery`. On a Linux installer deployment, the data directory is under the home directory of whichever user ran the installer, not necessarily `discovery`. To find the exact path on any deployment, run `systemctl show discovery-tool -p Environment` and look for `DT_DATA_DIR`. If `DT_DATA_DIR` is not listed, the data directory is `.local/share/DiscoveryTool` under the home directory of the user that runs the service.

Configure Kerberos authentication

Kerberos authentication is the recommended method for connecting to Windows servers from the discovery tool. The discovery tool VM uses native Amazon Linux 2023 Kerberos libraries to authenticate against your Active Directory domain.

The following are key points about Kerberos authentication on the discovery tool VM:

- Use the `kinit` command to obtain a Kerberos ticket and `klist` to verify the ticket.
- The Kerberos configuration file is located at `/etc/krb5.conf`.
- Before you configure the discovery tool, verify that `kinit` succeeds from the CLI on the discovery tool VM.

Kerberos prerequisites

Before you configure Kerberos authentication, verify that you have the following information and network connectivity.

1. Obtain the following information from your Active Directory administrator:
   - The Kerberos realm name (typically your domain name in uppercase, for example, `EXAMPLE.COM`).
   - The hostname or IP address of the Key Distribution Center (KDC), which is typically a domain controller (for example, `dc01.example.com`).

- A service account with permissions to authenticate against the target Windows servers.

2. Verify that the discovery tool VM has network connectivity to the following:
   - The KDC on port 88 (TCP and UDP) for Kerberos authentication.
   - The target Windows servers on Windows Remote Management (WinRM) ports (5985 for HTTP, 5986 for HTTPS).

Configure Kerberos

Complete the following steps to configure Kerberos authentication on the discovery tool VM.

1. SSH to the discovery tool VM.

   ```
   ssh discovery@<discovery-tool-vm-ip>
   ```

2. Edit the Kerberos configuration file at `/etc/krb5.conf`.

   ```
   sudo nano /etc/krb5.conf
   ```

   Add the following configuration, replacing the placeholder values with your environment details.

   ```
   [libdefaults]
       default_realm = EXAMPLE.COM
       dns_lookup_realm = false
       dns_lookup_kdc = true

   [realms]
       EXAMPLE.COM = {
           kdc = dc01.example.com
       }

   [domain_realm]
       .example.com = EXAMPLE.COM
       example.com = EXAMPLE.COM
   ```

> **Important**
>
> Kerberos is case-sensitive. The realm name must be in uppercase (for example, `EXAMPLE.COM`, not `example.com`). The domain name in the `[domain_realm]` section must be in lowercase.

Multiple Active Directory domains

The discovery tool supports multiple Kerberos credentials for different Active Directory domains. Each credential authenticates independently, so you can configure multiple credentials normally and isolation is automatic.

If you have servers in multiple domains, add entries for each realm in your `/etc/krb5.conf` file:

```
[libdefaults]
    default_realm = DEV.COMPANY.COM
    dns_lookup_realm = false
    dns_lookup_kdc = true

[realms]
    DEV.COMPANY.COM = {
        kdc = dc01.dev.company.com
    }
    PROD.COMPANY.COM = {
        kdc = dc01.prod.company.com
    }

[domain_realm]
    .dev.company.com = DEV.COMPANY.COM
    dev.company.com = DEV.COMPANY.COM
    .prod.company.com = PROD.COMPANY.COM
    prod.company.com = PROD.COMPANY.COM
```

3. Verify that you can obtain a Kerberos ticket by running the `kinit` command.

```
kinit username@REALM.COM
```

Enter the password when prompted. If the command completes without errors, authentication succeeded.

4. Verify the ticket by running the `klist` command.

```
klist
```

The expected output is similar to the following.

```
Ticket cache: FILE:/tmp/krb5cc_1000
Default principal: username@REALM.COM

Valid starting       Expires              Service principal
01/01/2025 12:00:00  01/01/2025 22:00:00  krbtgt/REALM.COM@REALM.COM
```

5. Configure the discovery tool with the same case-sensitive principal that you used with `kinit` (for example, `username@REALM.COM`).

An explicit `krb5.conf` configuration might not be required if your environment has DNS SRV records configured for Kerberos service discovery. For more information about Kerberos configuration options, see the MIT Kerberos krb5.conf documentation and the sample krb5.conf file.

Find Kerberos configuration from domain-joined machines

If you don't have the Kerberos configuration details, you can retrieve them from a Windows machine that is joined to the domain. Run the following commands from a command prompt on the domain-joined machine.

To find the domain name, run the following command.

```
echo %USERDNSDOMAIN%
```

Example output:

```
EXAMPLE.COM
```

To find the domain controller hostname, run the following command.

```
nltest /dsgetdc:EXAMPLE.COM
```

Example output:

```
           DC: \\dc01.example.com
      Address: \\10.0.1.100
     Dom Guid: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
     Dom Name: EXAMPLE.COM
  Forest Name: example.com
 Dc Site Name: Default-First-Site-Name
Our Site Name: Default-First-Site-Name
        Flags: 0xe00033fd
The command completed successfully
```

Map the output to your `krb5.conf` configuration as follows:

- **Realm** – Use the value from `%USERDNSDOMAIN%` in uppercase (for example, `EXAMPLE.COM`).
- **KDC** – Use the DC hostname from the `nltest` output (for example, `dc01.example.com`).

Using Kerberos with IP addresses

Kerberos identifies each target server by a service principal name (SPN). The SPN is built from the server's fully qualified domain name (FQDN), such as `WSMAN/server01.example.com`. If the server does not recognize the `WSMAN` SPN, the discovery tool falls back to `HTTP/server01.example.com`. Active Directory does not register SPNs for IP addresses.

When the discovery tool knows a server's FQDN, it uses the FQDN first for Kerberos credentials. The discovery tool learns the FQDN from guest metadata reported by VMware Tools on VMware, or by Integration Services on Hyper-V. Most servers that the discovery tool finds through VMware or Hyper-V discovery have an FQDN, so you do not need to do anything for those servers. A server that you import from a CSV file with only an IP address has no FQDN.

The discovery tool also tries any IP addresses it has for a server. An IP address works with Kerberos when the discovery tool can map that IP address to the FQDN with a reverse DNS lookup. Reverse DNS is a best-effort fallback, not a requirement. Set it up only if a server has no FQDN, and you want to use Kerberos with that server. The lookup needs one of the following:

- **A PTR record** – A PTR record for that IP address in your DNS or Active Directory. A PTR record maps an IP address to a host name. It is the reverse of the A record, which maps a host name to

an IP address. Many Active Directory domains add PTR records for you when a server joins, so the record might already exist.

- **An `/etc/hosts` entry** – An entry in `/etc/hosts` on the discovery tool VM, in the form `<ip-address> <fqdn> <short-name>`. Use this when you cannot change DNS. You do not need a DNS administrator.

Reverse lookups must also be turned on for the VM. They are on by default. Do not set `rdns = false` or `dns_canonicalize_hostname = false` in `/etc/krb5.conf`. If either setting is already `false`, Kerberos keeps the IP address in the SPN instead of replacing it with the FQDN, and authentication fails with a `Server not found in Kerberos database` error even when the reverse mapping exists. Remove the setting, or set it to `true`.

To check reverse lookups, run the following command on the discovery tool VM.

```
getent hosts <target-ip>
```

The command returns the server's FQDN when reverse lookups work. It reads both `/etc/hosts` and DNS. If it does not return the FQDN, add a PTR record, add an `/etc/hosts` entry, or use the FQDN instead.

> **Note**
>
> If no reverse mapping exists, authentication fails with `Server not found in Kerberos database`. This is a principal lookup failure, not a rejected password. It does not count toward Active Directory account lockout.

Configure vCenter access

1. On the **Discovery tool** page, under **Step 1. Configure discovery sources**, choose **Configure sources**.
2. On the **Configure discovery sources** page, provide the **Friendly name**, **vCenter FQDN/IP**, **Username**, and **Password**.
3. Choose **Save configuration**.

   The discovery tool begins to collect vCenter information, as described in Discovered Inventory.

After initial configuration choose **Edit vCenter access** in the **Discovery tool status** frame to change your vCenter access settings.

The discovery tool collects from all configured vCenter servers in parallel. If one vCenter server is unreachable during collection, the tool reports partial success and continues collecting from the remaining vCenter servers.

If a VM appears on multiple vCenter servers (for example, due to shared ESXi hosts or cross-vCenter vMotion), the discovery tool automatically deduplicates VMs. Each unique VM appears only once in the inventory.

Configure Hyper-V access

1. On the **Discovery tool** page, under **Step 1. Configure discovery sources**, choose **Configure sources**.
2. On the **Configure discovery sources** page, provide a **friendly name**, the **host FQDN or IP address**, the **authentication type (NTLM or Kerberos)**, the **WinRM username**, and the **WinRM password**.
3. Choose **Save configuration**.

   The discovery tool begins to collect Hyper-V information, as described in Discovered inventory.

Collection begins automatically after you save the credentials.

For Hyper-V failover clusters, you can add multiple hosts in the same cluster. The tool automatically deduplicates VMs that appear on more than one host.

Import servers

1. Navigate to the **Import servers** page from the Discovery tool homepage.
2. Prepare a CSV file with the following columns: `hostname_or_ip` (required), `os_credential_name` (optional), and `oracle_credential_name` (optional).

   - The `hostname_or_ip` value must be a valid IPv4 address or a fully qualified domain name (FQDN).
   - The `os_credential_name` value, if provided, must match the friendly name of an OS credential that you already configured (SSH, WinRM, or SNMP). Leave empty for servers where you have not yet configured an OS credential.

- The `oracle_credential_name` value, if provided, must match the friendly name of an Oracle credential that you already configured.

3. Upload the CSV file. The tool validates all rows and rejects the file if any row is invalid.

After a successful import, the tool automatically begins database, network and OS metrics collection for the imported servers, if OS credentials are configured. If you upload another CSV file, existing records are updated without creating duplicates and new records are merged into the inventory.

Import a self-signed certificate authority into the discovery tool (Optional)

This is required when you use WinRM over HTTPS and target servers using WinRM HTTPS certificates signed by a self-signed Certificate Authority (CA), and you want to enable "Validate server SSL certificate" on the discovery tool.

Prerequisites

1. Self-signed CA certificate that was used to sign the WinRM HTTPS certificates on target servers
2. Certificate in PEM format (.pem or .crt extension)

To import a self-signed certificate authority on the discovery tool VM:

1. Ssh to Discovery tool VM
2. Place the CA certificate(s) that signed your target servers' WinRM certificates into trust store directory `/etc/pki/ca-trust/source/anchors/` on the discovery tool VM. For example: `sudo cp winrm-ca.pem /etc/pki/ca-trust/source/anchors/winrm-ca.pem`. Note: If your target servers use certificates signed by different CAs, copy all relevant CA certificates to this directory.
3. Update the certificate trust store: `sudo update-ca-trust`
4. Reboot the VM
5. (Optional) To verify that certificates have been successfully imported, you can run the following command. `sudo trust list –filter=ca-anchors | grep -A 5 "<certificate_name>"`

See Installation and configuration for Windows Remote Management

Configure the discovery tool for OS access

Configure OS access so that the discovery tool can:

- Discover databases to perform database assessment and to assist in VM migration,
- Track network connections between servers in your inventory, including the process associated with each connection, to assist in application dependency mapping and wave planning. Only connections where both endpoints are in the discovery tool's inventory are included.

Enable discovery tool OS Access

1. Navigate to the **Set up OS access** page to provide Windows and Linux credentials.
2. Choose a protocol that you want to add credentials for.
3. Provide the required credentials for the selected protocol.
4. Select **Auto-connect** to enable the discovery tool to try all provided credentials on discovered servers until matching credentials are found for each server.

   See the section called “Using Auto-Connect Feature With Caution” for important security recommendations regarding the auto-connect feature.
5. Choose **Set up and connect**.

When the OS matching process is completed, you see a message that the data collection is in progress, and an error regarding servers for which a credentials match was not found.

Requirements for OS-level collection

OS-level collection — server and storage performance metrics, network connection tracking, and SQL Server and Oracle database discovery — runs remotely against your target servers. The discovery tool installs no agent or software on them. The following are the requirements for a target server to be collected.

> **Note**
>
> These requirements apply to the servers you are collecting data *from*. They are separate from the Linux distributions supported for the host you install the discovery tool *on*, which are listed in the section called “Supported distributions for the installer host”.

The following list describes the connectivity and account requirements for each type of target server.

Linux

SSH reachable from the discovery tool VM on TCP/22 (or a custom port), and an account that can run shell commands. Network connection tracking can also use SNMP on UDP/161 instead of SSH. sudo is optional. Without it, the discovery tool still collects most data, but the server UUID, LVM detection, and process-level network connection details are missing. For details, see the section called “Linux servers (SSH)”.

Windows

WinRM enabled and reachable on TCP/5985 (HTTP) or TCP/5986 (HTTPS), or a custom port. The discovery tool runs all Windows commands through PowerShell remoting, so PowerShell must be present on the target server. PowerShell 3.0 or later provides full data coverage. For details, see the section called “Windows servers (WinRM) — OS metrics”.

> **Important**
>
> If Windows PowerShell Transcription logging is enabled on your target servers, the discovery tool's frequent PowerShell commands can generate thousands of transcript files each day on each server. This can fill the target server's system disk quickly. See the section called “Set up WinRM and WMI” for the recommended mitigation before enabling collection on Windows targets.

Because collection uses standard operating system commands rather than an agent, the discovery tool works across a wide range of operating system versions, including older ones. The following operating systems are validated end to end. Other versions that meet the requirements in the preceding list are also collected.

- **Linux** – Amazon Linux 2, Amazon Linux 2023, RHEL 7.9, Rocky Linux 9.7, AlmaLinux 9.7, SLES 12 SP5, Debian 11, Ubuntu 16.04, and Ubuntu 22.04
- **Windows** – Windows Server 2012 R2, 2016, 2019, and 2022

Reduced data coverage on older operating systems

On older operating systems, the discovery tool substitutes alternative commands and some fields can be empty:

- Linux distributions released before `/etc/os-release` became standard — such as RHEL 6, CentOS 6, SLES 11, Ubuntu 14.04 and earlier, and Amazon Linux 1 — are identified from `/etc/*-release` or `uname` instead, which can report less detail.
- Windows Server 2003 does not include PowerShell. Because the discovery tool runs all Windows commands through PowerShell remoting, it collects no OS-level data from these servers unless you install PowerShell and WinRM on them.
- On servers that provide only PowerShell 2.0, such as Windows Server 2008, `Get-CimInstance` is unavailable. The discovery tool falls back to `Get-WmiObject` for basic server identity, but most other data, including performance and storage metrics, is unavailable.
- Servers whose SSH service offers only legacy key exchange or host key algorithms, which is common on RHEL 6 and other distributions of that era, can refuse the connection. If a server cannot be reached over SSH but is reachable manually, verify which algorithms its SSH service offers.

Supported protocols setup

You must set up WinRM, SSH, and SNMP protocols on target servers for the discovery tool to communicate with them.

Set up WinRM and WMI

WinRM is automatically installed with all currently-supported versions of the Windows operating system.

To verify or edit WinRM configuration, use the `winrm` command line tool:

- Verify installed WinRM listeners: `winrm enumerate winrm/config/listener`
- Verify WinRM configurations: `winrm get winrm/config`
- Example command to set up WinRM: `winrm quickconfig -transport:https`

Listener Ports

Default HTTP port is 5985; HTTPS is 5986. You can use other ports as needed. The ports must be open between the discovery tool and target servers.

Encryption

The discovery tool uses encrypted WinRM communication. We recommend that WinRM listeners on target servers also use encryption: `winrm set winrm/config/service '@{AllowUnencrypted="false"}'`

NTLM vs Kerberos

WinRM authentication protocols Kerberos and NTLM are supported by the discovery tool. NTLM can be used only with HTTPS and Kerberos can be used with both HTTP or HTTPS.

WMI Requirements

The discovery tool queries the following WMI namespaces. The WinRM account needs read access to each namespace relevant to your collection modules:

WMI namespace	Used by
`root\cimv2`	OS metrics, Hyper-V host metadata, SQL Server collection
`root\virtualization\v2`	Hyper-V VM inventory
`root\StandardCIMV2`	Network collection
`root\Microsoft\SqlServer\ComputerManagement*`	SQL Server collection
`root\Microsoft\SqlServer\ReportServer\*`	SQL Server collection (SSRS)

For network collection, ensure these conditions are met:

- Allow network connectivity via ICMP
- Allow network connectivity via TCP port 135 + ephemeral TCP port range (49152 - 65535)
- Disable UAC
- Remote DCOM permissions are set up

- Create a dedicated service account with minimal required permissions
- WMI namespace permissions are set up for Windows accounts with namespaces: `\\root\\standardcimv2`, `MSFT_NetTCPConnection` class

For SQL Server collection, a Windows account (local or domain) belonging to the **Local Administrator Group** is required because of complex WMI objects permission requirements.

PowerShell Transcription logging

If Windows PowerShell Transcription logging is enabled on your target servers, every command that the discovery tool sends writes a transcript file to that server's local disk. The Group Policy key `HKLM:\SOFTWARE\Policies\Microsoft\Windows\PowerShell\Transcription` controls this behavior, and a value of `EnableTranscripting = 1` turns it on. By default, transcription writes transcript files to the service account's Documents folder. If the Group Policy sets an `OutputDirectory`, transcription writes them to that path instead.

To check whether transcription is enabled on a target server, run the following command on that server in PowerShell:

```
Get-ItemProperty -Path "HKLM:\SOFTWARE\Policies\Microsoft\Windows\PowerShell
\Transcription" -ErrorAction SilentlyContinue
```

If the command returns no output, transcription is not enabled by Group Policy. If it returns a property named `EnableTranscripting` with a value of 1, transcription is active.

The discovery tool communicates with Windows targets approximately every 15 seconds for network data collection, generating thousands of transcript files each day on each server. This can fill the target server's system disk quickly, causing OS or application failures on monitored machines.

> **Important**
>
> **Recommended mitigation:** Before enabling collection on Windows targets, ask your IT or domain administrator to create an exemption for the Windows service account used by the discovery tool from the **Turn on PowerShell Transcription** Group Policy setting (**Computer Configuration** > **Administrative Templates** > **Windows Components** > **Windows PowerShell**). The specific exemption mechanism depends on your domain's Group Policy structure. For User Configuration policies, security group filtering can target

the account directly. For Computer Configuration policies, OU-scoped targeting is the typical approach.

If transcription logging is a compliance requirement, implement periodic cleanup of the transcript output directory to prevent disk exhaustion. In the default configuration, transcripts are written to the service account's Documents folder (`%USERPROFILE%\Documents`).

Set up SSH

- The default port is 22. Custom ports are supported. The configured port must be open between the discovery tool and target servers.
- For SSH network collection to work properly, provide a user configured for passwordless sudo.
- Ensure that the following commands are available on target Linux servers (installed by default on most distributions): `ss` or `netstat` for network collection, and `lsblk`, `iostat`, `dmidecode`, `smartctl`, `top`, `ps`, `free`, `ip`, and `df` for OS metrics collection.

The discovery tool supports two authentication methods for SSH:

Option 1: Username and password

Provide the SSH username and password. This is the default authentication method.

Option 2: SSH private key

Provide the SSH username and a private key in PEM format. To use this option, choose **SSH Key** from the **Authentication type** dropdown when configuring SSH credentials. If the private key is encrypted with a passphrase, enter the passphrase in the optional **Key Passphrase** field.

The following key formats are supported:

- RSA
- ECDSA
- Ed25519
- OpenSSH format
- PKCS#8 format

Both authentication methods support auto-connect. Credentials are stored encrypted at rest.

Set up SNMP

- The default port is 161/UDP. Custom ports are supported. The configured port must be open between the discovery tool and target servers.
- For SNMP v2: Provide a read-only community string that can access TCP connection OIDs.
- For SNMP v3: Provide username/password and auth/privacy details with read-only permission that can access TCP connection OIDs

The discovery tool requires access to:

- `"1.3.6.1.2.1.6.13.1.1." (tcpConnState)`
- `"1.3.6.1.2.1.6.19.1.8." (tcpConnectionProcess)`
- `"1.3.6.1.2.1.25.4.2.1.2." (hrSWRunName)`

Configure Oracle database access

Configure Oracle database access to collect detailed Oracle database metadata directly through SQL connections. Collected metadata includes CDB and PDB topology, feature usage, and installed options. This data helps you plan Oracle database migrations more accurately. You can collect Oracle Database 12c Release 1 (12.1) and later through direct SQL connections. OS-level fallback detection works with all Oracle versions.

Configure Oracle credentials in the discovery tool

1. On the **Discovery tool** page, in the sidebar, choose **Database access**.
2. Choose **Add Oracle credential**.
3. Provide the following information:
   - **Friendly name** – A descriptive name for this credential (for example, `Oracle Production`).
   - **Port** – The Oracle listener port (default 1521).
   - **Service name** – The Oracle service name for the target database.
   - **Username** – The Oracle service account username.
   - **Password** – The Oracle service account password.
   - **Auto-connect** – Turn on this option to try the credential against all servers in your inventory. Turn off this option to manually assign the credential to specific servers.

4. To add more credentials (for example, for different Oracle environments), choose **Add Oracle credential** again.
5. Choose **Save and collect**. The discovery tool immediately starts a full Oracle collection round for all servers in your inventory.

Credential modes

When you configure Oracle credentials, you can choose between two modes:

- **Manual** – Pin a credential to a specific server. The discovery tool uses that credential exclusively for that server. If the connection fails, no fallback occurs. Fix the credential configuration to resolve the issue.
- **Auto-connect** – The discovery tool tries each auto-connect credential against every server in your inventory. When a credential succeeds for a server, the discovery tool uses that credential for all subsequent collection rounds.

Detection flow

When you configure Oracle credentials, the discovery tool first tries a direct SQL connection. If all database credentials fail, the tool falls back to OS-level detection through SSH or WinRM, so you can still discover Oracle installations without database access.

Configure SQL Server database access

Configure SQL Server database credentials to collect additional information directly from SQL Server, such as database count, high-availability topology, and cluster role. Without database credentials, the discovery tool collects SQL Server information using your Windows credentials only.

Prerequisites

- The SQL Server instance must be configured for SQL Server authentication (mixed-mode authentication). The discovery tool uses SQL authentication only. Windows domain logins are not supported.
- The discovery tool appliance must have TCP access to the SQL Server port (default 1433) on each target server. Named instances on dynamic ports are not supported — the discovery tool connects by port number only.

- Only Windows servers are attempted. Linux servers are skipped and report "OS not supported" even when a SQL Server credential is assigned.
- SQL Server connections in this release are unencrypted. Servers with "Force Encryption" enabled cannot be collected using database credentials.

Required SQL Server grants

Create a dedicated SQL Server login with the following read-only grants:

- `VIEW SERVER STATE` – Required to read high-availability topology data. A missing grant leaves the HA fields blank with a **Warning** status.
- `VIEW ANY DEFINITION` – Required to read Always On availability group information. A missing grant causes the collection round to show a **Warning** status, with a message indicating the grant is likely missing.
- `VIEW ANY DATABASE` – Required to enumerate databases. A missing grant silently undercounts databases without raising an error.

Run the following statements on the SQL Server instance to create the login. Replace `YourStrongPassword` with a strong, unique password:

```
USE master;
CREATE LOGIN atx_discovery WITH PASSWORD = 'YourStrongPassword';
GRANT VIEW SERVER STATE TO atx_discovery;
GRANT VIEW ANY DEFINITION TO atx_discovery;
GRANT VIEW ANY DATABASE TO atx_discovery;
```

Configure SQL Server credentials in the discovery tool

1. On the **Discovery tool** page, in the sidebar, choose **Database access**.
2. Choose **Add SQL Server credential**.
3. Provide the following information:
   - **Friendly name** – A descriptive name for this credential (for example, `SQL Server Production`).
   - **Port** – The SQL Server port (default 1433).
   - **Username** – The SQL Server login name.

- **Password** – The SQL Server login password.
- **Auto-connect** – Turn on to try the credential against all Windows servers in your inventory. Turn off to assign the credential to specific servers manually.

4. To add more credentials, choose **Add SQL Server credential** again.
5. Choose **Save and collect**. The discovery tool immediately starts a full SQL Server collection round for all Windows servers in your inventory.

You can monitor the collection result for each server on the **Discovered inventory** page.

SQL Server database credentials use the same auto-connect and manual assignment modes as Oracle credentials. For details, see the section called “Using Auto-Connect Feature With Caution”.

Updating the discovery tool

You can update the discovery tool using the built-in self-update feature, or manually by updating in place or redeploying a new appliance image.

Self-update

The discovery tool can update itself to a newer version without redeploying the appliance. A self-update preserves all collected data, discovery source configuration, and credentials. Collection is paused during the update and resumes automatically when it completes.

> **Important**
>
> Self-update requires outbound internet access from the discovery tool appliance to download updates. It is disabled by default. If your appliance cannot reach the internet, use the manual update method described in the section called “Manual update”.

Prerequisites

- Outbound HTTPS access to the discovery tool's release location, `s3.us-east-1.amazonaws.com/atx.discovery.collector.bundle/releases/latest/`. If outbound traffic is restricted, allow this location. Because the discovery tool uses path-style Amazon S3 URLs, a filter that matches only on host name cannot scope below `s3.us-east-1.amazonaws.com`; use a filter that matches on the full URL path to limit access to this location. Path-based filtering requires a firewall that performs TLS inspection; because the

discovery tool validates the server certificate against the appliance's trust store, that firewall's certificate authority must be trusted on the appliance, or self-update fails with a TLS error. No proxy configuration is supported.

- Sufficient free disk space. The discovery tool checks available space before starting and refuses the update with an "insufficient space" error if there is not enough room for the installer, a database snapshot, and a safety margin. Free up disk space if you see this error.

Enabling self-update

1. On the **Discovery tool** page, choose **Software updates** in the sidebar.
2. Turn on **Automatically check for software updates**.
3. Make sure the discovery tool appliance has outbound internet access. No additional credentials are required to download updates.

> **Note**
>
> If self-update is not enabled or your appliance cannot reach the internet, the discovery tool shows a reminder banner after 30 days to prompt you to update manually.

Updating to a new version

1. When a new version is available, an **Update available** banner appears on the dashboard. Choose **Review update**.
2. On the **Software updates** page, review the available version and release date.

   > **Note**
   >
   > For major version updates, **Update now** is disabled until you confirm three readiness checkboxes: that you have exported a backup, that you understand collection halts during the update, and that you will not power off the appliance during the update.

3. Choose **Update now**. The discovery tool stops collection, installs the new version, and restarts. The **Software updates** page shows progress through each stage.
4. When the update completes, sign in again. The discovery tool resumes collection automatically.

> **Note**
>
> If you see the banner "Update to version X failed. The appliance restored version Y" — the update failed but the discovery tool automatically restored the previous version. Your collected data is intact. Choose **Open update page** to retry.

> **Important**
>
> If the page shows **Recovery required** — the browser timed out waiting for the update to complete. Choose **Reload page**. If the discovery tool is still not reachable after reloading, contact support.

Automatic rollback

If the new version fails to start correctly after installation, the discovery tool automatically restores the previous version. Your previously collected data is intact. The discovery tool does not offer the failed version again until you clear it. To clear it and retry, choose **Retry update** on the **Software updates** page.

Manual update

If self-update is not available (for example, on appliances with no internet access, sometimes called air-gapped), you can update manually.

There are two ways to update:

- **Update in place** – Updates the application inside your existing VM with the Linux installer. Your collected inventory, credentials, and configuration are preserved. This method needs operating system access to the discovery tool VM. We recommend this method.
- **Redeploy from a new image** – Deploys a new VM from the latest OVA for VMware or VHD for Hyper-V. A new deployment starts with an empty database. Your existing data and configuration are not carried over unless you migrate them yourself.

Not every discovery tool version can update in place to every newer version. See the section called “Minimum version for updating” for the minimum version each update method supports.

Updating in place with the Linux installer

This method runs on the existing VM. It keeps your database, which holds your collected data, discovery sources, and credentials, and it keeps the database encryption key. Use this method when you want to keep your existing configuration and collected data.

> **Note**
>
> This method updates only the discovery tool application. It does not update anything on the VM host, such as the operating system version, the preconfigured shell aliases, or the networking and firewall rules. To get host updates, redeploy from a new image instead. For more information, see the section called “Redeploying from a new image”.

> **Important**
>
> Complete the backup step before you update, so that you can restore your data if the update does not finish.

To update the discovery tool in place

1. Access the discovery tool VM through your hypervisor console, or through SSH after you run `enablessh`. For more information, see the section called “Accessing the discovery tool VM”.
2. Get the latest installer script onto the VM, and make it executable. Choose one of the following:

   - If the VM has outbound internet access, download it directly:

     ```
     curl -O https://s3.us-east-1.amazonaws.com/atx.discovery.collector.bundle/
     releases/latest/AWS-Transform-discovery-tool.sh
     ```

   - If the VM has no outbound internet access (for example, an air-gapped appliance), download the installer script on a machine that does have internet access, and then transfer it over SSH into the home directory of the discovery tool service user on the VM. If SSH access is disabled, first run `enablessh` on the discovery tool VM (see the section called “Accessing the discovery tool VM”).

```
chmod +x AWS-Transform-discovery-tool.sh
```

3. Stop the discovery tool service so that the backup in the next step captures a consistent snapshot of the database.

```
sudo ./AWS-Transform-discovery-tool.sh stop
```

4. Back up the data directory so that you can restore it if you need to. This command reads the data directory from the service's own configuration, so it works whether you are on the standard OVA/VHD appliance or a standalone Linux-installer host.

```
SVC=discovery-tool.service
DATA_DIR=$(systemctl show -p Environment --value "$SVC" | tr ' ' '\n' | sed -n 's/
^DT_DATA_DIR=//p')
if [ -z "$DATA_DIR" ]; then
  SVC_USER=$(systemctl show -p User --value "$SVC")
  DATA_DIR="$(getent passwd "$SVC_USER" | cut -d: -f6)/.local/share/DiscoveryTool"
fi
echo "Discovery Tool data directory: $DATA_DIR"
if [ ! -d "$DATA_DIR" ]; then
  echo "ERROR: data dir not found - do NOT proceed"
else
  BACKUP="$HOME/discovery-tool-backup-$(date +%F).tar.gz"
  sudo tar czf "$BACKUP" -C "$(dirname "$DATA_DIR")" "$(basename "$DATA_DIR")"
  echo "Backup created: $BACKUP ($(du -h "$BACKUP" | cut -f1))"; tar tzf "$BACKUP"
 | head
fi
```

The installer keeps the database encryption key. It does not change the key during an update.

5. Run the installer. It detects the existing installation and updates it in place, keeping your data directory, encryption key, and service user. The `--install-deps` flag installs any system packages that the new version requires and that are not already present, using your OS package manager (`yum`, `dnf`, `apt-get`, or `zypper`).

> **Note**
>
> On an update, the installer always installs required system packages regardless of whether you pass `--install-deps`. This step only reaches out to your OS package

repository if a required package is missing; it does not need network access if your appliance already has the required packages installed, which is typically the case on an appliance that is already running. If a package manager call does need network access and your appliance has none, host the packages in a local or mirrored repository that your package manager is configured to use, or install the missing packages manually before you run the installer.

```
sudo ./AWS-Transform-discovery-tool.sh install --install-deps
```

6. Start the discovery tool service.

```
sudo ./AWS-Transform-discovery-tool.sh start
```

7. Verify the update. Open `https://ip_address:5000` in a web browser, sign in, and confirm that the version is updated and that your discovery sources, credentials, and inventory are present.

Redeploying from a new image

Use this method if you do not have operating system access to the VM, or if you prefer to deploy a new appliance. A redeployed VM starts with an empty database.

To redeploy the discovery tool from a new image

1. Download the latest image file: the OVA for VMware or the VHD for Hyper-V.
2. (Optional) Delete the previous discovery tool image file before you deploy the latest one.
3. Deploy the new version. For VMware, see the section called “Deploy on VMware”. For Hyper-V, see the section called “Deploy on Hyper-V”.
4. Configure your discovery sources, credentials, and server imports again on the new VM.

> **Note**
>
> If you want to deploy a new discovery tool and keep the data and configuration that you already collected, contact AWS Support before you begin.

Minimum version for each update method

Not every version of the discovery tool can update in place to every newer version. Some older versions renamed internal configuration keys without carrying old values forward, so updating in place from a version older than the minimum shown below can silently drop collected data or settings. The following table shows, for each update method, the oldest version you can update **from**.

Update method	Minimum version to update from	Notes
Redeploy from a new image	1.0.0	Always available. Starts with an empty database, so nothing to preserve.
Manual update (in place)	1.0.2325	Updating in place from a version older than 1.0.2325 can silently stop scheduled collectio n for one or more modules, or cause the discovery tool to fail to start, depending on your starting version. If you are on an older version, redeploy from a new image instead.
Self-update	1.0.2810	Self-update is available starting in 1.0.2810. Appliances on an earlier version do not have the self-update feature and must update manually or redeploy.

Revoking access

You can revoke access for each discovery source independently. When you revoke access for one source, data from other sources is not affected.

- **Revoking vCenter access** – Deletes vCenter credentials and VMware-collected data. Does not delete Hyper-V data, imported server data, or OS credentials.
- **Revoking Hyper-V access** – Deletes Hyper-V credentials and Hyper-V-collected data only.
- **Deleting imported servers** – Removes imported servers from inventory. Downstream collection data (network, database) that was collected from those servers is retained.

Required permissions for the discovery tool

The discovery tool connects to your infrastructure using several protocols. Each protocol and collection module requires specific account permissions. This section describes the minimum permissions needed for complete data collection, and what data you lose if certain permissions are unavailable.

VMware vCenter

The discovery tool connects to VMware vCenter Server on port 443 (HTTPS) and performs read-only operations. No changes are made to your vCenter environment.

Minimum required role: Read-Only, assigned at the vCenter root level.

The Read-Only role provides all the access the discovery tool needs to collect VM inventory (name, UUID, CPU, memory, disk, network, power state, guest OS) and performance metrics (CPU utilization, memory utilization, disk IOPS and throughput).

Recommendation: Create a dedicated vCenter user with the built-in Read-Only role at the root datacenter level.

Hyper-V hosts (Windows)

The discovery tool connects to Hyper-V hosts over WinRM (port 5985 for HTTP, port 5986 for HTTPS) and runs PowerShell commands to collect VM inventory, host metadata, and storage performance.

Minimum required group memberships on each Hyper-V host:

Windows group	Why it's needed
Remote Management Users	Baseline WinRM remote access
Hyper-V Administrators	VM inventory collection (VM list, disk, network, memory, guest OS)
Performance Monitor Users	Storage I/O performance counters (read/write IOPS and throughput per VM)

The account also needs WMI read access to the `root\cimv2` namespace for host OS version and hardware UUID. This access is granted by default for local administrators but must be configured explicitly for non-administrator accounts.

Recommendation: Create a dedicated domain service account and add it to the Remote Management Users, Hyper-V Administrators, and Performance Monitor Users groups on each Hyper-V host. Avoid using domain administrator accounts.

Linux servers (SSH)

The discovery tool connects to Linux servers over SSH (port 22) to collect OS metrics, network connections, and server inventory. Most data collection runs as a regular user. A small number of commands attempt `sudo` and automatically fall back if sudo is unavailable.

Minimum required access: An SSH user account that can log in to the target server.

Recommended access: The same SSH user configured with passwordless sudo for complete data collection.

What requires sudo and what happens without it:

Data collected	Sudo needed?	What you lose without sudo
Server name, OS, CPU, memory, IP, disk count	No	Nothing — collected as regular user
CPU, memory, and network utilization	No	Nothing — collected as regular user
Disk IOPS, throughput, and space	No	Nothing — collected as regular user
Network interface configuration	No	Nothing — collected as regular user
Running processes (name, PID, command)	No	Nothing — collected as regular user
Server UUID and SMBIOS UUID	Yes	UUID fields are empty in the export
Hardware manufacturer (physical vs. virtual detection)	Yes	Resource type detection is less accurate on older distributions

Data collected	Sudo needed?	What you lose without sudo
LVM logical volume detection	Yes	LVM volumes not detected; volume type may show as "Unknown"
Network connections with process name and PID	Yes	Connections are still collected, but without process attribution (PID and process name columns are empty)
Oracle Database OS-level detection (fallback when Oracle credentials are not configured)	Yes	Basic Oracle presence detection (oratab, process monitor) works without sudo. Detailed collection (opatch, lsnrctl, ASM, Grid Infrastructure) requires passwordless `sudo -u oracle` access to the Oracle home binaries.

> **Note**
>
> The discovery tool never fails entirely due to lack of sudo. It collects what it can and reports partial results. However, for the most complete data — especially network dependency mapping with process names — we recommend passwordless sudo.

Required utilities on target servers: The following commands should be available (installed by default on most Linux distributions): `ss` or `netstat`, `lsblk`, `top`, `ps`, `free`, `ip`, `df`, `hostname`, `cat`, `nproc`, and `grep`. Optional for additional data: `iostat` (disk I/O detail), `dmidecode` (hardware UUID), `smartctl` (disk interface type), `lvdisplay` (LVM detection).

Windows servers (WinRM) — OS metrics

The discovery tool connects to Windows servers over WinRM to collect OS metrics (server inventory, performance, storage, network interfaces, and running processes).

Minimum required access: A WinRM-enabled user account with remote access to the target server.

Recommended access: A user in the local Administrators group for complete data collection.

Permission level	What it enables	What you lose without it
Remote Management Users (WinRM access)	All Windows data collection	No data collected from the server
WMI read access to `root\cimv2`	Server name, OS version, memory, UUID, BIOS serial, disk space	Server inventory fields are empty
Performance Monitor Users	CPU utilization, network throughput, disk IOPS and throughput	Performance metrics not collected
Local Administrator	Running process owner names	Process list collected but the user/owner column is empty

Note

Local Administrator membership implicitly grants Performance Monitor Users access and WMI read access, so it satisfies all requirements above.

Note

Oracle Database OS-level fallback detection on Windows uses registry queries and `Get-Service` and `Get-Process` commands when Oracle credentials are not configured. These commands require only standard Remote Management Users access. No additional permissions beyond OS metrics requirements are needed.

Windows servers (WinRM) — SQL Server collection

SQL Server collection discovers SQL Server instances, Reporting Services (SSRS), and Integration Services (SSIS) on Windows servers.

Recommended access: A user in the local Administrators group on each target Windows server.

Local Administrator is recommended because SQL Server discovery queries multiple WMI namespaces and requires elevated access for some operations:

What is discovered	Permission needed	What you lose without it
SQL Server Database Engine instances (version, edition, status)	WMI read access to `root\Microsoft\SqlServer\ComputerManagement*` namespaces	SQL Server instances not discovered
SQL Server Reporting Services (SSRS)	WMI read access to `root\Microsoft\SqlServer\ReportServer` namespaces	SSRS components not discovered
SSRS URL and port configuration	Elevated privileges (Local Admin)	SSRS URL reservation details missing
SQL Server Integration Services (SSIS)	Registry read access (HKLM)	SSIS version and edition missing
Port-to-service association	Access to TCP listener enumeration	Cannot associate listening ports with SQL Server services

Note

SQL Server collection is Windows-only. The discovery tool skips Linux servers for SQL Server discovery.

SQL Server direct database collection (optional)

SQL Server direct database collection requires a dedicated login with specific server-scoped grants. For setup instructions and the required grants, see the section called “Configure SQL Server database access”.

Oracle Database (SQL)

Oracle Database collection connects directly to Oracle instances on port 1521 (or a custom port) to collect database metadata. The discovery tool does not require DBA or SYSDBA privileges.

Minimum required access: A read-only Oracle service account with the SELECT_CATALOG_ROLE grant.

The discovery tool does not access Diagnostics Pack or Tuning Pack views, so no additional Oracle license is required for data collection.

SQL to create the account:

```
CREATE USER discovery_user IDENTIFIED BY <password>;
GRANT CREATE SESSION TO discovery_user;
GRANT SELECT_CATALOG_ROLE TO discovery_user;
```

OS-level fallback: If database credentials are not configured or the connection fails, the discovery tool uses existing SSH or WinRM OS credentials to detect Oracle installations. No additional Oracle-specific OS permissions are needed beyond what the OS metrics module requires.

Network collection

Network collection uses different protocols depending on the server's operating system:

Server OS	Protocol	Permissions needed
Linux	SSH	Regular user for connection data. Passwordless sudo recommended for process-level details (PID and process name).
Linux	SNMPv2	A read-only community string with access to the TCP MIB (tcpConnState, tcpConnectionProcess) and Host Resources MIB (hrSWRunName).
Linux	SNMPv3	A USM user with read access to the same MIBs as SNMPv2. Supports noAuthNoPriv, authNoPriv, and authPriv security levels.
Windows	WinRM	WMI read access to the `root\StandardCIMV2` namespace (MSFT_NetTCPConnection class).

Quick reference: Minimum permissions by use case

Use case	Account type	Minimum permissions
VMware VM discovery	vCenter user	Read-Only role at root datacenter level
Hyper-V VM discovery	Windows domain or local account	Remote Management Users + Hyper-V Administrators + Performance Monitor Users on each host
Linux OS metrics — full data	SSH user	Passwordless sudo
Linux OS metrics — partial data	SSH user	Regular user (no sudo). UUID, manufacturer, LVM, and network process info will be missing.
Windows OS metrics — full data	WinRM user	Local Administrator
Windows OS metrics — basic data	WinRM user	Remote Management Users + WMI read access to `root\cimv2`
SQL Server database discovery	WinRM user	Local Administrator
SQL Server direct database collection (optional)	SQL Server login	`VIEW SERVER STATE`, `VIEW ANY DEFINITION` , `VIEW ANY DATABASE`
Oracle database discovery (SQL)	Oracle service account	SELECT_CATALOG_ROLE (read-only, no DBA or SYSDBA)
Network collection — Linux (SSH)	SSH user	Passwordless sudo for process-level data; regular user for connection data only
Network collection — Linux (SNMP)	SNMP community string or USM user	Read access to TCP and Host Resources MIBs

Use case	Account type	Minimum permissions
Network collection — Windows	WinRM user	WMI read access to `root\StandardCIMV2`

Data collection

Discovery tool collection schedule

After your initial discovery collection, the discovery tool continues to run on a staggered schedule to avoid resource contention:

- VMware discovery – every hour (at :00 UTC)
- Hyper-V discovery – every hour (at :20 UTC)

The discovery tool also collects OS metrics through the following independent modules, each with its own staggered schedule:

- Network metrics – every 15 seconds, might be less frequent for large environments
- Server performance metrics – every 10 minutes (at :03, :13, :23, :33, :43, :53 UTC)
- Storage performance metrics – every 10 minutes (at :07, :17, :27, :37, :47, :57 UTC)
- Running processes – hourly (at :40 UTC)
- Server provisioning data – daily (at 00:05 UTC)
- Storage provisioning data – daily (at 00:35 UTC)
- Network interfaces – daily (at 01:05 UTC)
- SQL Server discovery – daily (at 03:05 UTC)
- Oracle database discovery – daily (at 05:05 UTC)

You can independently start, stop, or trigger each OS metrics module by using **Collect data now**.

To manually run a collection, from the **Actions** menu choose:

- **Start** – Enables the discovery module.
- **Stop** – Disables the discovery module.

- **Collect data now** – Starts discovery immediately. Use this option, for example, after you make a change in your network.

These actions apply per module. You can control OS metrics modules individually.

OS data collection attempts

When a new server is discovered, the discovery tool attempts each configured credential for each IP address and the hostname. After the discovery tool finds a valid credential, it continues to use that credential unless you add a new credential.

After a collection failure, the discovery tool attempts to collect networking data for a server after 3 minutes, 30 minutes, 2 hours, and then 6 hours. After 4 failed attempts, the discovery tool continues to try all configured credentials once every 6 hours.

How average, p95, and peak statistics are calculated

This section describes the average (`_avg`), 95th percentile (`_p95`), and peak (`_peak`) statistics in the *OS-collected* performance data: the columns in `server_performance_metrics.csv` and `server_storage_performance.csv`, and the `disk_*` statistic columns in `server_inventory.csv`. The VMware- and Hyper-V-sourced performance columns (for example, the MPA fields `avgCpuUsagePctDec` and `avgRamUtlPctDec`) are averaged on a different basis and carry no p95 or peak value.

Window – The statistics cover the entire export date range: the range you select at export time, up to the 30-day retention limit, or the tool's uptime if it is shorter.

Sampling – The discovery tool records one sample per metric per collection cycle. CPU, memory, and network samples come from the server performance module, and disk IOPS and throughput samples come from the storage performance module. Both modules run every 10 minutes, which is about 144 samples per metric per day, or about 4,320 over 30 days. For the full schedule, see the section called “Collection schedule”.

Computation – Samples accumulate each day during collection. At export, all samples in the selected window are pooled, and the statistics are computed over that full set:

- **Average** – the mean of the pooled samples.
- **p95** – the 95th percentile, computed with the inclusive method (equivalent to the Excel `PERCENTILE.INC` function). It never exceeds the maximum sample observed.

- **Peak** – for CPU, memory, network, and the per-volume rows in `server_storage_performance.csv`, the peak is the highest single sample observed. For the server-level `disk_*` columns in `server_inventory.csv`, the peak is the sum of each volume's peak.

The discovery tool computes the statistics from all samples in the window. A shorter window contains fewer samples. A 2-day export therefore produces different average and p95 values than a 30-day export.

Discovered inventory

After you configure a discovery source, the **Number of discovered servers** value in the **Discovery tool status** frame begins to increment. The discovery status for the configured source changes to **Enabled** in the **Collection module** frame. The inventory page shows servers from all configured sources: VMware VMs, Hyper-V VMs, and imported servers. Each server shows its source and collection status per module.

Navigate to the **Discovered inventory** page to see the servers that the discovery tool has found. From this page, choose **Download inventory** to download a ZIP file (`discovery_tool_export.zip`) that contains up to 30 days of collected data, including MPA files for all configured sources, performance utilization data, database information, and server-to-server communication information.

You can download the ZIP file while the discovery tool continues to work, and obtain partial results. Upload this file to Migration assessment to obtain a business case for migration.

Export options

When exporting data, you can customize the export with the following options:

Date range

Select a start date and end date to export only data collected within that time period. Both dates are inclusive. The maximum date range is 30 days.

> **Note**
>
> The discovery tool stores up to 30 days of collected data. If you need data spanning more than 30 days, run incremental exports every 30 days to capture all data.

Module selection

Choose which data modules to include in the export. You can export all modules or select specific ones:

Module	Description
VMware data	Virtual machine inventory from vCenter servers
Hyper-V data	Virtual machine inventory from Hyper-V hosts
Network data	Network connections between servers
Server inventory	Server hardware and OS information
Server performance metrics	CPU, memory, and network utilization
Server storage performance	Disk IOPS and throughput
Storage config	Disk and volume configuration
Network interfaces	Network adapter details
Process metrics	Running processes
SQL Server data	SQL Server database inventory
Oracle Database data	Oracle database inventory – CDBs, PDBs, features, options, and components

If you don't select any modules, the discovery tool exports all available data.

Understanding the export summary

Every export ZIP includes an `export_summary.json` file at its root that records the result of the export. Use it to confirm that an export completed, or to find what went wrong when it did not. The file includes the following:

- `overall_status` – `complete` when all requested data was exported, `partial` when data was exported but some records were skipped or a module reported errors, or `failed` when no data was exported because of errors.

- `start_date` and `end_date` – the date range that the export covered.
- `modules` – one entry for each exported data type (for example, server inventory, performance metrics, storage, and network connections), each with its own status, counts of what was exported, and a list of any errors.
- `errors` – entries for data that could not be exported, each identifying the affected server, the date, and the reason.

If `overall_status` is `partial` or `failed`, review each module's status and its errors to see which servers or data types were affected and why.

Data points collected

The discovery tool gathers comprehensive data across VMware, Hyper-V, OS metrics, database, and network components. The following sections detail the specific data points collected for each component.

VMware data collection

This table describes the VMware virtual machine information collected by the discovery tool:

Name	Type	Category	Sample value
vm_name	String	VM Info	"w2k22-snmpd-v2-en-us-mssql-2022-testcase4-1"
vm_id	String	VM Info	"vm-30920"
vm_uuid	String	VM Info	"4201ecf8-cc44-ee7 e-01da-34dfb2acf6c0"
powerstate	String	VM Info	"poweredOn"
host	String	VM Info	"esxi-70-node1.testlab.local"
primary_ip_address	String	VM Info	"192.168.0.52"
cpus	Integer	VM Info	2
memory	Integer	VM Info	4096

Name	Type	Category	Sample value
total_disk_capacity_mib	Integer	VM Info	32768
os_according_to_the_configuration_file	String	VM Info	"Microsoft Windows Server 2016 or later (64-bit)"
max_cpu_usage_pct_dec	Float	VM Performance	79.33
avg_cpu_usage_pct_dec	Float	VM Performance	45.06
max_ram_usage_pct_dec	Float	VM Performance	63.99
avg_ram_utl_pct_dec	Float	VM Performance	29.27

Hyper-V data collection

This table describes the Hyper-V virtual machine information collected by the discovery tool:

Name	Type	Category	Sample value
vm_name	String	VM Info	"win2022-hyperv-test-01"
vm_id	String	VM Info	"a1b2c3d4-e5f6-7890-abcd-ef1234567890"
powerstate	String	VM Info	"Running"
cpus	Integer	VM Info	4
memory_mb	Integer	VM Info	8192
disk_paths	String	Disk	"C:\\VMs\\disk1.vhdx"
disk_size_gb	Float	Disk	127.0

Name	Type	Category	Sample value
network_adapters	String	Network	"00:15:5D:01:02:03"
ip_addresses	String	Network	"10.0.1.50"
host_name	String	Host	"hyperv-host-01.example.com"
host_os_version	String	Host	"Windows Server 2022 Datacenter"
cluster_name	String	Host	"FailoverCluster01"
hypervisor	String	VM Info	"Hyper-V"

Imported server data

Imported servers are not auto-discovered. They are imported through a CSV file. The discovery tool does not collect hypervisor-level data for imported servers. Instead, it collects database, network, and OS metrics data by using the OS credentials associated with each server during import.

Discovery tool's OS-related data

The discovery tool collects server inventory, performance, storage, network interface, and process data through SSH (Linux) and WinRM (Windows). The following tables describe the data points collected.

Server inventory (server_inventory.csv)

Combines server provisioning (hardware and OS configuration) with aggregated storage performance. Collected every 24 hours.

Disk throughput columns end in `_mibps` and are reported in mebibytes per second (MiBps). The `_p95` columns hold the 95th percentile of the samples in the export window.

Name	Type	Category	Sample value
server_id	String	Server Info	"vm-web-server-01"

Name	Type	Category	Sample value
server_name	String	Server Info	"web-server-01"
resource_type	String	Server Info	"virtual_machine"
power_state	String	Server Info	"Running"
os_type	String	Server Info	"Linux"
os_name	String	Server Info	"Amazon Linux"
os_version	String	Server Info	"2023"
primary_hostname	String	Server Info	"web-server-01.example.com"
primary_ip_address	String	Server Info	"10.0.2.101"
netmask	String	Server Info	"255.255.255.0"
total_num_network_cards	Integer	Server Info	2
total_num_disks	Integer	Server Info	1
cpu_count	Integer	Server Info	4
total_memory_gb	Float	Server Info	15.88
server_uuid	String	Server Info	"4201ecf8-cc44-ee7 e-01da-34dfb2acf6c0"
smbios_uuid	String	Server Info	"4201ecf8-cc44-ee7 e-01da-34dfb2acf6c0"
cluster_name	String	Server Info	"production-cluster-01"
hypervisor_object_id	String	Server Info	"vm-30920"
hypervisor_type	String	Server Info	"VMware"
hypervisor_version	String	Server Info	"8.0.0"

Name	Type	Category	Sample value
hypervisor_hostname	String	Server Info	"esxi-node1.example.com"
hypervisor_host_id	String	Server Info	"host-1234"
hypervisor_id	String	Server Info	"4201ecf8-cc44-ee7 e-01da-34dfb2acf6c0"
disk_read_iops_avg	Float	Storage Performance	12.5
disk_read_iops_p95	Float	Storage Performance	198.0
disk_read_iops_peak	Float	Storage Performance	245.0
disk_write_iops_avg	Float	Storage Performance	8.3
disk_write_iops_p95	Float	Storage Performance	142.0
disk_write_iops_peak	Float	Storage Performance	180.0
disk_total_iops_avg	Float	Storage Performance	20.8
disk_total_iops_p95	Float	Storage Performance	340.0
disk_total_iops_peak	Float	Storage Performance	425.0
disk_read_throughp ut_avg_mibps	Float	Storage Performance	1.2

Name	Type	Category	Sample value
disk_read_throughput_p95_mibps	Float	Storage Performance	19.8
disk_read_throughput_peak_mibps	Float	Storage Performance	24.5
disk_write_throughput_avg_mibps	Float	Storage Performance	0.8
disk_write_throughput_p95_mibps	Float	Storage Performance	14.2
disk_write_throughput_peak_mibps	Float	Storage Performance	18.0
disk_total_throughput_avg_mibps	Float	Storage Performance	2.0
disk_total_throughput_p95_mibps	Float	Storage Performance	34.0
disk_total_throughput_peak_mibps	Float	Storage Performance	42.5

Server performance metrics (server_performance_metrics.csv)

CPU, memory, and network throughput utilization. Sampled every 10 minutes, aggregated over 30 days.

The _p95 columns hold the 95th percentile of the samples in the export window. Network throughput columns end in `_mbps` and are reported in megabits per second (Mbps).

Name	Type	Category	Sample value
server_id	String	Server Info	"vm-web-server-01"
data_source	String	Server Info	"OS"

Name	Type	Category	Sample value
cpu_utilization_avg_pct	Float	CPU	45.06
cpu_utilization_p95_pct	Float	CPU	72.14
cpu_utilization_peak_pct	Float	CPU	79.33
cpu_count	Integer	CPU	4
memory_total_gb	Float	Memory	15.88
memory_utilization_avg_pct	Float	Memory	29.27
memory_utilization_p95_pct	Float	Memory	58.42
memory_utilization_peak_pct	Float	Memory	63.99
network_in_avg_mbps	Float	Network	0.52
network_in_p95_mbps	Float	Network	9.84
network_in_peak_mbps	Float	Network	12.3
network_out_avg_mbps	Float	Network	0.31
network_out_p95_mbps	Float	Network	6.95
network_out_peak_mbps	Float	Network	8.7
network_total_avg_mbps	Float	Network	0.83
network_total_p95_mbps	Float	Network	16.79
network_total_peak_mbps	Float	Network	21.0

Storage performance (server_storage_performance.csv)

Per-volume disk I/O and space utilization. Sampled every 10 minutes, aggregated over 30 days.

Disk throughput columns end in `_mibps` and are reported in mebibytes per second (MiBps). The _p95 columns hold the 95th percentile of the samples in the export window.

Name	Type	Category	Sample value
server_id	String	Server Info	"vm-web-server-01"
data_source	String	Server Info	"OS"
disk_volume_id	String	Volume Info	"/dev/nvme0n1p1"
disk_mount_point	String	Volume Info	"/"
file_system	String	Volume Info	"xfs"
disk_total_gb	Float	Disk Space	30.0
disk_used_gb	Float	Disk Space	12.5
disk_free_gb	Float	Disk Space	17.5
disk_read_iops_avg	Float	Disk I/O	12.5
disk_read_iops_p95	Float	Disk I/O	198.0
disk_read_iops_peak	Float	Disk I/O	245.0
disk_write_iops_avg	Float	Disk I/O	8.3
disk_write_iops_p95	Float	Disk I/O	142.0
disk_write_iops_peak	Float	Disk I/O	180.0
disk_total_iops_avg	Float	Disk I/O	20.8
disk_total_iops_p95	Float	Disk I/O	340.0
disk_total_iops_peak	Float	Disk I/O	425.0
disk_read_throughput_avg_mibps	Float	Disk Throughput	1.2

Name	Type	Category	Sample value
disk_read_throughput_p95_mibps	Float	Disk Throughput	19.8
disk_read_throughput_peak_mibps	Float	Disk Throughput	24.5
disk_write_throughput_avg_mibps	Float	Disk Throughput	0.8
disk_write_throughput_p95_mibps	Float	Disk Throughput	14.2
disk_write_throughput_peak_mibps	Float	Disk Throughput	18.0
disk_total_throughput_avg_mibps	Float	Disk Throughput	2.0
disk_total_throughput_p95_mibps	Float	Disk Throughput	34.0
disk_total_throughput_peak_mibps	Float	Disk Throughput	42.5

Unmapped storage devices (unmapped_storage_devices.csv)

The discovery tool cannot always match a storage device to a mounted file system, and some devices report no usable capacity. The discovery tool does not omit these devices or report their capacity as zero. It writes them to `unmapped_storage_devices.csv` instead, and still reports performance data for them. Only the capacity columns are blank.

This file has the same columns as `server_storage_performance.csv`, in the same order, plus a `reason` column. The `reason` column can hold the following values.

Value	Meaning
`no_mounted_filesystem_match`	The discovery tool cannot match the device to a mounted file system.
`mounted_but_capacity_unavailable`	The device is mounted, but it does not report usable capacity.

When you export the storage performance module, the export includes both files.

> **Note**
>
> When you upload the export to AWS Transform, the assessments agent ignores unmapped storage devices unless you specifically instruct it to include them.

Storage configuration (storage_config.csv)

Physical disk hardware details. Collected every 24 hours.

Name	Type	Category	Sample value
server_id	String	Server Info	"vm-web-server-01"
disk_controller_id	String	Disk Info	"/dev/sda"
vmdk_vhd_file_name	String	Disk Info	"web-server-01.vmdk"
disk_volume_type	String	Disk Info	"Virtual"
disk_provisioned_gb	Float	Disk Info	30.0
disk_device_type	String	Disk Info	"SCSI HDD"
disk_interface_type	String	Disk Info	"SCSI"
disk_protocol	String	Disk Info	"LSI Logic SAS"

Network interfaces (network_interfaces.csv)

Network adapter configuration. Collected every 24 hours.

Name	Type	Category	Sample value
server_id	String	Server Info	"vm-web-server-01"
interface_name	String	Interface Info	"eth0"
interface_index	Integer	Interface Info	2
mac_address	String	Interface Info	"0A:1B:2C:3D:4E:5F"
adapter_type	String	Interface Info	"vmxnet3"
virtual_network_name	String	Interface Info	"VM Network"
virtual_network_id	String	Interface Info	"dvportgroup-1234"
virtual_switch	String	Interface Info	"vSwitch0"
ipv4_address	String	IP Config	"10.0.2.101"
ipv4_subnet_mask	String	IP Config	"255.255.255.0"
ipv4_gateway	String	IP Config	"10.0.2.1"
ipv6_address	String	IP Config	"fe80::a1b:2cff:fe3d:4e5f"
ipv6_prefix_length	Integer	IP Config	64
ipv6_gateway	String	IP Config	"fe80::1"
dns_servers	String	IP Config	"10.0.0.2"
dhcp_enabled	Boolean	IP Config	false
interface_status	String	Interface Info	"Up"
vlan_id	Integer	Interface Info	100

Name	Type	Category	Sample value
is_primary	Boolean	Interface Info	true

Running processes (process_metrics.csv)

Snapshot of running processes. Collected every hour, deduplicated over 30 days.

Name	Type	Category	Sample value
server_id	String	Server Info	"vm-web-server-01"
process_name	String	Process Info	"sshd"
process_id	Integer	Process Info	1234
process_command_line	String	Process Info	"/usr/sbin/sshd -D"
process_user	String	Process Info	"root"

Network collection

The Network collection module helps you discover dependencies among servers in your on-premises data center. This network data accelerates your migration planning by providing visibility into how applications communicate across servers.

This module collects network data for servers from all configured sources, including VMware, Hyper-V, and imported servers. It uses WinRM to collect data from Windows servers and uses SSH, SNMPv2, and SNMPv3 to collect data from Linux servers.

Network data collection

The Network collection module captures TCP IPv4 connections in ESTABLISHED or TIME_WAIT state between servers in your discovered inventory. A connection appears in the output only when both the source and target IP addresses belong to servers that the discovery tool has discovered or that you have imported. Connections to or from IP addresses outside your inventory — such as external services, cloud endpoints, or servers not yet added to the discovery tool — are not included.

This design focuses the network data on server-to-server dependencies within your environment, which is the information needed for application dependency mapping and migration wave planning.

These data points are collected for each connection:

Name	Type	Category	Sample value
Source IP	String	Connection	"192.168.1.10"
Source port	Integer	Connection	49152
Source process ID	Integer	Process	1234
Source process name	String	Process	"java"
Target IP	String	Connection	"192.168.1.20"
Target port	Integer	Connection	5432
Target process ID	Integer	Process	5678
Target process name	String	Process	"postgres"
State	String	Connection	"ESTABLISHED"
Transport protocol	String	Connection	"TCP"
IP version	String	Connection	"IPv4"
Count	Integer	Connection	42

Tip

To maximize the completeness of your network dependency map, configure all discovery sources (VMware, Hyper-V, and server CSV import) and add OS credentials before reviewing network data. The more servers in your inventory, the more connections the network module can capture.

Private address network collection

By default, the Network collection module only captures connections where both endpoints are servers in your discovered inventory. You can enable private address collection to also capture connections to and from RFC 1918 private IP addresses (10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16) that are not in your inventory.

To start private address collection

1. On the **Collector configuration** page, locate the **Collection modules** section.
2. Find **Network connections discovery** under **Application discovery**.
3. Open the **Actions** dropdown.
4. Choose **Start private address collection**.
5. If the module status was **Enabled**, you can see it change to **Enabled · Private Address on**.

To stop private address collection, open the **Actions** dropdown and choose **Stop private address collection**. Previously collected private address data is retained even after stopping.

Private address connections appear only in the full CSV export (inside the ZIP file), not in the MPA CSV files. If an IP address belongs to a server already in your inventory, it is always identified by its discovered server ID regardless of this setting.

This setting persists across restarts. You can start or stop private address collection at any time. Previously collected private address data is exported regardless of the current setting.

SQL Server collection

The SQL Server collection module gathers SQL Server information from Windows servers across all configured sources, including VMware, Hyper-V, and imported servers. The module uses WinRM to remotely connect to each Windows server and run PowerShell queries. It collects information about all installed SQL Server services (components) by using WMI namespaces, registry, and file properties.

A SQL Server component is a specific service or feature instance installed as part of a SQL Server deployment on a Windows server. The discovery tool collects Database Engine, Analysis Services, Reporting Services, and Integration Services.

SQL Server direct database collection

When you configure SQL Server database credentials, the discovery tool connects directly to each SQL Server instance and collects additional data that is not available through Windows credentials alone.

This additional data helps you assess your SQL Server instances for migration planning and right-sizing:

- **Database count** – The number of user databases on the instance.
- **Always On AG (Y/N)** – Whether the instance participates in an Always On availability group. Requires SQL Server 2012 or later.
- **Availability group name** – The name of the Always On availability group, if the instance participates in one. Requires SQL Server 2012 or later.
- **Cluster role** – The instance's role in a high-availability configuration (`PRIMARY`, `SECONDARY`, `RESOLVING`, or `STANDALONE`). Requires SQL Server 2012 or later. Cluster role reflects Always On availability group membership only. A failover cluster instance that is not in an availability group shows `STANDALONE`. To identify failover cluster instances, check the Is Clustered field (`Cluster/Oracle RAC (Y/N)` in the MPA file).
- **Read-only replica** – Whether the instance is a readable secondary replica. Requires SQL Server 2012 or later.

> **Note**
>
> Edition is also collected by the Windows-level path and already appears in the MPA export as "Source Engine Edition". The five fields listed above are the database-level additions that require a direct SQL Server connection.

When database credentials are configured

When you configure SQL Server credentials and a connection succeeds, the direct database path runs for that round instead of the Windows-level path. The five additional fields are collected alongside the standard Windows-level service inventory from an earlier or later round.

Windows-level fallback

If you have not configured SQL Server database credentials, or if an auto-connect credential fails to connect to the instance (for example, wrong password or network error), the discovery tool collects

SQL Server information using your Windows credentials. It discovers installed SQL Server services, versions, and configuration without direct database access. For these servers, the five database-level fields are blank in the export.

If you assign a SQL Server credential to a specific server (for example, through CSV import), the discovery tool uses only that credential for that server and does not fall back to Windows-level detection if the connection fails. The server shows an error status on the **Discovered inventory** page.

> **Note**
>
> The five database-level fields (Database Count, Always On AG (Y/N), Availability Group Name, Cluster Role, and Read-Only Replica) appear only in `sql_server_data_full.csv` inside the export ZIP file. They are not included in the MPA files. Servers collected through Windows-level detection have these fields blank.

> **Note**
>
> SQL Server database connections in this release are unencrypted. Ensure that your network environment is appropriately secured when enabling SQL Server database credentials.

For information about required permissions and how to configure SQL Server database credentials, see the section called “Configure SQL Server database access”.

SQL Server data collection

The SQL Server collection module gathers SQL Server component information. This table describes key data points collected:

Name	Type	Category	Sample value	Source
Engine Type	String	Component	sql_server	WinRM
Is Engine Component	Boolean	Component	Y	WinRM
Status	String	Service	Running, Stopped, StartPending	WinRM, SQL

Name	Type	Category	Sample value	Source
Version	String	Service	2015.131.5026.0	WinRM, SQL
Edition	String	Service	Developer Edition (64-bit)	WinRM, SQL
SQL Service Name	String	Service	MsDtsServer130, Mssql	WinRM, SQL
SQL Service Type	String	Service	SQL Server service, Integration Services service	WinRM, SQL
Instance Name	String	Instance	MSSQLSERVER	WinRM, SQL
Display Name	String	Service	SQL Server (MSSQLSERVER2017)	WinRM
Start Mode	String	Service	Automatic, Manual, Disabled	WinRM
Service Account Name	String	Service	NT Service/MsDtsServer130	WinRM
Is Clustered	Boolean	Configuration	N	WinRM, SQL
Database Count	Integer	Database	5	SQL
Always On AG (Y/N)	String	High Availability	Y	SQL (2012+)
Availability Group Name	String	High Availability	AG-Production	SQL (2012+)
Cluster Role	String	High Availability	PRIMARY, SECONDARY, RESOLVING, STANDALONE	SQL (2012+)

Name	Type	Category	Sample value	Source
Read-Only Replica (Y/N)	String	High Availability	N	SQL (2012+)

Note

WinRM = collected over WinRM without database credentials. SQL = requires SQL Server database credentials. (2012+) = also requires SQL Server 2012 or later; applies only to the Always On AG (Y/N), Availability Group Name, Cluster Role, and Read-Only Replica fields. Full format includes all service types. MPA format includes only database engine components and does not include SQL-only fields.

Oracle Database collection

With Oracle Database collection, you can discover Oracle database instances across all configured sources, including VMware, Hyper-V, and imported servers. The module collects the following data:

- Container database (CDB) and pluggable database (PDB) enumeration
- Version and edition
- Installed options
- Feature usage statistics
- Component inventory (DBA_REGISTRY)
- Datafile sizing
- Topology flags (RAC, Data Guard, and multitenant architecture)

Database-connected (SQL)

When you configure Oracle credentials, the discovery tool connects directly to the Oracle database using the read-only service account. This provides full SQL-level collection including PDB details, feature usage, and installed options. All CSV files contain complete data.

OS-level fallback

If you have not configured database credentials, or if the connection fails, the discovery tool uses SSH or WinRM to detect Oracle installations. It discovers Oracle homes, listeners, patches, version, and edition without database access. For OS-detected hosts, the exported CDB CSV contains instance name, host name, version, and edition. The PDB, features, options, and components CSV files contain no rows.

The Oracle collection produces the following CSV files in the export ZIP. Each table describes the data points collected per file.

CDB data (oracle_data_cdbs_full.csv)

One row per CDB instance. This table describes the CDB data points collected:

Name	Type	Category	Sample value	Source
Instance Name	String	Identity	"ORCL"	SQL, OS
Host Name	String	Identity	"oracledb01.example.com"	SQL, OS
DB Name	String	Identity	"ORCL"	SQL
DB Unique Name	String	Identity	"ORCL_PRIMARY"	SQL
DBID	Integer	Identity	1234567890	SQL
Version	String	Version	"19.0.0.0"	SQL, OS
Version Full	String	Version	"19.21.0.0.0"	SQL
Edition	String	Version	"Enterprise Edition"	SQL, OS
Database Type	String	Configuration	"SINGLE"	SQL
Database Role	String	Configuration	"PRIMARY"	SQL
Open Mode	String	Configuration	"READ WRITE"	SQL
Log Mode	String	Configuration	"ARCHIVELOG"	SQL
Platform	String	Configuration	"Linux x86 64-bit"	SQL

Name	Type	Category	Sample value	Source
CDB Flag	String	Configuration	"YES"	SQL
Is RAC	String	Topology	"YES" or "NO"	SQL
Is Data Guard Standby	String	Topology	"YES" or "NO"	SQL
Protection Mode	String	Topology	"MAXIMUM PERFORMANCE"	SQL
Data Guard Broker	String	Topology	"ENABLED"	SQL
NLS Characterset	String	Configuration	"AL32UTF8"	SQL
PDB Count	Integer	Sizing	3	SQL
Flashback On	String	Configuration	"YES"	SQL
Detection Path	String	Metadata	"phase2" or "phase1b"	SQL, OS

PDB data (oracle_data_pdbs_full.csv)

One row per PDB (pluggable database). This table describes the PDB data points collected:

> **Note**
>
> PDB data requires Oracle database credentials (SQL connection). OS-level fallback does not populate this CSV.

Name	Type	Category	Sample value	Source
CDB Instance Name	String	Identity	"ORCL"	SQL
PDB Name	String	Identity	"APPPDB1"	SQL
Open Mode	String	Status	"READ WRITE"	SQL

Name	Type	Category	Sample value	Source
Lifecycle Status	String	Status	"NORMAL"	SQL
Tablespace Count	Integer	Sizing	5	SQL
Datafile Count	Integer	Sizing	12	SQL
Total Size Bytes	Integer	Sizing	5368709120	SQL
User Schema Count	Integer	Sizing	8	SQL
DB Link Count	Integer	Connectivity	2	SQL
Components Installed	String	Configuration	"APEX;JVM;XML"	SQL
Encrypted Tablespace Count	Integer	Security	1	SQL

Feature usage data (oracle_data_features_full.csv)

One row per feature usage entry from DBA_FEATURE_USAGE_STATISTICS. This table describes the feature usage data points collected:

> **Note**
>
> Feature usage data requires Oracle database credentials (SQL connection). OS-level fallback does not populate this CSV.

Name	Type	Category	Sample value	Source
CDB Instance Name	String	Identity	"ORCL"	SQL
Name	String	Feature	"Partitioning (user)"	SQL
Detected Usages	Integer	Usage	42	SQL

Name	Type	Category	Sample value	Source
Currently Used	String	Usage	"TRUE"	SQL
First Usage Date	DateTime	Usage	"2024-01-15T00:00:00"	SQL
Last Usage Date	DateTime	Usage	"2026-06-01T00:00:00"	SQL

Options data (oracle_data_options_full.csv)

One row per V$OPTION entry per CDB. This table describes the installed options data points collected:

> **Note**
>
> Options data requires Oracle database credentials (SQL connection). OS-level fallback does not populate this CSV.

Name	Type	Category	Sample value	Source
CDB Instance Name	String	Identity	"ORCL"	SQL
Option Name	String	Option	"Advanced Analytics"	SQL
Is Installed	String	Option	"TRUE"	SQL
Container ID	Integer	Option	0	SQL

Components data (oracle_data_components_full.csv)

One row per DBA_REGISTRY component. This table describes the component data points collected:

> **Note**
>
> Components data requires Oracle database credentials (SQL connection). OS-level fallback does not populate this CSV.

Name	Type	Category	Sample value	Source
CDB Instance Name	String	Identity	"ORCL"	SQL
PDB Name	String	Identity	"APPPDB1"	SQL
Component ID	String	Component	"APEX"	SQL
Component Name	String	Component	"Oracle Application Express"	SQL
Version	String	Component	"22.1.0.15.0"	SQL
Status	String	Component	"VALID"	SQL
Schema	String	Component	"APEX_220100"	SQL

Managing discovery tool storage

The discovery tool stores collected data in a local database on the appliance or host. The tool collects data continuously, so the database grows with the size of your server inventory and the length of the collection period.

The discovery tool tracks its own disk usage and warns you as space runs low. It also pauses collection before the disk fills, so that you do not lose data. You manage storage on the **Settings** page in the discovery tool console.

Understanding disk usage states

The discovery tool responds to disk usage as follows.

Disk usage	State	Discovery tool behavior
Below 85%	**Ok**, or **Advisory** at 70% and above	Collection runs normally. Below 70%, the console reports **Collection running normally**. At 70% and above, it reports **Disk usage elevated**. No action is required.

Disk usage	State	Discovery tool behavior
85% to below 90%	**Warning**	Collection runs normally. The console reports **Disk space getting low**.
90% or above	**Critical**	The discovery tool pauses all collection. The console shows an error that you cannot dismiss.

The discovery tool checks disk usage every 5 minutes. These thresholds are fixed. You cannot change them.

A database write can fail because the disk is full. If that happens, the discovery tool pauses collection immediately. It does not wait for the next check.

While collection is paused, the discovery tool skips scheduled rounds. **Start** and **Collect data now** are not available. You can still export data you already collected. You can also change retention settings. For more information about restarting collection, see the section called “Resuming collection”.

Resuming collection

Collection does not resume on its own. After you free space, choose **Resume collection** on the **Settings** page.

You can resume collection only after usage falls below 85%. That is the warning level, not the 90% critical level. The lower threshold prevents collection from pausing again immediately. If usage is still 85% or higher, free more space first.

Prune data counts toward freeing space, even though the database file does not shrink. For more information, see the section called “Reviewing usage and pruning data”. The discovery tool tracks the free space inside the database, and new data reuses that space without growing the file. In most cases, pruning is enough to resume collection without a larger disk.

Configuring data retention

By default, the discovery tool keeps collected data for 30 days. It deletes older data once a day. You can set one period for all modules, or set it per module.

To configure data retention

1. Open the discovery tool console at `https://<discovery-tool-vm-ip>:5000`.
2. Choose **Settings**.
3. For **Global retention (days)**, enter the number of days to keep data.
4. (Optional) Set a different period for one or more modules.
5. Choose **Save**.

Shorter retention saves space, but it also reduces the range of data you can export. An export can only include data that the discovery tool still holds. We recommend that you keep retention as long as the range you plan to export.

You can change retention settings even while collection is paused.

Reviewing usage and pruning data

The **Settings** page shows current disk usage. It also shows free space inside the database, and a **Database usage by module** breakdown. This information shows which module holds the most data.

Prune data deletes data older than a period that you set. It applies only to the modules that you select. You can use your saved period, or set a different one for that run.

> **Important**
>
> Pruning deletes collected data permanently. If you set 0 days, the discovery tool deletes all data for the modules that you select. Export any data you still need first.

Excluding VMs from collection

You can exclude specific virtual machines from VMware and Hyper-V inventory by editing a skip list file on the discovery tool appliance. VMs whose names match a pattern in the skip list are excluded from VMware and Hyper-V inventory and do not appear in the discovered inventory. This is useful for excluding infrastructure or management VMs that are not part of your migration scope.

Note

The skip list applies only to VMware and Hyper-V auto-discovery. If you import the same server through a CSV file, it is collected regardless of the skip list.

Important

Configure the skip list before you add your VMware and Hyper-V sources. VMs collected before a pattern is added remain in the database and appear in exports that cover the dates before the pattern was added. You can limit the export date range to exclude earlier rounds, but it is simpler to set up the skip list first.

Skip list file locations

The following table lists the skip list file path for each supported virtualization source. The paths below are for the standard OVA and VHD appliance. On a Linux installer deployment, the path depends on which user ran the installer — run `systemctl show discovery-tool -p Environment` and look for `DT_DATA_DIR` to find the correct path.

Source	File path
VMware	`/home/discovery/.local/share/DiscoveryTool/vm-skip-list.txt`
Hyper-V	`/home/discovery/.local/share/DiscoveryTool/hyperv-vm-skip-list.txt`

The skip list file is created automatically from built-in defaults the first time the discovery tool starts. It is never overwritten by updates, so your customizations are preserved across upgrades.

VMware default skip patterns

The VMware skip list is seeded with the following built-in patterns that exclude common VMware infrastructure VMs:

```
# VMC-A Infrastructure VMs
```

```
*Interconnect-IX*
*Interconnect-NE*
*Interconnect-SDR*
*Interconnect-WO*
hcx_cloud_manager
NSX-Edge-*
NSX-Manager-*
srm
vr
vcenter

# vSphere Cluster Service VMs
vCLS-*
```

You can add your own patterns below these defaults. Do not remove them unless you intentionally want to collect these infrastructure VMs.

File format

- One pattern per line. Lines starting with # are comments and are ignored. Blank lines are ignored.
- * matches any sequence of characters. ? matches exactly one character.
- Patterns are anchored to the full VM name. For example, `vCLS-*` matches `vCLS-001` but does not match `my-vCLS-001`. To match anywhere in the name, use `*vCLS-*`.
- Matching is case-sensitive by default. The skip list file ships with a `# case-sensitive: true` directive on its first line. Because the first matching directive in the file wins, edit this existing line and change `true` to `false` to enable case-insensitive matching — adding a second directive elsewhere in the file has no effect.

Example skip list file (with case-sensitive matching enabled):

```
# case-sensitive: true
# Exclude VMware infrastructure VMs
vCLS-*
# Exclude management VMs
mgmt-*
vCenter-*
```

When changes take effect

The skip list file is read at the start of each collection cycle. Changes take effect on the next collection cycle without requiring a restart. Previously collected data for VMs that you add to the skip list is not removed retroactively.

> **Warning**
>
> A pattern of * alone matches every VM name and disables collection for the entire VMware or Hyper-V source.

Multi-host behavior (Hyper-V)

For Hyper-V, a single skip list file applies to all configured Hyper-V hosts. VM names are per host and might not be unique across hosts. A pattern such as `mgmt-*` excludes matching VMs on all configured hosts.

Security considerations

Securing the discovery tool VM

Discovery tool VM security is crucial because the discovery tool stores all credentials, logs, and customer data on the discovery tool VM.

The discovery tool VM has ssh access disabled by default and can only be accessed from client `vCenter UI -> "Launch Web Console"`.

The discovery tool VM comes with a default login password "password" for user "discovery".

- We recommend that you update this password immediately after deployment.
- We **require** you to update the password if you want to enable ssh using the command `enablessh` after logging in using vCenter `Launch Web Console`. Please note, every time this command is called, you need to reset it.

Securing Credentials

General best practices for Credential Management

- Store credentials securely

- Regularly rotate all credentials
- Use password managers or secure vaults
- Monitor credential usage
- Follow the principle of least privilege and only grant the minimum necessary permissions needed

SNMP v2 Credentials

- Use complex, non-default community strings
- Avoid common strings like "public" or "private"
- Treat community strings like passwords

SNMP v3 Credentials

- Enable both authentication and privacy
- Use strong authentication protocols (SHA preferred over MD5)
- Use strong encryption protocols (AES preferred over DES)
- Use complex passwords for both auth and privacy
- Use unique usernames (avoid common names)

WinRM Credentials

- Avoid disabling WinRM certificate check.
- We recommend that you create a dedicated service account with minimal required permissions.
- Avoid using domain administrator or local administrator accounts unless you need SQL Server collection. SQL Server collection requires Local Administrator access because it queries multiple WMI namespaces and uses elevated commands. For OS metrics without SQL Server discovery, a non-administrator account with Remote Management Users, Performance Monitor Users, and WMI read access to `root\cimv2` is sufficient. See the section called “Required permissions” for per-module details.

Hyper-V credentials

- Use dedicated service accounts with minimal Hyper-V management permissions.
- Avoid using domain administrator accounts.

- Hyper-V credentials support NTLM (HTTPS only) and Kerberos authentication.
- The discovery tool stores credentials encrypted at rest by using SQLCipher.

Oracle credentials

- Use a dedicated read-only service account with only SELECT_CATALOG_ROLE. Do not use DBA, SYSDBA, or SYSOPER privileges.
- The discovery tool performs only read operations and never writes to the Oracle database.
- The discovery tool does not access Diagnostics Pack or Tuning Pack views, so no additional Oracle license is required.
- Rotate the Oracle service account password regularly and update the credential in the discovery tool.

SQL Server database credentials

- Use a dedicated SQL Server login with only the minimum required grants: `VIEW SERVER STATE`, `VIEW ANY DEFINITION`, and `VIEW ANY DATABASE`. Do not use `sysadmin` or other elevated roles.
- The discovery tool performs only read operations and never writes to SQL Server.
- SQL Server direct database connections in this release are unencrypted. Ensure that your network environment is appropriately secured.
- Rotate the SQL Server login password regularly and update the credential in the discovery tool.

Credential storage

The discovery tool encrypts stored credentials at rest using a database encryption key. On systems with systemd 250 or later, this key is encrypted using systemd-creds. On older systems, the key is stored as a permission-protected file. In both cases, an attacker with root access to the discovery tool host could access the encryption key and decrypt stored credentials. Restrict access to the discovery tool host and treat it as a privileged system in your environment.

Using Auto-Connect Feature With Caution

The discovery tool uses two mechanisms to assign credentials to servers during *OS-level collection*: auto-connect and manual. OS-level collection includes the Network, SQL Server, Oracle Database,

and OS metrics modules. These modules connect to individual servers from all sources, including VMware VMs, Hyper-V VMs, and imported servers.

Manual: a server can be manually associated with a specific credential. In this case, the discovery tool uses that credential only, regardless of success or failure. You must manually monitor collection status for that server and make adjustments.

Auto-connect: if no credential is manually associated with the server, the discovery tool uses the auto-connect mechanism for that server. That means:

- At the start of each collection round, the discovery tool gets a list of credentials available for that server (based on OS types) and also configured to be "auto-connectable".
- The discovery tool then tests all credentials against the server in a loop.
  - If a working credential is found, the collection round for the server is successful. The discovery tool remembers it and tries it first next time.
  - If no working credential is found, the collection round for the server has failed.
    - Network module: the server uses a backoff schedule, starting next collection round in 3 min, 30 min, 2 hours, and 6 hours following each failure (similar to exponential backoff).
    - SQL Server collection: The discovery tool does not retry. It makes one attempt per server each day.

Impacts/Risks:

Only use auto-connect when you are sure that risks are mitigated in your system:

Risk 1: With auto-connect configured on multiple wrong credentials, automatically trying them against servers could trigger account lockouts for production environments where lockout policies are configured. For example, a data center can set its VMs to lock down after 3 failed SSH login attempts. In this case, if auto-connect is configured for 3 wrong SSH credentials, legitimate account lockouts would happen. If lockouts happen across multiple systems, critical business processes could be impacted, potentially causing cascading failures in dependent systems. In addition, Security Operations Centers could experience alert storms from mass authentication failure events, creating false positive security incidents that drain resources and may mask real attacks.

Risk 2: Actor with access to the discovery tool (knows discovery tool password) can brute force OS credentials on all servers, by configuring a large number of test credentials and use auto-connect to find the successful ones.

Mitigations:

Follow these guidelines:

- Make sure that the discovery tool password is properly secured and known only to authorized actors
- Ensure proper credentials are entered for the environment if lockout policies are in place. We recommend only configure known, working credentials even if account lockout policies are absent to ensure minimal operational load on individual VMs.
- "Auto-connect" is an opt-in feature. Do not select it and use manual credential assignment if account lockouts is a concern to the environment.

CSV Import Security

When you import servers by using a CSV file, consider the following security implications:

- The CSV file might contain hostnames or IP addresses of internal servers. Treat it as sensitive data.
- The `os_credential_name` and `oracle_credential_name` columns reference preconfigured credentials by friendly name. The CSV does not contain secrets.
- All-or-nothing validation: if any row in the CSV is invalid, the entire upload is rejected. This prevents partial imports that could create a confusing state.
- Imported servers are immediately visible in the inventory and eligible for collection. Ensure that OS and Oracle credentials are correctly scoped before you import.

Revoking Access Considerations

When you revoke access, deletion is scoped to the specific source:

- Revoking vCenter access deletes only vCenter data. It does not affect Hyper-V or imported server data.
- Revoking Hyper-V access deletes only Hyper-V data. It does not affect VMware or imported server data.
- Deleting imported servers removes them from inventory, but downstream collection data (network, database, OS metrics) is retained.

- To remove all source-specific inventory data, you must revoke or delete each source independently. Downstream collection data (network, database, OS metrics) is retained even after all sources are revoked.

Troubleshooting

Incomplete or failed export

If an export does not include all the data you expect, open the `export_summary.json` file in the export ZIP and check `overall_status` and each module's errors to identify which servers or data types were affected and why. For a description of the file, see the section called “Understanding the export summary”.

Verifying discovery tool connectivity to vCenter

When you experience VMware module configuration errors follow these steps to verify connectivity:

Access the discovery tool VM

- Log-in to the discovery tool VM, open Remote Console in vCenter
  - Username: discovery
  - Password: password

Test vCenter Connectivity

1. Test vCenter API Access:

```
curl -v --insecure -u <username>:<password> https://<vcenter-ip-or-hostname>:443/
mob
```

2. Expected Success Output:

```
[ec2-user@discoverytool ~]$ curl -v --insecure -u <user>:<password> https://vcsa/
mob > tmp.txt
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
```

```
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0*
 Trying 192.168.2.125:443...
* Connected to vcsa (192.168.2.125) port 443 (#0)
...
</xml>
* Connection #0 to host vcsa left intact
```

Test SSL Certificate

1. Run this command:

```
openssl s_client -showcerts -servername <hostname> -connect <hostname>:443
```

2. Expected Success Output:

   - Should show vSphere certificate details
   - Verifies SSL/TLS connectivity on port 443

```
[ec2-user@discoverytool ~]$ openssl s_client -showcerts -servername vcsa -connect
 vcsa:443
CONNECTED(00000003)
depth=0 CN = vcsa.onpremsim.env, C = US
verify error:num=20:unable to get local issuer certificate
verify return:1
depth=0 CN = vcsa.onpremsim.env, C = US
verify error:num=21:unable to verify the first certificate
verify return:1
---
Certificate chain
 0 s:/CN=vcsa.onpremsim.env/C=US
   i:/CN=CA/DC=vsphere/DC=local/C=US/ST=California/O=vcsa.onpremsim.env/OU=VMware
 Engineering
-----BEGIN CERTIFICATE-----
...
-----END CERTIFICATE-----
---
Server certificate
subject=/CN=vcsa.onpremsim.env/C=US
issuer=/CN=CA/DC=vsphere/DC=local/C=US/ST=California/O=vcsa.onpremsim.env/OU=VMware
 Engineering
```

```
---
```

WinRM Troubleshooting

If you're experiencing connectivity issues with WinRM, follow these steps to test the connection. These steps also apply to Hyper-V connectivity issues, because the discovery tool uses WinRM to communicate with Hyper-V hosts.

Test basic WinRM connectivity using ports 5985 (HTTP) and 5986 (HTTPS). We need to make sure that connectivity works on port 5986 (HTTPS)

```
# Check WinRM listener configuration
winrm enumerate winrm/config/listener

# Note: Replace <HOST> with the target computer's hostname or IP address. Adjust the
 username and password as needed.
# Test WinRM connection on port 5985 (HTTP)
$cred = Get-Credential
Test-WSMan -Computer <HOST> -Authentication Negotiate -Credential $cred -Port 5985

# Test WinRM connection on port 5986 (HTTPS)
Test-WSMan -Computer <HOST> -Authentication Negotiate -Credential $cred -Port 5986
```

If the above tests fail, try establishing a PowerShell session with certificate validation disabled:

```
$cred = Get-Credential
$so = New-PsSessionOption -SkipCACheck -SkipCNCheck -SkipRevocationCheck
Enter-PSSession -ComputerName <HOST> -Credential $cred -Port 5985 -SessionOption $so
```

Kerberos troubleshooting

If you experience Kerberos authentication failures when collecting data from Windows servers, use the following sections to diagnose and resolve common issues.

Verify network requirements

Before you troubleshoot Kerberos authentication, verify that the discovery tool can reach the required network endpoints.

To verify network requirements for Kerberos

1. Verify DNS resolution to the domain controller. Run the following command from the discovery tool VM:

   ```
   nslookup dc01.example.com
   ```

   Alternatively, you can use `dig`:

   ```
   dig dc01.example.com
   ```

   If DNS resolution fails, verify that the discovery tool VM is configured to use a DNS server that can resolve your Active Directory domain. Check `/etc/resolv.conf` and confirm that the nameserver entries point to your domain DNS servers.
2. Verify connectivity to the Key Distribution Center (KDC) on port 88. Run the following command:

   ```
   nc -zv dc01.example.com 88
   ```

   Expected output:

   ```
   Connection to dc01.example.com 88 port [tcp/kerberos] succeeded!
   ```

   If the connection fails, verify that no firewall rules block traffic from the discovery tool VM to the domain controller on port 88.
3. Verify connectivity to the target Windows servers on WinRM ports. Run the following commands:

   ```
   nc -zv <windows-server> 5985
   nc -zv <windows-server> 5986
   ```

   If the connection fails, verify that WinRM is enabled on the target server and that firewall rules allow inbound traffic on ports 5985 and 5986.

Common Kerberos issues

The following are common Kerberos issues and their solutions.

Server not found in Kerberos database

Symptom: You receive this error during authentication.

This error has two common causes:

- **Realm name not in uppercase** – Kerberos realms must be uppercase in your `krb5.conf` file. For example, use `EXAMPLE.COM` instead of `example.com`.
- **Reverse DNS not working for a server that has no FQDN** – Kerberos builds its service principal name from the FQDN, so the discovery tool must map the IP address to the FQDN first. This cause applies only when the discovery tool has no FQDN for the server, such as a server that you imported from a CSV file with only an IP address. Run `getent hosts <target-ip>` on the discovery tool VM. If the command does not return the FQDN, add a PTR record, add an `/etc/hosts` entry, or use the FQDN. For more information, see the section called "Using Kerberos with IP addresses".

The discovery tool tries a server's FQDN before its IP addresses. If the discovery tool has an FQDN for the server, it also tries the IP addresses as a fallback, so you might see this error for an IP address even though the FQDN attempt is the one that matters. In that case, look at the error for the FQDN attempt to find the cause.

Account lockout prevention

The discovery tool uses a backoff mechanism to prevent account lockouts from repeated failed authentication attempts. If your service account becomes locked out, you can reset the collection process by stopping and then starting the collection module through the discovery tool web UI at `https://<discovery-tool-vm-ip>:5000`.

kinit fails from CLI

The following table lists common `kinit` errors and their solutions.

Error	Cause	Solution
Cannot find KDC for realm	The KDC hostname or IP address is not reachable, or the realm is not configured in `krb5.conf` .	Verify that the `krb5.conf` file contains the correct KDC hostname and realm. Confirm DNS resolution and network connectivity to the KDC on port 88.

Error	Cause	Solution
Preauthentication failed	The password for the service account is incorrect.	Verify the password and try again. If the account is locked out, unlock it in Active Directory before you retry.
Client not found in Kerberos database	The principal name does not match any account in Active Directory.	Verify that the principal name matches the account name exactly, including case. Use the format `username@REALM` with the realm in uppercase.
Cannot resolve network address for KDC	DNS cannot resolve the KDC hostname.	Verify DNS configuration in `/etc/resolv.conf`. Confirm that the DNS server can resolve the KDC hostname. Test with `nslookup` or `dig`.

Collection fails despite successful kinit

If `kinit` succeeds but data collection still fails, check the following:

1. Verify that the principal name used for collection matches the case used during `kinit` exactly.
2. Verify that the service account has the required permissions on the target servers.
3. Verify that WinRM is enabled on the target servers.
4. Verify that the hostname used for collection matches the hostname registered in Active Directory.
5. If the discovery tool has no FQDN for the server, such as a server that you imported by IP address only, check that the VM maps that IP address to the server's FQDN. Run `getent hosts <target-ip>` on the discovery tool VM. Kerberos builds its service principal name from the FQDN, not the IP address. Without a reverse mapping, the handshake fails even when `kinit` succeeds.

Kerberos works for some servers but not others

If Kerberos authentication succeeds for some servers but fails for others, investigate the following areas:

If your servers span multiple Active Directory domains, configure a separate Kerberos credential for each domain. Make sure that your `/etc/krb5.conf` file includes entries for all realms. Each domain requires its own credential with the correct `username@REALM` principal.

Compare the WinRM configuration on a working server with a failing server. Run the following command on each server:

```
winrm get winrm/config
```

Test connectivity with Remote Desktop to isolate the issue. The discovery tool uses the format `username@DOMAIN`, while Remote Desktop uses the format `DOMAIN\username`.

Verify that the service account is a member of the local Administrators group on the failing server. Run the following command on the target server:

```
net localgroup Administrators
```

WMI requires local administrator privileges to access operating system information. SQL Server collection also requires that the service account has local administrator access on the target server.

FQDN unavailable for servers discovered through Hyper-V

For servers discovered through Hyper-V, the discovery tool resolves the server FQDN from guest metadata provided by Integration Services, which is required for Kerberos SPN matching. If Integration Services is not running in the guest, the FQDN is unavailable and Kerberos authentication fails. Confirm that Integration Services is running in the guest (see the section called "Guest agent requirements for OS-level collection").

Kerberos configuration checklist

Use the following checklist to verify your Kerberos configuration before you start data collection.

- The `krb5.conf` file exists on the discovery tool VM.
- The realm name is in uppercase in `krb5.conf`.
- Running `kinit` with the service account succeeds without errors.
- Running `klist` shows a valid, non-expired ticket.
- The principal name matches the Active Directory account name exactly.
- DNS resolution works for the KDC hostname.

- Network connectivity to the KDC on port 88 is confirmed.
- Network connectivity to target Windows servers on ports 5985 and 5986 is confirmed.
- (Multi-domain) Each Active Directory domain has its own credential configured in the discovery tool, and `krb5.conf` contains `[realms]` and `[domain_realm]` entries for all domains.
- (Servers with no FQDN) Running `getent hosts <target-ip>` returns the server's Active Directory FQDN, or the server is configured by FQDN instead.

Oracle Database troubleshooting

To diagnose Oracle Database collection issues, such as missing data or errors, check the following.

Connection refused or timeout

Symptom: Oracle collection status shows a connection error for a server.

To troubleshoot this issue, check the following:

- Verify that the Oracle listener is running on the target host: `lsnrctl status`
- Verify network connectivity from the discovery tool to the Oracle host on port 1521 (or your custom port): `nc -zv <oracle-host> 1521`
- Verify that firewall rules allow inbound connections on the Oracle listener port.
- Verify the service name by running `lsnrctl services` on the Oracle host. If the service name is incorrect, the Oracle listener rejects the connection.

Authentication failures (ORA-01017)

Symptom: Collection fails with an invalid username or password error.

To troubleshoot this issue, check the following:

- Verify that the Oracle service account exists and is not locked: `SELECT account_status FROM dba_users WHERE username = 'DISCOVERY_USER';`
- Verify the password is correct by connecting manually: `sqlplus discovery_user/<password>@<host>:1521/<service_name>`
- If the account is locked, unlock it: `ALTER USER discovery_user ACCOUNT UNLOCK;`

Insufficient privileges (ORA-01031)

Symptom: Connection succeeds but collection returns incomplete data.

To troubleshoot this issue, check the following:

- Verify that SELECT_CATALOG_ROLE is granted: `SELECT * FROM dba_role_privs WHERE grantee = 'DISCOVERY_USER';`
- Grant the required role if missing: `GRANT SELECT_CATALOG_ROLE TO discovery_user;`

Manual credential shows error but auto-connect works

When you pin a credential manually to a server, the discovery tool does not fall back if the connection fails. Verify that the port and service name that you configured on the credential match the Oracle listener on that specific server. If the server has a non-standard port or service name, update the credential configuration accordingly.

OS-level fallback not detecting Oracle

If no database credentials are configured and the OS-level fallback does not detect Oracle:

- Verify that SSH or WinRM OS credentials are configured and working for the server (check OS metrics collection status).
- For Linux hosts, verify that `/etc/oratab` exists or that Oracle process monitor (`pmon`) processes are running.
- For Windows hosts, verify that Oracle registry entries exist under `HKLM\SOFTWARE\Oracle` or that `oracle.exe` processes are running.

SNMP Troubleshooting

Access the discovery tool VM

- Log-in to the discovery tool VM, open Remote Console in vCenter
  - Username: discovery
  - Password: password

Install SNMP Tools (if needed)

- `sudo yum install net-snmp-utils -y`

Test SNMP Connection to Linux Servers

1. `snmptable -v 2c -c <COMMUNITY_STRING> <REMOTE_SERVER_IP> .1.3.6.1.2.1.6.13.1`
2. Example:

```
#SNMPv2c:
snmptable -v 2c -c public 192.168.1.100 .1.3.6.1.2.1.6.13.1

#SNMPv3 (with authentication):
snmptable -v 3 -u <username> -a MD5 -A <auth_password>
 192.168.1.100 .1.3.6.1.2.1.6.13.1

#SNMPv3 (with privacy):
snmptable -v 3 -u <username> -a MD5 -A <auth_password> -x DES -X <priv_password>
 192.168.1.100 .1.3.6.1.2.1.6.13.1
```

Network collection errors

a terminal is required to read the password

Error:

```
ss command failed on <host>: sudo: a terminal is required to read the password; either
 use the -S option to read from standard input or configure an askpass helper sudo: a
 password is required
```

The ss command is prompting for user password. The configured ssh user must be in the sudoers group and be configured with passwordless sudo for the ss/netstat command. To configure passwordless sudo:

1. Create a new sudoers file:

```
sudo vi -f /etc/sudoers.d/<username>
```

2. Add the line:

```
<username> ALL=(ALL) NOPASSWD: /usr/sbin/ss, /usr/bin/netstat
```

3. After this change, running `sudo ss -tnap` and `sudo netstat -tnap` should execute without prompting for a password

Network collection ran without sudo

If you see the following warning on the **Discovered inventory** page:

```
Network collection ran without sudo. Process-level connection data may be missing.
```

This warning indicates that the SSH user account does not have sudo access on the target server. Without sudo, the discovery tool can still collect network connection data, but it cannot determine which process owns each connection. To collect complete process-level connection data, ensure the SSH user has sudo access on the target server.

OS metrics collection errors

Missing server UUID for Linux servers

If the discovery tool is unable to collect the server UUID (showing as empty or missing) for Linux servers, verify that the SSH credentials configured for those servers have sudo privileges. The tool uses `dmidecode` to read the server UUID. If `dmidecode` is not installed, the tool falls back to reading `/sys/class/dmi/id/product_uuid`, which also requires sudo access. Without sudo, neither method can retrieve the UUID.

Resolution: Ensure the SSH user account provided to the discovery tool has sudo access on the target Linux servers.

Server appears in inventory but is never collected

Symptom

A server appears in **Discovered inventory** with no IP address, and its collection status never leaves pending. This typically means the guest agent is not supplying a guest address, so the discovery tool has nothing to connect to for OS-level collection.

Checks

- On Hyper-V, run `Get-VMIntegrationService -VMName` *`vm-name`* on the Hyper-V host and confirm that the Key-Value Pair Exchange service is running.

- On VMware, check the VMware Tools status for the virtual machine in the vSphere Client.

Resolution

- Install and start the guest agent (VMware Tools or Hyper-V Integration Services) in the guest, and then wait for the next discovery cycle.
- Alternatively, import the server from a CSV file with its IP address, which supplies the address directly. For more information, see the section called “Import servers”.

For background on the guest-agent dependency, see the section called “Guest agent requirements for OS-level collection”.

Access issues in Discovered inventory

If you see a message in **Server collection status** such as Missing credentials, or Access denied:

1. Select the server on the table of discovered servers.
2. Choose **Manage access credential** You can choose to:
   a. Select alternative credentials from the **Select credentials** dropdown.
   b. Select **Use new credentials** and provide new credentials.
3. **Save**.

The discovery tool retries the connection after you save your changes.

SSH key authentication troubleshooting

Testing SSH key connectivity from the discovery tool

If SSH key authentication fails, verify connectivity from the discovery tool to the target server:

1. Log in to the discovery tool (via vSphere console or SSH to the Linux host).
2. Test SSH connectivity using your private key:

```
ssh -i /path/to/private_key -o StrictHostKeyChecking=no <username>@<target_ip>
```

3. If the connection succeeds, the issue is with how the key was uploaded to the discovery tool. Re-upload the key and verify the username matches.

4. If the connection fails, check the error message in the following table.

Error message	Cause	Resolution
`Permission denied (publickey)`	The public key is not in the target server's `authorized_keys` file, or the username is wrong.	Add the public key to `~/.ssh/authorized_keys` on the target server for the correct user. Verify file permissions: `chmod 700 ~/.ssh && chmod 600 ~/.ssh/authorized_keys`.
`Connection timed out after 20s`	Port 22 is not reachable from the discovery tool.	Verify that port 22 is open between the discovery tool and the target server. Check firewalls and security groups.
`Connection refused`	SSH service is not running on the target server.	Start the SSH service: `sudo systemctl start sshd`.
`Invalid SSH key for credential 'name'`	The key format is not supported, the key data is corrupted, or the passphrase is missing or incorrect.	Verify the key format is RSA, ECDSA, or Ed25519 in PEM, OpenSSH, or PKCS#8 format. If the key is encrypted, make sure the passphrase is correct.

Linux installer troubleshooting

Port 5000 is already in use

Symptom: The discovery tool service fails to start after installation.

Resolution: Identify and stop the process using port 5000:

```
sudo ss -tlnp | grep :5000
```

Stop the conflicting process, then restart the discovery tool:

```
sudo ./AWS-Transform-discovery-tool.sh start
```

Common error messages

This table describes common error messages and their explanations:

Message	Location	Explanation
A password has already been created	Create password page	Race condition when two users create passwords at the same time; refresh
Export failed	Inventory page	Retry or send logs
An on-demand collection is already in progress	Inventory page	Race condition when two users start manual collections at the same time; try again after the current manual collection is finished
`Command timed out after 60s`	Server collection status	A command on the target server did not complete within 60 seconds. This can occur on heavily loaded servers. Retry the collection or investigate the target server's load.
One or more credentials contain unknown UUIDs	OS access page	Race condition when two users edit OS credentials at the same time; try again
Invalid password	Sign-in page	Incorrect password for logging in; contact admin or reach out
Your session has expired. Please log in again.	Sign-in page	Session has timed out, need to login again
An internal error occurred	Various pages	Retry or send logs

Release notes

This section describes the features, improvements, and bug fixes for the AWS Transform discovery tool, organized by release date.

Date	Version	Release notes
October 15, 2026	1.0.2810	**New features** • Added SQL Server direct database collection. When you configure SQL Server database credentials, the discovery tool connects directly to SQL Server instances and collects database count, Always On availability group membership, cluster role, and read-only replica status. These fields appear in `sql_server_data_full.csv` . The tool falls back to Windows-level collection when no auto-connect credential can connect. SQL Server database credentials are configured on the **Database access** page. • Added self-update. You can now update the discovery tool to a newer version in place without redeploying the appliance . Self-update preserves all collected data, discovery source configuration, and credentials. If the new version fails health checks after installation, the tool automatically rolls back to the previous version. Self-update is disabled by default and requires outbound internet access. **Improvements** • You can now update the discovery tool in place with the Linux installer, keeping your collected data, credentials, and database encryption key. This is available for appliances on version 1.0.2325 or later; see Minimum version for each update method. • Added VM skip lists for both VMware and Hyper-V. You can now exclude specific virtual machines from collection by editing a skip list file on the appliance. Both skip lists use the same file format, with glob-style pattern matching anchored to the full VM name. The skip lists also apply a set of built-in default patterns that exclude common infrastructure VMs (`vCLS-*`, `NSX-Edge-*` , `NSX-Manager-*` , and others).

Date	Version	Release notes
		For the full list of default patterns, see Excluding VMs from collection. • The discovery tool now skips non-routable addresses (link-local, loopback, unspecified, multicast, and `localhost`) when attempting OS-level collection. This prevents collection failures caused by unusable guest addresses and reduces unnecessary connection attempts. RFC 1918 private addresses are not affected. • Each export ZIP now includes an `export_summary.jso n` file at its root. This file reports the overall status of the export (`complete`, `partial`, or `failed`), the export time range, and per-module counts of the servers, devices, and connections that were exported, along with details of any records that were skipped or could not be exported. If an export is partial or fails, the discovery tool directs you to this file for details. • Oracle Database collection is now on by default. Previously you had to enable it explicitly. • Kerberos authentication now builds the service principal name as `WSMAN/<fqdn>` by default, falling back to `HTTP/ <fqdn>` if the server does not recognize the WSMAN SPN. This matches how Windows registers WinRM service principal names and reduces Kerberos authentication failures. • Saving or reassigning a SQL Server or Oracle database credential now starts collection for the affected servers immediately, instead of waiting for the next scheduled round. • The OVA and VHD appliance images now default to a 100 GB disk (up from 35 GB). The root volume is thin-provisioned on VMware, so the image still imports onto small datastores, and the file system grows to fill the disk on first boot. **Bug fixes**

Date	Version	Release notes
		• Fixed a SQL Server Reporting Services (SSRS) collection issue over WinRM that could cause SSRS components to be missed during collection. • Fixed a timing issue where an on-demand ("Collect now") SQL Server collection running longer than one hour could have its in-progress status cleared early, letting that night's scheduled round start at the same time and corrupt or drop results. On-demand collection can now safely run its full two-hour budget. • Fixed an issue where stopping SQL Server collection mid-round could mislabel a host's status as "Skipped (in backoff)" if that host's credential was deleted or rotated during the round. The host now keeps its prior status. • The discovery tool now shows a **Warning** status when a SQL Server login is missing the `VIEW ANY DEFINITION` grant and Always On is enabled on the instance, instead of silently reporting a **Success** status with an availability group member incorrectly shown as standalone. • Fixed the discovery tool's self-signed TLS certificate to cover the appliance's actual local network addresses, host name, and FQDN, instead of only `localhost` . • The **Discovered inventory** page now records the last collection time for collection that you trigger manually, such as choosing **Save and collect**, importing a CSV file, or assigning a credential, not only for scheduled rounds. CSV import now also reports every credential name in the file that does not match a configured credential, instead of only the first one.

Date	Version	Release notes
September 3, 2026	1.0.2325	**New features** • Added disk monitoring and storage management. The discovery tool now tracks its own disk usage and warns you as space runs low. It pauses collection at 90% usage to avoid filling the disk. You can set how long the tool keeps data, for all modules or per module. You can also see usage by module and delete older data on the **Settings** page. • Added `unmapped_storage_devices.csv` to the export. The discovery tool adds a device to this file when it cannot match the device to a mounted file system, or when the device reports no usable capacity. **Improvements** • Improved Kerberos authentication for Windows servers. When the discovery tool knows a server's fully qualified domain name (FQDN), it now uses the FQDN first for Kerberos credentials. That order matches how Active Directory registers service principal names. You can still use IP addresses. The discovery tool maps the IP address to the FQDN with a reverse DNS lookup. • Kerberos authentication now backs off after an unusable credential, such as an incorrect or expired password, instead of retrying every collection cycle. Threads that share one credential now make a single attempt between them. Both changes lower the risk of Active Directory account lockout. • Added 95th percentile (_p95) columns to the OS metrics CSV exports. • Renamed the disk throughput columns in the OS metrics CSV exports from `_mbps` to `_mibps`. The values do not change. They were already reported in mebibytes per second (MiBps).

Date	Version	Release notes
		• Improved the collection status for a virtual machine that reports no usable address. The discovery tool now reports why it cannot reach the machine, including the status of VMware Tools or Integration Services, instead of reporting only that no valid address was found. • Added `enablekrb5trace` and `disablekrb5trace` aliases on the appliance to turn Kerberos trace logging on or off for troubleshooting. The discovery tool writes trace output to `~/.local/share/DiscoveryTool/logs/krb5_trace.log`. Trace logging does not persist across a reboot, and a cron job rotates the log every 5 minutes to limit its size. **Bug fixes** • Fixed Linux disk IOPS and throughput values. The discovery tool now samples a live interval instead of reading the average since the host last started, which underreported both values. • Fixed exports that could fail on a large server inventory. The discovery tool now measures the memory available to its own service, estimates the size of network data more accurately, and runs one export at a time. • Fixed the loss of collected network connection data when the discovery tool service restarts.

Date	Version	Release notes
June 30, 2026	1.0.1646	**New features** • Added Oracle Database discovery. The tool collects CDB and PDB topology, version, edition, installed options, feature usage, and component inventory. The discovery tool supports direct SQL connections (using SELECT_CATALOG_ROLE) and OS-level fallback detection through SSH or WinRM. • Added support for multiple Kerberos credentials across different Active Directory domains. Each credential authenticates independently, enabling collection from Windows servers in multiple domains without authentication conflicts. **Improvements** • Improved OS metrics collection for legacy hosts. Added fallback data collection for older operating systems (RHEL 5, SLES 11, Windows Server 2008 R2) where modern commands are unavailable. Storage, performance, and network interface data is now collected using alternative system interfaces. **Bug fixes** • Fixed Kerberos authentication failures for servers discovered through Hyper-V. The discovery tool now correctly resolves the server hostname (FQDN) from Hyper-V guest metadata, which is required for Kerberos SPN matching.

Date	Version	Release notes
May 14, 2026	1.0.1363	**New features** • Added Linux installer for deploying the discovery tool on supported Linux distributions (Amazon Linux 2, Amazon Linux 2023, RHEL 8–9, Rocky Linux 8–9, AlmaLinux 9, Ubuntu 20.04–24.04, Debian 10–12, SLES 15 SP5). • Added support for configuring multiple VMware vCenter servers with automatic VM deduplication across vCenters. • Added SSH private key authentication as an alternative to username and password for Linux OS access. Supported key formats include RSA, ECDSA, Ed25519, OpenSSH, and PKCS#8. • Added private address network collection to capture connections to and from RFC 1918 private IP addresses not in your discovered inventory. • Added export options to select a custom date range and choose which data modules to include in the export. **Improvements** • Improved scheduled collection reliability by staggering collector schedules. • Improved OS metrics collection performance through SSH command batching. • Credential changes now immediately retry previously failed servers. • OS metrics collection now reports partial status when some commands return empty results. • Improved error visibility for vCenter connection issues in the dashboard.

Date	Version	Release notes
April 7, 2026	1.0.1074	**New features** • Added support for Microsoft Hyper-V discovery, including auto-discovery of VMs from Hyper-V hosts with failover cluster deduplication. • Added support for server import through CSV file upload. • Added OS metrics collection with six independent modules: server performance, storage performance, server provision ing, storage provisioning, network interfaces, and running processes. • Added Hyper-V deployment option using a VHD image file. • Improved on-premises security hardening by removing EC2-specific services and blocking access to the instance metadata endpoint.
March 26, 2026	1.0.1014	**Bug fixes and improvements** • Moved Kerberos domain capitalization validation from backend to frontend. • Switched from Python-kerberos to Python-pykerberos and made file-based cache the default. • Fixed case-insensitive bulk edit server ID validation. • Fixed export ZIP filename (now `discovery_tool_exp ort.zip`). • Fixed service running check and added debug logs for system credentials. • Added network configuration script to build artifact.

Date	Version	Release notes
December 16, 2025	1.0.718	**New features and improvements** • Added SMBIOS UUID support to prevent VM duplication in downstream services. • Introduced Min/Max/Avg tracking and expanded storage metrics for improved resource monitoring. • Added API validation improvements. • Added help panel functionality with contextual guidance. • Updated terminology from "collector" to "tool" throughout the interface.
October 31, 2025	1.0.0	**Initial release** Initial launch of the AWS Transform discovery tool. This release includes: • Offline collection (no dependency on an AWS account or Application Discovery Service). • VMware discovery and monitoring. • SQL Server database discovery. • Server-to-server communication collection through SSH, WMI, and SNMP.

Migrations (including VMware)

AWS Transform can help you migrate your virtual and bare metal server environments to Amazon EC2 by using generative AI, including VMware, Hyper-V, and other sources. An AI-powered agent accelerates each stage of your migration, from discovery and planning through network migration and server rehost. This reduces the manual effort traditionally required for large-scale migrations. You interact with the agent through a conversational interface that guides you step by step, so you can focus on decisions rather than execution.

Capabilities and key features

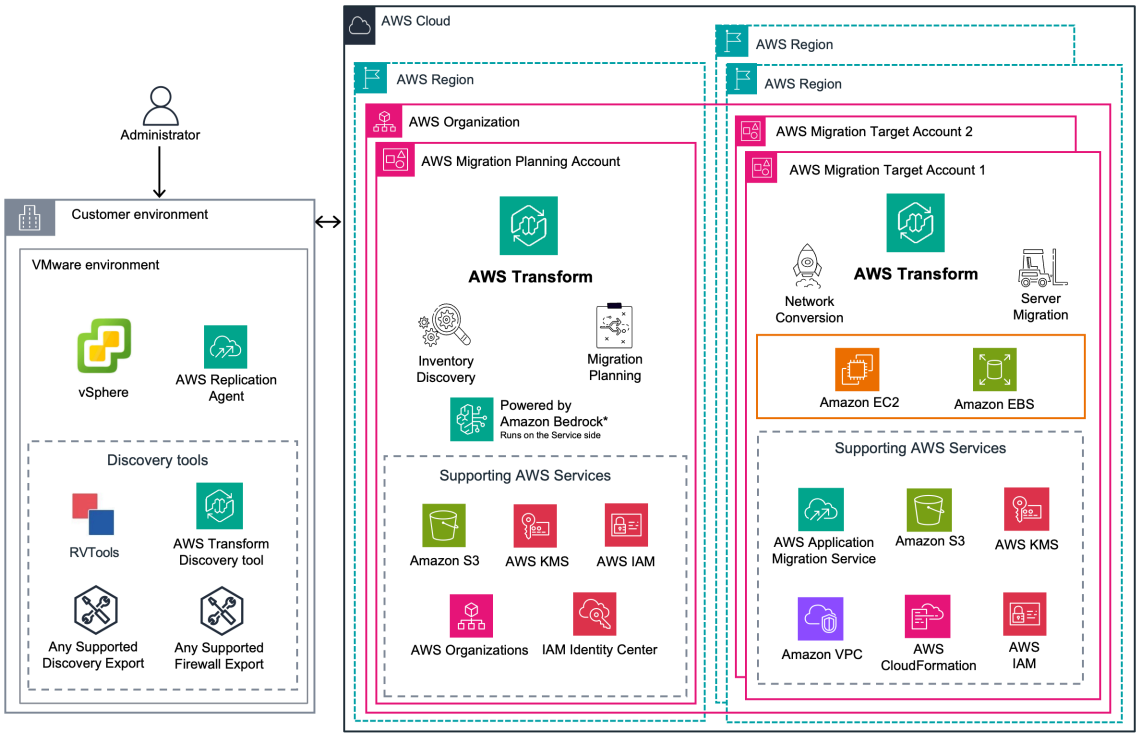
AWS Transform offers the following capabilities and key features for migrating your server environments to AWS.

- **Flexible discovery** that meets you where you are, with multiple options to capture your source environment:
  - Assisted discovery of your VMware environment by using collectors from AWS Application Discovery Service.
  - The open-source Export for vCenter tool (VMware environments).
  - Importing independently collected discovery data (any source environment).
- **AI-driven network migration** that translates your source network configuration into a production-ready Amazon VPC architecture, so you don't have to design it from scratch.
- **Intelligent migration planning** that automatically groups applications and sequences migration waves, helping you move faster with fewer planning cycles.
- **Automated server rehosting** to run your servers natively on Amazon EC2, with guided testing and cutover workflows.
- **Localized interface**, allowing you to use AWS Transform for migrations in your preferred language. For more information, see Language settings.

AWS Transform supports migrating Windows and Linux servers of supported operating systems. For the full list of supported operating systems, see Supported operating systems in the *AWS Transform MGN User Guide*.

AWS Transform migration architecture

AWS Transform orchestrates the entire migration lifecycle from a single interface. The following diagram displays an overview of the architecture.



Migration jobs

To start migrating, create a workspace and launch a migration job. A workspace is a logical container for one or more migration jobs, giving you a single place to manage your entire migration.

Getting a workspace

For information about creating a workspace, see Getting started.

Your workspace determines the AWS Region where your jobs, discovery data, and AWS Transform recommendations reside. To work in a different AWS Region, ask your administrator to create a workspace in that Region. For information about supported AWS Regions, see *Supported Regions*.

You can specify a different AWS Region as your migration target. This means you can run discovery in one Region but deploy your target environment in another. If you do that, some of your data is

transferred across AWS Regions. Server replication data is not transferred across Regions; it goes directly from your source environment into your target account and Region. For more information, see the section called “Connect target AWS accounts and regions”.

Job types

Whether you need an end-to-end migration or want to tackle a specific phase, AWS Transform offers several types of migration jobs to choose from. In addition to these preset options, you can add or remove any step from your job at any time to customize your migration workflow.

End-to-end migration

Best for complete migrations where you want AWS Transform to handle everything from discovery through rehost.

1. Perform discovery
2. Build migration plan
3. Connect target accounts
4. Build landing zone
5. Migrate network
6. Migrate servers

Discovery and migration planning

Use this when you want to assess your environment and build a wave plan before committing to execution.

1. Perform discovery
2. Build migration plan

Network migration

Use this when your landing zone and accounts are already set up and you need to migrate your network configuration to AWS.

1. Connect target accounts
2. Migrate network

Landing zone

Use this when you need to set up your multi-account AWS foundation before migrating workloads.

1. Connect target accounts
2. Build landing zone

Landing zone, network, and server migration

Use this when you already have discovery and planning complete and want AWS Transform to handle infrastructure setup and execution.

1. Connect target accounts
2. Build landing zone
3. Migrate network
4. Migrate servers

Migration planning and server migration

Use this when your network is already in place and you want to go from discovery straight to rehost.

1. Perform discovery
2. Build migration plan
3. Connect target accounts
4. Migrate servers

Creating and starting a job

To create a migration job, complete the following steps. For information about the different job types, see the section called “Job types”.

To create and start a new migration job

1. On your workspace landing page, choose **Create a job**.

2. Choose the migration option, and then specify the type of migration job that you want to create. For information about the steps included in each of the migration job types, see the section called “Job types”.
3. After you answer all the chat questions, choose **Create job**.

Downloading a workspace summary report

You can generate a workspace summary report as a downloadable PDF at any time during your migration. The report provides a consolidated view across all migration jobs in your workspace, including:

- Job statuses and current workflow steps
- User actions and approvals
- Wave planning details
- Network migration topology
- Landing zone configuration
- Rehost progress
- Containerization decisions
- Key artifacts produced

To generate a report, ask the agent in the chat. For example, “Give me a workspace summary of my migration progress across all jobs.” The agent compiles data from all jobs and delivers the PDF directly in chat.

Use expert mode

Expert mode minimizes interaction during a migration job. You provide inputs and preferences before execution, and the agent applies them automatically as the job progresses. The agent pauses only when it can't resolve a question or when an error requires action.

To use expert mode

1. Create the migration job and define its job plan. Before starting any step, enable expert mode by entering "Run this job in expert mode" in chat or by entering the /expert-mode-on command in the web application.

2. Start the job. The agent adds **Data collection** as the first step and displays the inputs that are available for your job plan.
3. Provide your inputs and preferences. You can include multiple values in one message or attach a UTF-8 text file (maximum 256 KB). For example, enter "Use multi-account migration, average Amazon EC2 sizing, shared tenancy, and a maximum of 20 servers per wave."
4. Optionally, specify review checkpoints or mode switches. For example, enter "Pause for my review after migration planning" or "Switch to standard mode after network migration."
5. Review the collected values and correct any values that the agent marks as invalid, and tell the agent to proceed.

The agent processes steps and waves sequentially, presenting intermediate results without waiting for confirmation unless you set a review checkpoint. You can send instructions in chat at any time and switch to standard mode on demand.

Limitations

- You must enable expert mode before any step starts. You can't enable it mid-job.
- The agent pauses when it can't resolve a question from collected inputs, when an input is invalid, or when an error requires action.
- Expert mode doesn't bypass approval requirements. Operations that require approval still wait for authorization in the **Approvals** tab.
- Collected settings apply globally to all relevant steps and waves. Wave-specific instructions are evaluated when that wave runs.
- Input text files must use UTF-8 encoding and can't exceed 256 KB. This limit doesn't apply to inventory or other migration artifacts.

Limitations

AWS Transform has the following limitations:

- If you stop a running migration job, and then ask the agent to restart it, the job will start again from the beginning and you will lose any progress you have made in the job. However, artifacts created in the job before restarting it will still be available.
- You can specify one target AWS Region per migration job. To migrate applications to different target Regions, create multiple migration jobs.

- Multi-account migration – Single region only – You can migrate to multiple accounts within a single AWS Region. For multi-Region migrations, you must create separate projects for each target region.
- Multi-account migration – One account per wave – Each migration wave can target only one account. Applications requiring different target accounts must be placed in separate waves.
- Multi-account migration – AWS Organizations required – All target accounts must be part of an AWS Organization.

Discover source data

Upload your on-premises server data and AWS Transform automatically detects file format and structure, parses and extracts structured entity records, removes duplicates across multiple files, validates data quality and reports issues, and then prepares a summary ready for downstream migration planning.

AWS Transform accepts data from the following sources:

- AWS Transform discovery tool (CSV and JSON in ZIP format)
- AWS Migration Evaluator collector
- RVTools (Excel file or a ZIP file containing CSV files)
- modelizeIT (CSV in ZIP format)
- AWS Migration Portfolio Assessment (MPA) format exports (for example, Cloudamize)

For each data source, AWS Transform accepts the primary server file individually. Supplementary files such as connections are only processed when included in a ZIP file with the server files or in an Excel file with the server sheet.

Upload the most detailed data available. Before you upload, review your export files to ensure data completeness and accuracy. Verify that all required files are included, and confirm that the data reflects your current environment state. This helps AWS Transform capture your on-premises environment more accurately and better support migration planning, including application grouping and wave planning. You can incrementally add data as you obtain it. AWS Transform automatically de-duplicates records across multiple uploads.

After upload, review your discovery data by expanding **Discover on-premises data** in the **Job Plan** and choosing **Inventory readiness summary**. You can ask questions in the chat to verify ingestion

or identify issues. For example, you can ask about operating system and versions. To correct mistakes, re-upload your data and AWS Transform merges the updated records automatically. You can also remove a previously uploaded file if you no longer want to use that inventory data.

Build migration plan

The Migration planning job step within AWS Transform for migrations is a collaborative chat-based experience for planning large migrations. AWS Transform agents apply AWS Prescriptive Guidance to guide customers from analysis of on-premises data to finalized migration wave plans.

After the Discover on-premises data job completes successfully, AWS Transform uses the discovery data to group applications into migration waves. AWS Transform guides you through steps to analyze and scope your servers, group them into applications, generate move groups, and build migration waves.

AWS Transform supports iterative planning throughout the process:

- You can ask questions to better understand how AWS Transform analyzed your installed software, for example, server dependencies and network architecture.
- You can adjust scope within application groups and waves at any time.
- You can re-upload discovery data, and AWS Transform automatically processes, de-duplicates, and merges new records with existing data.
- When changes are detected, such as newly discovered dependencies or infrastructure additions, AWS Transform flags impacted dependency groups and provides recommendations for wave plan adjustments.
- Migration planning can also make use of unstructured text data to enrich the planning process.

There are four migration planning stages:

- In **Scope and analyze**, you can review your discovery data, ask questions about your software and network environment and determine the resources in scope for migration.
- In **Group apps**, you can provide a combination of business and technical rules regarding, for example, hostname analysis, network dependencies, and business rules, so that migration planning can group your infrastructure into applications. If you already have an inventory of your applications, migration planning can use that instead.
- In **Generate move groups**, you can give migration planning your technical and business requirements so that it can determine which applications must be moved together. Technical

dependencies include databases, message queues, or other resources shared between multiple apps. Business and operational dependencies include business criticality, RPO and RTO, data center location, and application owners.

- Finally, in **Build waves**, you can give migration planning context about your timelines and priorities so it can build a wave plan that you can migrate. You can select move groups for inclusion in a wave based on factors such as priority score, move group size, user counts, and application complexity.

Migration planning terminology

Migration wave

A logical group of applications that are migrated together. Migration waves are comprised of one or more move groups.

Move group

A set of co-dependent applications that must be moved together. They might have technical dependencies such as a shared database or business dependencies such as supporting a shared business function.

Dependency

A relationship between systems. There are several types of dependencies:

- **Critical or hard dependencies**. Systems cannot operate without the dependency. Common examples include applications that depend on databases, other applications, or services.
- **Soft dependencies**. Not critical for the operation of the system. Common examples include latency-insensitive dependencies that can be migrated independently.
- **Non-technical dependencies**. Business, organizational, operational, and compliance dependencies related to your organization and its priorities. Examples include shared business function and organizational ownership.

Workflow

Migration planning is an interactive and iterative workflow. You can go back and make changes to previous steps at any time. A typical migration planning workflow is:

1. Migration planning starts by summarizing the discovery data that is available. Review the available data and return to the discovery step at any time to provide additional data.
2. Within the scoping and analysis step you can ask questions about your on-premises environment to validate the data you have collected. Example questions include:
   a. List my servers by operating system
   b. Summarize my on-premises network topology
   c. List the most common technologies running in my environment
3. As you analyze your environment, if you identify servers that should not be in scope for migration, you can tell AWS Transform to exclude those resources. Examples of this include:
   a. Remove all servers which have *legacy* in their hostname
   b. Remove all servers in the 10.0.2.0/24 subnet
   c. Remove all servers running versions of Windows older than 2022
4. After you have sufficiently explored your environment and determined your migration scope you can tell AWS Transform to move to the next migration planning step.
5. The next step is application grouping. If you already have your servers mapped to applications, you can tell AWS Transform to use that mapping and skip this step. If you do not have your applications predefined you can provide the technical and business logic that defines your applications. AWS Transform guides you through the application grouping process and suggests the data points you can provide to group your servers into applications. The more information you can provide about your on-premises applications, the more effectively AWS Transform can group your servers into apps. After you have provided sufficient information, you can then instruct AWS Transform to perform application grouping.
6. After application grouping has been performed, review the application groups. You can instruct AWS Transform to make any necessary changes, for example:
   a. Move server example-server to application-5
   b. Rename application-5 "HR App Test Environment"
   c. Remove all Linux servers from IIS Dev Farm
7. After your apps are grouped, instruct AWS Transform to move to the next step.
8. The next step is move grouping. In the move grouping step you identify applications that must be moved together. Provide context around your technical and non-technical dependencies. AWS Transform guides you through the process and suggests data points that you can provide to group your apps together. There are several considerations to make at this stage including:
   a. What should be the target size of move group?

b. Do you want to combine environments, for example dev, test, and prod, for each app, or split them?

c. How do you want to consider network dependencies? Are all dependencies critical or can some dependencies be considered soft dependencies and be split across move groups?

9. After you have provided your rules for move grouping, instruct AWS Transform to execute your move grouping strategy. You can then review and modify your move groups. After you have reviewed your move groups you can instruct AWS Transform to move to the final migration planning step.

10. Wave planning is the final step within migration planning. In this step, you group your move groups into migration waves and prioritize those waves. Within the wave planning step, AWS Transform guides you through providing the business prioritization required to group your move groups into waves and then prioritize those waves. Considerations within wave planning include:

a. The business criticality of each of your move groups

b. The migration timelines and the timelines for each move group

c. The risks associated with each move group

d. The number of servers to migrate per wave

11. After you have provided sufficient guidance on how to group into waves, instruct AWS Transform to execute the wave planning. You can then review your waves and modify them.

12. After you have finalized your wave plan, you can complete migration planning and move to execution. You can return to migration planning at any time to refine and iterate on your plan.

13. For each wave, you can assign a migration strategy: *rehost* (migrate servers to Amazon EC2) and *containerize* (containerize source code and deploy to Amazon Elastic Container Service or Amazon Elastic Kubernetes Service). When you assign a wave the strategy *containerize*, AWS Transform runs the source code containerization workflow for that wave during migration execution. For more information, see *Source code containerization*. To get AWS-recommended strategies across the 7Rs framework before you assign them, see the section called “7Rs recommendations”.

Migration strategy (7Rs) recommendations

During migration planning, AWS Transform can analyze your discovered inventory and recommend a migration strategy, one of the industry-standard *7Rs*, for every server and application. Each recommendation includes a suggested AWS target service, a confidence score, and the reasoning

behind it. This gives you a defensible starting point for your wave plan and surfaces modernization opportunities that you might otherwise miss.

The seven strategies are *rehost*, *replatform*, *refactor*, *repurchase*, *retire*, *retain*, and *relocate*. AWS Transform reads signals in your data, such as operating system, utilization, and workload type, to make explainable recommendations. For example, it can:

- Recommend retiring idle or end-of-life servers.
- Move SQL Server or open-source databases to Amazon RDS.
- Shift email and collaboration workloads to Microsoft 365.
- Retain specialized legacy systems.
- Rehost the remaining workloads on Amazon EC2.

Recommendations roll up from individual servers to applications and across your entire portfolio, with a criticality view by strategy and wave-level summaries.

To generate a migration strategy (7Rs) report:

1. Complete the *Discover on-premises data* step and group your servers into applications. Both are required so that AWS Transform can provide per-server and per-application recommendations.
2. In the migration planning chat, ask AWS Transform for a migration strategy (7Rs) report. For example, *Generate an R-strategy report*. After application grouping completes, AWS Transform also offers to generate the report for you.
3. AWS Transform analyzes your inventory, applies the 7Rs framework, and generates the report. You can provide details such as your organization name and any filters to scope the analysis.
4. Review the recommendations. To adjust a recommendation, tell AWS Transform in the chat. You can change a single server or application, or apply a change in bulk (for example, by operating system, environment, wave, or priority). AWS Transform updates the report with your changes.

By default, AWS Transform produces an interactive HTML dashboard that you can filter and explore. On request, you can also generate a PDF document to share with stakeholders, or a Microsoft PowerPoint (PPTX) slide deck to present. The PDF and PowerPoint outputs are point-in-time snapshots of the same analysis. The report includes:

- A recommended strategy and AWS target service for every server and application
- Confidence scores and reasoning

- Highlighted modernization opportunities
- A criticality view by strategy
- Wave-level summaries
- Risk flags

The strategy recommended in this report complements the per-wave strategy that you assign during wave planning. For source code containerization, see *Source code containerization*.

Diagramming and reporting

During migration planning, AWS Transform turns your discovered inventory and planning results into visual *diagrams* and analytical *reports*. You generate them conversationally in the planning chat. For example, "Show me a network topology diagram" or "Generate a risk report." AWS Transform builds them from the data you have already provided, with no setup required.

Diagrams are interactive and let you explore your environment visually. Available diagrams include:

- *Network topology* – An interactive map of your servers and their network relationships, designed to scale to large environments of thousands of nodes.
- *Application dependency* – Directed application-to-application relationships derived from your observed network connections.
- *Wave Gantt chart* – A timeline view of your migration waves showing schedules, dependencies, milestones, and progress (available after wave planning).
- *General charts* – Pie charts, bar charts, histograms, treemaps, and summary tables over the dimensions of your data that you choose.

Reports provide narrative analysis and recommendations over your migration data. Available reports include:

- *Risk assessment* – Scores each application across technical complexity, dependency risk, migration readiness, and operational risk, with per-application, per-wave, and portfolio-wide summaries.
- *Migration strategy (7Rs)* – Per-server and per-application strategy recommendations. For more information, see the section called “7Rs recommendations”.
- *General and custom reports* – Analytical reports for requests such as executive readiness summaries, cost-versus-risk tradeoffs, and containerization-candidate assessments.

Some outputs depend on how far you are in planning. For example, application dependency diagrams use your network connection data. The wave Gantt chart and wave-level report summaries become available after wave planning completes. If a prerequisite is missing, AWS Transform tells you what is needed and can run the required step first.

AWS Transform delivers diagrams as interactive HTML that you can filter and explore. By default, AWS Transform delivers reports as a self-contained HTML file. On request, AWS Transform can also generate a PDF document to share with stakeholders or a Microsoft PowerPoint (PPTX) slide deck to present. PDF and PowerPoint outputs are point-in-time snapshots of the same analysis.

Connect target AWS accounts and regions

Connect AWS Transform to your target AWS environment so the agent can deploy infrastructure, migrate networks, and rehost servers on your behalf. This involves three steps: select your migration type, provide your MAP agreement details (if applicable), and set up the connector. These settings apply across all migration stages, including network migration, landing zone, and server rehost.

Step 1: Migration type selection

AWS Transform supports both single-account and multi-account migrations. Choose the option that matches your target environment:

- **Single-account migration** – All workloads migrate to one target AWS account. The connector target account and the target account are the same.
- **Multi-account migration** – Migrate workloads across multiple accounts in your organization from a single workspace. The connector must be connected to the organization management account or a Delegated Administrator (DA) account registered for both AWS Transform MGN and CloudFormation StackSets.

Step 2: MAP agreement

If your migration is part of the **AWS Migration Acceleration Program (MAP 2.0)**, provide your Migration Program Engagement (MPE) ID. This is a 10-character code using uppercase letters and digits (for example, ABCDE12345). When you provide your MPE ID, the MAP tag is applied to all resources created across network migration, landing zone, and server rehost stages. The tag format is:

- **Key:** `map-migrated` **Value:** `mig`*`MPE_ID`*

You must apply MAP tags to receive MAP credit. For more information about MAP, see AWS Migration Acceleration Program.

Step 3: Connector configuration

You use the target account connector to connect your migration job to the AWS environment where your workloads reside after migration. Before you begin, verify that your target AWS account has the necessary permissions, quotas, and configurations to support your migrated infrastructure.

When you approve the connector request, you grant AWS Transform permissions to:

- Manage Amazon S3 bucket operations (read/write) for VMware migration, along with access to AWS Migration Hub and AWS Application Migration Service (MGN). This includes permissions for the following items, all restricted to resources within the target account that are tagged with `CreatedBy:AWSTransform` or `CreatedFor:AWSTransform`:
  - Manage migration waves.
  - Manage network configurations (Amazon EC2, VPC, Transit Gateway, Direct Connect, Load Balancers, Network Firewall).
  - Manage CloudFormation stack deployments.
  - Perform automated agent installations through Systems Manager.
- Migrate your on-premises workloads to the target AWS account and Region by using the information stored in the discovery Region.
- Provision and manage landing zone infrastructure in the target AWS account and Region. This includes permissions for the following items, restricted to resources that are tagged with `CreatedBy:AWSTransform` where applicable:
  - Perform Amazon S3 bucket operations (create, read, write, delete) for buckets that start with `transform-vmware-landing-zone-`.
  - Manage CloudFormation stack deployments and change sets for landing zone stacks.
  - Perform AWS Control Tower operations. You can manage landing zones, enable baselines, and enable controls.
  - Manage AWS Organizations. You can create and manage organizational units, create accounts, and move accounts.

- Manage service control policies (SCPs) through AWS Control Tower.
- Manage AWS Service Catalog provisioning artifacts.

> **Note**
>
> Connector types might be updated when new features require permission changes. The current version for the target account connector type is 2.0. When you create a new connector, it uses the latest version.

Before you set up the connector, understand the account roles involved in your migration:

Account	Description
AWS Transform account	Any member account in your AWS Organization where you set up AWS Transform. This is where your AWS Transform workspace runs. It does not need to be the management account.
Connector target account	The account your AWS Transform connector is configured to. This depends on your migration type: • **Single-account migration** – Connect to the account you are migrating workloads to. The connector target account and the target account are the same. • **Multi-account migration** – Connect to the organization managemen t account or a Delegated Administrator (DA) account. The DA account must be registered as delegated administrator for both MGN and CloudFormation StackSets in your AWS Organization. AWS Transform checks whether the connected account is the management account or a DA account and adjusts its behavior accordingly.
Target account	The AWS account where your workloads are migrated to. In a single-account migration, this is the same as the connector target account. In a multi-account migration, these are the individual member accounts receiving the migrated workloads.

Using a delegated administrator account

For multi-account migrations, you can also use a Delegated Administrator (DA) account instead of the organization management account. A DA account follows the principle of least privilege by limiting the scope of permissions required for migration operations.

For building landing zone:

To make an account a delegated administrator for AWS Organizations, save the following policy as `organizations-delegation-policy.json` and run the following command from the management account. Replace *`connector-account-id`* with the account that you intend to connect.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAWSTransformOrganizationsRead",
      "Effect": "Allow",
      "Principal": {
        "AWS": "<connector-account-id>"
      },
      "Action": [
        "organizations:ListRoots",
        "organizations:DescribeOrganization",
        "organizations:DescribeAccount",
        "organizations:DescribeOrganizationalUnit",
        "organizations:DescribePolicy",
        "organizations:ListAccountsForParent",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListParents",
        "organizations:ListTagsForResource"
      ],
      "Resource": "*"
    }
  ]
}
```

```
aws organizations put-resource-policy --content file://organizations-delegation-
policy.json
```

Important

This command replaces any existing delegation policy. If your organization already has a delegation policy, add this statement to it instead. To view the current policy, run `aws organizations describe-resource-policy`.

For network and server migration using a delegated administrator

If your plan also includes network migration and server migration, and you use a DA account rather than the management account, you must also register the DA account as a delegated administrator for MGN and CloudFormation StackSets. Run the following commands from the management account. Replace *DA-account-id* with your DA account ID.

```
aws organizations register-delegated-administrator \
  --account-id <DA-account-id> \
  --service-principal mgn.amazonaws.com

aws organizations register-delegated-administrator \
  --account-id <DA-account-id> \
  --service-principal member.org.stacksets.cloudformation.amazonaws.com
```

Note

You can enable trusted access for MGN and CloudFormation StackSets only from the management account. You can register up to five accounts as delegated administrators.

The key difference between the two options is:

- **Management account** – Can enable trusted access for MGN and CloudFormation StackSets across the organization. AWS Transform calls CloudFormation StackSets APIs with `CallAs: SELF`.
- **Delegated Administrator account** – Can manage MGN source servers, launch instances, and deploy CloudFormation StackSets across member accounts. AWS Transform calls CloudFormation StackSets APIs with `CallAs: DELEGATED_ADMIN`.

For more information, see Delegated administrator for MGN in the *MGN User Guide*.

IAM roles created during connector setup

During connector setup, AWS Transform creates the following IAM role in your target account:

- `AWSTransform-Connector-role-` – Created when you set up the target account connector, in the management account or the delegated administrator account of your AWS Organization. This role allows AWS Transform to connect to your target account and act on your behalf to run migration operations.

Target account connector setup

> **Important**
>
> During connector setup, an Amazon S3 bucket is created in your target AWS account. This bucket won't enforce HTTPS-only access (`SecureTransport`) by default. If you want the bucket policy to include secure transport, you must update the policy yourself. For more information, see Security best practices for Amazon S3.

To use an existing target account connector

1. In the **Job Plan** pane, expand **Choose target account**, and then choose **Create or select connectors**.
2. In the **Collaboration** tab, select an existing connector and then choose **Use connector**. If a connector is unavailable, its version isn't compatible with the job type you selected.

   > **Important**
   >
   > If you specify a connector with a target AWS Region that is different from the AWS Transform Region, AWS Transform will transfer your data across AWS Regions.

3. Choose **Continue**.

To create a new connector

1. In the **Job Plan** pane, expand **Connect target account**, and then choose **Create or select connectors**.
2. Specify the AWS account and AWS Region for your target, and then choose **Next**.

> **Important**
>
> If the target AWS Region differs from the discovery AWS Region, AWS Transform will transfer your data across AWS Regions.

3. Choose whether to use Amazon S3 managed keys for encryption. If you specify your own KMS key, you can use the default key policy or a less permissive one. For information about creating a KMS key, see Create a KMS key in the *AWS Key Management Service Developer Guide*.

   AWS Transform uses the `kms:DescribeKey` permission to verify the key exists, and `kms:GenerateDataKey` and `kms:Decrypt` to encrypt and decrypt job data in the Amazon S3 bucket. For more information, see Reducing the cost of SSE-KMS with Amazon S3 Bucket Keys.
4. Choose **Continue**.
5. Copy the verification link, share it with an administrator of the target AWS account, and ask them to approve the connection request.
6. After the administrator approves the request, select the newly created connector from the list and choose **Use connector**.
7. Choose **Send to AWS Transform**.

If you plan to modify the AWS Transform MGN template to enable post-launch actions, add the following permission to the target connector role. This JSON policy statement grants the `iam:PassRole` permission for the post-launch actions role. You can find the role name in the **Collaboration** tab after the connector is created. For information about adding permissions to a role, see Update permissions for a role in the *IAM User Guide*.

```
{
      "Sid": "MGNPostLaunchActions",
      "Effect": "Allow",
      "Action": [
        "iam:PassRole"
      ],
      "Resource": "arn:aws:iam::target-account-ID:role/service-role/
AWSApplicationMigrationLaunchInstanceWithSsmRole"
}
```

Supported target Regions

A migration target region is the AWS Region where migrated resources are deployed, including landing zones, network infrastructure, and server rehosting. When you create the connector, specify a target AWS Region. You can use any of the following AWS Regions:

- US East (N. Virginia)
- US East (Ohio)
- US West (N. California)
- US West (Oregon)
- AWS GovCloud (US-East)
- AWS GovCloud (US-West)
- Africa (Cape Town)
- Asia Pacific (Hong Kong)
- Asia Pacific (Taipei)
- Asia Pacific (Mumbai)
- Asia Pacific (Hyderabad)
- Asia Pacific (Tokyo)
- Asia Pacific (Seoul)
- Asia Pacific (Osaka)
- Asia Pacific (Singapore)
- Asia Pacific (Sydney)
- Asia Pacific (Jakarta)
- Asia Pacific (Melbourne)
- Asia Pacific (Malaysia)
- Asia Pacific (New Zealand)
- Asia Pacific (Thailand)
- Canada (Central)
- Canada West (Calgary)
- Europe (Frankfurt)
- Europe (Zurich)
- Europe (Ireland)

- Europe (London)
- Europe (Paris)
- Europe (Stockholm)
- Europe (Milan)
- Europe (Spain)
- Israel (Tel Aviv)
- Mexico (Central)
- South America (São Paulo)

> **Important**
>
> If you specify a target AWS Region that differs from the AWS Transform AWS Region, some of your data is transferred across AWS Regions.
> Note that your server replication data goes directly from your source environment to your target account and Region.
> You can specify AWS GovCloud (US-East) and AWS GovCloud (US-West) as migration targets only when you use AWS Transform in the AWS GovCloud (US) partition. These Regions aren't available as targets for workspaces in commercial AWS Regions.

Build landing zone

AWS Transform guides you through designing and deploying an AWS landing zone as part of your migration project. A landing zone is a multi-account AWS environment that serves as the foundation for your workloads. It puts organizational boundaries, governance controls, and account structure in place before any workloads arrive. AWS Transform analyzes your migration inventory and business requirements. Based on this analysis, it recommends an Organizational Unit (OU) and account structure, applies recommended Service Control Policies (SCPs), and generates and deploys the infrastructure as code (IaC). What typically takes weeks of manual planning and configuration, AWS Transform can complete in a single conversation.

The landing zone agent automates two phases:

- **Foundation setup** – Establish the core landing zone structure: AWS Control Tower, foundational OUs, and core accounts.

- **Workload account design** – Design and create workload OUs and accounts based on your migration waves, business units, and environment separation requirements.

AWS Transform supports both greenfield environments (no existing landing zone) and brownfield environments (existing OUs and accounts already deployed). In brownfield scenarios, AWS Transform detects your existing organization structure and recommends only the changes needed to fill gaps against AWS best practices. You don't have to start from scratch or perform a manual gap analysis.

AWS Transform supports two operational modes based on your connector configuration and preferences:

- **Full deployment mode** – AWS Transform analyzes your organization, generates recommendations, creates Infrastructure as Code (IaC) artifacts, and deploys the landing zone infrastructure directly to your AWS Organization.
- **IaC-only mode** – AWS Transform analyzes your organization, generates recommendations, and produces IaC artifacts for your review and deployment through your own pipeline. The agent does not deploy infrastructure directly. This mode is useful if you have existing deployment pipelines (such as AWS Landing Zone Accelerator, Terraform, or custom CI/CD) or prefer to manage landing zone changes through your internal change management process. For more information, see the section called “Generate landing zone IaC without deployment”.

Connector setup

Before the agent can proceed, you connect it to your organization's Management Account or Delegated Administrator account. The connector type determines your operational mode, available capabilities, and the permissions you grant to AWS Transform:

Connector Type	Capabilities and Workflow
Management Account	Provides full deployment capabilities. AWS Transform analyzes your organization, generates recommendations, creates Infrastructure as Code artifacts, and deploys landing zone infrastructure directly to your AWS Organization. You choose whether AWS Transform deploys automatically or only generates IaC artifacts for your review.

Connector Type	Capabilities and Workflow
	Use this option when you want AWS Transform to provision landing zone infrastructure directly, or when you want to generate IaC artifacts while retaining the option for direct deployment. **Permissions** – When you approve the connector request, you grant AWS Transform permissions to provision and manage landing zone infrastructure in the target AWS account and Region. Where applicable, permissions are restricted to resources tagged with `CreatedBy:AWSTransform` and `ATWorkspace:{workspace-id}`. This includes: • AWS Control Tower operations (setting up and managing landing zones, enabling baselines and controls) • AWS Organizations management (creating and managing organizational units, creating accounts, and moving accounts) • Service Control Policy (SCP) management through AWS Control Tower • CloudFormation stack deployments and change set management for landing zone stacks • AWS Service Catalog provisioning artifact management • S3 bucket operations (create, read, write, delete) for buckets starting with `transform-vmware-landing-zone-`

Connector Type	Capabilities and Workflow
Delegated Administrator (DA) Account	Provides IaC-only mode. AWS Transform analyzes your organization, generates recommendations, and produces Infrastructure as Code artifacts that you can download and deploy through your own pipeline. Direct deployment is not available with DA connectors due to AWS Control Tower's technical limitation that Delegated Admin accounts cannot provision AWS Control Tower landing zones. Use this option when: • Your security or governance policies restrict programmatic access to the Management Account • You have an established Delegated Administrator account for AWS Organizations operations • You prefer to deploy landing zone infrastructure through your own deployment pipeline (AWS Landing Zone Accelerator, Terraform, CloudFormation, CDK, or custom CI/CD) If you require direct deployment capability, you can reconfigure your connector to target the Management Account. **Permissions** – In IaC-only mode, permissions are limited to read and describe operations on the AWS Organization to support analysis and gap assessment. To create a Delegated Administrator account, follow the instructions to create a delegated administrator.

When you create the connector, you specify a target AWS Region. This is the Region where Control Tower is enabled, and it serves as your Control Tower home Region. For more information about Control Tower Regions, see How AWS Regions work with AWS Control Tower.

At the start of the landing zone setup, AWS Transform retrieves your connector configuration and presents the AWS Organization management account ID and target Region for confirmation. AWS Transform also detects your connector type and operational mode, informing you whether you are

proceeding with full deployment or IaC-only capabilities. For more information, see the section called “Connectors”.

Important

IAM Identity Center Region dependency – AWS Transform requires AWS IAM Identity Center. Your connector Region must match both your AWS Control Tower home Region *and* your IAM Identity Center Region. If IAM Identity Center is already configured in your organization and the connector targets a different Region, AWS Control Tower initialization fails. For more information, see Considerations for IAM Identity Center customers in the AWS Control Tower User Guide.

Note

Delegated Admin connectors and Control Tower initialization: If you use a Delegated Administrator account connector, AWS Transform operates in IaC-only mode. If Control Tower has not yet been initialized in your organization, AWS Transform generates IaC artifacts to initialize Control Tower, which you can review and deploy through your pipeline.

Foundation setup

Foundation discovery and planning

At the start of foundation setup, AWS Transform performs discovery to understand your current organizational state and determines the best approach for your landing zone. The agent asks you the following key questions:

- **Does your organization already have AWS Control Tower deployed?** AWS Transform checks if Control Tower is initialized in your organization. If Control Tower is already running, AWS Transform analyzes the existing landing zone configuration.
- **What foundation approach do you want to use?** If Control Tower is not yet deployed, AWS Transform offers you two options:
  - **Control Tower-based landing zone:** AWS Transform deploys AWS Control Tower as the foundation for your landing zone. This provides the most comprehensive governance with managed controls, mandatory guardrails, and automated OU and account management. This is the recommended approach for most organizations.

- **Organizations-only landing zone:** AWS Transform creates a landing zone using only AWS Organizations without deploying Control Tower. This gives you foundational OU structure and account organization but does not include Control Tower's managed controls and automated guardrails. Choose this option if your organization has specific requirements that prevent Control Tower deployment.

Based on your responses, AWS Transform tailors the foundation setup workflow and recommendations to match your chosen approach. For Control Tower-based landing zones, the full governance foundation is deployed. For Organizations-only landing zones, AWS Transform focuses on OU structure, account organization, and manual SCP configuration.

Foundation setup

The foundation setup phase establishes the core landing zone infrastructure using AWS Control Tower. The landing zone agent builds the full foundation OU structure. Part of it is the AWS Control Tower baseline, which Control Tower provisions automatically in your management account. AWS Transform adds the rest based on AWS best practices. You can customize the structure before creation.

The following table shows the foundation structure and where each component comes from:

Component	Type	Source	Purpose
Root	Root	AWS Control Tower baseline	The top-level parent that contains all OUs in your landing zone.
Security	OU	AWS Control Tower baseline	Centralized audit logging and monitoring. Isolating these services in dedicated accounts is designed to help keep your audit trail separate from workload teams.
Audit	Account	AWS Control Tower baseline	Read-only access to all accounts for security and compliance review.
Log Archive	Account	AWS Control Tower baseline	Centralized, immutable logging for all AWS API activity and resource changes across your organization.

Component	Type	Source	Purpose
Infrastructure	OU	AWS best practice addition	Parent OU for shared infrastructure that supports your entire organization.
Network	OU	AWS best practice addition	Shared networking infrastructure, including Transit Gateway, VPN, and DNS. Centralizing these services helps reduce duplication and gives your network team a single place to manage connectivity across all accounts.
Shared Services	OU	AWS best practice addition	Common services used across the organization, such as identity, logging, and monitoring solutions that support multiple workload accounts.
Sandbox	OU	AWS best practice addition	Developer experimentation with spending limits and restricted access, without risking production resources.
Sandbox	Account	AWS best practice addition	The account developers use for experimentation.
Workloads	OU	AWS best practice addition	Parent OU for your application workloads. It's created empty. Workload OUs and accounts, such as Productio n, Non-Production, and optionall y Regulated, are designed in the next phase based on your migration requirements. For more information, see the section called “Workload account design”.

In addition to the Security OU and its accounts, the AWS Control Tower baseline includes the following governance resources:

- **Mandatory controls (guardrails)** – Preventive and detective controls applied across your organization to enforce baseline governance policies. These cannot be disabled.
- **IAM Identity Center directory** – A cloud-native directory with preconfigured groups and single sign-on access for your landing zone users. For more information, see AWS IAM Identity Center.

> **Note**
>
> The Security OU with the Audit and Log Archive accounts is created as part of the AWS Control Tower setup. These accounts cannot be renamed or replaced after initial setup. AWS Transform creates the best practice additions separately after you confirm the structure.

Control Tower uses CloudFormation StackSets to deploy and manage its baseline resources consistently across all accounts and Regions in your organization. Don't modify or delete Control Tower managed resources outside of supported methods. Doing so can cause your landing zone to enter an unknown state.

In brownfield scenarios, AWS Transform compares your existing foundation against this structure and reports only the gaps. For example: "Your foundation has Security and Infrastructure OUs but no Sandbox OU."

Account email convention

AWS requires a unique email address for each account. These emails receive important notifications for the account. AWS Transform uses plus addressing to generate unique account emails from a single mailbox.

Format: `prefix+account-name@domain`

You provide a prefix (for example, `aws-admin`) and a domain (for example, `acme.com`), and AWS Transform derives all account emails automatically. For example:

- Audit account: `aws-admin+audit@acme.com`
- Log Archive account: `aws-admin+log-archive@acme.com`
- Sandbox account: `aws-admin+sandbox@acme.com`

In brownfield scenarios, AWS Transform inspects existing account emails to infer the plus-addressing convention already in use and offers to continue with the same pattern.

Service Control Policies (SCPs)

SCPs are organization-level permission guardrails that set the maximum permissions for all accounts in your AWS Organization. They don't grant access — instead, they define boundaries that no one in the account can exceed, even account administrators.

As part of the Control Tower deployment, baseline guardrails are applied automatically. AWS Transform also recommends additional SCPs designed to help strengthen your organization's posture. These are based on AWS best practices for a minimum viable landing zone.

SCPs can be applied to the Infrastructure, Sandbox, and Workloads OUs. The Security OU is managed by Control Tower and cannot be targeted by SCPs through this tool.

> **Important**
>
> The Security OU is a foundation OU managed by Control Tower. You cannot add accounts, SCPs, or any resources to it through the landing zone agent.

In brownfield scenarios, AWS Transform checks which SCPs are already applied and only recommends ones that would fill gaps.

> **Note**
>
> **Delegated Administrator account limitation in brownfield scenarios:** When using a Delegated Administrator account connector in a brownfield environment where AWS Control Tower landing zone is already implemented, AWS Transform cannot view or modify existing SCPs managed by Control Tower. This is because Delegated Administrator accounts have limited permissions for SCP operations on Control Tower-managed policies. You can still generate IaC artifacts for new SCPs that you want to apply outside of Control Tower's management scope, but existing Control Tower SCPs will remain unchanged. If you need to modify Control Tower-managed SCPs, reconfigure your connector to target the Management Account.

Foundation deployment

After the foundation design is complete, you choose how to deploy:

- **Deploy for me** – AWS Transform deploys the foundation OUs, accounts, and SCPs to your AWS Organization.
- **I'll deploy on my own** – AWS Transform generates Infrastructure as Code (IaC) artifacts for download in your preferred format (see the section called “IaC formats”).
- **Design workload accounts first** – Skip deployment and continue to the workload account design phase. You can deploy everything together later.

Control Tower initialization

If AWS Transform detects that AWS Control Tower is not yet initialized in your organization, it provides the user with a link to the AWS Transform console page. Generating the operation in the link creates a CloudFormation stack to bootstrap Control Tower in the CloudFormation console for your target Region. After the stack is created, AWS Transform continues with the deployment.

Workload account design

In the workload account design phase, AWS Transform designs the OU and account structure for your application workloads based on your migration inventory, business requirements, and environment separation preferences.

Migration planning context

AWS Transform retrieves data from your migration planning phase, including wave plans, server-to-application mappings, and shared context. If migration planning data is available, AWS Transform displays a summary and asks you to confirm or adjust it. If no migration planning data is available, AWS Transform asks discovery questions directly.

Discovery

AWS Transform asks questions to understand your workload requirements. You can skip any question. Topics include:

- Number of business units or teams using AWS
- Industry and any applicable frameworks (HIPAA, PCI-DSS, SOC2, FedRAMP)

- Whether workloads handle sensitive data (PII, PHI, financial)
- Environment separation preferences (dev/test/staging/prod as separate accounts or shared)
- Workload isolation requirements
- Business applications and their purposes
- Server grouping into applications
- Cost tracking and allocation needs (by business unit, project, environment)
- Expected growth in the next 12–24 months
- Account strategy preference (single app per account, grouped, or environment-based)

Proposed workload structure

Based on your answers and migration planning data, AWS Transform proposes an OU and account structure under the Workloads OU. The proposal includes the reasoning behind each design decision.

AWS Transform follows these design principles:

- All servers in a migration wave go to the same account — waves cannot be split across accounts. This is a rehost limitation during wave execution.
- If you request isolated environments, AWS Transform creates Workloads/Production and Workloads/Non-Production sub-OUs.
- If applicable frameworks are identified, AWS Transform creates Workloads/Regulated and Workloads/Standard sub-OUs.
- If multiple business units require different governance, AWS Transform creates business-unit-specific OUs under Workloads.
- Critical or sensitive-data applications get a single app per account. In this case, you might be asked to iterate on your wave plan.
- Tightly coupled applications with shared dependencies are grouped in one account.

Each proposed account includes: name, purpose, target OU, and business unit. AWS Transform shows the naming convention being used (for example, `<business-unit>-<environment>-<workload>`).

You can review and modify the proposed structure before AWS Transform applies the changes. After applying, you can iterate — making additional changes until you are satisfied.

Workload SCP configuration

After the workload structure is created, AWS Transform presents available SCPs and asks if you want to apply any to your workload OUs. You select which SCPs to apply and to which OUs. AWS Transform applies the SCPs and shows the updated organization tree with an SCP summary table.

Workload deployment

After the workload design is complete, you choose how to deploy:

- **Deploy for me** – AWS Transform deploys the workload OUs, accounts, and SCPs to your AWS Organization.
- **I'll deploy on my own** – AWS Transform generates IaC artifacts for download in your preferred format (see the section called “IaC formats”).

IaC formats

When you choose self-deployment, AWS Transform generates Infrastructure as Code artifacts in the following formats:

- **AWS Cloud Development Kit (AWS CDK)** – TypeScript project for programmatic infrastructure deployment.
- **HashiCorp Terraform** – Generates HashiCorp Configuration Language (HCL) templates for managing landing zone resources.
- **Landing Zone Accelerator (LZA)** – Configuration YAML files based on LZA Universal Configuration version 1.1.0. These enterprise-ready templates work with the Landing Zone Accelerator on AWS to establish multi-account AWS environments. The generated files include preconfigured settings for governance, organization structure, and networking. These settings align with AWS best practices. To learn more, see LZA Universal Configuration.

> **Note**
>
> When you deploy through the Landing Zone Accelerator (LZA) pipeline, your AWS Transform account and LZA installation must be in the same AWS Organization. Deployment fails if the Organizations IDs used in AWS Transform and LZA don't match.

To learn how to set up your LZA installation using Organizations, see AWS Organizations based installation.

After you select a format, AWS Transform generates the artifacts and makes them available for download.

To verify the downloaded file hasn't been corrupted or tampered with, generate and download a checksum, then compare it to a locally generated hash using:

```
openssl dgst -sha256 -binary <file.zip> | base64
```

Deployment approvals process

Landing zone deployment requests require explicit approval before execution. When you submit a deployment request, it automatically routes to authorized approvers through the AWS Transform Approvals tab.

Approvers review CloudFormation templates and landing zone configurations. Only users with the Admin role in AWS Transform can approve deployment requests. Each submission triggers a new review cycle, and deployments proceed only after receiving confirmation.

If an approver denies your request, contact them directly to discuss necessary modifications. The system tracks all approval decisions for audit purposes and maintains deployment history.

Tag landing zone resources

AWS Transform automatically tags all generated resources with `"CreatedBy": "AWSTransform"` along with definition and execution IDs for tracking purposes.

Automatic tags

All landing zone resources receive the following tags:

- `CreatedBy` – AWSTransform
- `ATWorkspace` – Workspace identifier

> **Note**
>
> If your migration is part of the AWS Migration Acceleration Program (MAP 2.0), you can include the required MAP tag. The key is `map-migrated` and the value is `migMPE_ID`, where MPE_ID is your MPE identifier. The MAP tag is requested during the connector setup phase, and AWS Transform applies these tags during landing zone deployment.

Reversing changes

Only non-deployed elements can be removed. After an OU or account is deployed, it cannot be removed through the landing zone agent.

When removing elements, order matters — you must remove children before parents:

1. Remove accounts first (by email).
2. Remove SCPs from OUs.
3. Remove child OUs — an OU cannot be removed if it still has accounts or nested OUs.

Related resources

- the section called “Connect target AWS accounts and regions”
- the section called “Migrate network”
- AWS Control Tower User Guide
- AWS Organizations User Guide
- AWS IAM Identity Center User Guide
- CloudFormation User Guide
- Landing Zone Accelerator on AWS
- AWS Well-Architected Framework
- AWS Prescriptive Guidance: Building Landing Zones

Generate landing zone IaC without deployment

The IaC-only flow enables you to use AWS Transform landing zone agent for analysis, recommendations, and Infrastructure as Code generation without direct deployment. This mode is available in two scenarios:

- **Delegated Administrator connector:** When you connect via a Delegated Administrator account, AWS Transform operates exclusively in IaC-only mode. This allows organizations with strict governance policies to use AWS Transform without granting access to the management account.
- **Management Account with IaC-only preference:** Even with a Management Account connector, you can choose IaC-only mode at the start of your job if you prefer to deploy through your own pipeline (AWS Landing Zone Accelerator, Terraform, CloudFormation, CDK, or custom CI/CD).

IaC-only workflow overview

In IaC-only mode, AWS Transform performs the same analysis and generates the same recommendations as full deployment mode, but outputs Infrastructure as Code artifacts instead of provisioning resources directly. The workflow includes:

- **Organization analysis:** AWS Transform discovers your existing organizational structure, identifies gaps against AWS best practices, and recommends landing zone configuration.
- **Foundation design:** AWS Transform proposes organizational unit (OU) hierarchy, account structure, and Service Control Policies (SCPs) based on your requirements.
- **Workload account design:** AWS Transform designs workload OUs and accounts based on your migration waves and business requirements.
- **IaC generation:** AWS Transform generates Infrastructure as Code artifacts in your preferred format. You download these artifacts and deploy them through your own pipeline.

Permissions in IaC-only mode

IaC-only mode requires minimal permissions compared to full deployment mode:

- **Read and describe permissions:** AWS Transform requires read-only access to your AWS Organization to discover existing structure, accounts, organizational units, and policies.
- **No write permissions needed:** Since AWS Transform does not provision resources, no write or create permissions are required for landing zone services like AWS Control Tower, Organizations, CloudFormation, or Service Catalog.

- **Delegated Administrator accounts:** When using a Delegated Administrator connector, these read-only permissions are sufficient for the agent to perform analysis and generate IaC artifacts, even though Delegated Administrators cannot deploy Control Tower landing zones directly.

Limitations in IaC-only mode

Be aware of the following limitations when using IaC-only mode:

- **No direct deployment:** AWS Transform does not provision landing zone infrastructure. You must deploy the generated IaC artifacts through your own pipeline.
- **Delegated Administrator and Control Tower:** When using a Delegated Administrator connector in a brownfield environment where AWS Control Tower is already deployed, AWS Transform cannot view or modify existing Control Tower-managed SCPs. Existing SCPs remain unchanged. You can still generate IaC for new SCPs outside Control Tower's scope. To modify Control Tower-managed SCPs, reconfigure your connector to target the Management Account.
- **No deployment step verification:** AWS Transform does not verify successful deployment of generated artifacts. After deploying through your pipeline, you are responsible for validating that landing zone infrastructure is created correctly and that all guardrails are in place.

Supported IaC formats

AWS Transform generates IaC artifacts in multiple formats:

- **AWS CDK:** TypeScript-based Infrastructure as Code for programmatic infrastructure deployment.
- **Terraform / HCL:** Terraform modules and HashiCorp Configuration Language (HCL) templates for managing landing zone resources.
- **Landing Zone Accelerator (LZA):** Configuration YAML files (accounts-config.yaml, organization-config.yaml, etc.) based on LZA Universal Configuration version 1.1.0. These enterprise-ready templates integrate with AWS Landing Zone Accelerator to establish multi-account AWS environments with governance, organization structure, and networking aligned to AWS best practices.

You can specify your preferred format upfront, and AWS Transform generates artifacts in that format without asking again during the workflow.

When to use IaC-only mode

Consider using IaC-only mode in these scenarios:

- **Strict governance policies:** Your organization restricts programmatic access to the management account. Using a Delegated Administrator connector with IaC-only mode bypasses this constraint.
- **Existing deployment pipelines:** You already manage infrastructure through AWS Landing Zone Accelerator, Terraform, or custom CI/CD pipelines. AWS Transform can generate artifacts that integrate with your existing workflow.
- **Review and customization:** You want to review and customize generated landing zone configuration before deployment.
- **Brownfield environments:** You have an existing landing zone and want to generate IaC for recommended changes without immediate deployment.
- **Change management processes:** Your organization requires all infrastructure changes to go through approved change management and deployment processes.

Migrate network

With AWS Transform, you can migrate your network to AWS in a fraction of the time it takes to design and deploy manually. AWS Transform uses an AI-powered agent to translate your source environment configuration into production-ready AWS network resources. You review and modify the generated configuration through a conversational interface before deployment.

AWS Transform supports two approaches to network migration. Choose the one that matches how much of your target network already exists in AWS:

- the section called “Map source network to new VPCs” – AWS Transform builds a complete target network from your source configuration, creating new VPCs, subnets, security groups, and connectivity. Choose this approach when you are provisioning your AWS network as part of the migration.
- the section called “Apply security posture to existing VPCs” – AWS Transform applies your source security groups and rules to VPCs that you have already provisioned in AWS, and leaves the rest of your network untouched. Choose this approach when your target VPCs and subnets already exist and you only need to bring over your security posture.

Map source network to new VPCs

Get a complete, production-ready target network generated from your source configuration in a fraction of the time. AWS Transform translates your source environment into AWS network resources, including VPCs, subnets, security groups, NAT gateways, transit gateways, elastic IPs, routes, and route tables, which you review and adjust before deployment. You can deploy directly with AWS Transform or choose self-deployment and receive Infrastructure as Code (IaC) in your preferred format: AWS Cloud Development Kit (AWS CDK), Landing Zone Accelerator (LZA), or HashiCorp Terraform.

Choose **Map source network to new VPCs** when you are provisioning your AWS network as part of the migration.

The AWS Transform agent guides you through the following steps, handling the analysis and generation while you make the decisions:

1. Upload your source network file.
2. Upload additional configuration files (optional, for RVTools environments).
3. Select a network topology.
4. Select a security groups mapping strategy.
5. Review and optimize your network.
6. Generate a network diagram (optional).
7. Configure resource tagging.
8. Deploy your network.

> **Note**
>
> For multi-account deployments, you must configure cross-account IAM roles and trusted access for AWS Organizations before you start the network migration. For more information about migration types, see the section called “Step 1: Migration type selection”.

Step 1: Source network mapping

AWS Transform generates target network infrastructure from a wide variety of source network configuration files. Upload one or more configuration files from your source environment, and AWS

Transform uses them to generate target networks, including Amazon VPCs, subnets, and security groups. AWS Transform does not generate firewall or load balancer resources. However, it can use configuration from these network elements as input to generate target networks.

AWS Transform accepts configuration files from the following source types:

- **Software Defined Networks (SDN):** Import/Export for VMware NSX network virtualization or Cisco ACI config for Cisco Application Centric Infrastructure.
- **VMware vSphere networks:** RVTools. When you use RVTools files, AWS Transform generates Amazon VPC configurations only. Security group configurations require additional input from firewall or software-defined network files. For more information about security group generation from additional files, see Additional configuration files.
- **Networks based on firewall configuration data:** Export files from Palo Alto Networks Firewall, Fortinet FortiGate Firewall, or Cisco ACI. For more information about supported versions and extraction instructions, see Configuration file extraction.
- **Hybrid networks that run both VMware and non-VMware workloads:** AWS Transform discovery tool or modelizeIT.
- **Other file types:** AWS Transform also accepts other network configuration files, such as Checkpoint and F5 configurations. If your configuration file is not one of the formats listed above, AWS Transform converts it automatically and uses it to generate target networks. This conversion can take up to two hours based on the file size and complexity.

> **Note**
>
> The maximum supported source network file size is 70 MB.

> **Note**
>
> To enable stale security group rule removal during network review, submit observed network traffic data from the AWS Transform discovery tool or modelizeIT alongside your source network configuration. For more information, see Guided network recommendations.

> **Warning**
>
> Download RVTools only from the official Dell site at https://www.dell.com/en-us/shop/vmware/sl/rvtools. Do not download RVTools from unofficial sources.

Each source network segment is mapped to its own distinct VPC. Network segmentation varies by source type:

- **vNetwork:** AWS Transform groups VMs by vSwitch and virtual LAN (VLAN). VLANs can appear under multiple vSwitches (except VLAN 0).
- **NSX networks:** AWS Transform segments the network based on Tier-1 routers, grouping the routers and collecting their segments.

Step 2: Additional configuration files

For RVTools source environments, you can optionally upload additional configuration files to enable security group generation. If you don't upload additional configuration files, no security groups are generated for your RVTools-based migration.

AWS Transform supports the following additional configuration file types. You can upload only one configuration file from one platform.

- Cisco Application Centric Infrastructure (ACI) provides network policy configurations.
- Palo Alto Networks provides firewall security policies.
- Fortinet FortiGate provides firewall security policies.

When you upload a firewall or Cisco ACI file, AWS Transform generates network infrastructure and security groups. When you upload an RVTools file alone, AWS Transform generates network infrastructure only.

For more information about supported versions and extraction instructions, see Configuration file extraction.

Step 3: Network topologies

During the network definition step, you select a network topology. You can choose the **Isolated VPCs** topology or the **Hub and Spoke** topology.

Isolated VPCs

What is deployed

Isolated VPCs are independent network environments that operate as separate units within AWS. Your VPCs are completely isolated, with no built-in communication pathways between them. This separation provides the highest level of network boundary protection.

AWS Transform creates the following resources:

- A dedicated VPC for each detected source network segment.
- Private subnets based on your source network configuration.
- Security groups (if you provided firewall or SDN configuration files).

Complete your setup

AWS Transform deploys the core network infrastructure for you. You complete the final connectivity and security configuration to match your organization's requirements.

To enable internet access for an Isolated VPC, complete the following steps:

1. Create an internet gateway and attach it to the VPC.
2. Create public subnets in each Availability Zone where you need internet access. Add a route for `0.0.0.0/0` pointing to the internet gateway. For more information about subnet configuration, see Subnets for your VPC.
3. Create NAT gateways in the public subnets (one per AZ for high availability). Allocate an Elastic IP for each NAT gateway.
4. Update your private subnet route tables. Add a route for `0.0.0.0/0` pointing to the NAT gateway in the same AZ.
5. Review your security group rules. Ensure outbound rules allow the traffic your workloads need (for example, HTTPS and DNS).

For VPC-to-VPC communication, set up VPC peering or a Transit Gateway and update route tables in each VPC to route traffic to the peering connection or TGW attachment.

Hub and Spoke

In this model, an AWS Transit Gateway acts as the central hub that connects multiple workload VPCs (the spokes).

What is deployed

AWS Transform creates the following resources:

- **Spoke VPCs:** One VPC per detected source network segment, with private subnets and a Transit Gateway attachment.
- **Inspection VPC:** Hosts your firewall appliance for traffic inspection. All cross-VPC traffic is routed through this VPC. The Transit Gateway attachment uses appliance mode, which ensures traffic flows symmetrically through the same appliance in both directions of a connection.
- **Inbound VPC:** Handles traffic entering your network from the public internet (north-south inbound). Includes an internet gateway and public subnets across multiple Availability Zones.
- **Outbound VPC:** Handles traffic leaving your network to the public internet (north-south outbound). Includes an internet gateway, NAT gateways with elastic IP addresses in each Availability Zone for high availability, and private subnets for the Transit Gateway attachment.
- **Transit Gateway route tables:** Two route tables steer traffic through the Inspection VPC:
  - The *Uninspected* table is associated with spoke VPCs, Inbound VPC, and Outbound VPC. It routes all traffic (0.0.0.0/0) to the Inspection VPC attachment and is the default association route table.
  - The *Inspected* table is associated with the Inspection VPC. It contains propagated routes from all spoke VPCs and is the default propagation route table.

For multi-account deployments, the Transit Gateway is shared across accounts through AWS Resource Access Manager (RAM).

Traffic flow

All cross-VPC traffic follows this path:

1. Traffic from a spoke VPC is sent to the Transit Gateway (default route 0.0.0.0/0).
2. The Uninspected route table routes the traffic to the Inspection VPC.
3. Your firewall in the Inspection VPC inspects the traffic and forwards it back to the Transit Gateway.
4. The Inspected route table routes the traffic to the destination spoke VPC using propagated routes.

For outbound internet traffic, the Inspected route table routes traffic to the Outbound VPC. NAT gateways translate private IP addresses before the traffic is forwarded to the internet gateway. The Outbound VPC public route table includes specific routes for each spoke VPC Classless Inter-Domain Routing (CIDR) range back to the Transit Gateway. These routes enable return traffic to reach the correct spoke VPC.

Inbound internet traffic enters through the Inbound VPC's internet gateway and follows the same inspection path to reach spoke VPCs.

Complete your setup

AWS Transform deploys the core network infrastructure for you, including the Transit Gateway, spoke VPCs, and traffic routing. You complete the firewall configuration and inbound service setup to match your organization's security requirements.

> **Note**
>
> By default, cross-VPC traffic passes through the Inspection VPC without inspection. You must deploy a firewall to enable traffic inspection.

Deploy a firewall: Create additional subnets in the Inspection VPC for the firewall endpoints. AWS Transform creates subnets for the Transit Gateway attachment only. Route traffic from the TGW attachment subnets to the firewall endpoints, and from the firewall subnets back to the Transit Gateway. You can deploy AWS Network Firewall or a third-party appliance. For more information about deploying a firewall with a Transit Gateway, see Creating a firewall with a Transit Gateway.

Verify connectivity: After you deploy the firewall, test outbound internet access from a spoke VPC instance (for example, `curl https://aws.amazon.com`). You can use Reachability Analyzer to troubleshoot connectivity issues.

Set up inbound services: To host public-facing services, deploy an Application Load Balancer or Network Load Balancer in the Inbound VPC public subnets. Configure target groups that point to instances in your spoke VPCs through the Transit Gateway, and verify that the Inspected route table has return routes to the Inbound VPC.

If you want fine-grained control over the communication between the VPCs, choose the **Isolated VPCs** option and modify the generated network to create the specific communication paths you require.

Step 4: Security groups mapping

Choose how your source security policies translate to AWS security groups. AWS Transform creates security groups based on your source environment configurations. Security policies, security policy rules, gateway policies, and gateway policy rules are converted to security groups.

> **Important**
>
> AWS Transform creates security groups on a best-effort basis to match your source environment. Review and modify the generated security groups to ensure that they meet your company's needs and security policies.

Security group referencing

When security groups are generated, AWS Transform uses security group referencing where supported. Security group referencing sets security rules based on another security group ID rather than specific IP address ranges (CIDR blocks). This approach provides more flexible and maintainable security configurations.

Security group referencing follows these rules:

- Rules can reference other security groups within the same VPC, or in a VPC that is connected within the same Region.
- Cross-account references are supported.
- You cannot reference a security group in a non-connected VPC or across Regions.
- For connected VPCs, only inbound rules support cross-VPC security group references. Outbound rules must use CIDR-based rules.

How AWS Transform creates security group rules depends on your chosen network topology:

- **Hub and Spoke:** Transit Gateway provides network connectivity between VPCs. AWS Transform uses referencing for both within-VPC and cross-VPC/cross-account ingress rules. Cross-VPC/cross-account egress (outbound) rules use CIDR-based rules.
- **Isolated VPCs:** Your VPCs have no network connectivity between them. AWS Transform uses referencing for within-VPC rules only. All cross-VPC and cross-account rules use CIDR-based rules.

CIDR-based rules are also used when source configurations are not symmetric.

Choose one of the following security group mapping strategies:

- **MAP:** Translates security rules from your source environment to AWS security groups and rules. Use this option for migrations using static IP addressing.
- **MAP_DHCP (Translate with DHCP support):** Translates security rules from your source environment with DHCP compatibility. DHCP assigns IP addresses dynamically from the subnet's CIDR range. As a result, cross-VPC egress rules are widened to match the full destination subnet CIDR. A narrower CIDR would block DHCP-assigned IPs that fall outside that range. Review these rules post-migration.

  Use this option for DHCP support with cross-VPC Transit Gateway communication. Also works with static IPs, but might produce broader rules than MAP.
- **SKIP:** Does not translate security rules. Configure AWS security groups manually post-migration. Works with both static IP and DHCP environments. For RVTools source environments without additional configuration files, AWS Transform automatically uses SKIP.

> **Note**
>
> Your mapping strategy determines your IP assignment options. MAP supports static IP only. MAP_DHCP and SKIP support both static and DHCP.

IP migration approaches

You have two network configuration choices for your migration:

For **network range selection**, choose one of the following options:

- **Keep Existing Ranges (IP Address Ranges Retention):** Keep original IP address ranges during migration. Ideal for migrations where you move applications to AWS without modification (lift-and-shift), especially with legacy applications that have hard-coded IP dependencies or existing firewall rules.
- **Update to new IP ranges (CIDR update):** You can modify each VPC CIDR range during migration, and AWS Transform automatically propagates changes to subnets, route tables, and security groups.

For **IP address assignment**, choose one of the following options:

- **Fixed IP addresses (Static):** AWS Transform assigns static IPs based on the CIDR. This is best for applications that require predictable network behavior, DNS management, or IP-based access control. IPs persist across instance restarts through Elastic Network Interfaces (ENIs), which are virtual network cards attached to your instances.
- **Dynamic IP assignment (AWS DHCP):** Automatically assign IPs from subnet pools at instance launch. Optimal for applications that are designed to run in the cloud and auto-scaling workloads. Reduces operational overhead but requires applications to use DNS or service discovery.

You can combine either range selection with either IP assignment method.

> **Note**
>
> The IP address assignment strategy is set at the wave level. You can assign different strategies to specific servers by customizing the wave file. For example, if you chose a static IP address approach for the wave but want to assign a dynamic approach to a specific server, you would use `[RESET_VALUE]` as described in Editing your configuration in the *MGN user guide*.

Step 5: Review and optimize your network

After AWS Transform generates your target network configuration, you can review the on-premises network segments converged to AWS infrastructure. Use the visual interface to review your network, and the chat interface to make changes and receive guided recommendations. AWS Transform performs cascading impact analysis and implements the required changes to maintain network consistency and compliance with best practices. You can also ask AWS Transform to analyze your network and suggest optimizations. For more information, see Guided recommendations.

Existing VPCs in your target account

If your target account already contains VPCs, from previous migration phases or parallel infrastructure projects, AWS Transform automatically detects them and displays them alongside your mapped VPCs during the review process. For multi-account migrations, AWS Transform detects existing VPCs across all accounts in your AWS Organization.

This visibility helps you understand how your planned network relates to your existing infrastructure, identify potential CIDR conflicts, and make informed decisions before deployment.

> **Note**
>
> AWS Transform detects existing VPCs only (not subnets or other resources). Detection is read-only. AWS Transform does not modify your existing VPCs.

Optimize your network

> **Note**
>
> These operations apply to workload VPCs only. For appliance VPCs in the Hub and Spoke topology (Inspection, Inbound, Outbound), only Change IP address is supported.
> You can't undo Delete, Merge, and Split operations. Review your configuration carefully before you apply these changes.

You can perform the following operations on VPCs:

- **Delete:** Permanently remove a VPC from the configuration. Use this for obsolete network segments that should not migrate to AWS.
- **Exclude:** Temporarily remove a VPC from the migration for phased migration strategies. Excluded VPCs are not deployed but can be re-included later.
- **Include:** Add a previously excluded VPC back into the migration.
- **Merge:** Combine two VPCs into one. The first VPC keeps its identity and absorbs all subnets from the second VPC. Security groups are moved to the merged VPC and their associations are rebuilt accordingly. The first VPC's CIDR expands to the smallest CIDR range that contains both original CIDRs, and routing is updated automatically. The second VPC is removed from the configuration.

  Merge requirements:

  - Subnet CIDRs must not overlap between the two VPCs.
  - The merged CIDR must not exceed /16.
  - For multi-account deployments, both VPCs must be assigned to the same account.
- **Change IP address:** Change the base IP address of a VPC CIDR while keeping the same prefix length. AWS Transform automatically translates all subnet CIDRs by the same offset.

For example, changing a VPC from `10.0.0.0/16` to `10.20.0.0/16` shifts a subnet from `10.0.1.0/24` to `10.20.1.0/24`.

Security group rules that exactly match the old VPC CIDR are updated automatically. Rules that partially overlap or don't match the old CIDR are not changed. Review these rules after the change.

- **Rename:** Change the name of a VPC to align with your organization's naming conventions for cost allocation, compliance tracking, and operational standards.
- **Resize:** Change the prefix length of a VPC CIDR to expand or reduce the IP address range.
  - **Prefix length decrease** (more IPs, for example /20 to /16): Subnets still fit within the larger range. No subnet changes needed.
  - **Prefix length increase** (fewer IPs, for example /16 to /20): Subnets that fall outside the new range must be resized first using the subnet resize operation.

  Security group rules that exactly match the old VPC CIDR are updated automatically. Rules that partially overlap or don't match the old CIDR are not changed. Review these rules after resizing.

  Resize requirements:

  - The new CIDR must be between /16 and /28.
  - The new CIDR must not overlap with other VPCs in the network (Hub and Spoke topology).
  - When reducing the CIDR, all existing subnets must fit within the new CIDR. Resize subnets first if needed.
- **Split:** Divide a VPC into two VPCs based on CIDR boundaries you provide. Subnets are assigned to the new VPC whose CIDR contains them. Security groups are cloned to both new VPCs, but security group rule CIDRs are not automatically updated. Review your rules after splitting to ensure cross-VPC communication works as expected. The original VPC is replaced by the two new VPCs.

  Split requirements:

  - You must provide exactly two CIDR ranges.
  - The two CIDRs must not overlap.
  - Each CIDR must be between /16 and /28.
  - Every subnet must fit in exactly one of the two CIDRs. If any subnet does not fit, the operation is rejected.

You can perform the following operations on subnets:

- **Change IP address:** Change the base IP address of a subnet CIDR while keeping the same prefix length.
- **Delete:** Permanently remove a subnet from the configuration without affecting the parent VPC.
- **Resize:** Change the prefix length of a subnet CIDR to expand or reduce the IP address range.

  Subnet resize requirements:
  - The new CIDR must be between /16 and /28.
  - The new CIDR must not overlap with other subnets in the same VPC.
  - The new CIDR must be within the parent VPC CIDR.

You can perform the following operations on security groups:

- **Create:** Add a new security group or a new rule to an existing security group.
- **Configure:** Edit the settings of a security group or one of its rules, such as the name, description, protocol, port range, or direction.
- **Delete:** Permanently remove a security group or one of its rules from the configuration.
- **Associate a CIDR with a rule:** Set or update the CIDR range that a security group rule allows traffic to or from.
- **Associate a source VM with a rule:** Set or update the source virtual machine (VM) for a security group rule by its universally unique identifier (UUID). The rule then follows that workload instead of a fixed IP address.
- **Enable referencing for a rule:** Turn on security group referencing for a specific rule. The rule then allows traffic from another security group instead of a CIDR range.

After each operation, AWS Transform re-evaluates security group referencing, which might convert CIDR-based rules to security group references or vice versa. Review your security group rules after you make changes to verify they meet your requirements.

Guided network recommendations

AWS Transform automatically analyzes your mapped network and surfaces prioritized recommendations through the chat interface, identifying optimizations that would typically require manual review by network architects. Recommendations are based on your network data and require your confirmation before any changes are applied.

AWS Transform might recommend the following optimizations:

- **CIDR conflict resolution:** Flags overlapping CIDR ranges between your mapped VPCs and existing VPCs across all accounts in your AWS Organization. Conflicting VPCs are shown first. You can resolve conflicts by re-addressing the mapped VPC, excluding or deleting it, or acknowledging the conflict and resolving it yourself after deployment.
- **Naming standardization:** Flags VPC names that don't follow a consistent pattern (for example, names containing hardware references). AWS Transform asks for your cloud naming convention before proposing replacements.
- **Scope review:** Identifies network segments that might not need to migrate to AWS, such as legacy systems or constructs pending decommission. AWS Transform asks for your confirmation before excluding any construct.
- **VPC capacity right-sizing:** Surfaces VPCs where the CIDR appears oversized or undersized for the subnets it contains. AWS Transform presents the current capacity data and lets you decide whether to resize.
- **Security review:** Flags security group rules that allow unrestricted inbound traffic (0.0.0.0/0) for your review.
- **Stale security group rule removal:** Identifies unused inbound firewall rules migrated from your on-premises environment and suggests removing them, so you don't carry forward security exposure that no longer serves a purpose. This recommendation works as follows:
  - To identify unused rules, AWS Transform uses observed network traffic data collected by the AWS Transform discovery tool or modelizeIT. You must submit this traffic data alongside your source network input.
  - AWS Transform compares your migrated firewall rules against the observed traffic over the observation window captured in that data. It flags a rule as unused when no inbound traffic matches it. The absence of observed inbound traffic is a reliable signal that a rule is unused.
  - Without traffic data, AWS Transform cannot determine which rules are unused and does not suggest removals.
  - AWS Transform removes only unused ingress (inbound) rules.

  Review the suggested removals before you apply them to confirm that they align with your security policies.
- **VPC consolidation:** Identifies fragmented VPCs that appear separated by physical infrastructure limits rather than logical isolation requirements, and suggests merging them.

- **Application alignment:** Identifies VPCs that host applications from multiple environments, such as development and production workloads that share a single VPC. When the environments can be cleanly separated along subnet boundaries, so that no environment has workloads on both sides of the proposed split, AWS Transform suggests splitting the VPC along that boundary. Subnets and server IP addresses remain unchanged. Separating environments into dedicated VPCs follows AWS best practices for workload isolation and environment-specific access control.

  This recommendation requires the application and environment data from your migration plan, so it is available only after migration planning.

> **Note**
>
> All recommendations require your explicit confirmation before AWS Transform applies any changes. AWS Transform presents trade-offs when a recommendation affects multiple aspects of your network.

Step 6: Network diagram

After reviewing the generated VPC configurations, you can optionally generate a network diagram to visualize your network topology. AWS Transform supports the following diagram formats:

- **Mermaid code (.mmd):** This format produces a text-based diagram definition file that you can render with Mermaid-compatible tools.
- **Image (.png):** This format produces a rendered image of your network topology.

Step 7: Configure resource tagging

Your network resources are tagged for launch and replication. You can also add custom tags and AWS Migration Acceleration Program (MAP) tags.

Automatic tags for launch and replication

AWS Transform automatically tags your migrated network resources (VPCs, subnets, security groups, and route tables) with the following tags:

- **Key:** `CreatedBy` **Value:** `AWSApplicationMigrationService`
- **Key:** `ATWorkspace` **Value:** *`workspace-id`*

These tags allow you to use the VPC and subnet to launch test and cutover instances in AWS.

> **Note**
>
> Your migrated VPCs and subnets don't include internet connectivity by default, so they aren't suitable as staging areas for replication.

To also use the VPC and subnet as a staging area (replication), manually add the following tags:

- **Key:** `CreatedFor` **Value:** `AWSTransform`
- **Key:** `ATWorkspace` **Value:** *`workspace-id`*

You can also apply these tags to any existing AWS network resource to make it available for replication.

Find your workspace ID in the AWS Transform web app URL: https://... /workspace/*`workspace-id`*/job/job-id

Custom tags

In addition to the tags applied automatically by AWS Transform, you can optionally add custom tags to organize, track costs, and manage compliance for your migrated network resources. You can apply custom tags at two levels:

- **Job-level tags:** Apply to all resources created by this job, including all VPCs, subnets, security groups, and route tables.
- **VPC-level tags:** Apply to a specific VPC and automatically cascade to all its associated resources (subnets, security groups, route tables).

> **Note**
>
> Maximum 40 tags per request. Each tag requires a key and value. AWS tagging conventions apply.

AWS Transform applies these tags when it generates the Infrastructure as Code templates.

AWS Migration Acceleration Program

If your migration is part of the **AWS Migration Acceleration Program (MAP 2.0)**, AWS Transform applies a MAP tag to your resources. If you provided your MPE ID earlier in the migration process, the tag is applied automatically. Otherwise, after you finish reviewing the generated VPC configurations, AWS Transform asks whether you have a MAP agreement and prompts you to provide your MPE ID. The MPE ID is a 10-character code using uppercase letters and digits (for example, ABCDE12345). The applied tag uses the format:

- **Key:** `map-migrated` **Value:** `mig`*`MPE_ID`*

Step 8: Deploy your network

After tagging, select your deployment strategy:

- **AWS Transform-managed deployment:** AWS Transform uses CloudFormation templates to deploy your network and runs Reachability Analyzer to check connectivity between subnets across multiple VPCs and within the same VPC.

  > **Note**
  >
  > You must get explicit approval before your network deployment request runs. See Deployment approvals process.

- **Self-deployment:** AWS Transform generates Infrastructure as Code (IaC) templates. CloudFormation templates are generated by default. You can also select additional output formats:
  - AWS CDK generates a TypeScript project for programmatic infrastructure deployment.
  - HashiCorp Terraform generates HashiCorp Configuration Language (HCL) templates for managing network resources.
  - Landing Zone Accelerator (LZA) generates a network-config.yaml file for LZA network configuration.

> **Note**
>
> When you deploy through the Landing Zone Accelerator (LZA) pipeline, your AWS Transform account and LZA installation must be in the same AWS Organization. Deployment fails if the Organization IDs don't match.

For details about the generated artifacts, including the output directory structure for each format, see Understanding the generated network artifacts.

Deployment approvals process

To ensure that network changes comply with your organization's security standards and architectural requirements, all deployment requests go through an approval workflow. You must get explicit approval before your network deployment request runs.

When you submit a deployment request, it automatically routes to authorized approvers through the AWS Transform Approvals tab. Approvers validate both CloudFormation templates and network configurations. Each submission triggers a new review cycle, and deployments proceed only after receiving confirmation. If an approver denies your request, contact them directly to discuss necessary modifications. AWS Transform tracks all approval decisions for audit purposes and maintains deployment history.

Delete deployed network resources

If you need to roll back a deployment, you can delete the network resources that AWS Transform deployed. You can delete resources immediately after deployment completes. If you modify the deployed network resources after deployment, the resources can't be deleted automatically.

- **AWS Transform-managed deployments:** AWS Transform removes all CloudFormation stacks that were created during deployment. This action requires approval through the AWS Transform Approvals tab.
- **Self-deployments:** You must manually delete the deployed resources through the AWS Management Console or AWS CLI.

Configuration file extraction

If your source environment uses Cisco ACI, Palo Alto Networks, or Fortinet FortiGate, you need to extract a configuration file to provide to AWS Transform. You can use these files as standalone

source files to generate network infrastructure and security groups, or as complementary files alongside an RVTools upload to add security group generation. The extraction process is the same in both cases.

To extract configuration files from your firewall and network environments, follow these procedures. Consult vendor documentation for the latest information.

Fortinet FortiGate

- The firmware version must be v7.0 or later.
- You need `super_admin` or `super_admin_readonly` privileges at the global level.
- Steps:
  1. Connect to the firewall by using SSH or built-in CLI client
  2. Run: `show | grep ""` (`| grep ""` disables pagination)
  3. Save all output to a file starting from the `show` command

Palo Alto Networks

- The firmware version must be 10.1 or later.
- You need the superadmin role.
- Connect to the firewall through SSH, run the following commands to disable pagination, set the output format, enter configuration mode, and export the configuration and predefined objects. Save the outputs:

```
set cli pager off
set cli config-output-format set
configure
show              # Save as palo-conf.txt
show predefined   # Save as palo-default.txt
```

Cisco ACI

- The firmware version must be 6.0 or later.
- You need the Admin role with all privileges and a configured Secure Copy Protocol (SCP), SSH File Transfer Protocol (SFTP), or File Transfer Protocol (FTP) destination.
- Steps:

1. Connect to the Application Policy Infrastructure Controller (APIC) through your browser
2. Open the **Admin** menu and choose **Config Rollbacks**
3. In the **Take a snapshot** dialog, select the remote location option and choose **Create a snapshot now**.
4. After receiving "Transfer successful" message, connect to the remote location server and retrieve the latest snapshot file (.gz file)

Apply security posture to existing VPCs

If you have already provisioned your AWS network outside of AWS Transform, you can automatically recreate your security rules on it. AWS Transform generates security groups from your source firewall rules and associates them with your existing VPCs, which saves you time and improves accuracy and consistency compared to recreating the rules manually.

Choose **Apply security posture to existing VPCs** when your target VPCs already exist and you want to apply your source security posture to them.

The AWS Transform agent guides you through the following steps, handling the analysis and generation while you make the decisions:

1. Upload your source network file.
2. Tag your existing VPCs.
3. Match source subnets to existing VPCs.
4. Specify your VPC topology.
5. Review the generated security groups.
6. Configure resource tagging.
7. Deploy the security groups.
8. Remove source rules after cutover.

> **Note**
>
> Because this flow applies security posture to VPCs that already exist, it does not include the network diagram step that is part of mapping a source network to new VPCs.

Step 1: Upload your source network file

Upload a source network file that contains security information. Because this flow extracts and applies your security posture, the file must include firewall rules. AWS Transform supports the following source file types for this flow:

- **Software-defined networks (SDNs):** VMware NSX, Cisco ACI.
- **Firewalls:** Palo Alto Networks, Fortinet FortiGate.
- **Hybrid networks (VMware and non-VMware workloads):** modelizeIT or the AWS Transform discovery tool.
- **Other file types:** If your file type is not one of the supported formats listed above, AWS Transform attempts to convert it automatically.

> **Note**
>
> RVTools is not supported in this flow. RVTools files do not contain firewall rules, which are required for migrating security posture.

You can provide the file in one of two ways:

- **Use file from User Uploads** – Enter the filename, including its extension, from User Uploads.
- **Upload a new file directly in the chat** – Use the file browser or drag and drop to upload a new file.

Step 2: Tag your existing VPCs

Tag your existing VPCs so that AWS Transform can discover them. Tagging a VPC is how you indicate that it should be included in the migration. Only VPCs that you tag are in scope for security group generation. Everything else in your account is left untouched.

AWS Transform provides a tagging page that lists the VPCs in your account with their name, VPC ID, CIDR block, and current tag status. Select the VPCs you want to include, confirm that you have reviewed them, and choose **Tag selected VPCs**. For each VPC you select, AWS Transform applies the following tags:

- **Key:** `CreatedFor` **Value:** `AWSTransform`

- **Key:** `ATWorkspace` **Value:** *`workspace-id`*

Any VPC that AWS Transform created is automatically tagged with **Key:** `CreatedBy` **Value:** `AWSTransform`, and is also in scope for the migration.

You can also apply or remove these tags manually from the Amazon VPC console. After you tag your VPCs, let the agent know, and AWS Transform discovers the tagged VPCs for you.

> **Note**
>
> The tagging page shows only the VPCs in the account you are currently logged in to. If you are not signed in to the AWS Management Console, the tagging page prompts you to sign in first. For multi-account migrations, tag the VPCs in each target account separately by signing in to that account.

Step 3: Match source subnets to existing VPCs

After you confirm your tagged VPCs, AWS Transform runs a coverage analysis that matches the source subnets in your network file to your existing VPCs by CIDR. Where a VPC's CIDR range covers a source subnet, AWS Transform associates the corresponding security groups automatically.

How your source subnets map to your existing VPCs depends on your migration approach:

- **Lift-and-shift (keep existing IP ranges):** To migrate your servers without changing their IP addresses, your existing VPCs must use CIDR ranges that match your on-premises subnets. Matching subnets are covered automatically.
- **Re-IP (change IP ranges):** To assign new IP addresses to your migrated servers, provide a CIDR mapping so AWS Transform can associate the source security posture with the correct target VPCs.

AWS Transform presents a coverage summary showing the total number of subnets and how many are covered or uncovered. For each subnet, the summary lists the on-premises subnet name, on-premises CIDR, matched VPC, matched VPC CIDR, and whether the subnet is covered.

For any uncovered subnet, you have two options:

- **Provide a CIDR mapping** – Through the chat interface, tell AWS Transform which on-premises CIDR maps to which target VPC CIDR (for example, `Map 10.68.101.0/24 to 172.31.101.0/24`). AWS Transform then associates the security groups with the correct VPCs.
- **Leave the subnet unmapped** – Uncovered subnets that you do not map are not migrated.

Each time you provide a mapping, AWS Transform stores it and shows the updated coverage. The coverage summary adds a **Mapped CIDR** column that shows the target CIDR each subnet was mapped to, along with the resolved matched VPC. For any subnets that remain uncovered, you can add more mappings or proceed to the next step. Subnets that are still uncovered when you proceed are not migrated.

When you provide CIDR mappings, keep the following guidance in mind:

- **Prefer wider mappings.** Where possible, supply a single broad mapping (for example, `10.68.0.0/16` to `172.31.0.0/16`) instead of many narrow /24 mappings. A wider mapping covers all subnets within that range at once, so you don't have to map each subnet individually.
- **Mappings also apply to your security group rules.** Only rule CIDRs that fall within a mapped CIDR range are carried over. Rule CIDRs outside your mapped ranges are not migrated, so broader mappings also help ensure your rules are fully covered.
- **Use the same prefix length on both sides.** A mapping must map to an equal-size range, such as /16 to /16 or /24 to /24. Mapping between different prefix lengths (for example, /24 to /20) is not supported.

When you provide a CIDR mapping, AWS Transform duplicates each affected security group rule: one (with the `[ONPREM]` prefix in its description) points to the source on-premises address space, the other to the target AWS address space. Having both rules lets a workload reach its peers whether they have migrated or still run on-premises.

Step 4: Specify your VPC topology

AWS Transform asks whether your existing VPCs are isolated or connected. Your answer determines how AWS Transform generates security group rules:

- **Isolated** – Each VPC operates independently with no cross-VPC routing. AWS Transform generates security group rules using CIDR ranges.

- **Connected** – Your VPCs are connected through Transit Gateways, VPC peering, or similar. AWS Transform generates security group rules using security group referencing, where a rule references another security group instead of a CIDR range.

AWS Transform needs to know your topology before it generates security groups because cross-VPC security group referencing works only between VPCs that have network connectivity. Within-VPC referencing does not require connectivity.

> **Important**
>
> Make sure your topology selection reflects your actual network. If you select **Connected** but your VPCs do not have connectivity, deployment of cross-VPC referencing rules fails because the referenced security groups cannot be resolved across unconnected VPCs.

> **Note**
>
> For VPCs connected through a Transit Gateway, security group referencing also requires that `SecurityGroupReferencingSupport` is set to `enable` on the Transit Gateway and on each Transit Gateway VPC attachment. For more information, see Transit gateway attachments to a VPC in the *AWS Transit Gateway Guide*.

Step 5: Review the generated security groups

AWS Transform runs the network mapping, the same as when you map a source network to new VPCs. When mapping completes, you review the generated security groups for your existing VPCs through a human-in-the-loop (HITL) review.

The review shows your existing VPCs, discovered by their tags, and only the security groups that AWS Transform generated and associated for them. For example, the review might show `Application VPC (172.30.0.0/16) — 3 security groups` and `Database VPC (172.31.0.0/16) — 1 security group`. Other details, such as subnets and route tables, are not shown, because this flow applies security posture only.

When you open a VPC, AWS Transform shows its IPv4 CIDR, account, source (listed as *Existing*), and description, along with a **Security groups** tab. Each security group lists its name, ID, and rule

count, and indicates the source rule it was generated from (for example, *Generated using: Web-Tier-HTTPS*), so you can trace each generated group back to your source configuration. You can search the security groups by name, ID, or description.

> **Note**
>
> This review is read-only. Your existing VPCs are managed outside AWS Transform and are not modified by this migration. AWS Transform adds only the generated security groups.

After you review the security groups, let the agent know to proceed.

Step 6: Configure resource tagging

You can optionally add custom job-level tags to the security groups that AWS Transform generates. Job-level tags apply to every security group created by this job, which helps you organize resources, track costs, and manage compliance.

> **Note**
>
> This flow supports job-level tags only. It creates only security groups, so VPC-level tagging doesn't apply.

AWS Transform applies these tags when it generates the security groups.

Step 7: Deploy the security groups

After you review the generated security groups, choose how to deploy them. As with mapping a source network to new VPCs, you can choose AWS Transform-managed deployment or self-deployment:

- **AWS Transform-managed deployment:** After you approve, AWS Transform deploys the security groups to your existing VPCs on your behalf.
- **Self-deployment:** AWS Transform generates the security groups as Infrastructure as Code (IaC) that you deploy yourself. The same output formats are available: CloudFormation, AWS CDK, HashiCorp Terraform, and Landing Zone Accelerator (LZA).

This flow deploys security groups only. Unlike mapping a source network to new VPCs, AWS Transform does not run Reachability Analyzer, because no new network infrastructure is created, and there is no automatic rollback of deployed resources.

For details about the generated artifacts, including the output directory structure for each format, see Understanding the generated network artifacts.

Step 8: Remove source rules after cutover

Once a workload has fully moved to AWS and no longer needs connectivity to its source network, remove the security group rules with the `[ONPREM]` prefix in their description. Keep the rules for any workloads that still run in the source environment.

Understanding the generated network artifacts

AWS Transform generates your target network as Infrastructure as Code (IaC) so you can review and deploy it using your own tools and processes. It produces the network in a fraction of the time, and you decide how and when to deploy it: check it against your own standards, adjust it to fit how your team works, deploy it the way you already deploy infrastructure, and keep it in version control so you can track changes and deploy again later.

You can find these artifacts in an Amazon S3 bucket in your target AWS account. AWS Transform writes AWS CDK and CloudFormation for every migration, plus any other formats you selected, to the same location. The bucket is in the AWS account and AWS Region that you specified for the connector and starts with `transform-vmware-target-`. For more information about the connector, see Step 3: Connector configuration.

Use the link provided in AWS Transform to open the generated artifacts in that bucket. The generated artifacts include a `README.md` file that explains how to use the generated templates.

> **Important**
>
> To verify the downloaded file hasn't been corrupted or tampered with, generate and download a checksum, then compare it to a locally generated hash using `openssl dgst -sha256 -binary <file.zip> | base64` command.

Output directory structure

Each IaC format has its own output directory (`cdk/` and `cfn/`, `cdk_l2/`, `terraform/`, or `lza/`). Each contains a `network_segments/` directory for network definitions and a `security/` directory for security definitions. The following trees show the structure of each format, where `<segment>` is a placeholder for each segment:

CloudFormation (`cfn/`) contains CloudFormation templates.

```
cfn/
### network_segments/
#   ### <segment>/                          # CloudFormation template JSON
### security/
    ### <segment>_security/
    ### <segment>_security_dependencies/   # only when the segment has cross-VPC rules
```

CDK L1 (`cdk/`) contains a AWS CDK project. It includes a `README.md` file.

```
cdk/
### README.md
### network_segments/
#   ### <segment>/
### security/
    ### security-interface.json
    ### validate-references.py
    ### <segment>_security/
    ### <segment>_security_dependencies/   # only when the segment has cross-VPC rules
```

CDK L2 (`cdk_l2/`) contains AWS CDK projects that use higher-level (L2) constructs from the AWS CDK Construct Library. L2 constructs provide sensible defaults and less boilerplate than the L1 constructs that CDK L1 uses. For more information, see Constructs in the *AWS Cloud Development Kit (AWS CDK) Developer Guide*. Workload segments are delivered as complete project zip files, and the deployment `README.md` is inside each zip.

```
cdk_l2/
### network_segments/
#   ### <segment>/                          # hub and spoke CDK app (.ts source)
#   ### <segment>/                          # workload segment (.zip)
### security/
    ### security-interface.json
```

```
    ### validate-references.py
    ### <segment>_security/                # .zip
    ### <segment>_security_dependencies/   # .ts source, only when the segment has
 cross-VPC rules
```

Terraform (`terraform/`) is delivered as a single zip file.

```
terraform/
### README.md, LICENSE.txt, main.tf, outputs.tf
### modules/                                # networking, vpc, workload-vpc
### network_segments/
#   ### <segment>/                          # main.tf, outputs.tf, variables.tf,
 versions.tf, terraform.tfvars
### security/                               # standalone Terraform root
    ### main.tf, locals.tf, variables.tf, terraform.tfvars
    ### security-interface.json
    ### validate-references.py
    ### modules/security/
    ### security_segments/
        ### <segment>_security/            # main.tf, variables.tf, terraform.tfvars
 (plus versions.tf for cross-account segments)
        ### <segment>_security_dependencies/   # only when the segment has cross-VPC
 rules
```

LZA (`lza/`) contains Landing Zone Accelerator configuration files and per-VPC AWS CDK applications.

```
lza/
### README.md
### network_segments/
#   ### network-config.yaml
#   ### replacements-config.yaml
### security/
    ### <segment>/                          # one CDK app per VPC segment
    #   ### <stack_name>.ts                 #   security group stack
    #   ### app.ts
    #   ### cdk.json
    #   ### tsconfig.json
    #   ### package.json
    ### <segment>_dependencies/             # same file set; only when the segment has
 cross-VPC rules
```

Security definitions

Within `security/`, security groups are split per segment. Each segment's own security groups are in its `<segment>_security/` directory. If a segment has rules that reference another segment, its cross-VPC rules are separated into a `<segment>_security_dependencies/` directory that deploys last, after the security groups it references exist.

A segment whose VPC already exists in the account keeps its original name under `security/<segment>/` with no suffix.

The `cdk/`, `cdk_l2/`, and `terraform/` outputs include two additional files in `security/`:

- `security-interface.json` lists the network identifiers each segment consumes, with one entry per segment. The identifier type depends on the format: CloudFormation export names for CloudFormation, CDK L1, and CDK L2, or SSM parameter names for Terraform. A segment that imports nothing has no entry.
- `validate-references.py` is a script you run before you deploy a segment. It confirms that the network identifiers in `security-interface.json` already exist in your AWS account. Running it requires the AWS CLI, with credentials configured for the target account and Region.

> **Note**
>
> Deploy the resources in `network_segments/` before those in `security/`. The security definitions reference the network resources' exports and parameters, which must already exist.

Multi-account output

A multi-account deployment adds one per-format manifest file at the root of the output directory. A single-account deployment has no manifest files. Each format has its own manifest:

- `manifest-cdk_l1.json` for the AWS CDK L1 format.
- `manifest-cdk_l2.json` for the AWS CDK L2 format.
- `manifest-terraform.json` for the Terraform format, at the `terraform/` root.
- `manifest-lza.json` for the LZA format.

Account assignment is not shown by the folder structure. It is carried in data fields: each manifest lists its segments with a target account (the manifest's `segments[].targetAccount` field), and each entry in `security-interface.json` has a `targetAccount` field.

Migrate servers

AWS Transform automates the rehosting of your servers to Amazon EC2 at scale. The AI-powered agent guides you through each migration wave, from inventory validation and replication agent deployment through testing and final cutover. AWS Transform handles the orchestration across hundreds of servers while you maintain control over configuration and approval decisions. Internally, AWS Transform uses AWS Transform MGN (MGN) for data replication. For more information about MGN, see What is AWS Transform MGN? in the *MGN User Guide*.

You can migrate servers from virtually any source environment, including on-premises data centers, other cloud providers, or other AWS Regions. AWS Transform supports both physical servers and virtual servers running on VMware, Hyper-V, KVM, or other virtualization platforms. The source infrastructure and hypervisor are not important as long as the source operating system is supported.

Server migration is organized by waves. Each wave represents a group of servers that are migrated together. For each wave, the agent walks you through the following phases:

1. **Prerequisites and Configure Migration Defaults**. Set up your target accounts and configure how instances are launched. AWS Transform provides intelligent defaults so you can get started quickly.
2. **Step 1: Set up migration wave**. The agent configures your target account, validates permissions, and sets up resource tagging automatically.
3. **Step 2: Validate and confirm inventory**. Review your server configurations, instance type recommendations, and network assignments before migration begins.
4. **Step 3: Deploy replication agents**. The agent can deploy agents across all servers in a wave without requiring manual access to each server individually.
5. **Step 4: Data replication**. Continuous block-level replication keeps your target environment synchronized with source servers until you are ready for cutover.
6. **Step 5: Testing**. Launch test instances to validate your migrated servers before committing to the final cutover.
7. **Step 6: Cutover**. Complete the migration with minimal downtime. The agent coordinates the final switchover and verifies success.

For waves with a *containerize* migration strategy, AWS Transform runs the source code containerization workflow instead of the rehost steps. The containerization workflow guides you through cloning source code, generating Docker artifacts, publishing container images, and deploying to Amazon Elastic Container Service or Amazon Elastic Kubernetes Service. For the full containerization workflow, see *Source code containerization*.

Prerequisites and Configure Migration Defaults

Prerequisites

If you completed all the steps of an end-to-end migration job in AWS Transform, your target accounts, inventory file, and network infrastructure are already prepared. You can proceed to Configure Migration Defaults.

If you are starting server migration independently, ensure you have the following in place:

- **Supported operating systems** – Source servers must run a supported operating system. For the full list, see Supported operating systems in the *MGN User Guide*.
- **Target accounts for migration** – The AWS account IDs where you need your servers to be migrated. You can use AWS Transform landing zone or any other tools to set up your infrastructure.
- **Network infrastructure in place** – VPCs, subnets, and security groups deployed and configured. You can use AWS Transform network migration or any other tools to set up your network infrastructure.
- **Inventory file** – Prepared with server details, wave assignments, target account information, and Amazon EC2 instance type preferences. You can use AWS Transform migration planning to generate this file.

Configure Migration Defaults

AWS Transform provides intelligent defaults for your migration configuration, including how Amazon EC2 instances are launched and how replication is set up. You can accept these defaults and start migrating immediately, or customize them through the chat interface or a visual review. Defaults apply across all your target accounts and can be overridden at the wave level during wave setup.

Amazon EC2 recommendation preferences

AWS Transform analyzes your source server utilization and recommends optimally sized Amazon EC2 instances, which helps you avoid overprovisioning from the start. You can configure your Amazon EC2 recommendation preferences to control how instance types are selected for your migrated servers.

For more information about generating Amazon EC2 recommendations, see Generating Amazon EC2 recommendations in AWS Migration Hub.

> **Note**
>
> You can modify the suggested Amazon EC2 instance types to include recommendations from the Migration Evaluator, AWS Optimization and Licensing Assessment (OLA), or an AWS Transform assessment job.

Migration initialization

AWS Transform automatically sets up the required migration infrastructure in your target accounts before migration begins. This includes initializing MGN for every AWS Region in which you plan to migrate, as well as all target accounts. During the initialization process:

- The required IAM roles and policies are created.
- The required default templates are configured.

For information about the initialization process, see Initializing AWS Transform MGN with the console in the *MGN User Guide*.

Amazon EC2 launch template

The launch settings include two parts: the general launch settings, and the Amazon EC2 launch template, which determines how a test or cutover instance is launched for each source server in AWS.

Launch settings, including the Amazon EC2 launch template, can be defined at the account level and are then applied to each source server automatically each time you add a source server to AWS Transform MGN. The launch settings defaults defined in this section can be applied to all your target accounts automatically.

AWS Transform presents the list of available launch template settings. You can choose to continue with the defaults or configure your launch template. If you choose to configure, AWS Transform provides a link to a human-in-the-loop (HITL) review that contains all the parameters of the launch template settings. You can also make modifications directly through the chat interface for any parameters you wish.

Source servers are created with the account launch template settings. After source servers are created with these default settings, you can change them at the source server launch settings level. You can change source server settings on any parameter using the chat interface, or for bulk operations using the inventory Excel file during Step 2: Validate and confirm inventory.

To review the full list of launch template settings and details, see Launch general settings in the *MGN User Guide*.

Additional Amazon EC2 launch template changes

Make additional Amazon EC2 launch template changes on the template ID for each target account. This option is available inside the wave setup. AWS Transform guides you through it and provides the appropriate link.

Post-launch actions

Post-launch actions automate modernization and validation tasks that run on each source server immediately after it launches as a test or cutover instance in AWS. AWS Transform Rehost takes the post-launch actions available in MGN and provides these capabilities so that the agent can recommend, configure, and run them on your behalf as part of the migration workflow. Actions are run through AWS Systems Manager (Systems Manager), and can be either one of the predefined actions available in MGN or a custom action created from an existing Systems Manager document.

Post-launch actions can be defined at the account level and are then applied to each source server automatically each time you add a source server to MGN. The post-launch action defaults defined in this section can be applied to all your target accounts automatically.

AWS Transform first presents the list of available post-launch actions across your target accounts. It then offers to define a new post-launch action and apply it across your target account migration. AWS Transform also provides AI-powered recommendations based on the operating system and best practices for your inventory. You can choose to continue with the defaults or configure your own actions.

You can also choose a post-launch action that is already defined in MGN. Prompt the agent to show the list of predefined post-launch actions available. For more information about this list, see Predefined post-launch actions in the *MGN User Guide*.

Creating a new post-launch action

If you choose to create a new post-launch action, the agent prompts you to provide the Systems Manager document name or Systems Manager ARN to build the post-launch action with. The Systems Manager document should be created in advance through the AWS Systems Manager console. For more information about creating a Systems Manager document, see Creating Systems Manager documents in the *AWS Systems Manager User Guide*.

The agent then generates a human-in-the-loop interface where you provide the required fields:

- **Post-launch action name** – The agent provides a default name, which you can change.
- **Post-launch action order** – The default order value is 1001, unless the account already has post-launch actions defined, in which case the new action is placed last in the run order. You can change the order. For more information about post-launch action order, see Post-launch settings in the *MGN User Guide*.
- **Required Systems Manager parameter values** – Provide the values for any parameters required by the Systems Manager document.

You can also make modifications directly through the chat interface for any parameters you wish.

Source servers are created with the account post-launch action settings. After source servers are created with these default settings, you can change them at the source server level. You can change source server settings on any action using the chat interface, or for bulk operations using the inventory Excel file during Step 2: Validate and confirm inventory.

Post-launch actions in the inventory file

The Rehost agent extends the inventory file with a post-launch action template defined for each source server. This allows customers to update or add post-launch actions per source server during the rehost to MGN import process. To delete specific post-launch actions from a source server, use the `active` column to indicate `FALSE`. Post-launch actions in the inventory file use the following naming convention:

`mgn:launch:post-actions:<ACTION_NAME>:<FIELD_NAME>`

A global toggle controls whether post-launch actions run:

`mgn:launch:post-actions:enabled` (TRUE/FALSE)

Fields per action

- `ssmDocumentName` (String, required) – The Systems Manager document to execute.
- `order` (Integer, required) – Execution order; must be between 1000 and 10000. Actions execute in ascending order — lower values run first.
- `active` (TRUE/FALSE, optional) – Whether the action is active.
- `mustSucceedForCutover` (TRUE/FALSE, optional) – Whether the action must succeed before cutover.
- `timeoutSeconds` (Integer, optional) – Timeout in seconds.
- `description` (String, optional) – Human-readable description.
- `parameters` (JSON, optional) – Systems Manager document parameters.

Parameters JSON structure

The `parameters` field is provided as a JSON structure:

```
{
  "parameters": {
    "Operation": [
      {"value": "Scan", "type": "String"}
    ]
  },
  "externalParameters": {
    "InstanceId": "ec2.InstanceId"
  }
}
```

- `parameters` – Maps document parameter names to a list of value references (each with a `value` and an optional `type`, which defaults to `String`).
- `externalParameters` – Maps document parameter names to dynamic path strings (no Systems Manager parameter is created).

Step 1: Set up migration wave

In this phase, AWS Transform prepares the migration wave by configuring the target account, verifying service permissions, setting up resource tags, adding networking data to your inventory, and configuring replication and launch settings.

Migration mode and account configuration

AWS Transform supports two migration modes:

- **Single-account migration** – All servers in the wave migrate to the same target account configured in your connector.
- **Multi-account migration** – Servers migrate to different target accounts specified in your inventory file. For multi-account migrations, your inventory file must include an `mgn:account-id` column with the target account ID for each server.

AWS Transform confirms the target account configuration and verifies that MGN is initialized in each target account. If MGN is not yet initialized, AWS Transform provides instructions to complete the initialization. During initialization, MGN creates the following IAM service roles for replication and launch operations:

- `AWSApplicationMigrationReplicationServerRole`
- `AWSApplicationMigrationConversionServerRole`
- `AWSApplicationMigrationMGHRole`
- `AWSApplicationMigrationLaunchInstanceWithDrsRole`
- `AWSApplicationMigrationLaunchInstanceWithSsmRole`
- `AWSApplicationMigrationAgentRole`

To learn more about these roles, see Initializing MGN with the console or Initializing MGN with the API in the *MGN User Guide*.

For multi-account migrations, AWS Transform also creates the following role during the initialization step: `AWSTransformRehostSharingRole_<management-or-delegated-admin-account-id>`. This role is deployed across all migration target accounts.

Resource tagging verification

After service permissions are confirmed, AWS Transform verifies that all required resources are properly tagged for the migration to be operated by the agent successfully. If any resources are missing required tags, AWS Transform provides a link to the tagging page where you can apply the missing tags before continuing. The following tags are required:

- Existing source servers must have tags `CreatedBy: AWSTransform` and `ATWorkspace: <workspace_id>`. If you have already started replication on source servers and created them in the AWS Transform MGN service, you need to tag these servers so that AWS Transform can correlate them with the source servers discovered from your on-premises environment, and avoid unnecessary creation of duplicate source servers. AWS Transform automatically correlates between them using the user-provided ID, FQDN, or hostname keys.
- Network resources must be properly tagged for both replication (staging area) and launch instances. AWS Transform displays the full list of network resources in your target account, with an indication of whether each resource is already tagged or not. You can review the list and select any untagged resources you want to add. For each resource you select, AWS Transform applies the relevant tag:
  - `CreatedBy: AWSTransform` or `CreatedFor: AWSTransform`, depending on the resource type.
  - `ATWorkspace: <workspace_id>` is applied to all selected resources.

  VPCs and subnets created by the AWS Transform network migration agent are automatically tagged.
- In addition to VPCs and subnets, AWS Transform also displays all existing Elastic Network Interfaces (ENIs) found in your target account. If you want AWS Transform to use them as part of your instance launch, they must be tagged with `CreatedFor: AWSTransform` and `ATWorkspace: <workspace_id>`. For more information on how to attach or add ENIs to the Amazon EC2 launch template, see Detailed considerations in the *MGN User Guide*.

Add networking data to inventory

AWS Transform adds networking information from your network migration to the inventory file. This step maps your servers to the appropriate target subnets and security groups based on the network configuration generated during the migrate network phase.

Replication and launch settings

Replication settings configuration

Replication settings determine how data is replicated from your source servers to AWS. Configure the replication settings in the replication template before adding source servers to AWS Transform MGN. AWS Transform shows you all the replication settings parameters — you can configure them through a dedicated HITL or through the chat interface.

For more details about the replication settings parameters, see Replication settings template in the *MGN User Guide*.

Launch template settings

The launch template allows you to control the way AWS Transform MGN launches instances in AWS. The default configuration defined in the template is automatically applied to every newly added server. You can configure launch template settings through a dedicated HITL or through the chat interface.

For more details about the launch template settings parameters, see Launch template in the *MGN User Guide*.

AWS Transform also provides a link to the Amazon EC2 launch template ID associated with the launch template, allowing you to change additional Amazon EC2 launch template attributes. To edit the Amazon EC2 launch template, follow the instructions in Launch template in the *MGN User Guide*.

IP assignment strategy

You choose how IP addresses are assigned to your migrated servers:

- **Static IP** – The source server's IP address is maintained. If CIDR transformation is required, AWS Transform automatically converts the IP address to match the new CIDR.
- **Dynamic IP (DHCP)** – Each server is assigned a new IP address from the subnet's IP pool.

> **Note**
>
> If you selected the MAP security groups mapping strategy during network migration, only static IP assignment is available. For more details, see Security groups mapping.

Step 2: Validate and confirm inventory

Before loading your server data into MGN, AWS Transform prepares the migration inventory file for your review. You must confirm the migration inventory file data and content as this sets the migration configuration. You can review the data in the file and update it using either of these methods:

- **Edit the migration inventory file directly** – Download the file in CSV or XLSX format and modify it locally in your preferred spreadsheet application. You can adjust any of the fields in the file, except for the read-only ones.
- **Use the chat interface** – Make targeted changes for specific columns on individual source servers without downloading the entire file.

The migration inventory file includes details such as server names, operating systems, Amazon EC2 instance type recommendations, target subnets, security groups, IP assignments, and licensing options. Required fields include:

- **Server information** – Server name, VMID, and source specifications.
- **Wave assignment** – Migration wave grouping.
- **Application grouping** – Logical application associations.
- **Target configuration** – Target account, Region, and Amazon EC2 instance type. You can control the operating system licensing options (BYOL or License Included) and tenancy by specifying the configuration in the inventory file columns `mgn:launch:placement:operating-system-licensing` and `mgn:launch:placement:tenancy`.
- **Network configuration** – Target subnet and security groups.

For more information about the fields, see Import parameters in the *MGN User Guide*.

After you review the migration inventory file, you can either accept it as shown or upload a modified version. AWS Transform then loads the data into MGN, which creates source server records for each server in the wave.

> **Note**
>
> Do not remove columns or change column headers in the inventory file. AWS Transform requires the original file structure to process the data correctly.

> **Note**
>
> AWS Transform allows one import to a given target AWS account and target AWS Region at a time. If you work on more than one wave simultaneously, or if there is more than one migration job running with the same target account, you must wait for an import to finish before you can perform another import in a different wave or job.

Step 3: Deploy replication agents

To begin replicating data from your source servers to AWS, you install the AWS Replication Agent on each source server. AWS Transform offers three installation methods:

- **Organization tools** – Use your organization's existing deployment tools (such as SCCM, Ansible, or Chef) to install agents across your servers. AWS Transform provides the installation commands with additional parameters for silent installation, including `--no-prompt`, `--aws-access-key-id`, `--aws-secret-access-key`, and `--aws-session-token`.
- **MGN connector** – Use an MGN connector to automate agent installation. The connector connects to source machines over SSH (Linux) or WinRM (Windows) and installs the replication agent automatically. After it is configured, a connector can be reused across multiple waves and different target AWS accounts. For more information about the MGN connector, see Set up the MGN Connector in the *MGN User Guide*.

  > **Note**
  >
  > Before using the MGN connector with AWS Transform, you must tag the connector's managed instance in AWS Systems Manager Fleet Manager with the following tags:
  >
  > - Key: `CreatedFor` Value: `AWSTransform`
  > - Key: `ATWorkspace` Value: *`workspace-id`*
  >
  > To tag the managed instance, open the AWS Systems Manager console, navigate to **Fleet Manager** under **Node Tools**, choose the managed instance of your MGN connector, and apply the tags above. Find your workspace ID in the AWS Transform web app URL: `https://.../workspace/`*`workspace-id`*`/job/`*`job-id`*.

- **Manual installation** – Install the agent directly on each source server. This method requires direct access to each server but gives you full control over the installation process.

AWS Transform MGN connector setup

The AWS Transform MGN connector automates the deployment of replication agents across your source servers, eliminating the need to log into each server individually. The connector is a lightweight client deployed on a dedicated Linux machine in your on-premises environment. It connects to source servers over SSH (Linux) or WinRM (Windows) to install and configure replication agents, eliminating the need to manually coordinate across multiple AWS services.

How the connector works

The connector operates through the following components:

- **Connector client** – Deployed on a dedicated Linux machine in your environment.
- **SSM Agent** – Installed on the same machine to enable secure communication with AWS.
- **SSM Hybrid Activation** – Links the connector machine to AWS Systems Manager for secure command execution.
- **Credentials management** – Retrieves source server credentials from AWS Secrets Manager.

When you deploy agents, AWS Transform sends an SSM document to the connector machine. The connector then does the following:

1. Retrieves source server credentials from AWS Secrets Manager.
2. Establishes a connection to each source server.
3. Validates that the source server meets prerequisites.
4. Installs and configures the replication agent.
5. Verifies successful installation.

Connector machine requirements

Requirement	Details
Operating system	Supported Linux operating system. For the full list, see MGN connector prerequisites in the *MGN User Guide*.
Network access	Must reach all source servers (Linux over SSH, Windows over WinRM)

Requirement	Details
Internet connectivity	Outbound HTTPS (443) to AWS endpoints (Systems Manager, Secrets Manager, MGN)
Disk space	Minimum 200 MB free
Permissions	Root or sudo access

> **Note**
>
> The connector must be installed on a Linux machine, but it can deploy agents to both Linux and Windows source servers.

Setup process

AWS Transform guides you through the following steps to set up the connector:

Step 1: Connector configuration

Provide a name for your connector, or use the auto-generated default name. The connector can be installed on the management account or on a delegated administrator account in MGN. For multi-account migrations, the connector can deploy agents to servers across member accounts.

Step 2: AWS resource setup

AWS Transform opens a setup page that runs in your browser using your AWS credentials. You must be logged in to the AWS Management Console with either your management account or your delegated administrator account. This must be the same account your AWS Transform target connector is connected to.

The setup page automatically creates the following resources:

- **IAM roles** (created idempotently — skipped if they already exist):
  - `AWSApplicationMigrationConnectorManagementRole` – Used during agent installation to access credentials.
  - `AWSApplicationMigrationConnectorSharingRole_<ACCOUNT-ID>` – Contains permissions for agent installation.

- **SSM Hybrid Activation** – 30-day expiration period. Links the connector machine to AWS Systems Manager and generates secure activation credentials.

Alternatively, you can download a CloudFormation template from the setup page to deploy the IAM roles yourself.

The setup page generates a one-line installation command with all necessary credentials and configuration.

> **Important**
>
> Keep the setup page open until installation is complete. Closing it will require restarting the process. All credentials exist only in your browser and are not stored by AWS Transform.

Step 3: Connector installation

Install the connector on a Linux machine in your environment:

1. Copy the installation link from the setup page.
2. SSH into your chosen Linux machine.
3. Paste and execute the installation command.
4. Wait for installation to complete (typically 2–3 minutes).

Step 4: Attach source servers

After installation, AWS Transform identifies all source servers that belong to the current wave and automatically attaches them to the MGN connector.

Step 5: Configure credentials

Provide AWS Secrets Manager ARNs for your source server credentials. AWS Transform offers three credential configuration options:

- **Single secret for Linux servers** – One shared secret containing SSH keys or username/password for all Linux source servers.
- **Single secret for Windows servers** – One shared secret containing username and password for all Windows source servers.

- **Multiple per-server secrets** – Different secrets per server or group of servers. Use this when servers have different credentials. AWS Transform generates a CSV file pre-populated with your server list. You fill in the `secret_arn` column for each server and upload the completed file.

> **Note**
>
> You can combine the Linux and Windows single-secret options if you have both server types with one shared secret each. The per-server secrets option is mutually exclusive with the single-secret options.

Credential secret format. For more information, see MGN connector credentials in the *MGN User Guide*:

```
{
  "WinConnectionProtocol": "HTTPS",
  "WinUserName": "windows_username",
  "WinPassword": "windows_password",
  "LinuxUserName": "linux_username",
  "LinuxPrivateKey": "linux_private_key",
  "LinuxHostKeyValidation": false
}
```

Agent deployment

After credentials are configured and verified, AWS Transform deploys replication agents to your source servers. You can deploy to all servers in the current wave or select specific servers.

The deployment process for each server:

1. AWS Transform sends deployment commands to the connector by using SSM.
2. The connector retrieves credentials from AWS Secrets Manager.
3. The connector connects to the source server using the configured credentials.
4. The connector validates that the source server meets all prerequisites required to run the replication agent.
5. The connector installs and configures the replication agent.
6. The connector verifies successful installation and connectivity.

You can monitor deployment progress in real time with per-server status tracking, including the current installation step, elapsed time, and estimated time remaining. If any servers fail, AWS Transform displays the failure reason and offers retry options per server. Successfully deployed servers can proceed independently while failed servers are retried.

Connector reuse and lifecycle

When deploying agents for subsequent waves, you can reuse an existing connector or create a new one. AWS Transform lists all connectors configured in your account, showing the connector name, status (Active or Expired), attached server count, and Hybrid Activation expiry date.

- **Active connector** – The Hybrid Activation is still valid. AWS Transform verifies IAM roles for the new wave and proceeds to credential configuration. No new Hybrid Activation is needed.
- **Expired connector** – The SSM Hybrid Activation has expired. Expired activations cannot be renewed. You must select a different connector or create a new one.

SSM Hybrid Activations expire after 30 days. The activation is required only for installing the connector on the Linux machine. Once the connector is installed, you can continue to use it to install replication agents on source servers even after the activation expires. If you need to install the connector on a new machine after the activation has expired, you need to create a new connector through the setup process.

Manual agent installation

For manual installation, you first generate AWS credentials (temporary or permanent) and then install the agent on each source server.

Credential options:

- **Temporary credentials (recommended)** – Create an IAM role with the `AWSApplicationMigrationAgentInstallationPolicy` managed policy, then use `aws sts assume-role` to generate temporary credentials. For more information, see Agent installation permissions in the *MGN User Guide*.
- **Permanent credentials** – Create an IAM user with the `AWSApplicationMigrationAgentInstallationPolicy` managed policy and generate an access key.

Installation steps:

For Linux servers, download and run the installer:

```
wget -O ./aws-replication-installer-init \
  https://aws-application-migration-service-region.s3.region.amazonaws.com/latest/
linux/aws-replication-installer-init
sudo chmod +x aws-replication-installer-init
sudo ./aws-replication-installer-init --region region --user-provided-id server-
identifier
```

For Windows servers, download and run the appropriate installer using PowerShell as Administrator:

```
Invoke-WebRequest -Uri "https://aws-application-migration-
service-region.s3.region.amazonaws.com/latest/windows/
AwsReplicationWindowsInstaller.exe" `
  -OutFile "C:\AwsReplicationWindowsInstaller.exe"
C:\AwsReplicationWindowsInstaller.exe --region region --user-provided-id server-
identifier
```

Important

The `--user-provided-id` parameter is required. Replace *`server-identifier`* with the exact value from the `mgn:server:user-provided-id` column in your inventory file. This identifier links the physical server to its MGN source server record.

For more information about agent installation, see Linux agent and Windows agent in the *MGN User Guide*.

After installation, AWS Transform verifies that all agents are successfully connected by checking that servers show a replication state of `INITIATING` or `INITIAL_SYNC`.

Note

AWS Transform does not support MGN agentless replication. For information about agentless replication, see Agentless replication overview in the *MGN User Guide*.

> **Note**
>
> You must install the replication agent on all servers in a wave. Disconnect and archive servers on which you don't install the replication agent. You can use the `disconnect-from-service` command to disconnect servers, and the `mark-as-archived` command to archive disconnected servers. The archiving command only works for source servers whose lifecycle state is `DISCONNECTED`.

For quotas related to replication, see MGN service quota limits in the *MGN User Guide*.

Step 4: Data replication

After the replication agents are installed, data replication begins automatically. AWS Transform uses continuous block-level replication to synchronize data from source servers to AWS.

The replication process consists of two phases:

- **Initial sync** – A complete copy of the source server data to AWS. Data is stored as Amazon Elastic Block Store (Amazon EBS) snapshots or on Amazon FSx for NetApp ONTAP (FSx for ONTAP) volumes in the target account, depending on your configured target storage type. For more information, see Target storage type in the *MGN User Guide*. Duration depends on data volume and network bandwidth.
- **Continuous replication** – Ongoing synchronization of changed blocks with minimal impact on source server performance. Maintains an up-to-date copy in AWS.

Replication servers are temporary Amazon EC2 instances deployed in the staging area subnet. They receive replicated data from source servers and are automatically managed by MGN. For more information, see Replication server settings in the *MGN User Guide*.

AWS Transform monitors the replication progress and provides status updates, including replication status, replication lag (the time difference between source and replicated data), and bandwidth usage.

During replication, each server progresses through the following states:

- **Not ready** – The server is undergoing the initial sync process and is not yet ready for testing.
- **Ready for testing** – The server has been successfully added and data replication has started. Test or cutover instances can now be launched.

After all servers in the wave have progressed beyond the NOT_READY state, the data replication phase is complete and you can proceed to testing.

You can control replication for individual servers or the entire wave at any time:

- **Pause replication** – Temporarily pause replication for specific servers or the entire wave.
- **Resume replication** – Resume previously paused replication.
- **Stop replication** – Permanently stop replication. Stopped replication can be restarted, but it begins from the initial sync.

Step 5: Testing

After data replication is complete, you can launch test instances to validate your migrated servers before performing the final cutover. For more information, see Launch test instances in the *MGN User Guide*. AWS Transform supports two testing options:

- **Full wave testing** – Launch test instances for all servers in the wave.
- **Selective testing** – Launch test instances for specific servers that you select by providing their user-provided IDs from the inventory file.

AWS Transform launches Amazon EC2 instances from the replicated data and provides the instance IDs so you can connect to and validate the test instances. After testing, you can:

- Proceed to cutover if testing is successful.
- Launch new test instances to retest.
- Terminate test instances and address any issues before retesting.

Step 5b: Mark applications as ready for cutover

After testing is complete and you are satisfied with the results, mark your applications as ready for cutover. AWS Transform reviews the replication status of each application and resolves any replication alerts before allowing you to proceed. Only applications with a clean replication status can be marked for cutover.

Step 6: Cutover

Cutover is the final migration step where your production workloads are moved to AWS. For more information, see Launch cutover instances in the *MGN User Guide*. Similar to testing, AWS Transform supports full wave cutover or selective cutover for specific servers.

During cutover, AWS Transform launches Amazon EC2 instances from the latest replicated data and provides the instance IDs for each server. After verifying the cutover instances, you finalize the cutover, which stops the ongoing source machine replication.

The cutover process includes the following steps:

1. **Launch cutover instances** – AWS Transform launches Amazon EC2 instances for the selected servers. You can choose full wave cutover or selective cutover.
2. **Verify cutover instances** – Connect to the launched instances and verify they are functioning correctly.
3. **Finalize cutover** – Confirm the cutover to stop source machine replication. You can finalize all servers in the wave or select specific servers. Finalization stops replication agents from sending data, removes replication agents from source servers, and locks the server lifecycle state. This action cannot be easily undone. For more information, see Finalize cutover in the *MGN User Guide*.
4. **Archive source servers (optional)** – After finalization, you can mark source servers as archived to free up source server quota in your account.

> **Important**
>
> Finalizing cutover stops the ongoing source machine replication. Make sure you have verified your cutover instances before finalizing.

> **Note**
>
> Downtime occurs between source shutdown and cutover instance availability. Plan your cutover window accordingly.

Server lifecycle states

During migration, each server progresses through the following lifecycle states. For more information, see Source server lifecycle in the *MGN User Guide*.

- **Not ready** – The server is undergoing the initial sync process and is not yet ready for testing.
- **Ready for testing** – Data replication has started and test or cutover instances can be launched.
- **Test in progress** – A test instance is currently being launched.
- **Ready for cutover** – The server has been tested and is ready for cutover.
- **Cutover in progress** – A cutover instance is currently being launched.
- **Cutover complete** – The server has been cutover. All data has been migrated to the AWS cutover instance.
- **Disconnected** – The server has been disconnected from MGN.

You can ask AWS Transform about the status of your servers at any time during the migration. AWS Transform provides an interactive wave status table that displays all relevant server information including migration lifecycle, replication status, and recommended next steps. You can also ask in natural language, for example:

- What is the status of my servers?
- What's the status of my wave?
- What's the status of the step that I'm currently in?

During wave migration, you can ask AWS Transform to update or change the status of individual servers. For example, if 9 out of 10 servers in your wave passed the test phase but one failed, you can allow AWS Transform to continue moving the 9 servers into the next phase while re-running the test on the failed server.

Deployment approvals

AWS Transform includes built-in approval workflows to ensure production changes go through your organization's review process. When an operation requires approval, AWS Transform routes the request to authorized approvers through the Approvals tab. Only users with the Admin role in AWS Transform can approve deployment requests. Deployments proceed only after receiving confirmation.

Run migrations with the AWS Transform MCP server

The AWS Transform Model Context Protocol (MCP) server lets you plan and run migrations from any MCP-compatible client, such as Kiro, Claude Code, Cursor, or Cline, without leaving your development environment. MCP is a standardized interface that gives the AI assistant in your client real-time, contextual access to your AWS Transform workspaces and jobs. With it, you can create workspaces, launch migration jobs, respond to human-in-the-loop tasks, and download artifacts through natural language.

The MCP server exposes the full migration lifecycle: discovery of your source environment, migration planning, connecting target AWS accounts, network migration, landing zone setup, and server rehost. It runs as a local process launched by your MCP client and communicates with AWS Transform over outbound HTTPS. The MCP client is the process that runs the server, such as Kiro or Claude Code. The AI assistant is the model experience in that client that you send prompts to. For a general introduction to AWS Transform developer tools, including the Kiro Power, agent plugins, and IDE plugin, see Developer tools.

Installing the MCP server

The Kiro Power and agent plugins install the MCP server automatically. Install the MCP server manually only if you want to use it without a Kiro Power or agent plugin.

Setting up the AWS Transform MCP server

Before you begin, make sure you have the following:

- Python 3.10 or later.
- An AWS Transform account with tenant access. For information about setting up AWS Transform, see Getting started with AWS Transform.
- An MCP-compatible client, such as Kiro, Claude Code, Claude Desktop, Cursor, or Cline.

Install the MCP server

You can install the AWS Transform MCP server from PyPI as `awslabs.aws-transform-mcp-server`. The following examples run the server with `uvx`.

Claude Code

```
claude mcp add awslabs.aws-transform-mcp-server -- uvx awslabs.aws-transform-mcp-
server@latest
```

Kiro, Cursor, VS Code, Cline, and Claude Desktop (macOS/Linux)

Add the following to your client's MCP configuration file. For Kiro, use `~/.kiro/settings/mcp.json`.

```
{
  "mcpServers": {
    "awslabs.aws-transform-mcp-server": {
      "command": "uvx",
      "args": ["awslabs.aws-transform-mcp-server@latest"],
      "env": {
        "AWS_REGION": "us-east-1",
        "FASTMCP_LOG_LEVEL": "ERROR"
      }
    }
  }
}
```

Windows

```
{
  "mcpServers": {
    "awslabs.aws-transform-mcp-server": {
      "command": "uvx",
      "args": [
        "--from",
        "awslabs.aws-transform-mcp-server@latest",
        "awslabs.aws-transform-mcp-server.exe"
      ],
      "env": {
        "FASTMCP_LOG_LEVEL": "ERROR",
        "AWS_REGION": "us-east-1"
      },
      "disabled": false
    }
  }
}
```

For configuration details for your specific MCP client, see the aws-transform-mcp-server README on GitHub.

Configure environment variables

You can set the following environment variables in the `env` block of your MCP client configuration.

Variable	Description
`AWS_REGION`	The AWS Region used for control plane API calls. Defaults to the profile Region, and then to US East (N. Virginia).
`AWS_PROFILE`	The credential profile used for control plane tools. Falls back through the standard credential chain if not set.
`FASTMCP_LOG_LEVEL`	The log level (`DEBUG`, `INFO`, `WARNING`, or `ERROR`). Defaults to `INFO`.
`AWS_TRANSFORM_MCP_WRITE_DIR`	The base directory that artifact downloads are confined to. A download path must resolve within this directory. If your client launches the server outside a shell, set this variable so that artifact downloads have a valid destination.

Authenticate

Most MCP tools use web API authentication (browser login). The `accept_connector` tool also requires AWS credentials, which the server detects automatically. AWS Transform supports the following authentication methods.

IAM Identity Center (SSO)

Ask the AI assistant to "Configure AWS Transform with SSO." The assistant prompts for your IAM Identity Center start URL and opens your browser to log in. The server then saves your credentials locally and loads them automatically when you restart it. Re-run this configuration when tools return authentication errors.

IAM role

Your environment provides AWS credentials automatically. Set `AWS_PROFILE` in your MCP client configuration to select a specific profile.

For more information about setting up authentication for AWS Transform, see the prerequisites earlier in this topic.

Verify the connection

After you install and configure the server, restart your MCP client. Then ask the AI assistant to "Check my AWS Transform connection status." A successful response confirms that the server is installed, configured, and authenticated.

Using AWS Transform for migrations via MCP

With the MCP server connected, you drive your migration by talking to the AI assistant in natural language. The assistant calls AWS Transform MCP tools on your behalf to manage workspaces, launch migration jobs, respond to human-in-the-loop tasks, and retrieve artifacts. A typical migration follows these stages.

To run a migration through the MCP server

1. **Create or select a workspace.** A workspace is a logical container for one or more migration jobs. The workspace determines the AWS Region where your jobs, discovery data, and AWS Transform recommendations reside. Ask the assistant to create a workspace or list your existing workspaces.
2. **Create and start a migration job.** Ask the assistant to create a migration job. The assistant discovers the available agents and starts the job. For information about the migration job types and the steps that each includes, see Job types.
3. **Run discovery.** Upload your on-premises server inventory so AWS Transform can parse, de-duplicate, and validate it. For the supported discovery data sources and formats, see Discover source data.
4. **Build the migration plan.** Work with the assistant to scope and analyze your inventory, group servers into applications, generate move groups, and build migration waves. You can ask questions about your environment and iterate on the plan at any time. For more information, see Build migration plan.
5. **Connect your target account.** Create a connector and have an administrator of the target AWS account approve it so AWS Transform can deploy infrastructure, migrate networks, and rehost servers. For more information, see Connect target AWS accounts and regions.
6. **Migrate the network, build the landing zone, and rehost servers.** Ask the assistant to run the remaining execution steps in your job plan.

7. **Respond to human-in-the-loop tasks.** AWS Transform pauses at review and approval points. The assistant surfaces the task and its artifact for you to review. After you decide, the assistant submits your response (approve, reject, or send for approval). Always review the task details and the agent artifact before you approve.
8. **Track status and download artifacts.** Ask the assistant for job status at any time, and have it download artifacts such as reports, diagrams, and workspace summaries to your local environment.

Because you interact with AWS Transform conversationally, you can move between these stages iteratively, ask clarifying questions, and refine your plan without switching tools.

Best practices

Follow these best practices when you run migrations through the AWS Transform MCP server.

- **Review before you approve.** Always inspect the task details and the agent artifact before you approve a human-in-the-loop task. Do not let the assistant auto-submit approvals for migration steps that create or modify AWS resources.
- **Use least-privilege credentials.** Configure the `AWS_PROFILE` environment variable to point at a profile scoped to only the permissions your migration requires. For multi-account migrations, prefer a Delegated Administrator account over the organization management account. For more information, see Using a delegated administrator account.
- **Confine artifact downloads.** Set the `AWS_TRANSFORM_MCP_WRITE_DIR` environment variable to a dedicated directory so that downloaded artifacts land in a known, contained location. This is especially important when a desktop or IDE client launches the server outside a shell.
- **Keep your session authenticated.** SSO tokens expire. Re-run SSO configuration when tools begin returning authentication errors, and verify your connection with a status check before starting long-running steps.
- **Set the correct Region.** Set `AWS_REGION` to the AWS Region where your workspace and discovery data reside. If your migration target Region differs from your discovery Region, some data is transferred across AWS Regions. For more information, see Supported target Regions.
- **Upload complete, current discovery data.** The quality of your migration plan depends on the quality of your inventory. Upload the most detailed data available and verify it reflects your current environment before you build waves.

- **Consider expert mode for large migrations.** When you want to minimize back-and-forth, provide your inputs up front and let the agent apply them automatically. For more information, see Use expert mode.

Example prompts

Running AWS Transform through the MCP server does more than move the web experience into your IDE. The AI assistant chains multiple tool calls, reads and writes files in your local workspace, and applies conditional logic. As a result, you can express multi-step, iterative, and batch workflows that go beyond a single web-application interaction. The following examples show sophisticated prompts. Adapt them to your environment.

Verification and monitoring loops

Have the assistant poll a running step, react to what it finds, and pause for you only when a decision is required. An autonomous loop like this is driven by the assistant, not the web application.

```
Start network migration for wave 1, then poll the job status every two
minutes until it completes or needs my input. If a human-in-the-loop task
appears, stop polling, summarize the task and its artifact for me, and wait
for my decision before doing anything else.
```

```
Check every active job across all my workspaces. For each one, give me a
one-line status, and flag any job that has been waiting on an approval for
more than 24 hours or has reported an error. Repeat this check every 15
minutes and only message me when something changes.
```

```
Re-run the inventory readiness check after my latest upload. If there are any
data-quality errors, list the affected servers grouped by issue and stop. If
it validates cleanly, proceed to application grouping using the rules in
./planning/grouping-rules.md.
```

Reports that combine assessment and planning data

Ask the assistant to cross-reference artifacts from different stages and synthesize a consolidated report. The assistant can then save it to your local workspace for review or version control.

```
Pull my migration strategy (7Rs) recommendations and my current wave plan.
```

```
Cross-reference them and produce a consolidated readiness report that
highlights every application whose recommended strategy conflicts with the
strategy assigned to its wave. Include the confidence score and reasoning for
each conflict, and save the report to ./reports/readiness.md.
```

```
Build an executive migration report that combines my risk assessment scores,
the 7Rs recommendations, and the wave schedule. Rank waves by combined risk,
call out the top five applications that need attention before cutover, and
export it as both a PDF for stakeholders and a PPTX deck for my review.
```

```
Compare the assessment I ran last month with my current plan. Tell me which
servers were newly discovered, which changed operating system or strategy, and
which moved between waves. Write the diff to ./reports/plan-change-log.md.
```

Per-application diagrams and batch generation

Iterate over the applications or waves in your plan and generate an artifact for each, saving them locally in one pass.

```
For each application in my plan, generate an application dependency diagram
and save it to ./diagrams/APP_NAME.html. When you finish, give me a table of
every application with the number of dependencies found, and list any
applications you could not generate a diagram for and why.
```

```
For each wave, generate a network topology diagram scoped to that wave's
servers plus their cross-wave dependencies, and save each one to
./diagrams/waves/wave-N.html. Highlight any dependency that crosses a wave
boundary, since those need sequencing attention.
```

Local-environment integration

Combine AWS Transform with files and tools in your local development environment, something the web application cannot do.

```
Read the decommissioning list in ./exclusions.csv, remove every matching
server from migration scope, then rebuild my waves and show me what changed in
the plan as a result.
```

```
Read the tagging standard in ./standards/tagging.md and apply it as the
```

```
resource tagging configuration for network migration and server rehost. Show
me the resulting tag set and flag any required tag the standard does not
specify a value for.
```

```
Download the workspace summary and the 7Rs report, commit them to the
migration-artifacts git repo in this workspace under ./artifacts/$(date +%F)/,
and write a short changelog entry describing what changed since the last
snapshot.
```

Conditional and gated automation

Express approval gates and branching so the assistant advances the migration only when your conditions are met.

```
Walk my end-to-end job through discovery and planning in expert mode using the
inputs in ./planning/inputs.txt. Pause for my review after wave planning
completes. Do not connect any target account or start execution until I
explicitly approve.
```

```
For wave 1 only, connect the target account, migrate the network, and rehost
the servers. After each step, verify it succeeded before starting the next. If
any step fails or produces a warning, stop immediately, show me the details,
and do not touch waves 2 or 3.
```

Release notes

The following release notes cover the latest changes to AWS Transform for migrations, including *Migration assessments* and *Migrations (including VMware)*. For a list of changes across the full AWS Transform service, see the changelog.

For supported AWS Transform regions, see Supported Regions. For supported target regions, see the account connector setup page.

October 2026

- AWS Transform landing zone agent now supports connecting through a Delegated Administrator account, allowing organizations with strict governance policies to use AWS Transform without granting direct access to the management account. When using a Delegated Administrator connector, the agent operates in IaC-only mode, generating Infrastructure as Code artifacts

(CDK, Terraform, or Landing Zone Accelerator format) for your own deployment pipeline instead of directly provisioning resources. This provides flexibility for organizations that prefer to manage landing zone changes through existing deployment processes. Learn more about connector setup and IaC-only mode.

September 2026

- AWS Transform landing zone agent now supports additional brownfield scenarios, including existing AWS Control Tower landing zones and landing zones built with AWS Organizations. You can run a foundation gap analysis and expand your landing zone on top of the existing landing zone in your target account. Learn more about foundation discovery and planning.
- AWS Transform migration assessments now assess self-managed Apache Kafka clusters for migration to Amazon MSK Express. AWS Transform recommends a broker configuration and estimates the total cost of ownership across broker hours, storage, data-in, and cross-Availability Zone transfer, along with a compatibility review of the source cluster. Learn more about analytics assessments.
- AWS Transform migration assessments now include Compute Savings Plans and Amazon EC2 Instance Savings Plans as pricing options for Amazon EC2 recommendations, alongside On-Demand and Reserved Instances, so you can compare committed-use discounts in your migration business case. Learn more about compute assessments.

August 2026

- AWS Transform for migrations now lets you refine the generated AWS security groups while reviewing your target network. You can create, configure, and delete security groups and their rules. You can also associate a rule with a CIDR range or a source , and enable security group referencing for a specific rule. Learn more about reviewing and optimizing your network.
- AWS Transform for migrations now supports for as a generally available block storage target, alongside Amazon EBS. You can migrate block storage directly to for within the same migration wave that handles compute and networks. Learn more about configuring for as a target storage type.
- AWS Transform for migrations now lets you apply your source security posture to Amazon VPCs you have already provisioned in AWS, instead of creating new Amazon VPCs. You upload a source network file that contains firewall rules, tag the existing Amazon VPCs you want in scope, and AWS Transform matches your source subnets to those Amazon VPCs by and generates the

corresponding security groups. Learn more about applying security posture to existing Amazon VPCs.

- AWS Transform for migrations now identifies unused inbound firewall rules migrated from your on-premises environment and suggests removing them as part of guided network recommendations. This helps you avoid carrying forward security exposure, such as open inbound access, that no longer serves a purpose. Removal is limited to unused ingress rules. Learn more about guided network recommendations.
- AWS Transform for migrations now flags Amazon VPCs that host applications from multiple environments, such as development and production, as part of guided network recommendations. When the environments can be cleanly separated, AWS Transform suggests splitting the Amazon VPC to isolate them. Learn more about guided network recommendations.
- AWS Transform for rehost now allows you to modify server configurations through the chat interface during the inventory validation step. In addition to editing the inventory file directly, you can specify changes for individual columns on specific source servers without downloading and editing the entire inventory file. This provides a more convenient way to make targeted updates to your server configurations. Learn more about inventory validation and modifications.

July 2026

- AWS Transform for migrations now offers expert mode, which lets you provide your inputs up front so the agent completes migration steps automatically with minimal back-and-forth. This helps you move faster through large migrations while keeping control of the decisions that matter. Learn more about expert mode.
- AWS Transform for migrations now supports migrating virtual and bare metal server environments from virtually any source, including VMware, Hyper-V, and other platforms. Migration capabilities, including discovery, migration planning, landing zone creation, network migration, and server rehost, are available regardless of your source infrastructure. The web application workflow has been renamed from "VMware Migrations" to "Migrations (including VMware)". Learn more about migrations.
- AWS Transform now lets you generate a workspace summary report as a downloadable PDF. The report consolidates data from all migration jobs in your workspace, including job statuses, workflow step progress, wave planning, network migration, landing zone configuration, rehost progress, and artifacts produced. Learn more about workspace summary reports.
- The AWS Transform discovery tool now collects detailed Oracle metadata directly from your databases across VMware, Hyper-V, and bare metal servers. You can discover Oracle instances

through direct SQL connections. The tool collects Container Database/Pluggable Database (CDB/PDB) enumeration, component inventory, and datafile sizing. Learn more about Oracle Database discovery.

- AWS Transform for migrations now provides enhanced capabilities for post-launch actions. You can now define post-launch actions at the account level and apply them to all source servers automatically, or customize them per source server during inventory validation. Post-launch actions automate modernization and validation tasks on each source server immediately after it launches as a test or cutover instance. Learn more about post-launch actions.

June 2026

- AWS Transform for migrations now supports localization. You can change the display language of the web application when working with migration workflows. Learn more about language settings.
- AWS Transform now allows you to configure your replication settings and launch settings for your migration. You can set the configuration per your target account or across specific source servers. Learn more about replication and launch settings.
- AWS Transform now allows you to attach existing Elastic Network Interfaces (ENIs) to your launch template. You can tag ENIs in your target account so they are available for use when instances are launched. Learn more about network resource tagging.
- AWS Transform for migrations now supports all AWS commercial regions as migration targets, excluding Middle East (Bahrain) and Middle East (UAE). Learn more about supported target regions.
- You can now use the AWS Transform discovery tool as a source for network migration alongside modelizeIT, enabling hybrid network migrations for environments running both VMware and non-VMware workloads. Learn more about network migration input sources.

May 2026

- AWS Transform now detects existing VPCs in your target account during network migration review. You can see existing VPCs alongside your mapped VPCs, identify CIDR conflicts, and resolve them before deployment. Learn more about existing VPC detection.
- AWS Transform supports replatforming source code repositories to containers during migration to AWS. Learn more about containerization.

- AWS Transform can now analyze your discovered inventory and provide a migration strategy recommendation for every server and application. Recommendations follow the industry-standard seven migration strategies (7Rs). Each recommendation includes a suggested AWS target service, a confidence score, and the reasoning behind it. Learn more about 7Rs recommendations.

April 2026

- Added landing zone creation directly within migration workflows. AWS Transform checks for an existing foundation, recommends Organizational Unit (OU) structures and target accounts, and offers automated deployment or Infrastructure as Code (IaC) output (CloudFormation, AWS CDK, or Landing Zone Accelerator (LZA) formats). Learn more about landing zone creation.
- Added support for DHCP with security group mapping during network migration. Learn more about security group creation.
- You can now generate interactive diagrams and analytical reports during migration planning. Diagram types include network topology maps, application dependency graphs, wave Gantt charts, and general charts. Report types include risk assessments, 7Rs recommendations, and custom analytical reports. Outputs are available as interactive HTML, PDF, or Microsoft PowerPoint (.pptx) files. Learn more about migration-planning diagrams and reports.

March 2026

- Added public APIs for network migration, enabling partner and programmatic access to network migration capabilities. See the network migration API reference.

February 2026

- Network migration now supports ingesting firewall or Software-Defined Networking (SDN) configuration files without requiring RVTools discovery data. You can use firewall or SDN configuration files independently, or combine them with RVTools exports.

January 2026

- Added support for Migration Acceleration Program (MAP) tagging and user-defined tagging on network migration elements, enabling you to track migration resources for MAP credit and organizational purposes.

December 2025

- Added a new migration experience with redesigned workflow and improved usability. For more information, see Accelerating VMware migration: AWS Transform's new experience.
- Added Landing Zone Accelerator (LZA) network configuration for automated landing zone setup.
- Added support for multiple target AWS accounts in migration execution, enabling you to migrate workloads across accounts.
- Added network migration support for Cisco ACI, Palo Alto, FortiGate, and ModelizeIT source formats.

October 2025

- Expanded migration execution to 16 target regions with the addition of Asia Pacific (Osaka).

September 2025

- Added support for licensing configuration in rehost waves, enabling you to specify Bring Your Own License (BYOL) or license-included options per wave.
- Added support for Terraform in network migration, enabling you to deploy network infrastructure using Terraform templates.
- Added support for updating replication subnet configuration, removing the requirement for a default VPC in the target account.

August 2025

- Expanded migration execution (network migration and rehost) to 15 target regions with the addition of US East (Ohio), Europe (Stockholm), and Europe (Ireland).

- Added flexible CIDR range capabilities for network migration, enabling you to customize IP address ranges during network deployment.

May 2025

- Initial service launch of *Migrations (including VMware)*, providing end-to-end migration capabilities including discovery, assessment, migration planning, network migration, and server rehost.
- Supported migration execution in 12 target regions: US East (N. Virginia), US West (Oregon), Canada (Central), South America (São Paulo), Europe (Frankfurt), Europe (London), Europe (Paris), Asia Pacific (Mumbai), Asia Pacific (Seoul), Asia Pacific (Tokyo), Asia Pacific (Singapore), and Asia Pacific (Sydney).

Source code containerization

AWS Transform supports replatforming applications to containers during migration to AWS. This chapter describes AWS Transform's agentic AI capabilities to automate the containerization of your source code. You can migrate and modernize in parallel, reducing the time and complexity of moving from on-premises to cloud-native architectures. You can containerize source code from GitHub, Bitbucket, GitLab, or .zip files, generate Docker images, publish to Amazon Elastic Container Registry (Amazon ECR), and deploy to Amazon Elastic Container Service (Amazon ECS) or Amazon Elastic Kubernetes Service (Amazon EKS). This brings containerization into the same workflow you use to plan and execute rehost migrations.

Capabilities and key features

AWS Transform offers the following capabilities for containerizing your applications.

- **AI-driven source code analysis**. Analyzes your application source code and automatically generates Docker artifacts, including Dockerfiles and related configuration files. No Docker expertise required.
- **Container image building and publishing**. Builds and tests container images automatically, and publishes them to Amazon Elastic Container Registry with automated vulnerability scanning.
- **Infrastructure as Code generation**. Generates production-ready deployment infrastructure for either Amazon Elastic Kubernetes Service (Helm charts) or Amazon Elastic Container Service (Terraform modules), with automated validation and security scanning. No need to write IaC from scratch.
- **Private dependency support**. You can optionally configure AWS CodeArtifact repositories (Maven, PyPI, npm) and private Amazon ECR base images as dependency sources for your builds.
- **Iterative test and cutover deployment**. Deploys test infrastructure for validation, then deploys finalized infrastructure for production cutover.

> **Note**
>
> Source code containerization is designed for applications that are not yet containerized. It requires access to your application source code and does not support migrating existing containerized workloads. To migrate existing containers to AWS, use standard deployment methods for Amazon Elastic Container Service or Amazon Elastic Kubernetes Service.

How containerization works

AWS Transform uses an AI-powered agent to guide you through the containerization process in a chat-based workflow. The agent handles the entire process from code analysis to deployed container, reducing days of manual Dockerfile authoring and infrastructure setup to hours. AWS Transform coordinates specialized tasks such as source code analysis, Docker image generation, and infrastructure creation. At key points in the workflow, you review and approve the output before proceeding.

You access source code containerization by creating a migration job. Within the job, you can run containerization as a standalone workflow, or as part of an end-to-end migration when a wave's migration strategy is set to *containerize*. For more information, see *Migrations (including VMware)*.

Prerequisites

Before you begin, ensure that you have the following:

- An AWS Transform workspace. For information about getting a workspace, see Getting started.
- Your application source code in a Git repository accessible through AWS CodeConnections, or packaged as zip files for upload. Individual files must not exceed 1 GB, and the total size of all source code must not exceed 8 GB.
- (Optional) An Amazon ECR repository for publishing container images. You can configure Amazon ECR access later in the workflow when you are ready to publish.
- For Amazon EKS deployments: An existing Amazon EKS cluster, or permissions to create the required infrastructure.
- For Amazon ECS deployments: Permissions to create Amazon ECS clusters, services, and related resources using Terraform, AWS CloudFormation, or AWS Cloud Development Kit (AWS CDK).

Containerization workflow

The containerization workflow consists of the following steps. The AWS Transform agent guides you through each step in the chat interface.

1. the section called “Step 1: Security disclaimer”. Review and accept the security disclaimer.
2. the section called “Step 2: Clone source code”. Provide your application source code.

3. the section called “Step 3: Containerize”. AI agent analyzes your code and generates Docker artifacts.
4. the section called “Step 4: Review artifacts”. Review generated artifacts and approve code changes.
5. the section called “Step 5: Publish images”. Publish container images to Amazon ECR.
6. the section called “Step 6: Generate IaC”. Generate Amazon EKS or Amazon ECS deployment templates.
7. the section called “Step 7: Deploy test infrastructure”. Deploy and validate test infrastructure.
8. the section called “Step 8: Clean up test infrastructure”. Tear down test resources.
9. the section called “Step 9: Deploy cutover infrastructure”. Deploy production infrastructure.

Starting a containerization job

To containerize your applications, you first create a VMware migration job in your AWS Transform workspace. From within the job, you can choose to run a standalone containerization workflow or an end-to-end migration flow that includes containerization.

- **Standalone containerization** — Containerize your source code without performing a full VMware migration. Choose this option when you want to containerize applications independently of any infrastructure migration.
- **End-to-end migration with containerization** — Run the full VMware migration workflow with a *containerize* migration strategy assigned to one or more waves. The containerization workflow runs as part of the migration for those waves. For more information about VMware migration, see *Migrations (including VMware)*.

To start a containerization job

1. On your workspace landing page, choose **Create a job**.
2. Choose **VMware migration**.
3. Choose whether to run a standalone containerization workflow or an end-to-end migration flow that includes containerization.
4. AWS Transform guides you through the workflow steps, starting with the section called “Step 1: Security disclaimer”.

Containerization permissions

During the containerization workflow, AWS Transform deploys an IAM role to your AWS account that uses to build container images and deploy infrastructure. You are asked to review and approve the creation of this role before the workflow continues.

The role is named `AWSTransformCodeBuildExecutionRole` and is deployed through an AWS CloudFormation stack. It includes the following managed policies.

Base policy

Provides core permissions for the containerization workflow:

- Amazon S3 — Read and write objects in `aws-transform-*` buckets
- Amazon ECR — Authenticate, pull images (tagged with `Project: atx-migration`), and push images to repositories tagged with `CreatedBy: AWSTransform`
- AWS CodeArtifact — Read from repositories tagged with `Project: atx-migration`, list repositories, obtain authorization tokens, and read repository endpoints. These permissions support both public dependency resolution and private dependency sources configured during the workflow
- Amazon CloudWatch Logs — Create log groups and streams, and write log events for and Amazon ECS log groups
- AWS KMS — Describe and decrypt keys (scoped to Amazon S3 via-service condition)
- Amazon EC2 — Create and manage network interfaces for VPC-enabled projects
- AWS CodeConnections — Use connections to access source code repositories through the configured CodeConnections ARN

Networking policy

Manages networking resources for deployed applications:

- Elastic Load Balancing — Create, describe, and manage load balancers, target groups, listeners, and rules (tagged with `CreatedBy: AWSTransform`)
- Route 53 — Create and manage hosted zones and DNS records
- AWS Cloud Map — Create and manage namespaces and services for service discovery (tagged with `CreatedBy: AWSTransform`)

Storage policy

Manages storage resources for deployed applications:

- Amazon S3 — Create and manage buckets, including encryption, versioning, lifecycle, and access policies
- Amazon EFS — Create and manage file systems, mount targets, and access points (tagged with `CreatedBy: AWSTransform`)
- Amazon EBS — Create and manage volumes and snapshots (tagged with `CreatedBy: AWSTransform`)

AWS KMS policy

Manages encryption key operations:

- Describe keys, list aliases, and read key policies
- Encrypt, decrypt, and generate data keys (scoped to Amazon CloudWatch Logs, Amazon EFS, Amazon EC2, and Amazon S3 via-service conditions)

Amazon ECS policy

Manages Amazon Elastic Container Service resources for container deployments:

- Create and manage clusters, services, and task definitions (tagged with `CreatedBy: AWSTransform`)
- Register and deregister task definitions, run tasks
- Pass IAM roles to Amazon ECS tasks and services
- Read IAM role information and ACM certificates
- Describe Amazon EC2 VPCs, subnets, security groups, and network interfaces

Amazon EKS policy

Manages Amazon Elastic Kubernetes Service resources for Kubernetes deployments:

- Access the Kubernetes API, describe clusters, and list add-ons

- Pass IAM roles to Amazon EKS

> **Note**
>
> All permissions are scoped to your AWS account and AWS Region. Resources created by AWS Transform are tagged with `CreatedBy: AWSTransform`, and write operations are restricted to resources with this tag where applicable.

Step 1: Review security disclaimer

Before the containerization workflow begins, AWS Transform presents a security disclaimer that you must review and accept.

What the disclaimer covers

The containerization process requires internet access to download application dependencies (such as packages from public registries like npm, PyPI, or Maven Central). The security disclaimer informs you that:

- Build environments will have outbound internet access to resolve and download dependencies specified in your application's build configuration.
- You are responsible for reviewing the dependencies that your application requires and ensuring they meet your organization's security policies.
- You can optionally configure private dependency sources (such as AWS CodeArtifact repositories or private Amazon ECR base images) in a later step to avoid downloading from public registries.

What you need to do

To review and accept the security disclaimer

1. Read the security disclaimer that AWS Transform presents.
2. Accept the disclaimer to proceed with the containerization workflow.

After you accept the disclaimer, AWS Transform proceeds to the next step where you provide your source code.

> **Note**
>
> The connector configuration for AWS CodeConnections and Amazon ECR is no longer required upfront. AWS Transform prompts you to configure access to these services later in the workflow, only when needed. For example, you configure CodeConnections when you choose to clone from Git repositories, and you configure Amazon ECR access when you are ready to publish container images.

Step 2: Clone source code

In this step, you provide your application source code to AWS Transform. You can provide source code from Git repositories or by uploading zip files.

Option 1: Git repositories

You can provide your source code from Git repositories. If you have not yet configured AWS CodeConnections, AWS Transform prompts you to provide your CodeConnections ARN at this point. You have two ways to specify your repositories:

- **Upload a CSV file** — Prepare a CSV file with columns `repo_name` and `branch`. The `repo_name` must be in `owner/repo-name` format. The `branch` column is optional; leave it empty to use the repository's default branch. Example:

```
repo_name,branch
owner/my-app,main
owner/api-service,develop
owner/another-repo,
```

- **Provide repository details in the chat** — Tell AWS Transform the repository name(s) and branch(es) directly in the chat instead of uploading a CSV.

AWS Transform validates that each repository exists and is accessible through your CodeConnections configuration, then clones the repositories.

Note

Individual files must not exceed 1 GB, and the total size of all repositories must not exceed 8 GB. AWS Transform clones multiple repositories concurrently for faster processing.

Important

Applications that span multiple repositories are not supported through the Git clone option. If your application requires source code from multiple repositories, package them together as a zip file and use the zip upload option instead.

Option 2: Zip file upload

If you did not configure CodeConnections, or prefer to upload your source code directly, you can upload zip files.

To upload source code as zip files

1. When prompted, upload one zip file per application. You can drag files into the chat or use the attachment button.
2. Each zip file should contain the source code for one application.
3. Individual files must not exceed 1 GB, and the total size of all uploads must not exceed 8 GB.

Optional: Configure private dependencies

If your application depends on packages from private registries, you can configure private dependency sources during this step. AWS Transform supports the following private dependency types:

- **AWS CodeArtifact repositories** — Configure Maven, PyPI, or npm repositories hosted in AWS CodeArtifact. AWS Transform presents a list of available repositories in your account for you to select from.
- **Private Amazon ECR base images** — If your Dockerfiles use base images from a private Amazon ECR repository, provide the repository details so that the build environment can pull these images.

When you configure private dependencies, AWS Transform prompts you to connect to the AWS account where these resources are hosted. This connector configuration is only required if you use private dependencies.

Confirming containerization preferences

After your source code is available, AWS Transform asks if you have specific instructions for how the containerization should be done. You can optionally specify:

- Whether to generate a new Dockerfile or reuse an existing one.
- Whether to process a specific service in a monorepo, or all services.
- Any other preferences for the containerization process.

The defaults are:

1. Use Amazon Linux 2023 base image (`public.ecr.aws/amazonlinux/amazonlinux:latest`).
2. Process all applications in the repository.
3. Reuse an existing Dockerfile if present, with minor modifications if needed.

You can confirm to proceed with defaults, or provide specific instructions.

Step 3: Containerize

In this step, the AWS Transform AI agent analyzes your application source code and generates Docker artifacts. AWS Transform examines your application's structure, dependencies, and runtime requirements to produce appropriate Dockerfiles and related configuration files.

What happens during containerization

The containerization agent performs the following tasks:

1. **Source code analysis** — AWS Transform examines your application's project files, dependencies, build configuration, and runtime requirements. AWS Transform also replaces hardcoded values (such as database IP addresses or DNS names), identifies environment variables, detects volumes, and attempts to redirect logs to standard output.

2. **Dockerfile generation** — Based on the analysis, AWS Transform generates a Dockerfile tailored to your application.
3. **Container image build and test** — AWS Transform builds the container image and runs a test to verify that the Dockerfile produces a working image. If the build fails, AWS Transform iterates on the Dockerfile automatically.
4. **Security analysis** — AWS Transform scans the generated artifacts for hardcoded values such as credentials, API keys, or other sensitive data, and flags any findings for your review.

If you provided multiple repositories or a monorepo with multiple services, the processes each service in parallel.

Supported application types

AWS Transform supports containerizing the following application types:

- **.NET 7 and later** — Optimized containerization for .NET applications running on version 7 or later.
- **General applications** — Containerization for applications built with any language or framework, including Java, Python, Node.js, Go, and others.
- **Unsupported applications** — Windows, Xcode, or any proprietary build environments are not supported.

What you need to do

This step runs automatically. AWS Transform displays progress updates as it analyzes and containerizes each application. If AWS Transform encounters issues, it may ask you for clarification or additional information.

When containerization is complete, AWS Transform moves to the next step where you review the generated artifacts. The outputs are saved in the Artifact Store.

Step 4: Review Docker artifacts and code changes

In this step, you review the Docker artifacts and code changes that AWS Transform generated during containerization.

Reviewing changes from Git repositories

If you provided source code from Git repositories, AWS Transform commits the containerization changes to a new branch in each repository. If you have multiple repositories, AWS Transform presents all changes as a batch for consolidated review.

To review and approve changes

1. Review the code diff links that AWS Transform provides. Each diff shows all files that were added or modified, including Dockerfiles, configuration files, and any source code changes.
2. When AWS Transform presents the approval dialog, approve or reject the changes. For multiple repositories, you approve all changes in a single batch operation.
3. If you approve, AWS Transform pushes the changes to the new branches in your repositories.
4. Confirm that you are ready to proceed to the next step.

Reviewing changes from zip uploads

If you uploaded source code as zip files, AWS Transform provides download links for the containerized artifacts. Download and review the artifacts, which include the generated Dockerfiles and any source code modifications.

Requesting changes

If the generated artifacts need modifications, you can provide feedback to the AWS Transform restarts the containerization step with your updated instructions and generates new artifacts for your review.

Step 5: Publish images

In this step, AWS Transform publishes your container images to Amazon Elastic Container Registry. If you have not yet configured Amazon ECR access, AWS Transform prompts you to provide your Amazon ECR repository details at this point.

What happens during publishing

AWS Transform performs the following tasks for each container image:

1. **Connector configuration** — If you have not already configured Amazon ECR access, AWS Transform presents a connector form where you provide your Amazon ECR repository ARN and connect to your AWS account.
2. **Approval request** — AWS Transform presents the list of images to be published and requests your approval. You can approve or reject the publishing operation.
3. **Image push** — After approval, AWS Transform publishes the container images to your Amazon ECR repository using . If you have multiple services, AWS Transform publishes all images in a single batch operation.
4. **Vulnerability scanning** — Amazon ECR automatically scans the published images for known vulnerabilities. AWS Transform reports the scan results. For more comprehensive analysis, enable AWS ECR enhanced scanning and AWS GuardDuty Runtime Monitoring.

> **Note**
>
> Image publishing runs asynchronously using . If your session is interrupted during publishing, AWS Transform automatically recovers the operation when you reconnect.

What you need to do

- Review the list of images to be published.
- Approve or reject the publishing operation. If you reject, AWS Transform skips image publishing and you can proceed to the next step without published images.
- Review the vulnerability scan results after publishing completes.

After publishing, AWS Transform provides the image URIs that are used in the infrastructure deployment steps.

Step 6: Generate Infrastructure as Code

In this step, AWS Transform generates AWS infrastructure templates for deploying your containerized application. You choose between Amazon Elastic Kubernetes Service and Amazon Elastic Container Service as your target platform.

Amazon Elastic Kubernetes Service (Kubernetes)

If you choose Amazon EKS, AWS Transform generates Helm charts with Kubernetes manifests for your application. The generated templates include:

- Deployments and services
- Ingress rules
- ConfigMaps and SecretProviderClass resources
- Service accounts
- Persistent volume claims (EBS and EFS)
- Route 53 DNS records
- Health check configurations

Optional: CloudWatch logging

AWS Transform asks if you want to include CloudWatch logging for your Amazon EKS deployment. If you opt in, AWS Transform generates a Fluent Bit DaemonSet configuration that collects pod logs and ships them to CloudWatch Logs.

If you choose CloudWatch logging, your cluster administrator must complete the following prerequisites before deployment:

1. Create the `amazon-cloudwatch` namespace in the cluster.
2. Create an IAM role for the Fluent Bit service account with CloudWatch Logs write permissions. You are prompted for the role ARN during deployment.

Amazon Elastic Container Service (Terraform)

If you choose Amazon ECS, AWS Transform generates Terraform modules for deploying your application. The generated templates include:

- ECS cluster
- ECS service with task definitions
- Application Load Balancer with access logging
- Amazon EFS for persistent storage

- Private hosted zones for service discovery
- Health check configurations

Automated validation and security scanning

AWS Transform automatically validates the generated infrastructure templates before presenting them to you. The validation includes:

- For Terraform: `terraform fmt` and `terraform validate` to check syntax and configuration correctness.
- For Helm: Chart validation to verify that the generated manifests are well-formed.
- Security scanning using Checkov to identify potential security misconfigurations in the generated infrastructure code.

AWS Transform reports any validation issues and resolves them before proceeding.

Generating IaC without published images

You can generate infrastructure templates even if you did not publish container images in the previous step. The generated templates use placeholder image URIs that you can update later with your actual image locations.

What you need to do

To generate infrastructure templates

1. When prompted, choose your target platform: **Amazon EKS** or **Amazon ECS**.
2. If you chose Amazon EKS, AWS Transform asks whether to include CloudWatch logging. Respond yes or no.
3. If prompted, provide health check endpoint details for your application (such as a `/health` path).
4. AWS Transform generates and validates the infrastructure templates. This step runs automatically and AWS Transform displays progress updates, including any validation or security scan results.
5. Review the generated templates when AWS Transform presents them.

Step 7: Deploy test infrastructure

In this step, AWS Transform deploys test infrastructure so that you can validate your containerized application before production cutover.

What happens during test deployment

AWS Transform deploys the infrastructure templates generated in the previous step to your AWS account. For Amazon EKS deployments, this uses Helm. For Amazon ECS deployments, this uses Terraform.

Before deploying, AWS Transform presents an infrastructure values form where you can configure deployment parameters such as resource names, replica counts, and environment-specific settings. AWS Transform also generates a deployment preview that shows what resources will be created, so you can review the changes before approving.

What you need to do

To deploy and validate test infrastructure

1. Complete the infrastructure values form with your deployment configuration. The form includes parameters such as cluster names, namespaces, replica counts, and other environment-specific settings.
2. Review the deployment preview that shows the resources that will be created or modified.
3. Approve the infrastructure deployment when AWS Transform requests it.
4. Wait for the deployment to complete. AWS Transform displays the deployment results.
5. Test your deployed application to verify that it runs correctly in a container and that the infrastructure meets your requirements.
6. Tell AWS Transform whether you are satisfied with the deployment, or whether you want to modify the infrastructure and redeploy.

Iterating on the infrastructure

If you need to modify the infrastructure templates, you can upload updated templates and redeploy:

1. Tell AWS Transform you want to modify the IaC.

2. Upload a zip file containing your modified infrastructure templates.
3. AWS Transform replaces the templates and performs an incremental update (Terraform apply or Helm upgrade).
4. Review the updated deployment results.

You can repeat this cycle until you are satisfied with the deployment.

Session recovery

Infrastructure deployments run asynchronously using . If your session is interrupted during a deployment, AWS Transform automatically recovers the operation when you reconnect and displays the deployment results.

Step 8: Clean up test infrastructure

After you complete validation, AWS Transform tears down the test infrastructure to avoid unnecessary costs.

What you need to do

To clean up test infrastructure

1. AWS Transform informs you that the test infrastructure will be torn down.
2. Approve the destroy operation when AWS Transform requests it.
3. Wait for the cleanup to complete. AWS Transform displays the results when all test resources have been removed.

For Amazon ECS deployments, this step removes all resources that were created during the test deployment, including compute resources, load balancers, storage, and networking components.

For Amazon EKS deployments, the cleanup is scoped by label to your specific project. Only resources tagged for this containerization workflow are removed. Other workloads running on the same Amazon EKS cluster are not affected.

Step 9: Deploy cutover infrastructure

In this final step, AWS Transform deploys the finalized infrastructure for production use. The cutover deployment uses the same infrastructure templates that you validated during the test deployment step.

What you need to do

To deploy cutover infrastructure

1. Complete the infrastructure values form with your production deployment configuration. You can adjust parameters from the test deployment or use the same values.
2. Review the deployment preview that shows the production resources that will be created.
3. Approve the production deployment when AWS Transform requests it.
4. Wait for the deployment to complete. AWS Transform displays the cutover deployment results.
5. Verify that your application is running correctly in the production environment.

> **Important**
>
> Review the generated infrastructure configurations with your team to ensure that they meet your organization's security, compliance, and business requirements. While AWS Transform provides automated configuration recommendations, you are responsible for validating and adjusting the settings to match your needs before proceeding with the cutover deployment.

Session recovery

Production deployments run asynchronously using . If your session is interrupted during deployment, AWS Transform automatically recovers the operation when you reconnect and displays the deployment results.

Modernization of mainframe applications

AWS Transform is designed to accelerate the modernization of legacy mainframe applications. It orchestrates the analysis of mainframe codebases, generate documentation, extract business logic, decompose monolithic structures, transform legacy code, and manage the overall journey with human inputs (HITL) when needed. The transformation capabilities of AWS Transform for modernizing and migrating mainframe applications empower you to modernize your critical mainframe application faster, while preserving your business-critical logic throughout the transformation process.

Topics

- Capabilities and key features
- High-level walkthrough
- Human in the loop (HITL)
- Supported file types for transformation of mainframe applications
- Supported Regions and quotas for AWS Transform mainframe
- High-level overview of mainframe modernization journey
- Transformation of mainframe applications
- Build and deploy your modernized application post-refactoring

Capabilities and key features

AWS Transform provides the following capabilities for mainframe modernization:

- Supports modernization of z/OS mainframe applications written in COBOL (Common Business-Oriented Language) and PL/I (Programming Language One) with associated JCL (Job Control Language), CICS (Customer Information Control System) transactions, BMS (Basic Mapping Support) screens, Db2 databases, and VSAM (Virtual Storage Access Method) data files.
- Supports refactoring of Fujitsu GS21 mainframe applications with PSAM (Presentation Service Access Method), Japanese character sets, and NDB (Network Data Base) data files.
- Performs goal-driven reasoning, analysis, decomposition, planning, documentation generation, and code refactoring.
- Automatically refactors COBOL-based mainframe workloads into modern, cloud-optimized Java applications.

- Orchestrates and integrates seamlessly with underlying tools executing analysis, documentation, decomposition, planning, and code refactoring.
- Helps you set up cloud environments for modernized mainframe applications by providing ready-to-use Infrastructure as Code (IaC) templates.

High-level walkthrough

Here's a high-level walkthrough of AWS Transform for modernizing and migrating mainframe applications.

1. Start a chat with AWS Transform, and enter an objective.
2. Based on your objective, AWS Transform proposes a modernization plan—breaking down the high-level goal into intermediate steps.
3. Depending on the goal you provided, AWS Transform can:
   - Assess and reimagine the codebase
   - Create a custom job based on the objective

Along the way, AWS Transform might request information from you to execute the tasks.

> **Note**
>
> For more information about AWS Transform for mainframe refactor capabilities, see Create a project in the AWS Transform for mainframe refactor documentation.

Human in the loop (HITL)

Throughout the transformation of mainframe applications, you can monitor the progress and status of the transformation tasks through the AWS Transform web experience.

AWS Transform will gather additional information from you in the following scenarios:

- When additional information is needed to execute tasks.
- When approval is required for intermediate artifacts (for example, domains decomposition or wave planning).

- When issues arise that AWS Transform cannot automatically resolve.

Supported file types for transformation of mainframe applications

The supported file types for z/OS include:

- COBOL artifacts and related CPY (Copybooks)
- PL/I source files
- JCL (Job Control Language) and JCL Procedure (PROC)
- CICS System Definition (CSD)
- BMS (Basic Mapping Support)
- Db2 databases
- VSAM (Virtual Storage Access Method)
- IMS TM (Transaction Manager)

The supported file types for Fujitsu GS21 include:

- PSAM (Presentation Service Access Method)
- ADL (AIM Definition Language)
- NDB (Network Data Base)

For more information about Fujitsu GS21 see these topics in the *AWS Transform for mainframe* migration guide:

- GS21
- Capture & Replay - GS21 Terminals
- Mainframe, AS400, Open VMS and GS21

Supported Regions and quotas for AWS Transform mainframe

For a list for supported Regions, see Supported Regions for AWS Transform.

> **Note**
>
> Your data might be processed in a different Region from the Region where you use AWS Transform. For information on cross-region processing, see the section called "Cross-region processing".

For the quota limits, see *Quotas*.

High-level overview of mainframe modernization journey

Modernizing mainframe applications to AWS is typically achieved in four phases: (a) assess, (b) mobilize, (c) migrate and modernize, and (d) operate, optimize, and innovate. Each of these phases are further described in detail with goals and associated activities to support your modernization journey.

Phases

- Phase 1: Assess
- Phase 2: Mobilize
- Phase 3: Migrate and modernize
- Phase 4: Operate, optimize, and innovate

Phase 1: Assess

Goal: To understand your modernization readiness and develop initial modernization plans.

With AWS Transform you can evaluate your mainframe applications, infrastructure, and operations to determine the right modernization approach for your business needs.

> **Note**
>
> Each phase contains a list of all possible activities you can do when modernizing your mainframe application. You can skip some of the activities that don't align with your use-case or goals.

Activities

- Start with an informal conversation about the modernization and work on opportunity qualification by getting the right people in the room
- Determine opportunity qualification
- Do a first-call presentation with application dependency mapping plan and application architecture
- Determine ROM (Rough order of magnitude) that includes infrastructure cost during migration, migration labor, post-production costs, project duration
- Rapid portfolio detection:
  - Execute pattern & tool guidance assessment
  - Do a source code analysis
  - Identify proof of concept (POC) candidates
- Create technical enablement training:
  - Immersion days
  - Workshops with general topics
- Create a Migration readiness assessment (MRA) report for large mainframe modernization projects or new AWS customers
- Create a statement of work for Mobilize phase

Phase 2: Mobilize

Goal: To build foundation and validate approach through successful proof of concept or pilot modernizations.

With the mobilize phase, you can establish the necessary frameworks, tools, and processes to support mainframe modernization at scale.

Activities

- Kick off the modernization plan with a pilot and the identified POC:
  - Analyze the pilot and set up the code for modernization
  - Build and test the pilot modernized data
  - Deliver results and review the pilot to create a detailed solution for larger data modernization

- For platform:
  - Create a security, compliance, and monitoring plan
  - Set up landing zone environments with AWS mainframe landing zone and account structure
  - Determine architecture infrastructure, application, integration, and more
- Think about people and processes:
  - Cloud Center of Excellence is a good resource
  - Operation model
- Create a comprehensive portfolio analysis for different application scenarios:
  - Comprehensive discovery that includes high-level documentation
  - Decomposition plan (for all applications)
  - Pattern and tool guidance for all applications
  - Come up with an initial modernization and migration plan
- Create a detailed solution considering pilot, platform, people and processes, and comprehensive portfolio analysis
- Determine the business case
- Complete successful pilot with solution for the statement of work for next phase

Phase 3: Migrate and modernize

Goal: To complete modernization and migration within planned timeline and budget.

During this phase, you can migrate and modernize your mainframe applications and data to AWS.

Activities

- Allocate resources to the production-ready application code
- Review and analyze the design using existing documentation or by interviewing subject matter experts (SMEs)
- Define test scenarios
- Set up modernization environments with appropriate runtime, databases, and third-party software
- Migrate and modernize source code and data:
  - You can modernize your application iteratively based on business domains in the application codebase

  - Scale modernization environments as needed to support your modernization workloads
- Establish CI/CD (continuous integration, continuous delivery) pipeline
- Migrate selected workloads to initiate your modernization.
- Transfer application programs and adjust environment scaling as needed:
  - You can scale the environments up or down depending on your need
  - Modernize work packages by running code sanity tests
  - Resynchronize with source code as necessary
- Run comprehensive testing and verify your migration:
  - Collect initial state data
  - Record test scenarios on mainframe
  - Stage environment, application, and data
  - Execute test scenarios on the cloud
  - Compare source and target test results
  - Build automated test scenario execution pipelines
  - Generate KPIs and reports
- Validate your data for production setup
- Validate your progress by:
  - Running integration and system testing
  - Running performance testing
  - Running user acceptance testing
  - Running HA/DR testing
- Define a roll-out plan with application owner and create a plan for training and knowledge transfer
- Consider the cutover and execute production cutovers
- Establish sustained support for post-production activities such as operations, monitoring, capability planning, and warranty for application functions

Phase 4: Operate, optimize, and innovate

Goal: To support operational excellence and continuous improvement of modernized applications.

After modernization, you can optimize your applications' performance, security, and costs while leveraging AWS services for innovation.

Activities

- Monitor application for:
  - Performance metrics
  - Security practices
  - Compliance guidelines
  - Costs
- Logging details
- Manage application operations:
  - Infrastructure management
  - Transactions and job management
  - Runtime environment support
- Application optimization and modernization
- Application development and maintenance:
  - Coding IDE
  - Build
  - Integration
  - Testing
  - Deployment

Transformation of mainframe applications

AWS Transform accelerates the transformation of your mainframe modernization applications. This topic describes the available capabilities.

Topics

- Prerequisite: Prepare project inputs in S3
- Mainframe source artifacts collection
- Sign-in and create a job
- Tracking transformation progress

- Set up a connector
- Assess and reimagine job plan
- Reimagine job plan
- Custom job plan

Prerequisite: Prepare project inputs in S3

AWS Transform is capable of handling complex mainframe codebases. To use codebase, make sure you have all the assets in your S3 location.

Key project inputs:

- **Source code:** You must upload your mainframe source code files to S3. This includes COBOL programs, JCL scripts, copybooks, and any other relevant source files.
- **Data files**: If you have any VSAM files or other data files that your mainframe applications use, these need to be uploaded to S3.
- **Configuration files**: Include configuration files specific to your mainframe environment.

Other project inputs

- System Management Facility (SMF) records: If applicable, upload SMF records to a new folder in the S3 bucket where the source code is stored, these records should be in a .zip file format.
- Formatting: For technical documentation generation, you can leverage an optional configuration file to generate PDF documents which aligns with your required formats and standards, including headers, footers, logos, and customized information.
- Glossary: AWS Transform AWS Transform leverages automation with Generative AI for documentation generation and business rule extraction. Including a glossary CSV file with information about important abbreviations and terminologies in the root directory of your zip file will help improve the generated documentation quality.
- Test data: If available, upload test data sets that can be used to validate the modernized application. This data should be stored in a new folder in the S3 bucket where the source code is stored.

Details on project inputs can be found here.

Mainframe source artifacts collection

This topic outlines the process for collecting mainframe source code and related artifacts to support mainframe modernization initiatives using AWS Transform. Proper collection and preparation of source artifacts are critical for successful analysis, migration planning, and implementation.

Source code inventory

Application source code types

The following application source code types are supported:

- COBOL programs
- JCL scripts
- Assembler programs
- PL/I programs
- CICS transactions
- Natural programs
- REXX scripts
- Easytrieve programs
- Copybooks and includes
- Proc libraries

Database assets

The following database assets are supported:

- DB2 DCLGEN and DDL scripts
- IMS database definitions (PSBs and DBDs)
- VSAM file definitions
- Physical and logical database schemas
- Database procedures and triggers

Source code file extensions

If you are not sure of the extension, leave it as blank or .txt and AWS Transform classifies it for you.

The following table lists common file extensions for mainframe source artifacts.

Mainframe source code file extensions

Language/Type	Common extensions	Description
COBOL	.cbl, .cob, .cobol	COBOL source programs
JCL	.jcl	Job Control Language scripts
Assembler	.asm	Assembler language programs
PL/I	.pl1	PL/I language programs
CICS Definition	.csd	CICS system definition (CSD)
Natural	.nat	Natural programs
REXX	.rex, .rexx	REXX scripts
Easytrieve	.ezt	Easytrieve report programs
Copybooks	.cpy	COBOL copybooks
BMS	.bms	Basic Mapping Support (BMS)
PL/I copybooks	.pl1_copy	PL/I copybooks
Db2 DCLGEN	.dcl	Db2 declarations generator
Db2 definition	.sql, .ddl	Db2 database definitions
IMS definition	.ims	IMS resource definition data set (IMS Stage 1)
MFS	.mfs	IMS MFS (Message Format Service)
PSB	.psb	IMS PSB (Program Specification Block)
DBD	.dbd	IMS Database Definitions

Language/Type	Common extensions	Description
JCL Proc	.prc, .proc	JCL Procedure libraries
JCL Includes	.inc	JCL Include files
Macros	.mac	Assembler macros
Control Cards	.ctl	JCL Utility control cards

Collection methods

Source code extraction

Use secure FTP clients (WinSCP/FileZilla) for interactive source code extraction from mainframe PDS libraries.

DB2 schema extraction

The DB2 LUW (Linux, UNIX, and Windows) client software includes the `db2look` utility as part of its installation package. The `db2look` utility extracts DDL (Data Definition Language) statements from a DB2 database.

Run the following command:

```
db2look -d DATABASE_NAME -i userid -w password -a -e -m -l -x -f -createdb -printdbcfg
 -o DATABASE_NAME.sql
```

The following list describes the parameters:

- `-d` *`DATABASE_NAME`*: Specifies the name of the database to extract schema from
- `-i` *`userid`*: User ID for database authentication
- `-w` *`password`*: Password for database authentication
- `-a`: Extracts authorization information (grants)
- `-e`: Extracts all database objects (tables, views, indexes, and so on)
- `-m`: Extracts statistics and physical characteristics
- `-l`: Extracts table and column comments
- `-x`: Extracts explain tables

- `-f`: Formats the output for better readability
- `-createdb`: Includes database creation statements
- `-printdbcfg`: Includes database configuration parameters
- `-o` *`DATABASE_NAME`*`.sql`: Specifies the output file name

Extracting catalog information from mainframe

The LISTCAT command in IDCAMS is a utility for retrieving catalog information about datasets on the mainframe. This is useful during mainframe modernization projects to inventory datasets and understand their characteristics.

```
//LSTCATJ JOB 'LISTCAT',CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
//*
//STEP1 EXEC PGM=IDCAMS
//*
//SYSPRINT DD DSN=AWS.M2.CATALOG.LIST,
//            DISP=(NEW,CATLG,DELETE),
//            SPACE=(CYL,(1,1),RLSE),
//            DCB=(RECFM=FBA,LRECL=133,BLKSIZE=0)
//*
//SYSIN DD *
  LISTCAT ENT('AWS.M2.CARDDEMO.*') ALL
  LISTCAT ENT('SYS1.*') ALL
/*
```

Extracting CICS definitions

The IBM-provided DFHCSDUP utility can extract CICS resource definitions from the CICS System Definition (CSD) file.

Use the following JCL to unload existing definitions from the DFHCSD:

```
//CSDEXTJ JOB 'EXTRACT CSD',CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
//*
//STEP1 EXEC PGM=DFHCSDUP,REGION=0M,
//          PARM='CSD(READONLY),PAGESIZE(60),NOCOMPAT'
//*
//STEPLIB  DD DSN=your-hlq.SDFHLOAD,DISP=SHR
//DFHCSD   DD DSN=your-hlq.DFHCSD,DISP=SHR
//*
//CBDOUT   DD DSN=your-hlq.CSD,
```

```
//            DISP=(NEW,CATLG,DELETE),
//            SPACE=(CYL,(1,1),RLSE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0)
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
  EXTRACT GROUP(cics-group) USERPROGRAM(DFH0CBDC) OBJECTS
/*
```

Extracting IMS definitions

To extract IMS definitions, you can generate the IMS Transaction file by using the DFSURDD0 utility. This utility creates stage 1 macro statements that you can use for migration purposes. Only IMS Stage 1 is currently supported.

Use the following JCL to extract IMS definitions:

```
//IMSEXTJ JOB 'IMS STAGE 1',CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
//*
//STEP1 EXEC PGM=DFSURDD0,MEMLIMIT=12G
//*
//STEPLIB  DD DISP=SHR,DSN=your-hlq.SDFSRESL
//*
//RDDSDSN DD DISP=SHR,DSN=your-rdds
//*
//SYSOUT   DD DSN=your-hlq.STAGE1.IMS,
//            DISP=(NEW,CATLG,DELETE),
//            SPACE=(CYL,(1,1),RLSE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0)
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
  OUTPUT=MAC
/*
```

Extracting CA-7 scheduler information

CA-7 job schedule information is comprised in LJOB reports. Use the LJOB utility with the `LIST=NODD` parameter to generate the reports on the mainframe. The reports must have a `.ca7` extension. Each report must have ANSI carriage control characters in column 1. You can create a report with carriage control characters in column 1 by specifying `DCB=RECFM=FBA` (or FA) in the JCL used to run the LJOB utility.

For example: `LJOB,JOB=ZBN*,LIST=ALL`

For more information, see the Broadcom documentation.

Extracting SMF data

System Management Facility (SMF) records provide activity metrics used by AWS Transform to analyze batch job and CICS transaction usage. The following SMF record types are supported: 14, 15, 30 (sub type 5), 64, 102, 110 (sub type 1 with data class 1, 3, and 4).

Format requirements for SMF records:

- Must include record types 30 and 110 (at minimum)
- Must be in raw binary EBCDIC format with RDW (Record Descriptor Word) bytes included
- Must be provided as one of the following:
  - A file up to 600 MB compressed in .zip format
  - A file up to 5 GB compressed in .tar.gz or .gz format
- Must be stored in a separate folder from your source code in Amazon S3

Provide a minimum of 13 months of SMF records to capture annual batch cycles and seasonal patterns that shorter timeframes might miss.

Option 1: Extract and sort SMF records

Use the following sample JCL to extract and sort SMF records:

```
//SMFEXTR  JOB (ACCT),'EXTRACT SMF',CLASS=A,
//             MSGCLASS=X,NOTIFY=&SYSUID
//*-------------------------------------------------------------------*
//* SMF RECORD EXTRACTION TEMPLATE
//* CUSTOMIZE: Job card, dataset names, date range, record types
//*-------------------------------------------------------------------*
//*
//* STEP 1: EXTRACT SMF RECORDS
//*
//EXTRACT  EXEC PGM=IFASMFDP
//SYSPRINT DD SYSOUT=*
//*
//* INPUT: Specify your SMF datasets (MAN files or GDG)
//*
//DUMPIN   DD DISP=SHR,DSN=SYS1.MAN1
//         DD DISP=SHR,DSN=SYS1.MAN2
```

```
//         DD DISP=SHR,DSN=SYS1.MAN3
//*
//* OUTPUT: Extracted SMF records
//*
//DUMPOUT  DD DSN=your.hlq.SMF.EXTRACT,
//            DISP=(NEW,CATLG,DELETE),
//            UNIT=SYSDA,
//            SPACE=(CYL,(500,100),RLSE),
//            DCB=(RECFM=VBS,LRECL=32760,BLKSIZE=27998)
//*
//* CONTROL CARDS
//*
//SYSIN    DD *
  INDD(DUMPIN,OPTIONS(DUMP))
  OUTDD(DUMPOUT,TYPE(14,15,30,110))
  DATE(yyyy/ddd,yyyy/ddd)
  START(0000)
  END(2359)
/*
//*
//* STEP 2: SORT EXTRACTED RECORDS (OPTIONAL)
//*
//SORT     EXEC PGM=SORT
//SORTIN   DD DISP=SHR,DSN=your.hlq.SMF.EXTRACT
//SORTOUT  DD DSN=your.hlq.SMF.SORTED,
//            DISP=(NEW,CATLG,DELETE),
//            UNIT=SYSDA,
//            SPACE=(CYL,(500,100),RLSE),
//            DCB=(RECFM=VBS,LRECL=32760)
//SORTWK01 DD UNIT=SYSDA,SPACE=(CYL,(50))
//SORTWK02 DD UNIT=SYSDA,SPACE=(CYL,(50))
//SORTWK03 DD UNIT=SYSDA,SPACE=(CYL,(50))
//SYSOUT   DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  SORT FIELDS=(11,4,CH,A,7,4,CH,A,15,4,CH,A),EQUALS
/*
```

Customize the following values in the preceding JCL:

1. Job card: Update account, class, and msgclass
2. DUMPIN: Point to your SMF datasets
3. DUMPOUT: Specify output dataset name and HLQ

4. TYPE: Select record types (14, 15, 30, 110 or others)
5. DATE: Format yyyy/ddd (for example, 2026/055,2026/056)
6. SPACE: Adjust based on expected data volume
7. Remove SORT step if not needed

Option 2: Extract SMF data with DFSORT

You can use DFSORT to consolidate and report historical SMF data. This is helpful if you store your SMF records in a GDG by day or week. The following job uses DFSORT to consolidate SMF records from the SORTIN file and filter based on the records specified in the INCLUDE COND parameter.

```
//EXTRSMF JOB 'EXTRACT SMF RECORDS',CLASS=S,MSGCLASS=H,
// MSGLEVEL=(1,1),REGION=0M,NOTIFY=&SYSUID,TIME=1440
//SORTJES2 EXEC PGM=SORT,REGION=0M
//SYSOUT DD SYSOUT=*
//SORTMSG DD SYSOUT=*
//SORTIN DD DISP=SHR,DSN=your.hlq.SMF.EXTRACT
//SORTOUT DD DSN=your.hlq.SMF.OUTPUT,
// DISP=(MOD,CATLG,DELETE),
// DCB=(BUFNO=20,
// RECFM=VBS,LRECL=32760,BLKSIZE=27998),
// SPACE=(CYL,(1400,900),RLSE)
//SORTWK1 DD UNIT=SYSDA,SPACE=(CYL,(30,50))
//SORTWK2 DD UNIT=SYSDA,SPACE=(CYL,(30,50))
//SORTWK3 DD UNIT=SYSDA,SPACE=(CYL,(30,50))
//SORTWK4 DD UNIT=SYSDA,SPACE=(CYL,(30,50))
//SYSIN DD *
OPTION COPY
INCLUDE COND=(6,1,BI,EQ,X'0E',OR,6,1,BI,EQ,X'0F',OR,
6,1,BI,EQ,X'1E',OR,6,1,BI,EQ,X'40',OR,
6,1,BI,EQ,X'50',OR,6,1,BI,EQ,X'59',OR,
6,1,BI,EQ,X'5C',OR,6,1,BI,EQ,X'65',OR,
6,1,BI,EQ,X'66',OR,6,1,BI,EQ,X'6E')
```

To download the data while keeping the RDW data, you can use an FTP server to receive the extracted file. Make sure the data is transferred in binary format and the RDW option is used so the RDW is included in the data transfer. The following job shows a sample FTP job to transfer this file.

```
//EXTRSMF JOB 'EXTRACT SMF RECORDS',CLASS=A,MSGCLASS=H,
// MSGLEVEL=(1,1),REGION=0M,NOTIFY=&SYSUID,TIME=1440
```

```
//STEP1 EXEC PGM=FTP,REGION=2048K
//SYSIN DD *
10.10.10.10
userid
password
type i
LOCSITE RDW
put 'your.hlq.SMF.OUTPUT' SMFREQS.SMF
quit
```

Uploading SCRT report

You can also upload a Sub-Capacity Reporting Tool (SCRT) report into AWS Transform.

Application inventory upload

After you collect all artifacts, complete the following steps:

1. Organize files into subfolders by artifact type (for example, /cobol, /jcl, /copybooks, /db2, /ims, /cics, /ca7).
2. Compress the top-level folder into a single .zip file.
3. Upload the .zip file to the Amazon S3 bucket connected to your AWS Transform workspace.

> **Note**
>
> SMF records must be placed in a separate folder from your source code within the Amazon S3 bucket, and must be in .tar.gz or .gz format.

For detailed instructions on connecting your Amazon S3 bucket to AWS Transform, see the section called “Set up a connector”.

Sign-in and create a job

To sign into the AWS Transform web experience, follow all the instructions in the Getting started with AWS Transform section of the documentation.

To create and start a job: Follow the steps in Start your project.

Create workspace: Name and describe your workspace where jobs, collaborators, and associated artifacts will be stored.

Create job: Create a job by selecting from preconfigured job plans, or customize the job plan based on your objective by selecting from the list of supported capabilities.

> **Important**
>
> AWS Transform will refuse operations from you if you don't have the proper permissions. For example, a contributor cannot cancel a job transformation of mainframe applications or delete a job. Only an administrator can perform these functions.

When you create your job you can select from the capabilities below, but the Kickoff step is always required as it is where the location of the source code for the project is located. In the first job set up in a workspace you are required to set up a connector to your Amazon S3 bucket.

Tracking transformation progress

You can track the progress of the transformation throughout the process in two ways:

- **Worklog** – This provides a detailed log of the actions AWS Transform takes, along with human input requests, and your responses to those requests.
- **Dashboard** – This provides high-level summary of the mainframe application transformation. It shows metrics on number of jobs transformed, transformation applied, and estimated time to complete the transformation of mainframe applications. You can also see details of each step including, lines of code by file types, generated documentation by each file type, the decomposed code, and the migration plan.

Set up a connector

AWS Transform uses a connector to access resources in your account required for mainframe modernization functions. Your connector is automatically configured with the first job you run in the workspace. Depending on the job plan you choose, AWS Transform guides you to create your connector.

Mainframe reimagine connector

The mainframe reimagine connector is for jobs running the assess and reimagine workflows. It uses an S3 bucket to access and store transformation resources, as well as an Amazon Neptune cluster to store extracted artifacts. The Neptune cluster serves as the unified knowledge graph that stores all extracted artifacts about your job, and the knowledge graph answers all questions in the AWS Transform chat interface.

Deploy the Amazon Neptune cluster in a VPC where AWS Transform can create elastic network interfaces (ENIs). These are the network connections that allow AWS Transform to securely communicate with Neptune in your VPC to load data and query your knowledge graph.

AWS CloudFormation

We recommend using the following CloudFormation template to create the resources required for your mainframe reimagine connector. Use the same S3 bucket that you plan to use for the connector when running the CloudFormation template. For more information about using CloudFormation templates, see Working with templates.

To download the CloudFormation template, choose neptune-kg-setup.yaml.

The template accepts the following parameters:

- **BucketName** – Required. S3 bucket name for Neptune bulk loader access. You must use the same S3 bucket that you plan to use in your AWS Transform connector.
- **KmsKeyId** – Optional. KMS key for Neptune encryption. Accepts key ID, ARN, alias name, or alias ARN. Defaults to the AWS managed key if omitted.

Provide the following details when you configure your connector:

- S3 bucket ARN
- Neptune cluster ARN
- Neptune cluster resource ID
- Application subnet IDs
- Application security group ID
- Neptune S3 bulk loader role ARN

> **Important**
>
> The Outputs tab of the CloudFormation template includes the details you need to provide to AWS Transform when configuring the connector.

Custom configuration

If you prefer to use an existing VPC or other tools to create the required infrastructure, your environment must meet the following requirements:

Neptune cluster

- Amazon Neptune Serverless with engine version 1.4.5.1 or later
  - IAM authentication enabled
  - Storage encryption enabled (AWS managed key or customer-managed KMS key)
- Serverless scaling configuration with a recommended range of 1–32 Neptune Capacity Units (NCUs)
- (Recommended) Read replica in a second Availability Zone for high availability and improved read performance

Networking

- A VPC with DNS support and DNS hostnames enabled
- At least two subnets in separate Availability Zones. AWS Transform creates ENIs in these subnets to access Neptune. The Neptune cluster can reside in the same subnets or in dedicated subnets. Size subnets at /20 to accommodate VPC endpoint and application ENIs.
- An S3 gateway endpoint attached to the route tables associated with your subnets. AWS Transform uses this endpoint to load data from the S3 bucket to the Neptune cluster.

Security groups

Security groups are attached to ENIs, not subnets. Even when AWS Transform ENIs and Neptune reside in the same subnet, separate security groups control access between them.

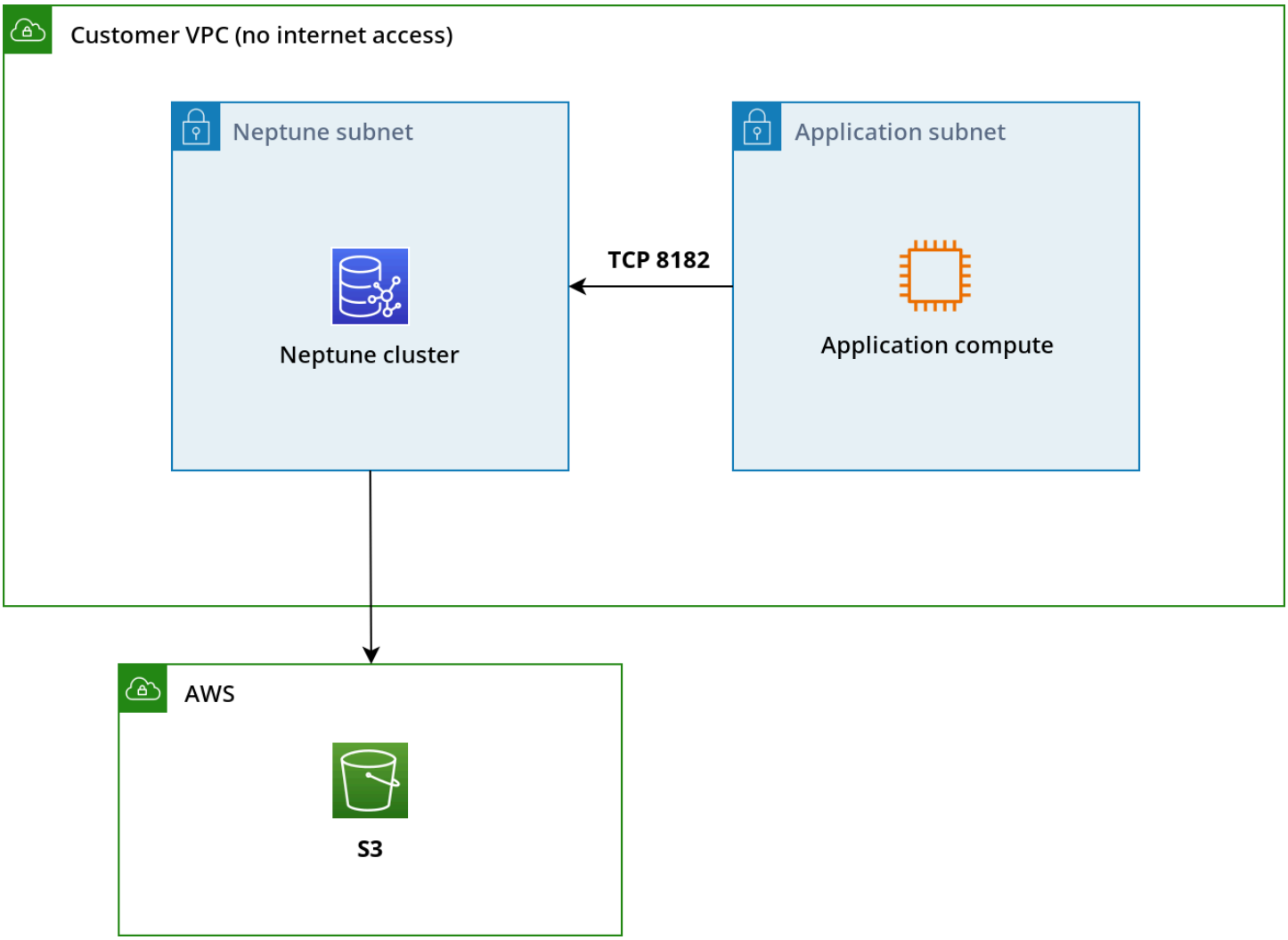
- A security group (attached to ENIs created by AWS Transform) that allows:
  - Outbound TCP 8182 to the Neptune security group

- A security group (attached to the Neptune cluster) that allows:
  - Inbound TCP 8182 from the application security group

IAM

- An IAM role with a trust policy for `rds.amazonaws.com` that grants `s3:GetObject`, `s3:ListBucket`, and `s3:GetBucketLocation` on your S3 bucket
- This role must be associated with the Neptune cluster for bulk loading knowledge graph data

The following diagram shows the recommended architecture.



S3 connector

For jobs with a custom plan, AWS Transform can use an S3 connector. The S3 connector uses an S3 bucket to access and store transformation resources, and an S3 vector bucket for indexed output.

> **Important**
>
> Your data is stored and persisted in the AWS Transform artifact store in your workspace and is used only for running the job.

Optional configuration of S3 vector bucket.

In Regions where S3 vector buckets are available, AWS Transform will store searchable vector encodings of job output in this S3 vector bucket in your account to provide an AI powered search and chat experience. Data is not used outside of this job, and is not used to train models. To enable this, you must create and provide S3 vector bucket. AWS Transform automatically creates and attaches a role with required permissions to write to this bucket.

S3 bucket CORS permissions

After your connector is configured, add the following CORS policy to your S3 bucket so you can view and compare artifacts directly in the AWS Transform console. If this policy is not set up correctly, you might not be able to use the inline viewing or file comparison functionalities of AWS Transform.

```
[
    {
        "AllowedHeaders": [],
        "AllowedMethods": [
            "GET"
        ],
        "AllowedOrigins": [
              "https://*.transform.eu-central-1.on.aws",
              "https://*.transform.ap-south-1.on.aws",
              "https://*.transform.ap-northeast-1.on.aws",
              "https://*.transform.ap-northeast-2.on.aws",
              "https://*.transform.ap-southeast-2.on.aws",
              "https://*.transform.ca-central-1.on.aws",
              "https://*.transform.eu-west-2.on.aws",
```

```
            "https://*.transform.us-east-1.on.aws",
            "https://*.transform.sa-east-1.on.aws"
        ],
        "ExposeHeaders": [],
        "MaxAgeSeconds": 0
    }
]
```

Assess and reimagine job plan

The Assess and reimagine job plan is a predefined workflow for modernizing mainframe applications. It guides you through two phases: assessing your codebase to identify business functions, and reimagining the functions you select. To choose individual capabilities instead, use the Custom job plan.

Assess

Mainframe modernization journeys typically start with evaluating the codebase to identify contents and understand relationships and dependencies. After this evaluation, you can start to decompose and select portions of the source code to modernize. The boundaries of decomposition are determined based on business functions.

The business function catalog decomposes your codebase into discrete business functions, which are built on a foundation of deterministic data paths that map each business function from start to finish. A data path is the route that data takes, starting with a business trigger through to the final write within the mainframe code. Rather than limiting analysis to a single entry point, the system spans batch jobs, CICS transactions, and their shared data stores to identify coherent units of business work.

A business function is a body of work that starts from a business trigger and ends at a measurable business outcome. AWS Transform provides a catalog of business functions that a line of business can act on, enabling informed decisions about what to modernize first.

Benefits of assessing your codebase to modernize using business functions:

- **Full-estate visibility** – Understand how your batch jobs, online transactions, and data stores connect into meaningful functions for your business.
- **Actionable boundaries** – Each identified business function is a self-contained unit that your line-of-business stakeholders can review and prioritize for modernization effort.

- **Consistency from code to specification** – The same code blocks that define a business function are used to generate its specifications, with no translation layer and no reconciliation required.
- **Reduced discovery time** – Automated detection replaces months of manual tribal-knowledge interviews with systematic, repeatable analysis.

Results of assessment provided through the chat

The results of assessment provided through the chat include a list of business functions, including the number of data paths, what each function spans in terms of batch and CICS transactions, and a business description. You are provided with a link to the artifacts generated in the Amazon S3 bucket, and you can also access them through the chat. Two artifacts are provided: business function details and business function summary.

- **Business function summary** – Provides a list of the business functions and a natural-language description of what functions the elements within the code are performing.
- **Business function details** – Provides an interactive graph that depicts the relationships across the source code.

To interact with the graph, select an element to show additional details about it, or open an element to interrogate its details. Within each page, there is a summary of all elements at that level and a graphical interface to interact with and learn more about each component, including the following:

- **Business function** – Depicts which business functions are connected to each other and provides an overview of each business function. When you select a business function, the following information is provided:
  - Number of data paths
  - Number of lines of code
  - Business description: an overview of the business function that describes, in natural language, what functions the elements within the code are performing.
  - Interfaces: details how data elements within the business function relate (for example, exchange or write).
- **Business function details** – Depicts the connections across the data paths within a single business function, allowing you to dive deeper into the composition of the business function. When you select an element, you see how that element connects across the business function, along with the following details:

  - Datastore: writers and readers for the datastore.
  - Other elements: an overview of the related data path that describes, in natural language, the actions and outcomes handled within the data path, a list of programs, and readers and writers.
- **Data paths** – Depicts the details of one data path and how the elements within the data path relate to each other.
  - Description: an overview of the data path that describes, in natural language, the actions and outcomes handled within the data path.
  - Contents of the data path:
    - Entry point
    - Reads
    - Writes
    - Programs

Through the chat, you can interrogate the outputs to understand the elements contained within each business function, and explore the boundaries through the business function graph.

After you identify the boundaries of your modernization, the chat prompts you to select the business functions that you want to reimagine. You can select a single business function or multiple business functions. AWS Transform sends the selected boundaries and the required assessment outputs to the reimagine flow to continue the modernization journey.

After you complete your first modernization, you can use the chat to select an additional set of business functions to modernize.

Reimagine

In the reimagine phase, you can select one or all of the business functions that assessment identified and prepare them for modernization. This triggers business logic extraction followed by requirement generation for the selected business functions. The generated requirements are the primary input for forward engineering, translating legacy system understanding into a precise specification of what the modernized application must deliver.

Extract business logic

After you select one or more business functions, AWS Transform begins generating business logic for the selected business functions. After extraction is complete, AWS Transform stores the results

in an Amazon S3 bucket in JSON format for downstream use. You can monitor the progress of the extraction and review any issues directly within the AWS Transform console.

To review business logic extraction results

1. In the left navigation pane, choose the **Extract business logic** step to expand it.
2. View the following from the step details:

   - **S3 bucket link**: The location where the extracted business logic results are stored in JSON format.
   - **Issues**: A list of any issues encountered during the business logic extraction process.

If any files encountered issues during extraction, the following details are displayed for each affected file:

- **File name**: The name of the file that encountered an issue.
- **File type**: The type of the file (for example, COBOL or JCL).
- **File path**: The location of the file within your application.
- **Status**: The current status of the file's extraction.
- **Details**: A description of the issue encountered.

> **Note**
>
> After the extraction step completes, AWS Transform automatically begins generating requirements. You are not prompted for any additional inputs before this process starts.

Generate requirements

After business logic extraction completes, the Generate Requirements agent consumes extracted artifacts (code analysis, data analysis, business rules) and produces modernization requirements. These requirements are formal functional specifications that are technology-agnostic and include testable acceptance criteria. AWS Transform produces one `requirements.md` file for each business function you select and stores the output in an Amazon S3 bucket. To view the Amazon S3 location of the output, choose the **Generate requirements** step.

Each `requirements.md` file is organized into the following parts:

- **Title** – Names the business function and the workflow it represents.
- **Global preconditions** – Conditions that must hold true across the entire workflow before and during processing. Configurable values appear as parameter placeholders.
- **Numbered workflow sections** – Each section represents a discrete stage of the business function. Every section contains the following:
  - A **user story** that states the goal as a role, the action that role wants to take, and the resulting outcome.
  - A list of **Requirements** in EARS (Easy Approach to Requirements Syntax) format, each with a unique identifier.

> **Note**
>
> Selecting a second set of business functions restarts the reimagine step for the newly selected functions. The console then shows results only for the new selection, so any issues with the business rules from your previous business functions, and the Amazon S3 links that store the outputs for the business logic and requirements, are no longer displayed. Your earlier outputs are not lost. You can still access them from the **Artifacts** tab or in your Amazon S3 bucket.

Traceability between transformation artifacts

AWS Transform maintains traceability between the artifacts it generates throughout the assess and reimagine workflow. Traceability lets you trace each requirement to your extracted business rules, or directly to the original source code that produced the requirement.

You can view all the traceability details for each business function using the `traceability.yaml` file. It describes how each business rule maps to a requirement.

For interactive traceability, you can use the IDE plugin in VS Code and Open VSX compatible editors. For more information, see Developer tools.

Reimagine job plan

For applications that you have scoped for modernization and are ready to reimagine, you can run a standalone reimagine job. The job analyzes your code and data, extracts business logic, identifies business domains, and generates modernization requirements for each business domain.

Custom job plan

A custom job plan lets you build your own modernization workflow by selecting the capabilities you want to run. It gives you full control over which capabilities are included. Choose from the following capabilities based on your modernization objective. Some capabilities depend on others being run first. For example, code analysis is required for most other capabilities.

Analyze code

After you share the Amazon S3 bucket path with AWS Transform, it will analyze the code for each file with details such as file name, file type, lines of code, and their paths.

> **Note**
>
> You can download the Analyze code results through the artifacts tab at the job or workspace level. At the job level go to the 'Artifacts' option in the left navigation menu, and open the 'results' folder, or at the workspace level find the name of the job, and open the 'results' folder. This will download a zip file that contains the classification file for manual classification workflow, assets list, dependencies JSON file, and list of missing files.

In the job plan select Analyze code in the left navigation pane to view your results. You can view your code analysis results in several ways:

- **List view** – All files in the Amazon S3 bucket you want to transform for mainframe
- **File type view** – All files in the Amazon S3 bucket displayed per file type. For a list of supported file types, see Supported files.
- **Folder view** – All files in the Amazon S3 bucket displayed in folder structure.

Within the file results, AWS Transform provides the following information depending on what file view you choose:

- Name
- File type
- Total lines of code
- File path

- Comment lines
- Empty lines
- Effective lines of code
- Number of files
- Cyclomatic Complexity - Cyclomatic complexity represents the number of linearly independent paths through a program's source code. AWS Transform will show a cyclomatic complexity for each of the files.

Missing files– Missing files from the mainframe modernization code analysis. These files ideally, should be added as a part of the source input in Amazon S3 bucket, and the analysis step should be re-run for better and cohesive results.

Identically named – AWS Transform gives you a list of files that share the same name, and possibly the same characteristics (e.g., number of lines of code). It will not have the ability to compare the difference between the contents of any two files at one time.

Duplicated IDs – With Cobol program, the **Program ID** field serves as the unique identifier of the file. This ID must be unique because it's used to call the program throughout your project. However, some projects might have COBOL files with different names but the same Program ID. Getting the list of those files during the assessment can help understand the dependencies among all programs.

> **Note**
>
> This is specific to COBOL code and files.

When you have programs with duplicated IDs, it's suggested to change the Program IDs of these files to have a unique identifier for each of these in the COBOL code. You can then re-run your job to get more accurate and comprehensive code analysis results.

By resolving duplicate Program IDs, you can:

- Improve code clarity and maintainability
- Reduce potential conflicts in program calls
- Enhance the accuracy of dependency mapping

- Simplify future modernization efforts

Codebase issues – Potential issues detected within the codebase that you should resolved before continuing with the modernization project. These issues could include missing references with associated statements or unsupported links in the code.

Update classification – With manual reclassification, you can reclassify files using the bulk update feature by uploading the JSON file with the new classification.

> **Important**
>
> This is only available for `UNKNOWN` and `TXT` files.

After reclassification, AWS Transform will:

1. Updates the classification results
2. Re-runs dependency analysis with the new file types
3. Refreshes all affected analysis results

> **Note**
>
> You can reclassify files only after the initial analysis loop completes.

Inline viewer and file comparison

The Inline viewer is a feature in the AWS Transform for mainframe capabilities that provides two key visualization capabilities:

- **File view**: View content of selected legacy files from jobs
- **File comparison**: Compare content of two legacy files side-by-side

Input file viewing

To view your files in the Analyze code step

- Under **View code analysis results**, select a file using the check box in the list.

Choose the **View** action button (enabled when 1 item is selected).

File content will be rendered on screen in the **File View** component.

File comparison

To compare files in the Analyze code step

1. Under **View code analysis results**, select two files using the check boxes in the list.
2. Choose the **Compare** action button (enabled only when 2 items are selected).
3. Files will be displayed side-by-side in the **File comparison** component.

> **Note**
>
> You can't select more than two files to compare files.

> **Important**
>
> If you're having issues with inline viewer or file comparison make sure that the S3 bucket is set up correctly. For more information on S3 bucket's CORS policy, see the section called "S3 bucket CORS permissions".

Data analysis

AWS Transform provides data analysis to help understand the impact of data relationships and elements to the mainframe modernization project. The two outputs provided are:

- Data lineage: Traces the lifecycle of data by mapping relationships between data sources, jobs, and programs
- Data dictionary: Serves as a repository documenting the structural metadata of legacy data elements

On completion of the analysis, there will be two tabs present for each output, and then multiple views available described below for both data lineage and dictionary.

> **Note**
>
> When you request data anlysis. AWS Transform performs code analysis, which is required in order to perform data analysis.

Data lineage

AWS Transform provides multiple views based on the data relationships that need to be understood across the code base being modernized. The four table views available in data lineage include:

- **Data sets**: This view provides a comprehensive impact analysis, including operation tracking to help distinguish between read, write, update, and delete
- **Db2 tables**: Provide the impact analysis related to Db2 tables and the operations
- **Program-to-data**: Identify which COBOL programs reference each dataset
- **JCL-to-data relationships**: Identify which JCL scripts reference each dataset

Summary provides an overview of data sources, and their relationships to programs and JCLs, as well as summary information about how the data sources are leveraged and total operations found in the codebase.

Data dictionary

Understanding the data elements within the data sources present in the codebase is the next step to realize how the various data sources are dependent. Data dictionary is a data catalog providing field level metadata with business language descriptions for accurate transformation mapping.

- **COBOL data structure**: Provides field information across COBOL copybooks and INLINE references present in the codebase, including field properties and business definition
- **Db2 tables**: Provides column and table properties across the Db2 tables present in the codebase, including primary and foreign keys, schema and index information, and data types

Relationship between data lineage and dictionary is available by selecting the data source, and then utilizing the data lineage or data dictionary button in order to dive deeper on the relationship between the data source and the data elements. The navigation between data lineage and

dictionary, provides integrated data visibility with data lineage providing the "what" - structure and meaning, alongside the "where" - usage and relationship.

Activity metrics analysis

Activity metrics analysis lets you analyze System Management Facility (SMF) records of type 14, 15, 30, 64, 102, and 110. The analysis provides insights into how elements are used within your mainframe application. It can help with retiring unused code or with decisions about the target architecture for the modernized application. If you include activity metrics analysis in your job, completing the code analysis step first provides richer SMF outputs, although it is not required.

> **Note**
>
> When you negotiate the job plan for mainframe modernization, completing the code analysis and decomposition steps first provides richer outputs.

Onboard SMF records for analysis

You must provide the SMF record location within your Amazon S3 bucket as the initial step for SMF analysis. Provide a minimum of 13 months of records to capture annual occurrences that shorter time frames might miss. Although 13 months is recommended, any time frame with detectable SMF records produces analysis results.

Your SMF extract must meet these format requirements:

- Include types 14, 15, 30 (sub-type 5), 64, 102, and 110.
- Use a raw binary file in EBCDIC format.
- Include RDW bytes.

Provide a link to the SMF records within your Amazon S3 bucket, ensuring the records are in a folder separate from your source code. Format options include .zip (up to 600 MB) or .tar.gz (up to 5 GB) compressed. If you provide a link to a folder where SMF records are not detected, you receive an error message and no analysis is completed.

Analysis output

The analysis output contains two components, depending on the type of record: a tabular view and a .csv output (available in your Amazon S3 bucket). Within AWS Transform, outputs are displayed

for type 30 and 110. Through the artifacts in your Amazon S3 bucket, you can explore the analysis on other types of records. In the user interface, the header displays the time range of your provided records so that you can identify any missing key dates. For example, if your records span May 1 to October 31 but exclude the busy day after Thanksgiving, you can see that key records are missing. Timestamps in the web application reflect your system's time zone and the SMF records.

Artifacts available in Amazon S3 provide analysis for any record type found in the records provided.

Tabular view

The tabular view, available only for SMF 30 and 110, contains up to three main components for batch jobs and CICS transactions:

- **Summary** – Provides key jobs and transactions.
- **Batch job/CICS transactions analysis** – Provides aggregated analysis across jobs and transactions.
- **Code analysis comparison (batch only)** – Provides comparison data when the code analysis step runs prior to SMF analysis.

The discovery summary provides three groups of jobs and transactions that help you quickly identify items for deeper analysis.

Batch job and CICS transaction analysis provides aggregated analysis across jobs and transactions. Some columns in the analysis results are hidden by default. Choose the gear icon to display additional fields.

The transaction key generates unique data aggregation for CICS transactions, combining four fields into a single key value:

- Transaction ID
- Program name
- SysPlex ID
- SysID

You can search by the complete key or by any component of the key in the analysis output.

Code analysis comparison highlights jobs found in either the SMF records or the analyze code step, but not present in both. This shows jobs that have not run during your SMF record time frame or

were not present in the analyze code step. This output is available only when you run the code analysis step before SMF analysis in your job.

Best practices

To get comprehensive outputs for batch jobs (type 30), ensure that your JCL file name matches the job name to get meaningful outputs in the code analysis comparison.

Generate technical documentation

You can generate technical documentation for your mainframe applications undergoing modernization. By analyzing your code, AWS Transform can automatically create detailed documentation of your application programs, including descriptions of the program logic, flows, integrations, and dependencies present in your legacy systems. This documentation capability helps bridge the knowledge gap, enabling you to make informed decisions as you transition your applications to modern cloud architectures.

> **Note**
>
> When you request generation of technical documentation, AWS Transform performs code analysis, including code dependency analysis, which are required in order to generate the documentation.

To generate technical documentation

1. In the left navigation pane, under **Generate technical documentation**, choose **Select files and configure settings**.
2. Select the files in the Amazon S3 bucket that you want to generate documentation for, and configure the settings in the **Collaboration** tab.

   > **Note**
   >
   > Selected files should have the same encoding type (that is, all in the same CCSID - UTF8 or ASCII). Otherwise, generated technical documentation might have empty fields or sections.

3. Choose the documentation detail level:

- **Summary** – Provides a high-level overview of each file in the scope. Also, gives a one-line summary of each file.
- **Detailed functional specification** – Provides comprehensive details for each file in the mainframe application transformation scope. Some details include logic and flow, dependencies, input and output processing, and various transaction details.

> **Note**
>
> Documentation can be generated only for COBOL and JCL files.

4. Choose **Continue**.
5. Once AWS Transform generates documentation, review the documentation results by following the Amazon S3 bucket path in the console, where the results are generated and stored.
6. Once the documentation is generated, you can also use AWS Transform chat to ask questions about the generated documentation and decide the next steps.

Add user information into the documentation

```
ARTIFACT_ID.zip
### app/
    ### File1.CBL
    ### File2.JCL
    ### subFolder/
    #   # File3.CBL
### glossary.csv
### pdf_config.json
### header-logo.png
### footer-logo.png
# ...
```

Optional files can be added in the zip file to help improve the generated documentation quality and provide customized PDF cover page. Some of these can be:

- **glossary.csv file**: You can choose to provide and upload an optional glossary in the zip file in the S3 bucket. The glossary is in CSV format. This glossary helps create documentation with relevant descriptions in line with the customer vocabulary. A sample `glossary.csv` file looks like:

```
LOL,Laugh out loud
ASAP,As soon as possible
WIP,Work in progress
SWOT,"Strengths, Weaknesses, Opportunities and Threats"
```

- **pdf_config.json**: You can leverage this optional configuration file to generate PDF documents which align with their company's formats and standards, including headers, footers, logos, and customized information. A sample `pdf_config.json` looks like:

```
{
    "header": {
    "text": "Acme Corporation Documentation",
    "logo": "header-logo.png"
  },
  "customSection": {
    "variables": [
      {
        "key": "business Unit",
        "value": "XYZ"
      },
      {
        "key": "application Name",
        "value": "ABC"
      },
      {
        "key": "xxxxxxxxxx",
        "value": "yyyyyyyyyyy"
      },
      {
        "key": "urls",
        "value": [
          {
            "text": "Product Intranet Site",
            "url": "https://example.com/intranet"
          },
          {
            "text": "Compliance Policies",
            "url": "https://example.com/policies"
          }
        ]
      }
    ]
```

```
  },
  "footer": {
    "text": "This document is intended for internal use only. Do not distribute
 without permission.",
    "logo": "footer-logo.png",
    "pageNumber": true
  }
}
```

- **Header:**
  - For the cover page PDF file, the default text will be the project name.
  - For each program PDF file, the default text will be the program name.
  - There is no default logo. If a header logo is not configured, no logo will be displayed.
  - The font size and logo size shall be dynamically changed based on the number of words or logo file size.
- **Custom section:**
  - If the custom section is not configured, it will be omitted from the PDF.
  - The link has to be click able.
- **Footer:**
  - There is no default text or logo for the footer.
  - The page number will be displayed in the footer by default, unless explicitly configured otherwise.
  - The font size and logo size shall be dynamically changed based on the number of words or logo file size.

Generate documentation inline viewer

You can view the PDF files in the generate technical documentation step.

To view the PDF files

1. Navigate to the **Review documentation results** tab.
2. Locate the PDF in the table listing generated PDFs.
3. Either select the rile and then select **View** or select the link element overlaid on the file name.

The PDF opens in AWS Transform, with the option to expand the screen in the top right.

Note

AWS Transform also gives you the ability to download either an XML of PDF version of the generated technical documentation.

Important

If you're having issues with documentation inline viewer, make sure that the S3 bucket is set up correctly. For more information on S3 bucket's CORS policy, see the section called "S3 bucket CORS permissions".

Extract business logic

You can extract essential business logic from your mainframe applications that are undergoing modernization. AWS Transform automatically analyzes your code to identify and document critical business elements, including detailed process flows, and business logic that is embedded in your applications. This capability serves multiple stakeholders in your modernization journey. Business analysts can leverage extracted logic to create precise business requirements and identify gaps or inconsistencies in current implementations. Developers gain the ability to quickly comprehend complex legacy system functionality without extensive mainframe expertise.

Note

When you request business logic extraction. AWS Transform performs code analysis, including code dependency and entry point analysis, which are required in order to perform business logic extraction.

To extract business logic

1. In the left navigation pane, under **Extract business logic**, choose **Configure settings**.
2. In the **Collaboration tab** select how you want to extract business logic:

   - **Application level**: Generates business documents for all business functions, transactions, batch jobs, and files. This selects all of the files in the application.
   - **File level**: Generates business documents only for files you select from the file table.

Note

- For either option you can select **Include detailed functional specification** so that AWS Transform includes control flow and comprehensive business rules for the selected files.
- Selected files should have the same encoding type (that is, all in the same CCSID - UTF8 or ASCII). Otherwise, generated documentation might have empty fields or sections.
- Documentation can be generated only for COBOL and JCL files.
- For application level, programs used by CICS transactions and batch jobs are grouped together, while all other programs are categorized as *Unassigned*.

3. Choose **Continue**.
4. Once AWS Transform extracts business logic it stores the results in an Amazon S3 bucket in JSON format so that you can view them online.

Note

The number of generated business rule files might be larger than your initial selection. Some selected files may trigger business rule extraction to include additional dependent files, which will also appear in the results table.

View the extracted business documentation inline

You can view the business logic in the Extract business rule step. To do this,

1. Navigate to **Review business logic extraction results**.
2. Select the document you want to view from the table, and then click the **View result** button.

The business documentation page opens in a new browser tab.

Decomposition

You can decompose your code into domains that account for dependencies between programs and components. This helps the related files and programs to be grouped appropriately within the same domain. It also helps maintain the integrity of the application logic during the decomposition process.

> **Note**
>
> When you request decomposition, AWS Transform performs code analysis, including code dependency analysis, which are required in order to perform decomposition. We also recommend that you perform business logic extraction before decomposition for better results.

To get started with decomposing your application:

1. Select **Decompose code** from the left navigation pane.

   > **Note**
   >
   > One domain, Unassigned, is automatically created for all files not associated with a domain. On initial navigation, all files should be associated to Unassigned, unless domains were proposed from application level business logic extraction.

2. Create a new domain through the Actions menu, and then choose Create domain.
3. In Create domain, provide domain name, optional description, and mark some files as seeds.
   - CICS configured files (CSD) and scheduler configured files (SCL) can be used for automatic seed detection.
   - You can also set one domain only as a common component. The files in this domain are common to multiple domains.
4. Choose Create.

   > **Note**
   >
   > You can create multiple domains with different files as seeds.

5. After confirming all domains and seeds, choose Decompose.

6. AWS Transform will check the source code files and then decompose into domains with programs and data sets with similar use cases and high programming dependencies.

   AWS Transform gives you a tabular and graph view of decomposed domains as dependencies. Graph view has three options:

   - **Domain view** – Can view how different domains are related to each other in visual format.
   - **Dependency view** – Can view all files in each domain as a complex dependency graph. If a node that was added to a domain didn't receive information from a seed in the same domain, then this node will either be predicted into unassigned (node didn't receive any information), disconnected (in a sub graph that didn't receive seed information) or into another domain (node received information from at least that domain).
   - **Subgraph view** - User can create subgraphs to visualize a subset of nodes to more clearly understand the relational impact and boundaries of that collection of nodes.
     - To create a subgraph select a group of nodes and select the Extract subgraph option from the tool bar
     - Merging subgraphs is available, when merging a new third subgraph is created in addition to the two subgraphs you are merging.

   > **Note**
   >
   > Repeat these steps to add more domains or to reconfigure your already created domains with a different set of seeds if you don't like current domain structure.

7. When completed, choose **Continue**.

Seeds

Seeds are the foundational inputs for the decompose code phase. Each component or file (e.g., JCL, COBOL, Db2 tables, CSD, and scheduler files) can be assigned as a seed to only one domain, ensuring clear boundaries and alignment during the decomposition process.

The identification of the seeds depends on the structure of the application or portfolio. In the case of a typical mainframe legacy application, seeds can often be determined by adhering to established naming conventions, batch-level grouping in the scheduler, and transaction-level grouping defined in the CICS system. Additionally, database tables can also serve as seeds, providing another layer of structure for decomposition.

Import and/or update dependencies files

During decomposition, you can upload a JSON file for the dependencies that replaces the existing files generated by the dependencies analysis AWS Transform performs.

Export dependencies function allows you to download the dependencies json file generated in the decomposition step. After downloading, you can modify the file per your requirement. Then, you can **import dependencies** using the AWS Transform's upload functionality which allows you to upload the new JSON file of the dependencies that replaces the file generated by the dependencies analysis. After that, the graph in the decomposition step will be updated.

To export, modify, and import dependencies

1. On the **View decomposition results** page, choose **Actions**.
2. In the dropdown list, choose **Update dependencies file** option under **Other actions**.
3. In the **Update dependencies file** modal,

    a. Download the dependency file AWS Transform created from the existing analysis results.

    b. In the downloaded file, modify the dependencies based on what you want to achieve.

    c. After modifying, save and upload this file using the **Upload dependency file** button.

    > **Note**
    >
    > The only accepted file format is JSON file.

4. Next, choose **Import**.

AWS Transform will import the dependency file and create a new dependencies graph based on your input.

Import and/or Update Domains

For customers that have domains, seed, and/or file relationships mapped prior to the decomposition step, you can upload this domain definition through the Import domains file function available through the Actions menu. Some examples of when this function may be utilized:

- Bring forward decomposition from another job
- Subject matter experts providing this mapping

Once the domains file has been imported, the user can either run decomposition against the domain definition, or if satisfied can Save and then Submit the domain definition.

Parent/child/neighbor files

In a dependencies graph, programs relate to each other through different types of connections. Understanding these relationships helps you analyze program dependencies during transformation of your mainframe applications. It also helps with understanding the boundaries of a domain. For example, if you select a domain, and then select parent one level, it will show you the connected nodes.

Parent relationships – A parent file calls or controls other programs. Parents sit above their dependent programs in the hierarchy. You can select parent at one level or at all levels.

Children relationships – A child file is called or controlled by the parent program. Children sit below their parent in the file hierarchy.

Neighbor relationships – Neighbors are files at the same hierarchical level. They share the same parent program and might interact with each other directly.

Reforge code

Reforge uses Large Language Models (LLMs) to improve the quality of refactored code. The initial COBOL-to-Java transformation preserves functional equivalence while retaining COBOL-influenced data structures and variable names from the legacy system. Reforge restructures this code to follow modern Java practices and idioms, replacing COBOL-style constructs with native Java collections and naming conventions. This makes the code more readable and maintainable for Java developers.

> **Note**
>
> Quotas for reforge are:
>
> - 3,000,000 lines of code per job
> - 50,000,000 lines of code per user per month

Reforge your code after refactoring by following these steps:

1. Choose **Reforge java code** in the left navigation pane and then select **Configure code reforge**.

2. Provide the S3 location to your zipped buildable source project and choose **Continue**. Use this zip structure:

```
input.zip
        ### PROJECT-pom
        ### PROJECT-entities
        ### PROJECT-service
        ### PROJECT-tools
        ### PROJECT-web (optional)
        ### pom.xml
```

   AWS Transform analyzes your zip package to locate files within the PROJECT-service directory so that it can provide a selectable list of classes that you can reforge. These classes have the suffix `ProcessImpl.java`.

3. Complete the **Select classes to reforge** page and choose **Continue**. Track the reforge status on the **Worklog** tab.
4. View the results of your completed reforge on the **View results** page, which displays the reforge status per class. It also specifies where to find the Reforge result in your S3 bucket.

Once AWS Transform gets this input from you it gives you a downloadable file with the **Reforge results**.

This is the zip structure resulting from a successful reforge:

```
reforge.zip
### maven_project
### reforge.log
###tokenizer_map.json
```

- **maven_project** contains the reforged source code.
  - Files that have been refactored but whose compilation was not successfully finalized are located at `/src/main/resources/reforge/originalClassName.java.incomplete` and are named `originalClassName.java.incomplete`. Compare these to the original versions of the files to choose reforged functions you want to save.
  - Source files provided to AWS Transform that were refactored successfully are backed up to `src/main/resources/reforge/originalClassName.java.original` and are named

`originalClassName.java.original`. The refactored versions of the files replace the source files provided to AWS Transform.

> **Note**
>
> The `originalClassName.java` files are replaced with the reforged files only if the reforging process is successful. Otherwise, they retain the original content.

- **reforge.log** contains logs that you can use to diagnose job failures or provide to AWS support in case of an issue.
- **tokenizer_map.json** contains a mapping of token IDs to your data, such as file paths and class/method names, that are tokenized in the logs for privacy protection. You can provide this file to AWS support in case of an issue.

Plan your modernized applications testing

You can create and manage test plans for your mainframe modernized applications based on extracted code attributes, job complexity, and scheduler paths. AWS Transform helps prioritize which jobs to test and identifies the specific artifacts needed for each test case. The test planning process is divided into three main phases: configuration, scoping, and review.

To create a test plan

1. **Configure test plan settings**
   a. In the left navigation pane, under **Plan your testing**, choose **Configure settings**.
   b. (Optional) Provide S3 paths for your Business Logic Extraction (BLE). These artifacts enhance test plan quality. Without BLE artifacts some fields in the test plan may remain incomplete.
2. **Define test plan scope**
   a. Select entry points, such as batch jobs, to include in your test plan.
   b. Filter and sort jobs based on multiple attributes:
      - Business functions (extracted from BRE BLE)
      - Domains (from decomposition phase)
      - File paths and locations

- Custom search criteria

c. Select individual jobs or entire groups for testing.

d. Review job relationships and dependencies.

3. **Review and adjust the test plan**

   The generated test plan provides comprehensive information including:

   - Preferred execution order based on dependencies
   - Job group assignments from scheduler
   - Complexity scores, which are aggregate scores for test cases
   - Business domain associations
   - Cyclomatic complexity metrics
   - Dataset and table dependencies
   - Lines of code metrics
   - Business function mappings

Test plan customization options

Your test plan can be customized to address specific needs. For example, you can:

- Create new test cases by selecting multiple entry points
- Merge existing test cases to combine related functionality
- Split test cases for more granular testing
- Remove unnecessary test cases
- Add or remove entry points for existing test cases
- Modify test case descriptions and attributes
- Adjust execution order

Detailed test case information

Each test case provides details that describe its content and the related dataset or datafiles:

- Comprehensive description of test scope
- Complete list of entry points included

- Aggregate metrics showing complexity and size
- Business rules and automated test case guidance
- Dataset and table dependencies with direction (input/output)
- Interactive dependency graph visualization
- Execution prerequisites and requirements

Data management features

These details help you understand the data related to the test case. You can:

- Filter data sets by input/output direction
- Identify required artifacts for test execution
- Track data dependencies between test cases
- Monitor data set usage across test plans

> **Note**
>
> AWS Transform automatically analyzes scheduler dependencies and assigns complexity scores to help prioritize testing efforts. Higher complexity scores indicate jobs that may require more thorough testing or isolation during the testing process.

Business rules and test case guidance

The AWS Transform test plan provides recommendations for your test case plan based on Business Rule Extraction:

- Automatic processing of business rules using LLM
- Generation of synthetic test cases
- Testing guidance for specific business scenarios
- Traceability between rules and test cases

The final test plan is stored in the specified S3 location and includes all of the necessary information to execute your testing strategy effectively. You can export the test plan for integration with other testing tools or documentation systems.

> **Note**
>
> While synthetic test case guidance is provided, the actual test artifacts must be created separately based on the guidance. The test plan serves as a comprehensive blueprint for your testing strategy but does not generate the test data or execution scripts.

Test case creation rules - Summary

General rules

- A test case contains 1 to many JCLs in schedule execution order
- Test cases are created from valid supported schedulers (CA7 and Control-M)
- The test executes JCLs, not the scheduler itself or scheduled tasks
- A scheduled task is unique and executes only one JCL
- One JCL can be executed by multiple scheduled tasks
- One JCL can exist without being in a schedule

Default test case creation rules

- If a single JCL is in a test case:
  - JCL is not involved with a schedule
  - JCL is executed by a scheduled task that diverges into multiple branches
  - A branch in the schedule contains only this JCL
- If multiple JCL in a test case: represents a linear execution path sequence within a schedule branch
- If JCL is on a divergent scheduled task: JCL becomes its own separate test case
- If JCL is on a convergent scheduled task: JCL can start a new test case but won't be included in previous branches
- Missing JCLs are skipped and execution continues (with planned future enhancement to cut test case at missing JCL point)

User operations rules

- **Create Test Case**: user selects from available JCLs

  - Succeeds if JCLs exist in schedule execution branch sequence
  - Follows schedule execution order
- **Add a JCL to Test Case**: user selects from available JCLs
  - Succeeds if JCL can exist in the test case's schedule execution branch/path
- **Remove JCL from Test Case**: user can remove any JCL from a test case
  - Allowed even if it causes execution path gaps
- **Merge Test Cases**: user selects two test cases to combine
  - Succeeds if JCLs can exist together in same schedule execution branch
  - Maintains schedule execution order
- **Split Test Case**: user selects one JCL in a test case for split
  - Creates new test case from split point forward
  - Original test case modified to exclude JCLs past split point
- **Delete Test Case**: user can delete any created test case

> **Note**
>
> You cannot create, add, or merge test cases from different scheduler branches/path. A future enhancement is planned to allow operations beyond divergent/convergent tasks in the scheduler.

Generate test data collection scripts

You can generate JCL scripts to collect test data from your mainframe systems based on the test plan created in the previous step. AWS Transform automatically creates data collection scripts for datasets, database tables, and sequential files needed for comprehensive testing. The data collection process is divided into four main phases: input configuration, test case selection, script configuration, and script generation.

To generate test data collection scripts

1. **Provide test plan input**

   a. In the left navigation pane, under **Test data collection**, choose **Provide test plan input**.

b. Specify the S3 path to your test plan JSON file from Plan your modernized applications testing.

c. The input field is pre-populated if the test plan was generated in a previous job step.

d. You can also select test plan from other jobs by specifying the appropriate S3 location.

2. **Select test cases for data collection**

a. Review the complete list of test cases from your test plan.

b. Filter and sort test cases based on multiple attributes:

- Business functions and domains
- Database table dependencies
- Data set requirements
- Complexity metrics
- Custom search criteria

c. Select individual test cases or use bulk selection options.

d. Review test case details including entry points, metrics, and business rules by clicking on one Test Case to see details.

3. **Configure data collection scripts**

a. Download sample templates and configuration files for reference.

- AWS Transform provides sample templates for Db2 database unloads, VSAM file REPRO and sequential dataset processing to be used as guidance on the kinds of template expected by the process.
- Standards may vary from site to site so the expectation is that customers will modify or replace these templates that conform to their own standards.
- These modified templates need to be uploaded to a S3 bucket where the test data collection can process them.

b. Provide variable configuration file (JSON format) containing:

- User prefixes and environment-specific constants
- Database configuration parameters
- Destination endpoint settings and data transfer parameters
- Other required parameters defined by user and to be used in JCL templates

c. Upload JCL templates for different data collection methods:

- **Db2 template**: For database table unloading (customize for BMC, IBM DSN, or other unload utilities)
- **VSAM template**: For VSAM file processing (typically uses REPRO utility)
- **Sequential datasets template**: For processing sequential datasets, partitioned datasets, GDGs etc.

4. **Review and manage generated scripts**

The generated scripts provide comprehensive data collection capabilities including:

- Separate scripts for "before" and "after" test execution data collection
- Organized script structure by test case and data type
- Scripts are automatically stored into S3 bucket for easy access and transfer
- Generated JCL scripts are ready for mainframe execution
- Variable substitution is based on user configuration defined in the templates

Script generation features

Generated scripts are automatically customized based on your templates and configuration:

- **Template-based generation**: Uses your provided JCL templates with variable substitution
- **Environment**: Incorporates your specific mainframe configuration
- **Data type handling**: Creates appropriate scripts for sequential datasets, VSAM files, and database tables
- **Collection for test cases**: Generates both "before" and "after" data collection scripts
- **Sequential dataset processing**: AWS provided sample provides for a file transfer function, but this can be customized to compression utilities available at your site or utilities such as Connect Direct or managed file transfer etc.

Data collection strategy

The generated scripts support comprehensive data collection strategies:

- **Sequential dataset collection**: REPRO and copy utilities for VSAM and flat files
- **Database table unloading**: Customizable Db2 unload processes

- **Sequential dataset processing**: Customizable post processing of sequential datasets such as compression, managed file transfer services etc.
- **Dependency management**: Coordinated collection based on test case definition

> **Note**
>
> AWS Transform generates scripts based on your templates and configuration. Review all generated JCL before executing on your mainframe environment to ensure compatibility with your specific system configuration and security requirements.

Template customization and best practices

ATX Test Data Collection provides flexible template customization capabilities:

- **Multi-utility support**: Adapt templates for different mainframe utilities (BMC, IBM, DSN)
- **Variable-driven configuration**: Use constants for environment-specific parameters
- **Reusable templates**: Create standardized templates for consistent script generation
- **Data handling**: Incorporate organization-specific data handling requirements
- **Security integration**: Include appropriate security and access controls
- **Performance optimization**: Configure for efficient data collection and transfer

Generated output structure

The generated scripts are organized in your S3 bucket with the following structure:

- **Test case organization**: Scripts grouped by associated test cases
- **Collection timing**: Separate folders for "before" and "after" data collection
- **Data type classification**: Scripts organized by sequential datasets, database tables, and transfers
- **Metadata files**: Summary information and execution guidance
- **Ready-to-transfer format**: JCL formatted for direct mainframe deployment

The final script collection is stored in the specified S3 location and includes all necessary JCL to execute your data collection strategy effectively. You can download the scripts for transfer to your mainframe environment or integrate with automated deployment processes.

> **Note**
>
> While comprehensive JCL scripts are generated, actual execution must be performed on your mainframe environment. The scripts serve as ready-to-use data collection tools but require appropriate mainframe access and execution permissions.

Test automation script generation

You can generate test automation scripts to execute test cases on your modernized application based on the test plan created in the previous step. AWS Transform automatically creates comprehensive test scripts that utilize the data collected from the test data collection process. The test automation script generation process consists of three main phases: input configuration, test case selection, and script generation results.

To generate test scripts

1. **Provide test plan input**
   a. In the left navigation pane, under **Test automation script generation**, choose **Provide test plan input**.
   b. Specify the S3 path to your test plan JSON file from Plan your modernized applications testing.
   c. The input field is pre-populated if the test plan was generated in a previous job step.
   d. You can also select test plan from other jobs by specifying the appropriate S3 location.
   e. The system uses this test plan as the foundation for generating automation scripts.
2. **Select test cases for script generation**
   a. Review the complete list of test cases from your test plan.
   b. Filter and sort test cases based on multiple attributes:
      - Business functions and domains
      - Database table dependencies
      - Data set requirements
      - Complexity metrics
      - Custom search criteria

c. Select individual test cases or use bulk selection options with the **Check All** and **Uncheck All** buttons.

d. Review test case details including entry points, metrics, and business rules by clicking on individual test cases.

The selected test cases will have automation scripts generated for execution on the modernized application.

3. **Review and manage generated test automation scripts:**

- The system displays a success message confirming script generation completion.
- Generated test scripts are automatically stored in your specified S3 bucket location.
- Access the complete list of generated test scripts with their respective S3 locations.
- Each test case has its corresponding automation script stored in individual S3 locations.
- Scripts are ready for deployment and execution on your modernized application environment.

Test automation script features

The generated automation scripts provide comprehensive testing capabilities:

- **Modernized application testing**: Scripts are specifically designed to execute test cases on your transformed application
- **Data integration**: Utilizes the test data collected from the previous test data collection step, these data need to be copied to folders for each test cases
- **Automated execution**: Scripts can be used to set up data sinks, run test cases, and compare results, some parameters will have to be set according to your deployment environment
- **Organized structure**: Scripts are systematically organized by test case in your S3 bucket
- **Ready-to-deploy format**: Scripts are formatted for direct deployment to your testing environment

Generated output structure

The generated test automation scripts are organized in your S3 bucket with the following structure:

- **Test case organization**: Each test case has its dedicated script stored in individual S3 folder

- **Execution-ready format**: Scripts are formatted for immediate deployment and execution after setting up some variable depending from your environment
- **Centralized access**: All scripts are accessible from a single S3 bucket location for easy management

Test execution strategy

The generated scripts support comprehensive test execution workflows:

- **Environment setup**: Scripts include capabilities for setting up initial data to execute the test
- **Data preparation**: Integration with collected test data from test case data collection step
- **Test case execution**: Automated execution of individual test cases on the modernized application
- **Result comparison**: Built-in capabilities for comparing test results and validating application behavior

> **Note**
>
> AWS Transform generates test automation scripts based on your test plan and selected test cases. The scripts are designed for execution on your modernized application environment and utilize the test data collected in the previous step. Review all generated scripts before deployment to ensure compatibility with your specific application configuration and testing requirements.

Best practices for test automation

- **Environment validation**: Ensure your modernized application environment is properly configured before script execution
- **Data verification**: Validate that the required test data from the collection phase is available and accessible
- **Script customization**: Review and customize generated scripts as needed for your specific testing requirements
- **Execution monitoring**: Implement appropriate monitoring and logging during test script execution
- **Result analysis**: Establish processes for analyzing test results and identifying application issues

The final collection of test automation scripts provides a complete testing framework for validating your modernized application functionality. The scripts can be integrated into your continuous testing processes or executed as part of your application validation workflow.

Deployment capabilities in AWS Transform

AWS Transform helps you set up cloud environments for modernized mainframe applications by providing ready-to-use Infrastructure as Code (IaC) templates. Through the AWS Transform chat interface, you can access pre-built templates that create essential components like compute resources, databases, storage, and security controls. The templates are available in popular formats including CloudFormation (CFN), AWS Cloud Development Kit (AWS CDK), and Terraform, giving you flexibility to deploy your infrastructure.

These templates serve as building blocks that reduce the time and expertise needed to configure environments for your modernized mainframe applications. You can customize these templates to fit your needs, giving you a foundation to build your deployment environment.

To retrieve the IaC templates, ask in the AWS Transform chat for the Infrastructure-as-Code templates clarifying your preferred modernization pattern (such as AWS Transform for mainframe refactor), your preferred topology (standalone vs high availability), and your preferred format (CloudFormation vs Cloud Development Kit vs Terraform).

Build and deploy your modernized application post-refactoring

After you complete the refactoring process with AWS Transform, you can build and deploy your modernized Java application. This guide walks you through retrieving your modernized code, configuring your environment, and deploying and testing your application.

> **Note**
>
> In addition to the guidance provided here, the AWS Transform generated code package will include an *Set up the AWS Automated Refactor Development Environment* document which provides instructions to set up a IDE (Integrated Development Environment) with Developer Runtime.

Topics

- Prerequisites

- Step 1: Retrieve the modernized code
- Step 2: Build the modernized application
- Step 3: Configure the test environment
- Step 4: Deploy the modernized application
- Step 5: Test the modernized application
- Additional example

Prerequisites

Before you begin, make sure you have:

- Successfully completed a refactoring job with AWS Transform.
- Access to your modernized code.
- Installed and configured build software tool stack on your development machine, such as Apache Maven or Apache Tomcat. For more information on Runtime versioning, see AWS Transform for mainframe Runtime release notes.
- Installed and configured Amazon Corretto or a version of Java runtime. For more information on installing Amazon Corretto, see Amazon Corretto 24.
- Access to create and configure Amazon Aurora PostgreSQL databases for Runtime components, if necessary. For more information on Creating the Aurora PostgreSQL database, see Working with Amazon Aurora PostgreSQL.
- Administrative access to deploy applications to your runtime environment.
- Reviewed the AWS Transform for mainframe Runtime concepts for fundamental concepts on applications modernized with AWS automated refactoring solution.

Step 1: Retrieve the modernized code

To retrieve the modernized code

1. Download and extract the generated code package.
2. Open `codebase/app-pom/pom.xml` and note the required runtime engine version. For example `<gapwalk.version>4.6.0</gapwalk.version>`.
3. Locate the *Set up the AWS Automated Refactor Development Environment* document from the downloaded code package for reference.

Step 2: Build the modernized application

To build your modernized application

1. Access the Runtime from the AWS Transform for mainframe Toolbox.
2. Download and install the appropriate runtime version (identified in the section called "Step 1: Retrieve the modernized code") on your local development machine.

   > **Note**
   >
   > Additional information for installing the runtime dependencies on your local machine is available in section 3.1 of the *Set up the AWS Automated Refactor Development Environment* document.

3. Open the command prompt and navigate to your application's root directory.
4. To build deployable packages for the modernized application run the Maven build command:

   ```
   mvn package
   ```

   Refer to the Application Organization page for details on the basic organization of the modernized code. For instance, for modernized application containing a front-end web application, you may expect at-least the following deployable `.war` aggregates in addition to the runtime components:

   - **Service project**: Contains legacy business logic modernization elements

     ```
     <business-app>-service.*.war
     ```

   - **Web project**: Contains the modernization of user interface-related elements

     ```
     <business-app>-web.*.war
     ```

Step 3: Configure the test environment

To configure your test environment

1. Configure your modernized application runtime. For more information, see the Set up configuration for Runtime section in the *AWS Mainframe Modernization user guide*.

> **Note**
>
> Refer section 5 of the *Set up the AWS Automated Refactor Development Environment* guide for runtime component specific configuration examples.

2. Prepare input and output (I/O) data sets for modernized applications. Modernized applications may process sequential I/O data sets, VSAM data sets, or others.

> **Note**
>
> Refer section 6 of the *Set up the AWS Automated Refactor Development Environment* for examples.

3. A runtime environment - You can use your existing runtime environment or create a new runtime environment.
   - To configure a non-managed runtime environment, see Set up a non-managed application.
   - To configure a managed runtime environment, see Set up a managed application.

After configuring the test environment, you move to the next step of deploying the modernized application.

Step 4: Deploy the modernized application

Deploy the application artifacts in the runtime you created and/or configured in the the section called “Step 2: Build the modernized application” and the section called “Step 3: Configure the test environment” sections.

Additional guidance for deploying the modernized application can be found using these links:

- Deploy on Amazon EC2
- Deploy on containers on Amazon ECS and Amazon EKS
- Create an AWS Mainframe Modernization application

Step 5: Test the modernized application

After deployment,

1. Review the available Runtime APIs for ways to interact with the modernized applications.
2. Test your application to align its functional equivalence with legacy application. For example, see Test a sample application in the *AWS Mainframe Modernization user guide*.

Additional example

For a specific example of modernizing mainframe application with AWS Transform, see Modernize the CardDemo mainframe application.

Full-stack Windows modernization

AWS Transform provides full-stack Windows modernization, including .NET modernization and SQL Server modernization.

The AWS Transform agent for .NET can help you modernize your .NET applications for compatibility with cross-platform .NET. You can modernize .NET applications at scale from a web application, or work on individual solutions from Visual Studio IDE.

AWS Transform for SQL Server modernization is an AI-powered service that automates the full-stack modernization of Microsoft SQL Server databases and their associated .NET applications to Amazon Aurora PostgreSQL. The service orchestrates the entire migration journey from schema conversion, data migration and modifying application code to match the new target PostgreSQL, making your teams more productive by automating complex and labor-intensive tasks.

Topics

- Modernizing .NET with AWS Transform
- SQL Server modernization

Modernizing .NET with AWS Transform

AWS Transform for .NET is a generative AI-powered agent that helps you modernize your .NET applications. You can modernize legacy .NET Framework applications to cross-platform .NET, and upgrade .NET applications to later versions.

Multiple experiences

AWS Transform is available in multiple experiences, including a web application, Visual Studio IDE, or AI code companions via MCP. Refer to How to work with the .NET agent for recommendations based on role and scenario.

Capabilities and key features

The scope of the .NET agent is .NET-to-.NET code transformation. If your transformation needs include non-.NET languages and frameworks, such as Web Forms to React, use AWS Transform custom.

AWS Transform for .NET has these capabilities:

- Connect to source control systems (Azure Repos, Bitbucket, GitHub, GitLab)
- Assess .NET repositories for modernization and provide a customizable modernization plan
- Modernize legacy .NET Framework applications to modern cross-platform .NET
- Update modern .NET applications to later versions
- Convert Azure cloud-native application code to use AWS equivalents for popular services
- Validate transformed code with local build verification and unit test execution
- Produce a transformation report and next steps guidance after transformation

Supported versions and project types

The .NET agent can transform the following versions, languages, and project types:

Transform from	.NET Framework 3.5, .NET Core 3.1, .NET 5.x+ through .NET 10
Transform to	.NET 8, .NET 10, .NET Standard (class libraries)
Languages	C#, VB.NET
Supported project types	Class libraries, Console apps, ASP.NET (MVC, Web API, Web Forms), Unit test projects (NUnit, xUnit, MSTest), Windows Communication Foundation (WCF) services, Desktop (WinForms, WPF), Mobile (Xamarin), ASMX web service

Limitations

For more information on quotas and limitations for AWS Transform, see Quotas for AWS Transform.

AWS Transform does not transform the following:

- Win32 DLLs that don't have core compatible libraries
- Repositories that do not contain any .NET solutions.
- Web site projects without a project file (must be converted to a Web application).

Human intervention

During the porting of .NET Framework applications to cross-platform .NET, you may be requested to provide input or approvals in the following scenarios:

- Set up a connector to your source code and permissions
- Validate the proposed modernization plan
- Upload missing package dependencies as NuGets
- Review and accept the transformed code

More information

You can modernize your .NET code by using either the AWS Transform web application or the AWS Toolkit for Visual Studio.

- Modernizing .NET code with the AWS Transform web application
- Modernizing .NET in the IDE
- Best practices for .NET transformations

How to work with the .NET agent

Experiences

AWS Transform .NET modernization is available in several experiences:

- **Web console**: For large-scale transformation of repositories.
- **Visual Studio IDE**: Developer-led transformation of a solution or project working alongside agent interactively and iteratively. Requires a Windows development machine.
- **Command line**: Transform .NET solutions with the ATX CLI using an AWS-managed transformation.

You can also invoke AWS Transform from AI code companions:

- **Kiro**: transform from Kiro using the AWS Transform for Kiro power.
- **Other AI code companions**: transform from your preferred AI coding assistants using AWS Transform MCP agents.

A common pattern is to first modernize with AWS Transform for .NET, then hand off to an AI code companion for last mile work.

Area	Web application	Visual Studio IDE	ATX CLI
Target role	IT professional	Developer	Developer
Operating system	Any	Windows	Linux, macOS, Windows
Source control access	• Repository (AWS CodeConnections) • Repository (Personal Access Token) • Code zip file (Amazon S3) • Code zip file (Direct Upload)	Local file system	Local Git repository
Job scope	Up to 5,000 repositories per job	One solution at a time	One solution at a time
Transformed code location	Written to a new writeable repository branch	Replaces original code on your file system	Replaces original code on your file system
Unit of work	Repository	Project	Project
Assessment	• Analyze repositories • Assessment reports (multiple formats) • Global modernization plan	• Analyze projects • Assessment report • Modernization plan	• Analyze projects • Assessment report • Modernization plan
Customize modernization plan	• Customize global plan	• Customize plan	• Customize plan

Area	Web application	Visual Studio IDE	ATX CLI
	• Upload steering docs	• Upload steering docs	• Provide steering doc
Transformation modes	Autonomous or Interactive	Autonomous or Interactive	Autonomous or Interactive
Checkpoint reviews	Review each repository after transformation	Review each project after transformation	Review each project after transformation
Iterative modernization	• Ask for changes at checkpoint • Ask for changes after transformation • Retry repo with revised instructions • Assess additional repos in same job	• Ask for changes at checkpoint • Ask for changes after transformation • Retry project with revised instructions • Review code diffs • Edit code at checkpoint	• Ask for changes at checkpoint • Ask for changes after transformation • Retry project with revised instructions • Review code diffs
Downloadable artifacts	• Transformed code • Assessment report • Modernization plan • Transformation report • Next Steps markdown	• Transformed code • Assessment report • Modernization plan • Transformation report • Next Steps markdown	• Assessment report • Modernization plan • Transformation report • Next Steps markdown
Unit test validation	Ports and runs existing unit tests	• Ports and runs existing unit tests • Generates missing unit tests for modernized code	Ports and runs existing unit tests

Recommended experiences

Use the experience that best fits your scenario.

Scenario	Recommended experience
Work at repository level when transforming	Web application
Perform large-scale transformations	Web application
Transform from a Mac	Web application
Transform unattended	Web or IDE, autonomous mode
Work at project level when transforming	Visual Studio IDE
Local build to confirm transformed code builds on local environment	Visual Studio IDE
Agent leads the modernization, I review	Web or IDE, autonomous mode
I lead the modernization, agent assists	Web or IDE, interactive mode, customize plan
Complex modernization	IDE, interactive mode
Portfolio owner starts transformation, hands off to developers for review	Web application; use beaming to hand off repositories to developers, who review in the IDE
Script modernization in an existing pipeline or workflow	ATX CLI

Transform .NET solutions with the ATX CLI

You can modernize a .NET solution from the command line using the ATX CLI (AWS Transform custom) and the AWS-managed transformation `AWS/dotnet-modernization`.

Use the following command form:

```
atx custom def exec -n AWS/dotnet-modernization -p <path-to-solution> [-q] [-x] [-t]
```

The following command-line options are available:

- **-q** (quiet): show agent conversation only, hide tool execution details
- **-x** (autonomous): run in autonomous mode without user interaction
- **-t** (trust): trust all tools without prompts

The following table shows example commands.

Command line example	Description
`atx custom def exec -n AWS/dotnet-modernization -p C:\dev\InventoryService -t -q`	Transform the solution in interactive mode, showing agent conversation only, and trust tool updates.
`atx custom def exec -n AWS/dotnet-modernization -p C:\dev\InventoryService -t`	Transform the solution in interactive mode, showing all tool activity, and trust tool updates.
`atx custom def exec -n AWS/dotnet-modernization -p C:\dev\InventoryService -t -x`	Transform the solution in autonomous mode and trust tool updates.

Modernizing .NET code with the AWS Transform web application

You can modernize your .NET code using the AWS Transform web application (described here) or the Visual Studio IDE. The web application is best suited for IT professionals interested in large-scale transformation. It works at repository level, and can transform as many as 5,000 repositories in a single job.

To access the web application, navigate to the web link you received with your AWS Transform email invitation or contact your AWS Transform administrator.

Web transformation workflow

To modernize your .NET repositories using the AWS Transform web application, navigate to the web application and follow these steps:

1. **Start transformation**

- Create a .NET modernization job
- Connect to source code

2. **Discovery and assessment**
   - Discover and confirm repositories
   - Set transformation mode
   - Assessment
3. **Prepare for transformation**
   - Resolve missing package dependencies
   - Review and customize transformation plan
4. **Transformation and checkpoint reviews**
   - Transform your .NET code
5. **End of transformation**
   - Review transformation artifacts
   - Validate and finalize with AI code companion

Developer hand-off

Hand-off of repositories to developers is common, and can happen in two ways:

1. **Early hand-off:** For complex repositories, have developers perform the transformation in IDE instead of using the web application. Complex repositories are best transformed by developers working on one solution at a time in Visual Studio, interactively and iteratively, where they can directly review the transformed code and guide the agent at project level. AWS Transform will recommend handing off complex repositories to developers after assessment.
2. **Post-transformation hand-off:** After repositories are transformed in the web application, hand them off to developers to review, validate, and debug the transformed application - typically assisted by an AI code companion. See Beam a transformed repository to a developer.

Create a .NET modernization job

After you create or select your workspace, create a .NET modernization job:

1. On the **Jobs** tab, choose **Create job**.
2. In the chat pane, choose **Windows modernization** and then **.NET modernization agent**.

3. Follow the prompts from AWS Transform in the chat pane using natural language.

After your job starts, a Job Plan tab opens and you are guided to set up your job. This is one of 4 views you can select using the icon buttons at left.

- **Job Plan:** This tab is for job set up, monitoring, and conversing with the agent. You and the agent collaborate on this tab to plan and execute .NET modernizations.
- **Approvals:** Workflow approvals are handled on this tab, such as administrator approval of a repository source code connection.
- **Artifacts:** Artifacts can be downloaded or uploaded to this tab, such as assessment reports, modernization plans, transformed code, transformation report, and next steps.
- **Worklog:** This tab shows the agent's current activity in natural language.

Take note of the job's browser URL. You can use it to directly return to the job. If you ever need to report an AWS Transform issue, providing the job browser URL helps AWS identify and analyze your job.

Connect to source code

After Create a .NET modernization job, on the **Job Plan** tab, the left pane of the AWS Transform window lists the phases of the transformation job. The first phase is *Connect source code*. In this phase, you connect to your source code.

Before you begin, make sure that you have an AWS account with permissions to create the source code connection. Depending on the connection method that you choose, you also need access to AWS CodeConnections, a personal access token (PAT) stored in AWS Secrets Manager, or an Amazon S3 bucket in the same AWS Region as your transformation job. Creating a connector requires approval by an AWS administrator.

You can connect to a source code repository or provide code in a zip file. You have 4 options for connecting to source code:

- **Code Connection**: Connect to source code repository using AWS CodeConnections (GitHub, GitLab, Bitbucket, Azure DevOps).
- **PAT**: Connect to source code repository using a Personal Access Token stored in AWS Secrets Manager.

- **S3**: Provide an Amazon S3 URI to a code archive zip file (for example, `s3://my-bucket/repos.zip`).
- **Direct Upload**: Drag and drop a code archive zip file into the chat pane.

Select a code connection method and the agent will guide you through set up.

Connect to a repository with AWS CodeConnections

If you choose the Code Connection option, the AWS Transform agent uses AWS CodeConnections to download your .NET codebase from Azure Repos, Bitbucket, GitHub, or GitLab. Transformed code is written to a new writable branch. You must set up AWS CodeConnections in the same Region as your AWS Transform job.

> **Note**
>
> If you have an existing source code repository connector, AWS Transform will notify you. Select **Use existing connector** to use this connector. If you do not wish to use the existing connector, see Deleting an existing connector before Adding a new connector. You can only have one source repository connector per job.

Adding a new connector

To create a new source code repository connector:

1. Enter the AWS account number that you would like to use for the AWS CodeConnections connector.
2. If you have not set up AWS CodeConnections for the same AWS account, Set up AWS CodeConnections.
3. Use the AWS Developer Tools Console to create a connection to your Bitbucket, GitHub, GitHub Enterprise Server, GitLab, or AzureDevOps repository.
4. Copy the Amazon Resource Name (ARN) of the connection you created.
5. Return to AWS Transform, and paste the ARN of the connection you created.
6. Enter a name for the connector.

> **Note**
>
> Do not enter personal information as part of the connector name.

7. Select **Initiate connector creation.**
8. AWS Transform makes the request to create a connector for AWS account you entered, in the same Region as the current transformation job, and notifies you that an approval request is ready to be approved by your AWS administrator in the AWS Management Console. Select **Copy the verification link**.
9. If you are the AWS administrator, sign into the AWS account and go to the verification link to approve the connection request. If you are not the AWS administrator, provide the verification link to your AWS administrator.
10. After the AWS administrator approves the connector request, select **Finalize connector** to complete the connector creation process. Should you wish to use a different connector, select **Restart**, to restart the connector creation process.

The Collaboration tab also includes details about your connector, which include:

- Status -- Approved or Pending approval
- AWS account ID
- AWS Region
- Connection ARN

Deleting an existing connector

> **Note**
>
> If you have an existing source code repository connector, AWS Transform will notify you. Select either **Use existing connector** or **Delete and create a new connector**.

If you choose to delete an existing source code repository connect, AWS Transform will warn you before actually deleting the connector.

1. You can delete an existing connector a couple of ways:

a. If AWS Transform prompts you to, you can select **Delete and create a new connector**.

b. You can also select **restart** in the prompt, **To modify this connector, you must restart**.

2. AWS Transform warns you that restarting will delete the connector. To delete the connector, select **Restart** again.
3. AWS Transform warns you again that deleting the connector will remove the connection to your third party repository, such as Bitbucket, GitHub, and GitLab. Also, any AWS Transform jobs that are using this connector will fail if the connector is deleted. To confirm deletion, type *delete* and select **Delete**.
4. To add a new source code repository connector, see Adding a new connector.

Connect to a repository with a Personal Access Token

You can connect to a repository using a Personal Access Token (PAT). Follow these steps:

1. When the agent prompts for a source code connection, choose **PAT** in the chat pane.
2. Read the displayed instructions about generating a PAT and storing it in AWS Secrets Manager.
3. Generate a PAT from your source control.
4. Store your PAT in AWS Secrets Manager and record the Amazon Resource Name (ARN). For example, a GitHub token is formatted like this:

```
github_pat_XXXXXXXXXXXXXXX
```

5. In chat, choose the **Set up connector** link to open a connector form.
6. In the connector form, enter a connector name, your AWS account ID, the secret ARN, and optionally a KMS key ARN. Then submit the form.
7. The agent provides an approval link. Copy the link and send to an administrator to approve the connection. After approval, continue with the agent.

Connect to source code in Amazon S3

You can connect AWS Transform to source code in Amazon S3 as an alternative to connecting a source code repository.

S3 bucket organization

Original source code and transformed code are stored in a common S3 bucket, organized as shown below. You must set up your S3 bucket in the same region as your AWS Transform job. Upload your source code to the bucket as a zip file at root level. The zip file must contain a top level folder for each repository.

During transformation, AWS Transform creates a transform-output folder, and stores the transformation results in that folder. Transformation creates a zip file named *transformed-code.zip* containing the transformed code. This includes a code differences report file name diff.txt that highlights file changes at a project level.

```
<customer-bucket>/
### source-code.zip
         ### repo 1
         ### repo 2
         ### ......
         ### repo n
### transform-output/
      ### transformed-code.zip
```

Adding a new S3 Connector

To create a new S3 source code repository connector:

1. In the AWS console, create an S3 bucket.
2. Upload your source code as a zip file to the S3 bucket.
3. In the AWS Transform web app, start a .NET transformation job. In the **Connect a source code repository** step, select **Connect your source code in S3 bucket** and choose **Save**.
4. On the next page, enter your S3 bucket details and choose **Submit**.
   a. Connector name
   b. AWS Account ID
   c. S3 Bucket Arn
   d. S3 Bucket Encryption Key (optional)
5. Approve the connector:
   a. On the next page, copy the verification link.

b. Have an approver browse to the link to reach the AWS Transform connector creation request page.
c. Choose to create a new role or use an existing role.
d. After reviewing the request, choose **Save** and **Approve** to approve it.

6. Specify the S3 zip file location:
   a. In the AWS Transform web app, wait for status to show **Approved**.
   b. On the **Specify asset location** page, enter an S3 URL for the code zip file in the format `s3://bucket-name/zip-file-name` and choose **Send to AWS Transform**.
   c. The job proceeds to the **Discovery** step to continue the transformation.

Deleting an S3 Connector

To delete an existing S3 connector:

1. In the AWS Transform web app, select **Manage connectors** at the top right.
2. In the **Manage connectors** window, select the connector.
3. Choose **Delete**.

Discover and confirm repositories

After Connect to source code, the agent will scan your source code to discover .NET repositories. In the *Discovery* phase, you can:

- Review discovered repositories and select which ones to assess for transformation.
- Download a repository report in HTML or JSON.
- Upload steering documents to customize your transformation.

Transformation settings

Transformation mode

You can transform in autonomous or interactive mode.

- Use **autonomous mode** for unattended transformation, suitable for overnight or long-running jobs. You'll still have an opportunity to review results and ask for changes after transformation completes.

- Use **interactive mode** to work alongside the agent in chat. This gives you high visibility into transformation, a customizable transformation plan, and checkpoint reviews of repositories as the agent transforms them. You can iterate to refine results or change direction until satisfied.

To view or change transformation settings, select the **Transformation settings panel** or choose the **Transformation settings** link in chat. From the right panel, select **Autonomous** or **Interactive** and then **Save Changes**. You can freely switch transformation modes throughout the course of the job. For example, you could begin in interactive mode, review and customize the transformation plan, and then switch to autonomous mode for the remainder.

Assessment

After you confirm repositories for assessment in the *Discovery* phase, your job will enter the *Assessment* phase. AWS Transform analyzes your repositories for modernization suitability based on the following considerations.

- Framework version and upgrade path
- Code complexity and solution structure
- NuGet dependencies (public vs. private)
- Database connections and ORM patterns

After analysis is complete, you'll see a summary in chat showing the following for each repository:

- Repository name
- Complexity
- Number of solutions
- Number of projects
- Lines of code

You can now review the assessment details, select repos for transformation, or assess additional repos.

You can ask questions about the assessment in chat. When you are done reviewing the assessment, instruct the agent to generate a modernization plan for some or all of the repositories.

Assessment Summary and Report

To review the assessment, choose the **Assess > General** assessment step at left, or the assessment details link in agent chat. The right panel shows additional detail, including language, complexity rating, and Linux readiness rating. The ratings identify which repositories require minor, moderate, or significant effort to transform.

Detailed assessment information is available in assessment reports. Using the **Download report** button, you can download assessment reports in 5 formats:

- **Excel assessment report:** package compatibility findings.
- **HTML assessment report:** detailed assessment including executive summary, portfolio details, repositories, project analysis, findings, Linux incompatibilities, package and API compatibility, and recommendations.
- **Markdown assessment report:** assessment detail in markdown format.
- **Spreadsheet assessment report:** tabular assessment detail in CSV format.
- **JSON assessment report:** assessment detail in JSON format.

Resolving package dependencies to prepare for transformation

In the *Prepare for transformation* phase, you will:

- Select repositories for transformation.
- Resolve any missing dependencies.
- Review and optionally customize the generated modernization plan.

If AWS Transform finds missing package dependencies, you must complete this step. You can run a Windows PowerShell script to get the missing package dependencies from the same device as your Visual Studio development environment, or you can retrieve the missing packages manually. Then, upload the missing packages.

The Missing Package Dependencies can be updated in two ways:

- **Resolve using Artifact Connector:** Resolve by configuring an Artifact Repository (ADO NuGet Connector) through a dedicated Connect Artifact Repository HITL workflow, which is displayed when an Artifact Connector is not configured and packages are missing.

- **Update missing package:** AWS Transform lists the missing packages in the Missing package dependencies table. You can search for a missing package by name in the search box. This table includes the following details about the missing packages:
  - Name
  - Associated repositories
  - Framework version status
  - Core version status

To resolve the missing package dependencies, Upload the missing packages.

Upload the missing packages

1. If you choose, you can download a Windows PowerShell helper script to retrieve the missing package dependencies from within your Visual Studio development environment. Or you can find the missing packages manually.

   To use the Windows PowerShell script:

   a. Select **Download Windows PowerShell script**.

   b. Run the script locally with an active connection to the repositories that contain the missing package dependency files.

   c. This script allows you to download the missing package dependencies to your local environment.
2. The script will create a single zip file for you to upload which includes all of the dependencies in one archive. You can also upload individual `.nupkg` or zip files for each dependency.
3. Select **Upload package files.**
4. In the **Upload dependency files** modal, select **Choose files** and browse to the location of the compressed missing package files on your device.
5. Select **Upload**.
6. AWS Transform validates the files you uploaded. During validation, you cannot make any updates. AWS Transform reports the validation status above the *Missing package dependencies* table.
7. AWS Transform also updates the status columns in the *Missing package dependencies* table from *Missing* to *Resolved*. If a package fails validation, its status becomes *Invalid*. For invalid files do the following:

a. In the *Missing package dependencies* table, select the invalid package using the check box.

b. Select **Remove uploaded file**.

c. This changes its status back to *Missing*.

8. After you have uploaded the missing packages and resolved the package dependencies, select **Proceed to review**.

   If you select **Proceed to review** without resolving the missing package dependencies, AWS Transform asks if you would like to start the transformation job without the missing packages. If you select **Ignore the missing package dependencies**, AWS Transform will use assembly references to transform the code. Proceeding with this action can affect related resources.

Folder Structure Requirements

For NuGet uploads AWS Transform requires either:

- The `.nuspec` file to be placed at the root level for manually uploaded packages, or
- Use of the provided PowerShell script to generate the correct structure

Version matching requirements

- Framework status: AWS Transform attempts to match the version of the dependency specified in the `.csproj` file (source code) with the version of the uploaded dependency. An exact match results in a **Success** status.
- Core status: AWS Transform expects a core version specification in the dependencies section of the uploaded `.nuspec` file. If one is found then the core status is **Success**.

  Example format:

```
<dependencies>
    <group targetFramework="net8.0" />
</dependencies>
```

  Common versions include:

  - .NET 5.0 (net5.0)
  - .NET 6.0 (net6.0)
  - .NET 7.0 (net7.0)

- .NET 8.0 (net8.0)

Review and customize transformation plan

In the *Prepare for transformation* phase, after selecting repositories and resolving missing dependencies, AWS Transform will generate a modernization plan for review.

The web application uses a single global plan to modernize your selected repositories, which you can customize. The modernization plan provides details about the transformation. It includes these sections:

1. **Repository topology:** waves of repos in dependency order with project and dependency information.
2. **Feature & Package Migration Map:** package actions with before and after package names and versions.
3. **Transformation Strategy:** project type strategies, transformation execution order, risks and mitigations.

Discuss the plan with the agent for clarifications or to explore options.

Customize the plan

There are several ways to customize the transformation plan:

- **Chat**: Discuss the plan with the agent in chat and give it revised instructions (*Prepare for transformation* phase).
- **Edit plan**: Edit the transformation plan markdown in the web application (*Prepare for transformation* phase).
- **Upload customized plan**: Download the transformation plan markdown, review it, and upload a revised plan (*Prepare for transformation* phase).
- **Steering document**: Upload one or more steering documents to tell the agent about your organization's requirements and preferences (*Assessment* or *Prepare for transformation* phase).

After you are satisfied with the modernization plan and have resolved any missing dependencies, confirm the repositories to transform and transformation will begin.

Steering Documents

You can give the agent one or more steering documents to tailor the transformation plan. Steering documents can describe your organization's requirements and preferences such as those listed below. You can also upload application information such as a README file or specification to tell the agent more about the solution. To upload a steering document, drag and drop a file into the chat area and explain the document to give the agent context.

- Technology stack decisions
- Preferred .NET version(s)
- Preferred packages
- Front-end UI framework
- Back-end web service framework
- Organization conventions
- Examples of preferred coding patterns

A sample steering document follows.

```
# Organizational preferences

## Tech Stack for modernized .NET applications

We target the latest LTS .NET release (currently .NET 10)

For class libraries, we target .NET Standard 2.0 so they can be used by both .NET
 Framework and .NET applications.

We use the C# programming language. We convert legacy VB.NET code to C#.

We run our front-end web UIs on ASP.NET Core MVC projects with Razor Views.

We run our back-end web services on ASP.NET Web API projects.

## Package preferences

We use System.Text.Json for serialization.

We use the log4net .NET logger.
```

Transform your .NET code

In the *Transform* phase, the agent transforms your repositories.

Monitoring progress

You can view job progress in several ways:

- **Dashboard:** choose the Dashboard panel on the left side of the Job Plan window to see a dashboard showing repository status.
- **Chat:** the agent will let you know when repository transformations start and complete.
- **Transform panel:** Expand the Transform panel at left to view steps and progress for each repository.
- **Worklog tab:** The Worklog tab shows current agent activity in natural language.

Checkpoint reviews

In interactive mode, you can set checkpoints for some or all repositories. Checkpoints allow you to work alongside the agent, pausing for human review after a repository is transformed. You can review transformed code, discuss it with the agent, and ask for further changes.

Checkpoints are set on the Dashboard panel. Each repository has a checkpoint toggle. After configuring checkpoints, choose **Save checkpoints**.

The agent will pause for review of each repository that has a checkpoint set. While paused at a checkpoint, you can:

- Download the transformed code and review it in your IDE.
- Ask the agent to explain changes it made.
- Ask the agent to make further changes. You can ask for minor refinements, or a full retry of the transformation with revised instructions.

You can iterate at a checkpoint until satisfied.

Whether or not you use checkpoints, you also have an opportunity to review results and ask for changes when transformation completes.

Access transformed repositories

Repositories are transformed in parallel, grouped in a batch. You can access transformed repositories in several ways:

- **Source control:** After the agent transforms a repository, the modernized code is written to a new writable branch. If you uploaded a code zip file from an Amazon S3 bucket, a zip file of the transformed code is added to the bucket.
- **Download:** After the agent transforms a repository, the modernized code and artifacts are available for download from the dashboard or the **Artifacts** tab.
- **Beaming:** After the agent completes the transformation batch, you can beam transformed repositories to developers in the *Final summary* phase.

Final summary

When transformation completes, the *Final summary* phase shows a summary of transformation. Artifacts are available for download and you can transform further, hand off transformed repositories to developers, or complete the job.

Transformation artifacts

The following artifacts are available for download from the **Artifacts** tab:

- A zip file of the original and transformed source code.
- An HTML transformation report that documents what was changed and why.
- A Next Steps markdown file for hand-off to an AI code companion.

End of transformation actions

At the end of transformation, you can:

- **Beam to IDE:** hand off the transformed repositories to developers to review and finalize.
- **Ask questions:** ask the agent about changes it made.
- **Transform more repos:** ask the agent for additional changes and start a new batch.
- **Complete job:** mark the job as complete and close this session.

Beam a transformed repository to a developer

You can hand off transformed repositories to developers with a feature called **beaming**. This allows developers to locally build and review transformed applications.

Beaming a repository from the web application

The agent transforms your repositories in batches. When a batch completes, the agent will prompt whether you want to beam repositories to IDE.

1. In chat, choose **Beam to IDE**.
2. Open the link to the selection panel.
3. Select the repositories to beam and choose the **Beam to IDE** button.

Accessing a beamed repository from IDE

Developers follow these steps to retrieve a beamed repository:

1. In Visual Studio IDE, sign in to AWS Transform on the AWS Toolkit Getting Started page.
2. On the AWS Transform Dashboard, select the same workspace where the repository was beamed from.
3. Navigate to the dashboard's Beamed Repositories tab, which lists the beamed repositories.
4. Select a beamed repository and choose **Load**.
5. In the folder selection dialog that appears, select a file location for the repository files.
6. The repository solution opens in Visual Studio along with an in-progress AWS Transform job.

After a beamed repository is retrieved, it is no longer visible to other developers.

Modernizing .NET in the IDE

AWS Transform includes an interactive, agentic AI assistant in the AWS Toolkit for Visual Studio. It enables you to modernize applications through a conversational, step-by-step guided experience directly in your IDE. To transform a .NET solution or project, the AWS Transform agent analyzes your codebase, determines the necessary updates to port your application, and generates an assessment report and transformation plan before the transformation begins.

The Visual Studio IDE experience supports authentication by IAM Identity Center only. For more information, see Adding users in IAM Identity Center.

The IDE experience offers autonomous and interactive modes. Use autonomous mode for unattended transformation, suitable for overnight or long-running jobs. Use interactive mode to work alongside the agent in chat, with high visibility into transformation, a customizable transformation plan, and step-by-step review of each project. You can iterate to refine results or change direction until satisfied.

When a transformation job is active, you can interact with three windows:

- The **Chat** window is your primary interaction point with the transformation agent. Use chat to ask questions, discuss options, steer the transformation, and iterate.
- The **AWS Transform Job Plan** window shows progress steps and is where you can download or upload artifacts or view code change differences.
- The **Worklog** window shows a log of transformation activity in natural language.

AWS Transform performs four key tasks to port .NET applications to Linux:

- Upgrades .NET and language versions – Upgrades older .NET C# code to current .NET and language versions.
- Migrates from .NET Framework to cross-platform .NET – Migrates projects and packages from Windows dependent .NET Framework to cross-platform .NET compatible with Linux.
- Rewrites code for Linux compatibility – Refactors and rewrites deprecated and inefficient code components.
- Generates Linux compatibility recommendations – Assesses Linux readiness and provides next-step tasks for addressing issues.

For more information about how AWS Transform performs .NET transformations, see How AWS Transform modernizes .NET applications.

To modernize your .NET code using AWS Transform from Visual Studio IDE, refer to the following:

- Modernizing .NET using AWS Transform in Visual Studio
- How AWS Transform modernizes .NET applications
- Port and generate unit tests with AWS Transform
- Troubleshooting issues with .NET transformations in the IDE

Quotas

For AWS Transform .NET transformation quotas in the IDE, see Quotas for AWS Transform.

Modernizing .NET using AWS Transform in Visual Studio

Complete these steps to port a Windows-based .NET application to a Linux-compatible cross-platform .NET application with AWS Transform in Visual Studio 2026 or 2022.

Step 1: Prerequisites

Before you continue, make sure you've downloaded and installed the AWS Toolkit extension with Amazon Q from the Visual Studio Marketplace. Review the AWS Transform capabilities and limitations for .NET modernization to confirm that your code can be transformed.

Step 2: Authenticate in Visual Studio

To connect to your AWS accounts from the Toolkit for Visual Studio, open the Getting Started with the AWS Toolkit User Interface (connection UI):

1. From the Visual Studio main menu, navigate to **Extensions > AWS Toolkit > Getting Started** to open the **Getting Started: AWS Toolkit page**.
2. Select **AWS Transform** authentication.
3. Choose an existing profile or create a new one. When creating a new profile, enter a name for the profile and your AWS Transform start URL, which was emailed to you after you were added to IAM Identity Center by your administrator.
4. Choose **Connect**. Follow the prompts to sign in and grant access permissions through your web browser. In Visual Studio, you should see **Connected with IAM Identity Center**.
5. After signing in, you are taken to the **AWS Transform Dashboard**. From this landing page, you can do the following:

   - Select an AWS Transform profile (region)
   - Select or create a workspace
   - View status of transformation jobs

Step 3: Transform your application

To transform your .NET solution or project, complete the following procedure:

Start transformation

1. Open any C# or VB.NET based solution or project in Visual Studio that you want to transform.
2. From **Solution Explorer**, right click a solution or project you want to transform, and choose **Port with AWS Transform**.
3. The **Port with AWS Transform** dialog appears.

    a. In the **Choose a workspace** dropdown, create a workspace or select an existing one. Workspaces make your activity visible in the AWS Transform web application, where you can invite collaborators to view and chat about transformation jobs.

    b. In the **Choose a .NET target** dropdown menu, choose the .NET version you want to upgrade to.

    c. For **Transformation Mode**, select **Autonomous** for unattended transformation or **Interactive** for interactive transformation.

    d. Choose **Start** to begin the transformation.
4. Once the transformation job starts, the **AWS Transform Job Plan**, **Chat**, and **Worklog** windows appear.

Assessment and transformation plan

1. After AWS Transform analyzes your code, an assessment report and transformation plan are shown in chat. In interactive mode, chat will pause for you to review them. If you prefer not to review in chat, you can use these alternatives:

    - To view assessment or plan in the code editor, choose **Show assessment** or **Show transformation plan** in the AWS Transform Job Plan window.
    - To download assessment or plan, use the download icon in the Job Plan window.
2. Optionally modify the transformation plan. You can discuss the plan with chat, such as exploring other options. You make plan changes by giving instructions in chat, or by uploading a revised transformation plan using the download icon in the Job Plan window. You can freely add, remove, or modify steps. Chat will confirm its understanding of your instructions with a revised plan.
3. Optionally upload steering documents. In the chat window, use the upload icon to upload steering documents such as application specs, organizational tech stack preferences and conventions, or coding examples. After uploading a steering document, describe it in the chat window to give the agent context.

4. Once you are satisfied with the plan, give approval in chat and the transformation will begin.
5. In interactive mode, you can set checkpoints to stop after a project is transformed for review. The steps under **Transforming code** in the Job Plan window will have round checkpoint icons. All projects have checkpoints enabled by default. Use the check boxes to clear or set a checkpoint.

Transformation and checkpoint reviews

1. AWS Transform begins transforming your code. You can monitor progress of transformation steps in the AWS Transform Job Plan window. The Job Plan window displays the estimated maximum time remaining and your Job ID.
2. As projects are transformed, summaries are displayed in chat and code changes are applied.
3. In interactive mode, when a checkpoint is reached, the agent will pause so you can review the results. In the Job Plan window, you'll see a **View Results** button for the step it is paused on where you can review code differences. After reviewing the changes, you can ask for further changes in chat or make code changes yourself in Visual Studio.
4. You can iterate at the checkpoint as much as desired. When you're satisfied, instruct chat to continue, and it will move on to the next project.

End of transformation

1. After transforming your projects, the .NET agent will trigger a local build in Visual Studio, to confirm that the code builds successfully on your local machine. If there are errors, it will work to resolve them.
2. Once transformation is completed, there may be additional steps needed — such as transformation tasks AWS Transform could not perform, or changes needed for Linux compatibility. A next steps markdown file (*NextSteps.md*) is available for download which you can use as prompts to an AI code companion. Use the Job Plan window's download icon button to download *NextSteps.md*.
3. You can continue to interact in chat to ask questions or request additional changes.
4. Tell chat when you are ready to end your transformation job.

How AWS Transform modernizes .NET applications

Review the following sections for details about how .NET transformation with AWS Transform works.

Analyzing your application and generating a transformation plan

Before a transformation begins, AWS Transform builds your code locally to verify that it is buildable and configured correctly for transformation. If the code does not build, AWS Transform will prompt whether to continue with transformation. AWS Transform then uploads your code to a secure and encrypted build environment on AWS, analyzes your codebase, and determines the necessary updates to port your application

During this analysis, AWS Transform divides your .NET solution or project into code groups. A code group is a project and all its dependencies that together generate a buildable unit of code such as a dynamic link library (DLL) or an executable. Even if you didn't select all project dependencies to be transformed, AWS Transform determines the dependencies needed to build your selected projects and transforms them too, so that your transformed application will be buildable and ready for use.

After analyzing your code, AWS Transform generates a transformation plan that outlines the proposed changes that it will make, including a list of code groups and their dependencies that will be transformed.

Transforming your application

To start the transformation, AWS Transform builds your code again in the secure build environment to verify if it is buildable remotely. AWS Transform then begins porting your application. It works from the bottom up, starting with the lowest level dependency. If AWS Transform runs into an issue with porting a dependency, it stops the transformation and provides information about what caused the error.

The transformation includes the following updates to your application:

- Replacing outdated C# versions of code with Linux-compatible C# versions
- Upgrading .NET Framework to cross-platform .NET, including:
  - Identifying and iteratively replacing packages, libraries, and APIs
  - Upgrading and replacing NuGet packages and APIs
  - Transitioning to cross-platform runtime
  - Setting up middleware and updating runtime configurations

- Replacing private or third-party packages
- Handling IIS and WCF components
- Debugging build errors

- Rewriting code for Linux compatibility, including refactoring and rewriting deprecated and inefficient code to port existing code

Reviewing transformation report and accepting changes

After the transformation is complete, AWS Transform provides a transformation report with information about the proposed updates it made to your application, including the number of files changed, packages updated, and APIs changed. It flags any unsuccessful transformations, including affected files or portions of files and the errors encountered during an attempted build. You can also view a build summary with build logs to learn more about what changes were made.

You can review the proposed changes AWS Transform made in a diff view before accepting them as in-place updates to your files. After updating your files, your application is ready to run on cross-platform .NET.

You can download a *Next Steps* markdown file with prompts for continued transformation with AWS Transform or an AI code companion. To download the file, choose the **Download Next Steps** button at the top right of the report.

Port and generate unit tests with AWS Transform

AWS Transform can port and run existing unit tests, and can generate missing unit tests after transformation.

Port and run existing unit tests

If your original solution contains unit test projects (MSTest, NUnit, xUnit), AWS Transform ports them to the target .NET version. The agent then executes the ported unit tests and summarizes the results in the transformation report. You can also run the ported tests yourself using Visual Studio Test Explorer or the `dotnet test` command.

Unit test porting and execution are automatic.

Generate unit tests for transformed code

AWS Transform can optionally generate unit tests for your transformed code. The agent assesses existing unit test coverage (MSTest, NUnit, xUnit) and automatically generates missing unit tests.

The generated unit tests target the testable classes in your transformed .NET application, such as business logic and controllers.

> **Note**
>
> Unit tests are generated **after** modernization. They do not validate original code functionality in the modernized code.

This is an opt-in feature. You can enable unit test generation either at the start of a job or after transformation:

- **To generate unit tests during transformation:** Start a job. In the **Port solution with AWS Transform** dialog box, choose the **Generate unit tests** option. The generated unit tests are included in the local build verification step.
- **To generate unit tests after transformation:** Use interactive mode. After transformation, the agent prompts you whether to generate unit tests. These unit tests are not included in the local build verification step, which has already completed. Generating unit tests at this point allows you to make post-transformation code changes prior to generating unit tests.

The transformation report describes the resulting unit test coverage.

Troubleshooting issues with .NET transformations in the IDE

Use the following sections to troubleshoot common issues with .NET transformations in the IDE with AWS Transform.

How do I know if a job is progressing?

You should see regular activity in the Worklog window. If AWS Transform appears to be spending a long time on a step in the **AWS Transform Job Plan** window, you can check whether the job is still active in the output logs. If diagnostic messages are being generated, the job is still active.

To check the outputs, choose the **Output** tab in Visual Studio. In the **Show output from:** menu, choose **Amazon Q Language Client**.

The following screenshot shows an example of the outputs AWS Transform generates during a transformation.

Why are some projects not selected for transformation?

AWS Transform can only transform supported .NET project types. For a list of supported project types and other prerequisites for transforming your .NET projects, see Step 1: Prerequisites.

How can I get support if my project or solution isn't transforming?

If you aren't able to troubleshoot issues on your own, you can reach out to Support or your AWS account team to submit a support case.

To get support, provide the transformation job ID so AWS can investigate a failed job. To find a transformation job ID, choose the **Output** tab in Visual Studio. In the **Show output from:** menu, choose **Amazon Q Language Client**.

How can I prevent my firewall from interfering with transformation jobs?

If your organization uses a firewall, it might interfere with transformations in Visual Studio. You can temporarily disable security checks in Node.js to troubleshoot or test what is preventing the transformation from running.

The environment variable `NODE_TLS_REJECT_UNAUTHORIZED` controls important security checks. Setting `NODE_TLS_REJECT_UNAUTHORIZED` to "0" disables Node.js's rejection of unauthorized TLS/SSL certificates. This means:

- Self-signed certificates will be accepted
- Expired certificates will be allowed
- Certificates with mismatched hostnames will be permitted
- Any other certificate validation errors will be ignored

If your proxy uses a self-certificate, you can set the following environment variables instead of disabling `NODE_TLS_REJECT_UNAUTHORIZED`:

```
NODE_OPTIONS = –use-openssl-ca
```

```
NODE_EXTRA_CA_CERTS = Path/To/Corporate/Certs
```

Otherwise, you must specify the CA certs used by the proxy to disable `NODE_TLS_REJECT_UNAUTHORIZED`.

To disable NODE_TLS_REJECT_UNAUTHORIZED on Windows:

1. Open the Start menu and search for **Environment Variables**.
2. Choose **Edit the system environment variables**.
3. In the **System Properties** window, choose **Environment Variables**.
4. Under **System variables**, choose **New**.
5. Set **Variable name** to NODE_TLS_REJECT_UNAUTHORIZED and **Variable value** to 0.
6. Choose **OK** to save the changes.
7. Restart Visual Studio.

Porting UI layer code

AWS Transform can port the UI layer of ASP.NET websites to UI frameworks compatible with ASP.NET Core, which can run on Linux. The following transformations are available:

- MVC Razor UI porting
- Web Forms to Blazor UI porting

MVC Razor UI porting

ASP.NET MVC projects with Razor Views UI can be ported to MVC Razor Views on ASP.NET Core, remaining on the same UI framework. While Razor views are similar between ASP.NET and ASP.NET Core, there are differences that require porting code changes:

- ASP.NET MVC Razor Views use the System.Web.Mvc namespace and are closely tied to the .NET Framework.
- ASP.NET Core Razor Views are modular, built on Microsoft.AspNetCore.Mvc, with improved dependency injection.
- ASP.NET Core has new features including Tag Helpers and View Components.

AWS Transform detects and ports common incompatible components and constructs in Razor Views.

Web Forms to Blazor UI porting

ASP.NET Web Forms UI does not have a counterpart on ASP.NET Core. AWS Transform can port Web Forms UI layer code to Blazor Server on ASP.NET Core. UI layer code is rewritten for the target UI framework.

To enable or disable Web Forms to Blazor UI porting, check or clear the Transform UI ASP.NET Web Forms to Blazor job setting.

Transformation details

Your Web Forms project is converted to a Blazor project with server-side hosting. The following project and file changes are made during transformation:

File transformation mapping

From	To	Description
*.aspx, *.ascx	*.razor	.aspx pages and .ascx custom controls become .razor files
Web.config	appsettings.json	Web.config settings become appsettings.json settings
Global.asax	Program.cs	Global.asax code becomes Program.cs code
*.master	*layout.razor	Master files become layout.razor files

Post-transformation project structure

After transformation, the transformed files reside in the following project locations:

- Components folder: All razor components (*.razor) are placed in a Components folder created at the same level as the project's .csproj file.
- Pages mapping: Web Forms pages (*.aspx) are converted to Components/Pages folder structure.
- Controls mapping: Web Forms controls (*.ascx) are placed directly under the Components folder.

Limitations and unsupported features

The following features are not supported at present. If your workloads use these features, that code will need to be ported manually or with an AI code companion post-transformation.

- .svc files (service files)
- .ashx files (generic handler files)
- Service references & web references (reference.map, reference.cs, .wsdl files)
- Mixtures of .aspx (Web Forms) files and .cshtml (Razor) files in the same project
- Dependencies on project types not yet supported by AWS Transform for .NET
- Third-party vendor UI libraries/controls
- Web Site projects

Only Web Application projects (containing a .csproj file) are supported. To convert a Web Site project to a Web Application project, see Microsoft guidance Converting a Web Site Project to a Web Application Project.

Convert cloud-native code to AWS services

When moving cloud-native applications from other clouds to AWS, AWS Transform can help update code to use equivalent AWS services. For example, an Azure Functions serverless function that uses Azure Cosmos DB can be converted to an AWS Lambda function using Amazon DynamoDB. This feature converts application code only and does not handle migration of cloud infrastructure.

Cloud-native code conversion is automatic for eligible modern .NET applications. The agent shares details in the assessment report and modernization plan, which you can customize. To opt out of cloud-native code conversion and leave SDK code from other clouds unchanged, edit the modernization plan or give instructions in agent chat.

Code that cannot be converted to an equivalent on AWS is noted in TODO comments, stubbed code, or the Next Steps markdown file that AWS Transform generates.

Only the specific cloud services listed on this page are converted. To convert code for other cloud services, consider using AWS Transform custom or an AI code companion such as Kiro.

Azure to AWS conversions

AWS Transform can convert Azure application code for these services:

Service category	From	To	Code conversion details
Serverless functions	Azure Functions	AWS Lambda	Azure Functions are rewritten as AWS Lambda functions
Object storage	Azure Blob Storage	Amazon S3	`Azure.Storage.Blobs` SDK code → `Amazon.S3` SDK code
Message queuing	Azure Queue Storage	Amazon SQS	`Azure.Storage.Queues` SDK code → `Amazon.SQS` SDK code
NoSQL database*	Azure Cosmos DB	Amazon DynamoDB	`Microsoft.Azure.Cosmos` SDK code → `Amazon.DynamoDBv2` SDK code

* preview feature

Customizing code conversion

You can control the code conversion behavior by giving instructions in agent chat, editing the modernization plan, or providing a steering document. Examples of customizations you can ask for include the following:

- Opt out of some or all service code conversions.
- Specify variable names or naming conventions to use.
- Add specific comments or change the verbosity of comments.
- Redirect default conversion targets, such as converting Azure Blob Storage code to use Amazon DynamoDB instead of Amazon S3.

Transformation Report

After a transformation job finishes, transformation reports are available for both web console and IDE users.

- **Web console users:** a transformation report is available for each transformed repository. You can download transformation reports from the dashboard or the **Artifacts** tab.

- **IDE**: download the transformation report from the **AWS Transform Job Plan** window using the download icon.

The transformation report is an interactive HTML report. It contains in-depth information about the transformation, including the reasons for file changes and actionable error details. The information is arranged in the following hierarchy:

- Job Information
- For each solution:
  - Solution Transformation Summary
  - Transformation Overview
  - Projects
  - For each project:
    - Project Summary
    - Build Errors Summary
    - NuGet package changes
    - File Changes
    - Build Errors
  - Unit Test Results

For a walk-through of the transformation detail report sections with examples, refer to the blog post Announcing AWS Transform for .NET detailed transformation reports.

Hand-off to AI code companion

A common pattern is to transform with AWS Transform, then use an AI code companion to finalize the application such as Kiro. Use an AI code companion to:

- Perform post-transformation tasks that AWS Transform recommends
- Address any runtime errors you encounter as you validate the transformed application
- Address any styling issues you notice as you validate the transformed application
- Enhance the modernized application with additional features

Next Steps

After transformation is complete, additional steps may be necessary — such as transformation tasks AWS Transform could not perform, or changes needed for Linux compatibility. AWS Transform provides a next steps markdown file (NextSteps.md) for download which you can use as a prompt to an AI code companion.

Best practices for .NET transformations

Follow this guidance for better .NET modernization outcomes with AWS Transform. These best practices apply to the .NET conversational AI assistant, currently available for Visual Studio IDE.

Topics

- Before you transform
- Assessment and planning
- Transform interactively or autonomously
- Validation and finalization

Before you transform

Review the following recommendations before you start a .NET transformation job.

Use the .NET conversational AI assistant

Use the new .NET agent for best transformation results. The conversational AI assistant is available for Visual Studio through the AWS Toolkit for Visual Studio. It is also available for Kiro or other IDEs through the Kiro power for AWS Transform and AWS Transform MCP agents. The new .NET agent is not yet available in web console.

Use other tools if the objective is not .NET to .NET transformation

AWS Transform for .NET is appropriate for .NET Framework to .NET code modernization and .NET upgrades to later versions. It is not suitable for transformations that include non-.NET languages, such as Java, or non-.NET frameworks, such as React. It also does not provide application architectural refactoring. Use AWS Transform custom or Kiro for those kinds of modernization.

Use a stable edition of Visual Studio, preferably VS2026

Use a GA edition of Visual Studio rather than a preview or Insiders edition to avoid complications from preview software. The AWS Toolkit can be used with Visual Studio 2026 or 2022. VS2026 is recommended because it performs better with reduced memory consumption.

Use the latest AWS Toolkit for Visual Studio

Stay on the latest version of AWS Toolkit for Visual Studio for bug fixes and feature updates. You can check your toolkit version and upgrade at **Extensions** > **Manage Extensions** > **Installed**.

Work in Visual Studio IDE for medium or higher complexity projects

The IDE experience gives you the deepest view of transformation, where you can work alongside the .NET agent as it transforms projects. For medium, high, or critical complexity projects, use the IDE rather than web console.

Have a validation plan

Decide how you will validate your application for correctness after it is transformed. Post-transformation, you'll need to confirm that it behaves and looks correctly when run, and that security controls, business logic, and data storage are intact. Will validation be conducted manually and/or using test automation? If your solution contains unit tests, AWS Transform will port them for you and execute them, and results will appear in the transformation report.

Provide NuGet private package dependencies

If your solution depends on private packages, ensure they are available on your local machine so the IDE and AWS Transform can find them.

Specify package replacements or ask for agent recommendations

For packages that do not have modern .NET versions, you can either customize the transformation plan to specify your intended package substitutions, or ask the agent for recommendations.

Do not end your job until you are sure you do not want any more changes

After transformation completes, do not end your job right away in case you want additional changes. Transformation is not complete until you say it is. You can ask for changes up until you mark the job complete.

After transforming, hand off to an AI code companion to finalize your app

Expect to combine AWS Transform with an AI code companion such as Kiro. A common pattern is to first transform with AWS Transform for .NET, then hand off to an AI code companion to finalize the application. That finalization work might include identifying and correcting runtime errors, behavioral issues, and UX styling.

In the modernization journey, think of AWS Transform as the express travel that takes you a long way towards your objective, and an AI code companion as the local travel that takes you to your final destination.

To aid hand-off from AWS Transform to AI code companions, AWS Transform provides an HTML transformation report and a Next Steps markdown document at the end of transformation.

Assessment and planning

After you start a transformation job, AWS Transform analyzes your code and produces an assessment report and transformation plan. Use the following best practices to get the most from this phase.

Review assessment and ask questions to set your expectations

Review the assessment findings carefully to understand which projects will be challenging to transform and why. The agent presents projects with a complexity level (low, medium, high, critical) and explains why they have that rating.

Ask questions to understand more about challenging areas. For example:

- Ask the agent what its confidence level is in a project transformation or use case.
- Ask the agent what post-transformation tasks will likely be needed with an AI code companion.

Review the plan and discuss options with the agent

The agent provides a transformation plan for your review. You can discuss and modify this plan. Start by understanding your options. Ask the agent what alternatives there are to any part of the plan.

For example, the transformation plan might propose transforming WCF services to ASP.NET Web APIs. However, your organization prefers to stay with WCF. You ask the agent for options, and learn that an alternative is to transform them to CoreWCF.

Upload steering documents to tailor a plan for your organization

The original transformation plan is created without an understanding of your organization's requirements and preferences. You can inform the agent of organizational conventions, application specifications, preferred technology stack, and preferred coding pattern examples.

After uploading a steering document, explain it to give the agent context. With the additional info from steering documents, the agent can prepare a plan tailored to how your organization likes to work.

Freely customize the plan

The .NET agent is flexible and you can freely customize the transformation plan, within the domain of .NET to .NET transformation. You can ask for plan changes in chat, or if you prefer you can work with markdown files. Review and refine the plan until satisfied.

If a plan needs to be reviewed by multiple stakeholders, you can download the plan as a markdown file, share it, and later upload a customized plan.

For example, you review the transformation plan, and decide you want the class library projects transformed to .NET Standard instead of .NET 10. In chat you say, "Revise the plan to transform all class libraries to .NET Standard 2.0" and the agent presents you with a revised plan.

Transform interactively or autonomously

The Visual Studio IDE experience offers autonomous and interactive modes to control the degree of interactivity during transformation.

Use interactive and autonomous modes to control the degree of interactivity

You select the mode when starting a transformation, and you can change it anytime through a dropdown at the top of the chat window.

- Use **autonomous mode** for unattended transformation, suitable for overnight or long-running jobs. You still have the opportunity to review results and ask for changes at the end of the job.
- Use **interactive mode** to work alongside the agent in chat throughout the transformation process with a customizable transformation plan, high visibility into transformation details, and step-by-step review of each project. You can iterate to refine results or change direction until satisfied.

You can combine modes. For example, start in interactive mode to review and adjust the transformation plan, then switch to autonomous mode to transform the projects automatically.

Ask the agent to explain code changes

If you see code changes you do not understand, ask the agent to explain them.

Use checkpoints to stop at projects for review or changes

In interactive mode, after approving your plan, you can set or clear checkpoints for every project. When a project with a checkpoint has been transformed, the transformed code is applied to the project in IDE and the agent pauses for your review. You can review the results, and if you do not like them you can ask for more changes.

While stopped at a checkpoint, you can:

- Review results as presented in chat and ask questions in chat. For example: "Why did you change the cryptography API in file X?"
- View code changes, by choosing the **View Results** button for the project in the AWS Transform Job Plan window.
- Refine the transformation, by asking for additional changes. For example: "Change this to use logger X" or "Change this to use package X instead of package Y".
- Retry the transformation, doing it over with different instructions. For example: "Change this class library to target .NET Standard instead of .NET 10" or "Retry this transformation and change the target from Blazor to MVC Razor Pages".
- Make your own code changes in the IDE.
- Tell the agent to continue on to the next project checkpoint.

Iterate until satisfied

Modernization is inherently iterative, and you can expect to iterate with the agent. You can make changes at project checkpoints, and also at end of transformation. Iterate as needed until you are satisfied. Transformation is not complete until you say it is.

Code alongside the agent when stopped at a checkpoint

You can make your own code changes as you work with the agent. When stopped at a checkpoint, you can edit code files in Visual Studio. Your changes will be synced when you direct the agent to continue on.

Validation and finalization

After transformation completes, validate your transformed application and finalize it for production use.

Review the transformation report and Next Steps to understand what is done and what remains

Review the HTML transformation report to understand what has been done. Review the Next Steps markdown file to understand what tasks still remain. Ask the agent to clarify anything that is not clear.

Use an AI code companion to validate, debug, and finalize your application

Review your transformed application for correctness. Use an AI code companion such as Kiro to resolve issues.

Validate unit tests

If your solution contains unit test projects (NUnit, xUnit, or MSTest), AWS Transform will port them to modern .NET and run the tests for you. Check the HTML transformation report for unit test results. Correct failing tests with an AI code companion.

Validate application functionality

Run your application and confirm its functionality matches the original. Note functional issues and runtime errors. Correct with an AI code companion.

Validate application UI

Run your application and confirm it looks and behaves like the original including navigation and routing. If your application has no styling, it might be that .css files are not in the correct location. Correct with an AI code companion.

SQL Server modernization

AWS Transform for SQL Server Modernization is an AI-powered service that automates the full-stack modernization of Microsoft SQL Server databases and their associated .NET applications to Amazon Aurora PostgreSQL. AWS Transform now converts SQL Server storage objects, powered by AWS DMS, and code objects (stored procedures) using an agentic, interactive experience.

The service orchestrates the entire migration journey from schema conversion, data migration and modifying application code to match the new target PostgreSQL, making your teams more productive by automating complex and labor-intensive tasks.

Supported regions

AWS Transform for SQL Server is available in US East (N. Virginia) - us-east-1

Cross-Region Usage: For databases in unsupported regions, you can clone the database to a supported region for transformation, then deploy the results back to your target region.

Capabilities and key features

Database transformation

- **Schema conversion:** Automatically converts SQL Server schemas to Aurora PostgreSQL, including tables, views, indexes, constraints, and relationships
- **Stored procedure transformation:** Converts T-SQL stored procedures to PL/pgSQL with AI-enhanced accuracy
- **Data migration:** Migrates data with integrity validation using AWS Database Migration Service (DMS)
- **Database objects:** Supports triggers, functions, views, computed columns, and identity columns
- **Validation:** Automated data integrity verification and referential integrity checks

Application transformation

- **Entity Framework transformation:** Updates Entity Framework 6.3-6.5 and EF Core 1.0-10.0 configurations for PostgreSQL
- **ADO.NET transformation:** Converts ADO.NET data access code from SQL Server to PostgreSQL providers.
- **Connection string updates:** Automatically updates all database connection strings to the new target PostgreSQL database
- **Database provider changes:** Replaces SQL Server providers with Npgsql (PostgreSQL provider)
- **ORM configuration updates:** Modifies data type mappings, identity columns, and database-specific configurations

Orchestration & validation

- **Wave-based modernization:** Organizes large estates into logical migration phases
- **Dependency mapping:** Identifies relationships between applications and databases
- **Human-in-the-loop (HITL) checkpoints:** Provides review and approval gates at critical stages
- **Automated validation:** Tests schema compatibility, data integrity, and application functionality
- **CI/CD integration:** Integrates with existing development pipelines

Deployment

- **Amazon ECS and Amazon EC2 deployment:** Automated containerized deployment with auto-scaling support
- **Infrastructure-as-code generation:** Creates CloudFormation or AWS CDK templates
- **Automated deployment validation:** Verifies successful deployment with health checks
- **Rollback capabilities:** Supports rollback procedures if issues arise

Supported versions and project types

SQL Server versions

AWS Transform supports the following SQL Server versions:

SQL Server Version	Support Status
SQL Server 2022	Supported
SQL Server 2019	Supported
SQL Server 2017	Supported
SQL Server 2016	Supported
SQL Server 2014	Supported
SQL Server 2012	Supported
SQL Server 2008 R2	Supported

Note

All SQL Server editions are supported (Express, Standard, Enterprise). SQL Server can be hosted on AWS (Amazon RDS for SQL Server or SQL Server on Amazon EC2) or hosted outside of AWS.

.NET versions

.NET Version	Support Status
.NET 10	Supported
.NET 8	Supported
.NET 7	Supported
.NET 6 (Core)	Supported
.NET Framework 4.x and earlier	Not Supported

Important

Legacy .NET Framework 4.x and earlier versions are not supported. If your application uses .NET Framework, you must first upgrade to .NET Core 6+ using AWS Transform for .NET modernization before using SQL Server transformation capabilities.

Entity Framework versions

Framework	Supported Versions
Entity Framework 6	6.3, 6.4, 6.5
Entity Framework Core	1.0 through 10.0
ADO.NET	All versions (GA)

Source code repositories

AWS Transform supports the following source code platforms:

- GitHub and GitHub Enterprise Server
- GitLab.com and GitLab Self-Managed
- Bitbucket Cloud and Bitbucket Data Center
- Azure DevOps and Azure DevOps Server
- Amazon S3

Target database

AWS Transform targets Amazon Aurora PostgreSQL (PostgreSQL 15+ compatible) with support for the latest Aurora features and optimizations.

Technical requirements

Database requirements

- Microsoft SQL Server version 2008 R2 through 2022
- SQL Server hosted on AWS (Amazon RDS for SQL Server or SQL Server on Amazon EC2) or hosted outside of AWS
- For AWS-hosted databases, the database and AWS Transform must be in the same AWS Region.
- For databases hosted outside of AWS, network connectivity to the AWS Transform service is required.
- Database user with VIEW DEFINITION and VIEW DATABASE STATE permissions
- Database passwords using printable ASCII characters only (excluding '/', '@', '"', and spaces)
- VPC containing the source SQL Server must have subnets in at least 2 different Availability Zones (required for DMS replication subnet groups)

Application requirements

- .NET 6, 7, 8, or 10 applications
- Entity Framework 6.3-6.5 or Entity Framework Core 1.0-10.0, or ADO.NET

- Database connections discoverable in source code
- Applications successfully build and run
- Source code in supported repository platforms

AWS account requirements

- AWS account with administrator access
- IAM Identity Center enabled
- Required service roles created (see setup instructions below)
- VPC with appropriate network configuration

Data processing and storage

Processing location

- Schema processing occurs in a DMS instance within your VPC
- Data migration is optional and can be excluded if required
- Transformation artifacts are stored in the AWS Transform service region

Stored artifacts

The following items are stored in the service region:

- Agent logs
- Assessment results
- SQL schema files
- DMS output artifacts

> **Important**
>
> **Important for Data Residency:** Even when data migration is opted out, metadata and processing artifacts are stored in the service region. This is important for organizations with strict data residency requirements.

Artifact management

- Customer option for encryption using your own KMS keys
- Defined TTL (time-to-live) period for all artifacts
- Artifacts can be downloaded for offline storage

Getting started with a modernization job

After you create a SQL Server modernization job in AWS Transform and the agent starts, the agent asks how you want AWS Transform to access your source SQL Server schema. Your response to this first message determines which workflow runs for the remainder of the job. Choose one of the following:

- **Connect to database** — AWS Transform connects directly to your running SQL Server. Choose this when a live network connection to the database is available and you want AWS Transform to migrate real data.
- **Upload DDL files** — You extract your schema definition (DDL) files with a provided script and upload them to AWS Transform. Choose this when a direct database connection is not available or not desired. No live connection is required, and real data is not migrated.

Both workflows target Amazon Aurora PostgreSQL and share the same supported versions, limitations, and data-handling behavior described elsewhere on this page. Choose the workflow that matches your environment.

> **Note**
>
> When you choose **Connect to database**, the online workflow starts. When you choose **Upload DDL files**, the offline workflow starts.

Connect to database (online workflow)

In this workflow, AWS Transform connects directly to your source SQL Server and uses AWS Database Migration Service (DMS) to convert the schema and migrate data to Amazon Aurora PostgreSQL. This is the standard end-to-end workflow when a live database connection is available.

The workflow proceeds through the following phases:

1. **Prerequisites** — AWS Transform confirms the DMS prerequisites are in place (IAM roles, secrets, and network access to the source database).
2. **Connect** — You provide connection details, and AWS Transform establishes a connection to the source database.
3. **Validate** — AWS Transform validates connectivity and permissions.
4. **Discovery and assessment** — AWS Transform discovers the databases on the server, you choose which to modernize, and AWS Transform assesses them.
5. **Per-database transformation** — For each selected database, AWS Transform provisions an Aurora PostgreSQL target, converts the schema, and migrates data. Choose to migrate real data, generate synthetic data, or skip data migration.
6. **Code assessment and transformation** (optional) — If a source code repository is connected, AWS Transform assesses and transforms the associated .NET application.
7. **Deploy** (optional) — AWS Transform deploys the transformed application.

> **Note**
>
> This workflow migrates your real data using AWS DMS. Ensure the network and IAM prerequisites are satisfied before you begin. See the technical requirements in the preceding section.

SQL Server requirements

Database requirements

Externally hosted databases

Databases hosted outside of AWS are supported. To use an externally hosted database, you must ensure network connectivity between the database and the AWS Transform service.

SSIS workloads

- **Limitation:** SQL Server Integration Services (SSIS) transformation is not supported.
- **Workaround:** Use AWS Glue or other ETL services for SSIS migration.

Linked servers

- **Limitation:** External database connections via linked servers require human configuration.
- **Workaround:** Reconfigure linked servers after migration or use alternative approaches such as database links or application-level integration.

Complex T-SQL patterns

- **Limitation:** Some advanced T-SQL patterns may need human intervention.
- **Workaround:** Use Amazon Kiro CLI for complex SQL conversions. AWS Transform flags these for review during the transformation process.

Application requirements

Legacy .NET Framework

- **Limitation:** .NET Framework 4.x and earlier versions are not supported.
- **Workaround:** Use AWS Transform for .NET to upgrade to .NET Core 6+ first, then use SQL Server transformation.

Entity Framework versions

- **Limitation:** Only Entity Framework 6.3-6.5 and EF Core 1.0-10.0 are supported.
- **Workaround:** Upgrade to a supported Entity Framework version before transformation.

VB.NET applications

- **Limitation:** VB.NET is not supported.
- **Workaround:** Convert to C# or use AWS Transform custom to convert from VB.NET to C#.

Cross-database dependencies

- **Limitation:** Challenges when database schemas interact across multiple databases.
- **Workaround:** Review and refactor cross-database queries before migration. Consider consolidating databases or using PostgreSQL schemas.
- **Impact:** May require human intervention for complex cross-database scenarios.

Repository-database coupling

- **Limitation:** Challenges when a single repository serves multiple databases.
- **Workaround:** Consider repository restructuring or phased migration approach.
- **Impact:** May require additional planning for wave-based migrations.

Infrastructure requirements

Single account/region per job

- **Limitation:** Each transformation job targets one AWS account and region.
- **Workaround:** Create multiple transformation jobs for multi-account or multi-region deployments.

Deployment targets

- **Limitation:** Amazon ECS and Amazon EC2 deployments are supported.

Repository requirements

Private NuGet packages

- **Limitation:** Private NuGet packages require additional configuration.
- **Workaround:** Configure private NuGet feeds in transformation settings before starting the job.

SQL Server setup

Perform these steps in your SQL Server environment to enable AWS Transform modernization.

Database setup and configuration

Step 1: Create database user with required permissions

Create a dedicated database user for AWS Transform with the necessary permissions. If you already have a DMS Schema Conversion user, you can reuse it.

Connect to your SQL Server instance and run the following commands:

```
-- Create the login in master database
```

```
USE master;
CREATE LOGIN [atx_user] WITH PASSWORD = 'YourStrongPassword123!';

-- Switch to your application database
USE [YourDatabaseName];
CREATE USER [atx_user] FOR LOGIN [atx_user];

-- Grant required permissions
GRANT VIEW DEFINITION TO [atx_user];
GRANT VIEW DATABASE STATE TO [atx_user];
ALTER ROLE [db_datareader] ADD MEMBER [atx_user];

-- Grant master database permissions
USE master;
GRANT VIEW SERVER STATE TO [atx_user];
GRANT VIEW ANY DEFINITION TO [atx_user];
```

Note

Repeat the database-specific commands (USE, CREATE USER, GRANT) for each database you want to modernize.

The db_datareader role is only necessary for data migration, not for schema conversion alone.

- The db_datareader role grants read access to all tables in the database
- This role is required ONLY when performing data migration.
- For schema conversion only (without data migration), the db_datareader role is NOT required
- The other permissions (VIEW DEFINITION, VIEW DATABASE STATE, etc.) are sufficient for schema conversion

Step 2: Store credentials in AWS Secrets Manager

Store your database credentials securely in AWS Secrets Manager. Skip this step if you already have a secret created for DMS.

1. Navigate to AWS Secrets Manager in the console
2. Choose **Store a new secret**
3. Configure the secret:

- **Secret type:** Credentials for other database
- **Database**: Microsoft SQL Server
- **Username**: atx_user (or your chosen username)
- **Password**: The password you created
- **Server name**: Your SQL Server endpoint
- **Database name**: Your database name
- **Port**: 1433 (or your custom port)

4. Choose **Next**
5. Enter secret name: atx-db-modernization-sqlserver
6. Add required tags (these tags are mandatory):
   - Key: Project, Value: atx-db-modernization
   - Key: Owner, Value: database-connector
7. Choose **Next** through remaining screens
8. Choose **Store**
9. Note the Secret ARN for use in the next step

> **Important**
>
> Database passwords must use printable ASCII characters only, excluding '/', '@', '"', and spaces. Secrets scheduled for deletion can cause transformation failures.

Step 3: Create required DMS roles

AWS Transform requires specific IAM roles for DMS operations. Deploy these roles using the CloudFormation template below.

> **Note**
>
> If your AWS account already has existing DMS-related roles, modify this template to reuse those resources rather than creating duplicates.

Create a file named dms-roles.yaml with the following content:

```
AWSTemplateFormatVersion: '2010-09-09'
Description: 'DMS Service Roles for AWS Transform SQL Server Modernization'

Resources:
  DMSCloudWatchLogsRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: dms-cloudwatch-logs-role
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - dms.amazonaws.com
                - schema-conversion.dms.amazonaws.com
            Action: sts:AssumeRole
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonDMSCloudWatchLogsRole

  DMSS3AccessRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: dms-s3-access-role
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - dms.amazonaws.com
                - schema-conversion.dms.amazonaws.com
            Action: sts:AssumeRole
      Policies:
        - PolicyName: S3TaggedAccess
          PolicyDocument:
            Version: '2012-10-17'
            Statement:
              - Effect: Allow
                Action:
                  - s3:GetBucketLocation
                  - s3:GetBucketVersioning
                  - s3:PutObject
```

```
                - s3:PutBucketVersioning
                - s3:GetObject
                - s3:GetObjectVersion
                - s3:ListBucket
                - s3:DeleteObject
              Resource: arn:aws:s3:::atx-db-modernization-*
              Condition:
                StringEquals:
                  aws:ResourceAccount: !Ref AWS::AccountId

  DMSSecretsManagerRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: dms-secrets-manager-role
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - dms.amazonaws.com
                - schema-conversion.dms.amazonaws.com
            Action: sts:AssumeRole
      Policies:
        - PolicyName: SecretsManagerTaggedAccess
          PolicyDocument:
            Version: '2012-10-17'
            Statement:
              - Effect: Allow
                Action:
                  - secretsmanager:GetSecretValue
                  - secretsmanager:DescribeSecret
                Resource: '*'
                Condition:
                  StringEquals:
                    secretsmanager:ResourceTag/Project: atx-db-modernization
                    secretsmanager:ResourceTag/Owner: database-connector

  DMSVPCRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: dms-vpc-role
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
```

```
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - dms.amazonaws.com
                - schema-conversion.dms.amazonaws.com
            Action: sts:AssumeRole
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonDMSVPCManagementRole

  DMSServerlessRole:
    Type: AWS::IAM::ServiceLinkedRole
    Properties:
      AWSServiceName: dms.amazonaws.com
      Description: 'Service Linked Role for AWS DMS Serverless'

Outputs:
  DMSCloudWatchLogsRoleArn:
    Description: ARN of the DMS CloudWatch Logs Role
    Value: !GetAtt DMSCloudWatchLogsRole.Arn
  DMSS3AccessRoleArn:
    Description: ARN of the DMS S3 Access Role
    Value: !GetAtt DMSS3AccessRole.Arn
  DMSSecretsManagerRoleArn:
    Description: ARN of the DMS Secrets Manager Role
    Value: !GetAtt DMSSecretsManagerRole.Arn
  DMSVPCRoleArn:
    Description: ARN of the DMS VPC Role
    Value: !GetAtt DMSVPCRole.Arn
    Export:
      Name: !Sub ${AWS::StackName}-VPCRole

  DMSServerlessRoleArn:
    Description: ARN of the DMS Serverless Role
    Value: !Sub 'arn:aws:iam::${AWS::AccountId}:role/aws-service-role/
dms.amazonaws.com/AWSServiceRoleForDMSServerless'
    Export:
      Name: !Sub ${AWS::StackName}-ServerlessRole
```

Deploy the CloudFormation stack using AWS CLI:

```
aws cloudformation create-stack \
  --stack-name dms-roles \
```

```
  --template-body file://dms-roles.yaml \
  --capabilities CAPABILITY_NAMED_IAM \
  --region us-east-1
```

Or deploy using the AWS Console:

1. Navigate to CloudFormation in the AWS Console
2. Choose **Create stack**
3. Select **Upload a template file**
4. Upload the dms-roles.yaml file
5. Enter stack name: dms-roles
6. Acknowledge IAM capabilities
7. Choose **Create stack**

Step 4: Configure network security

Ensure proper network connectivity between AWS Transform, your SQL Server database, and other AWS services.

Security group configuration (recommended approach)

Recommended Approach: Use security group-based access control rather than IP-based rules. This provides better security, easier management, and works seamlessly with AWS Transform's architecture.

Why Security Group-Based Access Control?

- DMS Schema Conversion creates Elastic Network Interfaces (ENIs) within your VPC
- Your databases do not need to be publicly accessible
- AWS Transform does not expose private IP addresses, making IP-based rules complex
- Security group references provide dynamic, automatic updates as resources scale

Configure your SQL Server security group

When you configure the DMS Schema Conversion Instance Profile in AWS Transform, you specify a Security Group for the DMS SC instance. Your database security group should allow inbound traffic from this DMS SC security group.

Step-by-step configuration:

1. Identify the DMS Schema Conversion Security Group:
   - This is specified when creating the Instance Profile in AWS Transform
   - Note the Security Group ID (e.g., sg-0123456789abcdef0)
2. Update your SQL Server Security Group inbound rules:
   - **Type**: Custom TCP
   - **Port**: 1433 (or your custom SQL Server port)
   - **Source**: The DMS Schema Conversion Security Group ID
   - **Description**: "Allow DMS Schema Conversion access"
3. For Aurora PostgreSQL target (after creation):
   - **Type**: PostgreSQL
   - **Port**: 5432 (or your custom PostgreSQL port)
   - **Source**: The DMS Schema Conversion Security Group ID
   - **Description**: "Allow DMS Schema Conversion access"

> **Important**
>
> **Important for Least Privileged Security Models:** If your organization uses a "least privileged" security model that blocks all traffic by default, you must explicitly allow inbound traffic from the DMS Schema Conversion Security Group to your database port. Do not open port 1433 to all sources or IP ranges.

Required AWS service connectivity

Ensure your VPC can communicate with:

- AWS Transform service endpoints
- AWS DMS endpoints
- Aurora PostgreSQL endpoints
- S3 endpoints for artifact storage
- AWS Secrets Manager endpoints
- AWS CodeConnections endpoints

VPC Endpoints: For private networks, configure VPC endpoints for required AWS services to avoid internet gateway dependencies.

Requirements for externally hosted databases

If your SQL Server database is hosted outside of AWS, ensure the following prerequisites are met and then complete the setup steps before you begin the modernization.

Prerequisites

- An AWS account with a VPC
- Network connectivity between the VPC and the external database. For information about configuring network connectivity, see Configuring network connectivity in the AWS DMS User Guide.

Setup steps

1. Create a secret in AWS Secrets Manager with the connection details for the external database. For more information, see the section called “Step 2: Store credentials in AWS Secrets Manager”.
2. When prompted, provide the VPC ID and security group ID for connecting to the external database. AWS Transform prompts you for this information because the database hostname in the secret cannot be resolved within the AWS account.

AWS Transform setup

Begin your modernization journey by accessing the AWS Transform console and initiating a new modernization project. This step establishes your workspace and sets up the foundational configuration for your full-stack transformation.

Actions to Complete:

- Navigate to AWS Transform service in the AWS Management Console
- Select **Microsoft Modernization** from the service options
- Choose **Create new modernization project**
- Provide project name and description
- Select your target AWS region (must match your SQL Server location)
- Review and confirm project settings

> **Note**
>
> **Best Practice:** Use descriptive project names that include the application name and target environment (e.g., 'CustomerPortal-Production-Modernization') to easily identify projects in multi-wave scenarios.

Create SQL Server modernization job

Connect database and source code repo

Configure database connector

Establish secure connectivity to your SQL Server databases by configuring database connectors. The connector performs environment analysis, dependency discovery, and enables AWS Transform to access your database schemas and metadata for assessment and conversion.

Actions to Complete:

- Select **Configure Database Connector** from the setup wizard
- Choose connection method: **New Connection** or **Existing Connection**
- Provide SQL Server connection details (endpoint, port, database names)
- Configure authentication and test connectivity
- Review discovered databases and confirm selection

> **Note**
>
> **Network Connectivity:** Ensure your SQL Server security groups and NACLs allow inbound connections from AWS Transform service endpoints.

Connect source code repository

Enable AWS Transform to access your .NET application source code. AWS Transform supports three methods for providing source code: a Personal Access Token (PAT) connector (recommended), AWS CodeConnections, or Amazon S3. This integration allows the service to analyze your

application code, identify database dependencies, and perform automated code transformations for PostgreSQL compatibility.

Actions to Complete:

- Navigate to **Connect Source Code Repository** section
- Choose your authentication method: **PAT Connector** (recommended), **AWS CodeConnections**, or **Amazon S3**
- Select repositories containing .NET applications
- Specify branch for analysis (typically main/master or development)
- Validate repository access and code structure

> **Note**
>
> **Repository Discovery:** AWS Transform automatically scans your repositories to identify .NET projects, Entity Framework configurations, database connection strings, and SQL dependencies.

> **Note**
>
> **Security Note:** AWS Transform only requires read access to your repositories and creates new feature branches for transformed code. Your main branch remains untouched.

Human in the loop

AWS Transform uses human-in-the-loop (HITL) mechanisms to ensure quality and allow you to review and approve critical transformation decisions. The following checkpoints require your attention:

Wave plan review and approval

What you review: The proposed migration waves, including which databases and applications are grouped together and the sequence of waves.

You can:

- Approve the wave plan
- Customize waves by moving databases between waves
- Modify wave sequence
- Split or merge waves

After approval, AWS Transform proceeds with schema conversion.

Schema conversion review

What you review: Converted database objects including tables, stored procedures, functions, and triggers. Action items highlight objects that require attention.

You can:

- Accept converted code
- Modify converted code
- Flag for human review after transformation
- View side-by-side comparison of original and converted code

What happens after approval: AWS Transform applies the schema to Aurora PostgreSQL and proceeds with data migration (if configured).

Application code review

What you review: All application code changes including Entity Framework configurations, connection strings, data access code, and stored procedure calls.

You can:

- Accept changes for each file
- Modify transformed code
- Reject changes (not recommended)
- Add comments for your team
- Download transformed code for local review

What happens after approval: AWS Transform commits the changes to a new branch in your repository and proceeds with validation.

Validation results review

What you review: Automated validation results including schema compatibility, data integrity checks, and application build status.

You can:

- Review passed tests
- Investigate failed tests
- Address warnings
- Proceed to deployment or return to fix issues

What happens after approval: AWS Transform prepares for deployment to your target environment.

Deployment approval

What you review: Deployment configuration including infrastructure-as-code templates, ECS service configuration, and deployment settings.

You can:

- Review and customize deployment settings
- Approve deployment to proceed
- Delay deployment for additional testing
- Download infrastructure code for review

After approval, AWS Transform deploys your modernized application and database to the target environment.

Assessment and wave planning

Set up landing zone

Configure the target infrastructure environment where your modernized applications will be deployed. The landing zone setup includes Aurora PostgreSQL provisioning, networking configuration, and reference deployment configurations.

Actions to Complete:

- Select **Set up Landing Zone** from the configuration menu
- Configure Aurora PostgreSQL target (version, instance class, Multi-AZ)
- Configure network and security settings (VPC, subnets, security groups)

> **Note**
>
> **Cost Consideration:** Aurora PostgreSQL instances incur ongoing costs. Consider using Aurora Serverless v2 for development/testing environments.

Assessment

Conduct comprehensive analysis of your database schemas, application code, and dependencies. The assessment phase provides transformation complexity ratings, identifies potential challenges, and generates detailed reports.

Actions to Complete:

- Select Repository Branch for code analysis
- Select Repositories and Databases for transformation
- Review Assessment Results and complexity ratings
- Generate Assessment Report with effort projections
- Export report for stakeholder review and approval

> **Note**
>
> **Assessment Insights:** The assessment provides detailed analysis of schema objects, stored procedures, application dependencies, and estimates the level of human intervention required.

Wave planning

Organize your modernization into manageable waves based on transformation complexity, business priorities, and application dependencies. AWS Transform generates intelligent wave recommendations that you can customize.

Actions to Complete:

- Review AI-Generated Wave Plan and recommendations
- Customize Wave Configuration based on business priorities
- Validate Dependencies between applications and databases
- Finalize Wave Plan using conversational interface
- Set wave execution priorities and assign responsibilities

> **Note**
>
> **AI-Powered Recommendations:** You can interact with the AI through natural language to adjust recommendations: 'Move CustomerDB to Wave 1' or 'Combine these two applications into the same wave.'

> **Note**
>
> **Best Practice:** Start with a pilot wave containing 1-2 low-complexity databases to establish processes and build team confidence.

Data migration for each wave

Schema conversion for each wave

Transform SQL Server database schemas to PostgreSQL-compatible equivalents using AI-enhanced conversion capabilities. This process converts tables, stored procedures, functions, triggers, and other database objects.

Actions to Complete:

- Provide Schema Conversion Targets and configure preferences
- Execute Automated Conversion using DMS Schema Conversion service
- Review Conversion Results and identify human intervention items
- Handle Manual Interventions through HITL prompts
- Validate human corrections and re-run conversion if needed

> **Note**
>
> **Common Manual Interventions:** Linked servers, user-defined types, advanced T-SQL patterns, and vendor-specific SQL features typically require human review and correction.

Data migration for each wave

Transfer data from SQL Server to Aurora PostgreSQL using AWS Database Migration Service (AWS DMS) integration. Choose between production data migration or synthetic data generation for testing purposes.

Actions to Complete:

- Choose **Production Data Migration** or **Synthetic Data Generation**
- Configure DMS replication instance for data transfer
- Monitor migration progress and handle data inconsistencies
- Validate data integrity and referential constraints
- Confirm successful completion before proceeding

> **Note**
>
> **Automated DMS Integration:** AWS Transform abstracts the complexity of DMS configuration, automatically handling endpoint creation, task configuration, and data type mappings.

> **Note**
>
> **Data Validation:** The service performs empirical testing by running identical queries against both source and target databases to ensure data consistency.

Code migration for each wave

Transform .NET application code to work with Aurora PostgreSQL, including Entity Framework configuration updates, connection string modifications, SQL query adaptations, and framework-specific adjustments.

Actions to Complete:

- Execute Automated Code Analysis for SQL Server dependencies
- Perform Framework Transformation (Entity Framework provider updates)
- Adapt SQL Query syntax to PostgreSQL-compatible SQL
- Configure Target Branch Management for transformed code
- Handle Manual Interventions for complex patterns

SQL Server modernization workflow

This section provides a step-by-step walkthrough of the complete SQL Server modernization process using AWS Transform.

Step 1: Create SQL Server modernization job

Begin your modernization journey by creating a new transformation job in the AWS Transform console.

1. Sign in to the AWS Transform console
2. Choose **Create modernization job**
3. Select **Windows modernization** job and then select **SQL Server modernization**
4. Enter job details:
   - **Job name**: Descriptive name for your project
   - **Description**: Optional description
   - **Target region**: AWS region for deployment
5. Choose **Create job**

> ⚠ **Important**
>
> Do not include personally identifiable information (PII) in your job name.

Step 2: Connect to SQL Server database

Connect AWS Transform to your SQL Server database to enable schema analysis and conversion.

Create a database connector

1. In your SQL Server modernization job, navigate to Connect to resources
2. Choose **Connect to SQL Server database**
3. Choose **Create new connector**
4. Enter the connector information:
   - C**onnector name**: Descriptive name
   - **AWS account ID**: Account where SQL Server is hosted
5. Once confirmed, you will receive a link for approval. Copy the approval link to get approval from your AWS admin for the account. Once they have approved, you can proceed to the next step.
6. After your administrator has approved the connector request, click Submit to proceed to the source code connection setup.

Step 3: Connect source code repository

AWS Transform needs access to your .NET application source code to analyze and transform the code that interacts with your SQL Server database. AWS Transform supports three methods for providing source code.

Choose your authentication method

Personal Access Token (PAT) connector (recommended)

Best for teams that need custom permission scopes, self-hosted provider support, or access to provider-specific APIs such as GitHub Secrets. You create a PAT in your source code provider with custom permissions, store it in AWS Secrets Manager, and AWS Transform retrieves it when needed. You are responsible for managing token rotation and expiration.

AWS CodeConnections

Best for teams that want automated credential management. AWS CodeConnections uses a managed provider integration that handles authentication through an OAuth 2.0 authorization flow. AWS manages the entire credential lifecycle, including automatic token refresh and rotation. No manual credential management is required.

Amazon S3

Upload your source code directly to an Amazon S3 bucket. AWS Transform accesses the code from the bucket during the transformation job.

Feature	PAT connector (recommended)	AWS CodeConnections
Credential management	Manual (customer-managed)	Automatic (AWS-managed)
Token lifecycle	Manual rotation required	Automatic refresh
Permission flexibility	Fully customizable scopes	Fixed permissions
Self-hosted provider support	Supported	Not available
Setup complexity	Moderate (manual token creation and storage)	Low (one-time authorization)
Token storage	Customer's AWS Secrets Manager	AWS-managed

Set up a PAT connector (recommended)

With a PAT connector, you create a Personal Access Token in your source code provider with custom permissions, store it securely in AWS Secrets Manager, and AWS Transform retrieves it when needed. You are responsible for managing the token lifecycle including rotation and expiration. AWS Transform automatically creates the necessary IAM role with permissions to access your secret.

PAT connector supports the following providers, including self-hosted and custom DNS/URL versions:

- GitHub and GitHub Enterprise Server
- GitLab.com and GitLab Self-Managed
- Bitbucket Cloud and Bitbucket Data Center
- Azure DevOps and Azure DevOps Server

Create a Personal Access Token

Create a PAT in your source code provider. The required permissions vary by provider. Choose the tab for your provider.

> **Important**
>
> Copy the token immediately after creation. You cannot view it again. Set the expiration for the duration of the transformation job. Do not set the expiration to never expire.

> **Warning**
>
> Never commit PAT tokens to code repositories or share them through insecure channels. Always store them in AWS Secrets Manager.

GitHub

Navigate to **Settings**, **Developer settings**, **Personal access tokens**, **Fine-grained tokens**. Select the repositories to transform and grant the following permissions.

Repository permissions

Permission	Access	Purpose
Contents	Read and write	Reads source code and writes transformed code back to repository
Metadata	Read-only	Accesses basic repository information

Organization permissions (required for organization repositories)

Permission	Access	Purpose
Members	Read-only	Lists organizations accessible to the token for repository discovery

GitLab

Navigate to **Edit Profile**, **Access Tokens**. Select the following scopes.

Scope	Purpose
`read_api`	Reads repository metadata, project information, user details, and lists groups and branches
`read_repository`	Reads source code files and repository structure for analysis
`write_repository`	Writes transformed code back to repository

Bitbucket

Navigate to **Account settings**, **Security**, **Create and manage API Tokens**. The required scopes depend on your token type.

Workspace/Repository Token (ATCT — Bearer auth, no username required)

Permission	Access	Purpose
Repositories	Read and write	Lists repos, reads branches, and writes transformed code via git push

Account API Token (ATAT — Basic auth with email) or App Password (ATBB — Basic auth with username)

Scope	Purpose
`read:account`	Identifies the authenticated user to resolve repository membership
`read:workspace:bitbucket`	Lists the workspaces the token can access so AWS Transform can enumerate their repositories. Not required if you specify a workspaces list in the secret.

Scope	Purpose
`read:repository:bitbucket`	Lists repositories and reads metadata and branch information
`write:repository:bitbucket`	Writes transformed code back to repository via git push

Azure DevOps

Navigate to **User settings**, **Personal access tokens**. Select **Custom defined** scopes. For organization scope, choose **All accessible organizations** (recommended) or specify a single organization.

Scope	Access	Purpose
Code	Read & write	Reads source code, lists repositories and branches, and writes transformed code back
User Profile	Read	Validates token access and discovers user identity for organization lookup
Member Entitlement Management	Read	Lists organizations accessible to the token for repository discovery

Store the PAT in AWS Secrets Manager

1. Open the AWS Secrets Manager console.
2. Choose **Store a new secret**.
3. For **Secret type**, choose **Other type of secret**.
4. Add key-value pairs based on your provider and hosting type:
   - **Cloud-hosted providers** — Add a key named `token` with your PAT as the value.
     - For Azure DevOps with a specific organization, also add a key named `organization` with your organization name.

- For Bitbucket app passwords (ATBB), also add a key named `username` with your Bitbucket username. For Bitbucket account API tokens (ATAT), add a key named `email` with your Bitbucket email address.

- **Self-hosted and custom DNS/URL providers** — Add the following keys: `host` (your server URL, for example `https://github.mycompany.com`), `provider_type` (`github`, `gitlab`, `bitbucket`, or `ado`), and `token` (your PAT).
  - For Azure DevOps with a specific organization, also add a key named `organization` with your organization name.
  - For Bitbucket app passwords (ATBB), also add a key named `username` with your Bitbucket username. For Bitbucket account API tokens (ATAT), add a key named `email` with your Bitbucket email address.

The following example shows how a secret looks in AWS Secrets Manager for a GitHub cloud-hosted provider:

```
{
  "token": "your-github-personal-access-token"
}
```

The following example shows a Bitbucket app password (ATBB):

```
{
  "token": "your-bitbucket-app-password",
  "username": "my-bitbucket-username"
}
```

The following example shows a self-hosted GitLab instance:

```
{
  "host": "https://gitlab.mycompany.com",
  "provider_type": "gitlab",
  "token": "your-gitlab-personal-access-token"
}
```

5. Choose **Next**.
6. Enter a secret name, for example `github-pat-myproject`.
7. (Optional) Select a customer-managed KMS key for encryption.

8. Complete the wizard and choose **Store**.
9. Copy the Secret ARN. You need this value when you configure the AWS Transform job.

If you use a customer-managed KMS key to encrypt your secret (instead of the default AWS-managed key), you must update the KMS key policy to allow AWS Transform to decrypt the secret. Add the following statement to your customer-managed KMS key policy:

```
{
  "Sid": "Allow AWS Transform to decrypt secrets",
  "Effect": "Allow",
  "Principal": {
    "Service": "transform.amazonaws.com"
  },
  "Action": [
    "kms:Decrypt",
    "kms:DescribeKey"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:ViaService": "secretsmanager.REGION.amazonaws.com",
      "kms:EncryptionContext:SecretARN": "YOUR-SECRET-ARN"
    }
  }
}
```

Replace *REGION* with your AWS Region (for example, `us-east-1`) and *YOUR-SECRET-ARN* with the ARN of your secret. The `kms:ViaService` condition ensures the KMS key can only be used through the AWS Secrets Manager service. The `kms:EncryptionContext:SecretARN` condition restricts decryption to your specific secret.

To update your KMS key policy:

1. Open the AWS KMS console at `https://console.aws.amazon.com/kms`.
2. In the navigation pane, choose **Customer managed keys**.
3. Select your KMS key.
4. On the **Key policy** tab, choose **Edit**.
5. Add the policy statement to the existing policy.
6. Choose **Save changes**.

> **Note**
>
> If you use the default AWS-managed key (`aws/secretsmanager`), you do not need to modify any KMS key policy.

Configure the AWS Transform job

1. In your AWS Transform job, navigate to **Connect to resources**.
2. Choose **Connect source code repository**.
3. Select **PAT Connector** as the authentication method.
4. Enter the Secret ARN from Step 2.
5. (Optional) Enter the KMS Key ARN if you used a customer-managed KMS key.
6. Select your repository and branch.
7. Choose **Continue**.

AWS Transform automatically creates an IAM role with the permissions required to access your secret.

Token rotation and maintenance

You are responsible for rotating PAT tokens before they expire. To rotate a token:

1. Generate a new PAT in your source code provider with the same permissions.
2. Update the secret value in AWS Secrets Manager.
3. Verify that your AWS Transform job can access the repository with the new token.
4. Revoke the old PAT in your source code provider.

Troubleshoot PAT connector issues

Access Denied — Invalid PAT

Verify that the PAT has not expired. Confirm that the PAT has the required scopes for your provider. Check that the PAT is correctly stored in AWS Secrets Manager.

Cannot retrieve secret

Verify that the Secret ARN is correct. Check the job logs to confirm that AWS Transform created the IAM role. If you are using a customer-managed KMS key, verify the key policy.

Insufficient permissions

The PAT might lack the required scopes for the operation. Regenerate the PAT with the required scopes and update the secret value in AWS Secrets Manager.

Set up AWS CodeConnections

AWS CodeConnections uses a managed provider integration that automatically retrieves temporary OAuth credentials through an OAuth 2.0 authorization flow. Permissions are configured in the provider app and managed entirely by AWS. You authorize the app once, and AWS handles all credential management.

1. In your SQL Server modernization job, navigate to **Connect to resources**.
2. Choose **Connect source code repository**.
3. If you don't have an existing connection, choose **Create connection**.
4. Select your repository provider:
   - GitHub / GitHub Enterprise
   - GitLab.com
   - Bitbucket Cloud
   - Azure Repositories
5. Follow the authorization flow for your provider.
6. After authorization, choose **Connect**.

Select your repository and branch

1. Select your repository from the list.
2. Choose the branch you want to transform (typically main, master, or develop).
3. (Optional) Specify a subdirectory if your .NET application is not in the repository root.
4. Choose **Continue**.

> **Note**
>
> AWS Transform creates a new branch for the transformed code. You can review and merge the changes through your normal code review process.

Repository access approval

For GitHub and some other platforms, the repository administrator must approve the connection request:

1. AWS Transform displays a verification link.
2. Share this link with your repository administrator.
3. The administrator reviews and approves the request in their repository settings.
4. After the administrator approves the request, the connection status changes to *Approved*.

> **Important**
>
> The approval process can take time depending on your organization's policies. Plan accordingly.

Step 4: Create deployment connector (optional)

If you want to deploy the transformed applications into your AWS account, you have the option to select a deployment connector.

Set up deployment connector

1. Select **Yes** if you want to deploy your applications. Selecting **No** will skip this step.
2. Add your AWS account where you want to deploy the transformed applications.
3. Add a name that helps you remember the connector easily
4. Submit the connector request for approval.

Deployment connector approval

Your AWS account administrator must approve the connection request for deployment connector.

1. AWS Transform displays a verification link
2. Share this link with your AWS account administrator
3. The administrator reviews and approves the request in their repository settings
4. Once approved, the connection status changes to *Approved*

> **Important**
>
> The approval process can take time depending on your organization's policies. Plan accordingly.

Step 5: Confirm your resources

After connecting to your database and repository, AWS Transform verifies that all required resources are accessible and ready for transformation.

What AWS Transform verifies

- **Database connectivity:** Connection is active, user has required permissions, databases are accessible, version is supported
- **Repository access:** Repository is accessible, branch exists, .NET project files detected, database connections discoverable
- **Environment readiness:** VPC configuration supports DMS, required AWS service roles exist, network connectivity established, region compatibility confirmed

Review the pre-flight checklist

1. Navigate to **Confirm your resources** in the job plan
2. Review the checklist items:
   - # Database connection verified
   - # Repository access confirmed
   - # .NET version supported
   - # Entity Framework or ADO.NET detected
   - # Network configuration valid
   - # Required permissions granted

3. If all items show as complete, choose Continue
4. If any items show warnings or errors, address them before proceeding

Step 6: Discovery and assessment

AWS Transform analyzes your SQL Server database and .NET application to understand the scope and complexity of the modernization.

What gets discovered

- **Database objects:** Tables, views, indexes, stored procedures, functions, triggers, constraints, data types, computed columns, identity columns, foreign key relationships
- **Application code:** .NET project structure, Entity Framework models and configurations, ADO.NET data access code, database connection strings, stored procedure calls, SQL queries in code
- **Dependencies:** Which applications use which databases, cross-database dependencies, shared stored procedures, common data access patterns

Discovery process

- AWS Transform begins discovery automatically after resource confirmation
- Discovery typically takes 5-15 minutes depending on database size and application complexity
- Monitor progress in the worklog
- AWS Transform displays real-time updates as objects are discovered

Review discovery results

After discovery completes, navigate to **Discovery and assessment** to review:

Database Analysis:

- **Object count**: Number of tables, views, stored procedures, functions, triggers
- **Complexity score**: Assessment of transformation complexity (Low, Medium, High)
- **Action items**: Objects that may require human attention
- **Supported features**: Database features that will convert automatically
- **Unsupported features**: Features that require workarounds

Application Analysis:

- **Project type**: ASP.NET Core, Console App, Class Library, etc.
- **.NET version**: Detected .NET Core version
- **Data access framework**: Entity Framework version or ADO.NET
- **Database connections**: Number of connection strings found
- **Code complexity**: Assessment of transformation complexity

Dependency Map:

- Visual representation of application-to-database relationships
- Cross-database dependencies
- Shared components

Understanding complexity assessment

AWS Transform classifies your modernization into three categories:

Complexity	Characteristics	Expected Outcome
Low (Class A)	Standard SQL patterns (ANSI SQL), simple stored procedures, basic data types, Entity Framework with standard configurations	Minimal human intervention expected, high automation success rate
Medium (Class B)	Advanced T-SQL patterns, complex stored procedures with business logic, user-defined functions, computed columns	Some human intervention required, expert review recommended
High (Class C)	CLR assemblies, linked servers, Service Broker, complex full-text search	Significant human refactoring required, consider phased approach

Assessment report

AWS Transform generates a detailed assessment report that includes:

- Executive summary with high-level overview
- Complete database inventory
- Application inventory
- Transformation readiness percentage
- Effort estimation
- Risk assessment and mitigation strategies
- Recommended approach

You can download the assessment report for offline review and sharing with stakeholders.

Step 7: Generate and review wave plan

For large estates with multiple databases and applications, AWS Transform generates a wave plan that sequences the modernization in logical groups.

What is a wave plan?

A wave plan organizes your modernization into phases (waves) based on:

- Dependencies between databases and applications
- Business priorities
- Risk tolerance
- Resource availability
- Technical complexity

Each wave contains a group of databases and applications that can be modernized together without breaking dependencies.

Review the wave plan

1. Navigate to **Wave planning** in the job plan
2. Review the proposed waves
3. For each wave, review:

- Databases included
- Applications included
- Dependencies on other waves
- Estimated transformation time
- Complexity level
- Deployable applications

Customize the wave plan

You can customize the wave plan to match your business needs in 2 ways:

Using JSON:

1. Choose **Download all waves** to get a JSON file with all the waves
2. Modify waves in the JSON by:
   - Moving databases between waves
   - Splitting waves into smaller groups
   - Merging waves together
   - Changing wave sequence
   - Adding or removing databases from scope
3. Upload the JSON file back to the console by choosing **Upload wave plan**
4. AWS Transform validates your changes and warns if dependencies are violated
5. Choose **Confirm waves** to update the wave plan

Using Chat:

You can modify the wave plans by chatting with the agent and asking it to move the repositories and databases to specific waves. This approach works well if you need to make minor edits to the waves.

> **Important**
>
> Ensure that dependencies are respected when customizing waves. Transforming a dependent application before its database can cause issues.

Single database modernization

If you're modernizing a single database and application, AWS Transform creates a simple plan with one wave. You can proceed directly to transformation without wave planning.

Approve the wave plan

1. After reviewing and customizing (if needed), choose **Approve wave plan**
2. AWS Transform locks the wave plan and proceeds to transformation
3. You can still modify the plan later by choosing **Edit wave plan**

Step 8: Schema conversion

AWS Transform converts your SQL Server database schema to Aurora PostgreSQL, including tables, views, stored procedures, functions, and triggers.

How schema conversion works

AWS Transform uses AWS DMS Schema Conversion enhanced with generative AI to:

- Analyze SQL Server schemas and relationships
- Map data types from SQL Server to PostgreSQL equivalents
- Transform T-SQL to PL/pgSQL
- Handle identity columns, computed columns, and constraints
- Validate conversion and referential integrity
- Generate action items for objects requiring human review

Supported conversions

Automatically converted:

- Tables, views, and indexes
- Primary keys and foreign keys
- Check constraints and default values
- Most common data types
- Simple stored procedures
- Basic functions and triggers

- Identity columns (converted to SERIAL or GENERATED)
- Most computed columns

May require human review:

- Complex stored procedures with advanced T-SQL
- SQL Server-specific functions (GETUTCDATE, SUSER_SNAME, etc.)
- Computed columns with complex expressions
- Full-text search indexes
- XML data type operations
- HIERARCHYID data type (requires ltree extension)

Not automatically converted:

- CLR assemblies
- Linked servers
- Service Broker
- SQL Server Agent jobs

Start schema conversion

1. Navigate to Schema conversion in the job plan
2. Review the conversion settings:
   - Target PostgreSQL version
   - Extension options (ltree, PostGIS, etc.)
   - Naming conventions
3. Choose **Start conversion**
4. Monitor progress in the worklog
5. Conversion typically takes 10-30 minutes depending on the number of database objects

Review conversion results

After conversion completes, navigate to Review schema conversion:

Conversion Summary:

- **Objects converted**: Count of successfully converted objects
- **Action items**: Objects requiring human attention
- **Warnings**: Potential issues to review
- **Errors**: Objects that could not be converted

Review by Object Type:

- **Tables**: Data type mappings, constraints, indexes
- **Stored procedures**: T-SQL to PL/pgSQL conversion
- **Functions**: Function signature and logic changes
- **Triggers**: Trigger syntax and timing changes

Review action items

1. Choose **View action items**
2. For each action item, review:
   - **Object name**: The database object
   - **Issue type**: What requires attention
   - **Severity**: Critical, Warning, or Info
   - **Recommendation**: Suggested resolution
   - **Original code**: SQL Server version
   - **Converted code**: PostgreSQL version
3. For each action item, you can:
   - **Accept**: Use the converted code
   - **Modify**: Edit the converted code
   - **Flag for later**: Mark for human review after transformation

Example: Stored procedure conversion

SQL Server T-SQL:

```
CREATE PROCEDURE GetProductsByCategory
```

```
    @CategoryId INT,
    @PageSize INT = 10
AS
BEGIN
    SET NOCOUNT ON;

    SELECT TOP (@PageSize)
        ProductId,
        Name,
        Price,
        DATEDIFF(DAY, CreatedDate, GETUTCDATE()) AS DaysOld
    FROM Products
    WHERE CategoryId = @CategoryId
    ORDER BY Name
END
```

Converted PostgreSQL PL/pgSQL:

```
CREATE OR REPLACE FUNCTION get_products_by_category(
    p_category_id INTEGER,
    p_page_size INTEGER DEFAULT 10
)
RETURNS TABLE (
    product_id INTEGER,
    name VARCHAR(255),
    price NUMERIC(18,2),
    days_old INTEGER
) AS $$
BEGIN
    RETURN QUERY
    SELECT
        p.product_id,
        p.name,
        p.price,
        EXTRACT(DAY FROM (NOW() - p.created_date))::INTEGER AS days_old
    FROM products p
    WHERE p.category_id = p_category_id
    ORDER BY p.name
    LIMIT p_page_size;
END;
$$ LANGUAGE plpgsql;
```

Changes made:

- Procedure converted to function returning TABLE
- Parameter names prefixed with p_
- TOP converted to LIMIT
- DATEDIFF converted to EXTRACT
- GETUTCDATE() converted to NOW()
- Column names converted to lowercase (PostgreSQL convention)

Approve schema conversion

1. After reviewing all action items and making necessary modifications
2. Choose **Approve schema conversion**
3. AWS Transform prepares the converted schema for deployment to Aurora PostgreSQL

> **Note**
>
> You can download the converted schema as SQL scripts for offline review or version control.

Step 9: Data migration (optional)

AWS Transform provides options for migrating data from SQL Server to Aurora PostgreSQL. Data migration is optional and can be skipped if you only need schema and code transformation.

Data migration options

Option 1: Production Data Migration

Migrate your actual production data using AWS DMS:

- Full initial load of all data
- Continuous replication during testing (CDC)
- Minimal downtime cutover
- Data validation and integrity checks

Option 2: Skip Data Migration

Transform schema and code only:

- Useful for development/testing environments
- When data will be migrated separately
- For proof-of-concept projects

Configure data migration

1. Navigate to **Data migration** in the job plan
2. Choose your migration option:
   - **Migrate production data**
   - **Skip data migration**
3. If migrating production data, configure:
   - **Migration type**: Full load, or Full load + CDC
   - **Validation**: Enable data validation
   - **Performance**: DMS instance size
   - Choose **>Start migration**

Production data migration process

If you choose to migrate production data:

1. **Initial sync**: AWS DMS performs full load of all tables
2. **Continuous replication**: (If CDC enabled) Keeps data synchronized
3. **Validation**: Verifies row counts and data integrity
4. **Cutover preparation**: Prepares for final synchronization

Migration Timeline:

- Small databases (< 10 GB): 30 minutes - 2 hours
- Medium databases (10-100 GB): 2-8 hours
- Large databases (> 100 GB): 8+ hours

Data validation

AWS Transform validates migrated data with the following checks:

- Row count comparison (source vs target)
- Primary key integrity
- Foreign key relationships
- Data type compatibility
- Computed column results
- Null value handling

Step 10: Application code transformation

AWS Transform transforms your .NET application code to work with Aurora PostgreSQL instead of SQL Server. It asks for a target branch name in your repositories to commit the transformed source code. Once you enter the branch name, AWS Transform will create a new branch and initiate the transformation to match the PostgreSQL database.

What gets transformed

Entity Framework Changes:

- **Database provider**: UseSqlServer() → UseNpgsql()
- **Connection strings**: SQL Server format → PostgreSQL format
- **Data type mappings**: SQL Server types → PostgreSQL types
- **DbContext configurations**: SQL Server-specific → PostgreSQL-specific
- **Migration files**: Updated for PostgreSQL compatibility

ADO.NET Changes:

- **Connection classes**: SqlConnection → NpgsqlConnection
- **Command classes**: SqlCommand → NpgsqlCommand
- **Data reader**: SqlDataReader → NpgsqlDataReader
- **Parameters**: SqlParameter → NpgsqlParameter
- S**QL syntax**: T-SQL → PostgreSQL SQL

Configuration Changes:

- Connection strings in appsettings.json
- Database provider NuGet packages
- Dependency injection configurations
- Startup/Program.cs configurations

Start code transformation

1. Navigate to Application transformation in the job plan
2. Review the transformation settings:
   - **Target .NET version** (if upgrading)
   - **PostgreSQL provider version**
   - **Code style preferences**
3. Choose **Start transformation**
4. Monitor progress in the worklog
5. Transformation typically takes 15-45 minutes depending on codebase size

Step 11: Review transformation results

Before proceeding to deployment, review the complete transformation results to ensure everything is ready for testing.

You can download the transformed code from the repository branch for:

- Local testing and validation
- Code review in your IDE
- Integration with your CI/CD pipeline
- Version control commit

You can also download the transformation summary to review the natural language changes that are made by AWS Transform as part of the transformation.

Transformation summary

1. Navigate to **Transformation summary** in the job plan

2. Review the overall results:
   - **Schema conversion**: Objects converted, action items, warnings
   - **Data migration**: Tables migrated, rows transferred, validation status
   - **Code transformation**: Files changed, lines modified, issues resolved
   - **Readiness score**: Overall readiness for deployment

Generate transformation report

AWS Transform generates a comprehensive transformation report:

1. Choose **Generate report**
2. Select report type:
   - **Executive summary**: High-level overview for stakeholders
   - **Technical details**: Complete transformation documentation
   - **Action items**: List of human tasks required
3. Choose **Download report**

The report includes:

- Transformation scope and objectives
- Objects and code transformed
- Issues encountered and resolutions
- Validation results
- Deployment readiness assessment
- Recommendations for testing

Step 12: Validation and testing

Before deploying to production, validate that the transformed application works correctly with Aurora PostgreSQL.

Validation types

Automated Validation: AWS Transform performs automated checks:

- Schema validation against source database
- Data integrity verification
- Query equivalence testing
- Connection string validation
- Configuration validation

Human Validation: You should perform additional testing:

- Functional testing of application features
- Integration testing with other systems
- Performance testing and benchmarking
- User acceptance testing
- Security testing

Run automated validation

1. Navigate to **Validation** in the job plan
2. Choose **Run validation**
3. AWS Transform executes validation tests:
   - Database connectivity
   - Schema compatibility
   - Data integrity
   - Application build
   - Basic functionality
4. Review validation results:
   - Passed: Tests that succeeded
   - Failed: Tests that need attention
   - Warnings: Potential issues to review

Testing checklist

Database Functionality:

- All tables accessible
- Stored procedures execute correctly
- Functions return expected results
- Triggers fire appropriately
- Constraints enforced properly
- Indexes improve query performance

Application Functionality:

- Application starts successfully
- Database connections established
- CRUD operations work correctly
- Stored procedure calls succeed
- Transactions commit/rollback properly
- Error handling works as expected

Data Integrity:

- Row counts match source
- Primary keys unique
- Foreign keys valid
- Computed columns correct
- Null handling appropriate
- Data types compatible

Performance:

- Query response times acceptable
- Connection pooling configured
- Indexes optimized
- No N+1 query issues

- Batch operations efficient
- Resource utilization reasonable

Step 13: Deployment

After successful validation deploy your modernized application and database to production.

Deployment options

- Amazon ECS and Amazon EC2 Linux

Pre-deployment checklist

Before deploying to production:

- All validation tests passed
- Performance testing completed
- Security review completed
- Backup and rollback plan documented
- Monitoring and alerting configured
- Team trained on new environment
- Stakeholders informed of deployment
- Maintenance window scheduled

Deploy to Amazon ECS

1. Navigate to **Deployment** in the job plan
2. Choose **Deploy to ECS**
3. Configure deployment settings:
   - **Cluster**: Select or create ECS cluster
   - **Service**: Configure ECS service
   - **Task definition**: Review generated task definition
   - **Load balancer**: Configure ALB/NLB
   - **Auto-scaling**: Set scaling policies

4. Review infrastructure-as-code (CloudFormation template or AWS CDK code)
5. Choose **Deploy**

Monitor deployment

AWS Transform deploys your application:

1. Creates Aurora PostgreSQL cluster
2. Applies database schema
3. Loads data (if applicable)
4. Deploys application containers
5. Configures load balancer
6. Sets up auto-scaling

Monitor deployment progress and verify:

- Infrastructure provisioning
- Database initialization
- Application deployment
- Health checks passing
- Application accessible
- Database connections working
- Logs showing normal operation

Post-deployment validation

After deployment:

Smoke Testing:

- Verify critical functionality
- Test key user workflows
- Check integration points

- Monitor error rates

Performance Monitoring:

- Track response times
- Monitor database queries
- Check resource utilization
- Review application logs

User Validation:

- Conduct user acceptance testing
- Gather feedback
- Address any issues
- Document lessons learned

Rollback procedures

If issues arise after deployment:

Immediate Rollback:

- Revert to previous application version
- Switch back to SQL Server (if still available)
- Restore from backup if needed

Partial Rollback:

- Roll back specific components
- Keep database changes
- Revert application code only

Forward Fix:

- Apply hotfix to Aurora PostgreSQL version

- Deploy updated application code
- Monitor for resolution

> **Important**
>
> Keep your SQL Server database available for a period after cutover to enable rollback if needed.

Post-deployment optimization

After successful deployment:

Performance Tuning:

- Optimize slow queries
- Adjust connection pool settings
- Fine-tune Aurora PostgreSQL parameters
- Review and optimize indexes

Cost Optimization:

- Right-size Aurora instance
- Configure auto-scaling appropriately
- Review storage settings
- Optimize backup retention

Monitoring Setup:

- Configure CloudWatch dashboards
- Set up alerting
- Enable Enhanced Monitoring
- Configure Performance Insights

Documentation:

- Update runbooks
- Document architecture changes
- Train operations team
- Create troubleshooting guides

Test and deploy after transformation

Deploy your transformed applications to the landing zone and validate the connectivity between your applications and the transformed database, and optionally set up CI/CD pipelines. Sample CI/CD pipeline configurations are generated as reference to help you implement your deployment automation. This final phase validates the complete modernization.

Configure resources

You need to configure resources used in deployment such as IAM roles, instance profiles and S3 bucket. AWS Transform does not have permissions to create S3 buckets or to make changes in your IAM. This action is needed to be performed by a user who has permissions to create new roles. Usually it is account administrator.

To simplify the process, we provide a setup script that automatically:

- Executes the necessary CloudFormation templates
- Creates the ECS Service-Linked Role
- Sets up all required IAM resources

> **Important**
>
> Run the provided setup script as is - do not modify any resource names in the templates or script as this may impact the deployment process.

For Linux:

```
./setup.sh --region <YOUR-Region>
```

For Windows:

```
./setup.ps1 -Region <YOUR-Region>
```

Check security groups in your VPC

AWS Transform deploys the applications to the same subnet and same security group as Aurora PostgreSQL instance.

In order for the application to connect to the Aurora instance, make sure that this security group has outbound rules allowing connection to the same security group and inbound rules allowing connection from the same security group.

Select and configure applications

The AWS Transform UI shows a list of applications that can be deployed. Select application and click **Configure and Commit** to open a configuration window where you can specify:

- Target infrastructure for the applications: select ECS or one of the EC2 instances
- Application or instance parameters

After you complete and close the configuration window, AWS Transform:

- Generates deployment artifacts based on your configuration
- Commits these artifacts to your repository, including:
  - Deployment scripts
  - CloudFormation templates
  - Sample CI/CD pipeline configurations (which you can use as reference for setting up your deployment automation)

Once the commit is complete, the application status changes to *Configured and committed*. You can then select the applications and use the **Deploy** button in the top right to initiate deployment.

ECS deployment

All applications with target ECS are deployed into a new ECS cluster that is created in your account. Each application is deployed in its own dedicated container - multiple applications per container are not supported. You configure container parameters for each application individually, such as required CPU and Memory.

EC2 deployment

You can select one of proposed EC2 instances and can configure the its parameters such as instance type. Multiple applications can be deployed to the same EC2 instance:

- To deploy multiple applications to a single instance, select the same target EC2 instance for each of these applications during configuration.
- All applications that have the same instance selected as a target are deployed to that instance and assigned different random ports.
- Note that if you edit instance parameters for one application, and then edit another application targeting the same instance, the changes affect all applications deployed to that instance.

Deployment success

Upon successful deployment, your applications are running on Aurora PostgreSQL with reduced licensing costs, improved performance, and cloud-native scalability. The job stays open unless explicitly stopped. This allows the option for a possible re-deployment with updated deployment parameters.

Upload DDL files (offline workflow)

In this workflow, AWS Transform modernizes your schema entirely from uploaded DDL files — no direct connection to your database is required. You run a provided extraction script against your SQL Server, upload the resulting schema files, and AWS Transform discovers, assesses, converts, and deploys the schema to Amazon Aurora PostgreSQL. Choose this workflow when your database cannot be exposed to a direct connection, or when you want a feasibility assessment and schema conversion before granting network access.

> **Important**
>
> The offline workflow converts and deploys your schema only. It does not migrate real data. Where available, you can generate synthetic test data to validate the target. If you need to migrate real data, use the AWS DMS console to create a new data migration project.

Prerequisites for the offline workflow

- A supported SQL Server version (2008 R2 through 2022, all editions).

- **PowerShell 7 or later** to run the extraction script. PowerShell is pre-installed on Windows. On macOS or Linux, install PowerShell 7+ and run the script with `pwsh`. See the Microsoft installation guide.
- The `sqlcmd` utility available on the PATH. See sqlcmd utility.
- A SQL Server authentication login (username and password) with at least `VIEW SERVER STATE` and `VIEW ANY DEFINITION` permissions. The script does not support Windows or domain authentication.
- An AWS account for the later phases (creating the Aurora PostgreSQL target and deploying the converted schema), with IAM Identity Center or IAM authentication enabled.
- No inbound or outbound database connectivity to AWS is required for the schema analysis and conversion phases.

Workflow steps

1. **Upload DDL files** — AWS Transform provides a download button for an extraction script in the chat. Download and run the script against your SQL Server. Optionally limit extraction to specific databases with a comma-separated include list (case-insensitive). The script produces a timestamped `.zip` of SQL DDL files. Upload the `.zip` (or individual `.sql` files) using the chat upload button.
2. **Discovery** — AWS Transform parses the uploaded files and catalogs the database objects (tables, stored procedures, views, triggers, indexes, constraints, sequences, and more), grouped by database. You review a per-database summary and choose which databases to assess. If some statements cannot be parsed, AWS Transform reports the issues and lets you continue with the objects that parsed, choose specific databases, or upload corrected files.
3. **Assessment** — AWS Transform evaluates conversion feasibility for the selected databases and produces a downloadable report with an executive summary, a per-database object summary, and a level-of-effort (LOE) estimate. Customize the LOE scores, upload additional files, or proceed to conversion.
4. **Schema conversion** — AWS Transform converts the SQL Server schema (including T-SQL stored procedures) to Aurora PostgreSQL, one database at a time. A single database converts automatically; with multiple databases, you choose which to convert. Databases convert independently and in parallel.
5. **Target provisioning** — For each converted database, AWS Transform creates or reuses an Amazon Aurora PostgreSQL cluster in your AWS account. You choose networking and target configuration through guided prompts.

6. **Schema deployment** — AWS Transform applies the converted schema to the Aurora cluster. After deployment, fix remaining objects in the chat or in an IDE, generate synthetic test data (where available), or finish.
7. **Application code modernization** (optional) — AWS Transform can update associated .NET 6+ applications to work with Aurora PostgreSQL. The service updates database connections in the source code and modifies ORM code in Entity Framework and ADO.NET to be compatible with Aurora PostgreSQL in a unified workflow with human supervision. Connect your source code repository using AWS CodeConnections or a Personal Access Token (PAT) for GitHub and GitLab, or provide an S3 URI for a zip file containing the source code.

> **Note**
>
> Amazon Aurora PostgreSQL is the only supported target engine for this workflow.

Service limits for the offline workflow

The following default limits apply to the offline workflow. Some limits are configurable per account; contact your AWS account team to request an increase.

Limit	Default	Applies to
Upload size per job	1,024 MB (1 GB), cumulative across all uploads in the job	Upload DDL files
Databases converted per month	10 per account	Schema conversion
Stored procedures per database	250	Schema conversion
Concurrent schema conversions	5 databases at a time	Schema conversion
Concurrent running jobs	5	Per account

- The upload size limit is a per-job cumulative budget. If you reset input setup, the budget is cleared (this discards the files already uploaded and re-runs discovery and assessment within the same job).
- When you choose more than 5 databases for conversion, the additional databases are queued and start automatically as running conversions finish.
- Assessment is never blocked by the stored-procedure-per-database limit; only schema conversion is affected.

Troubleshooting the offline workflow

Extraction script

- **`pwsh: command not found` (macOS/Linux)** — PowerShell 7+ is not installed. Install it and run `pwsh ./ExtractDatabaseMetadata.ps1`.
- **Errors referencing `sqlcmd`** — The `sqlcmd` utility is not on the PATH. Install it and ensure it is on the PATH.
- **Login or permission errors** — Use a SQL Server authentication login with `VIEW SERVER STATE` and `VIEW ANY DEFINITION`. Windows/domain authentication is not supported.
- **Only some databases exported** — The include list narrowed the scope. Re-run without it, or specify the correct comma-separated database names.

Upload

- **Upload rejected with a size message** — The per-job upload budget is exceeded. Upload a smaller file, or ask the agent to reset input setup to clear the budget within the same job. Ask the agent for the current remaining budget rather than estimating.
- **File not accepted** — Upload the `.zip` produced by the script, or individual `.sql` files. Zip multiple files into a single archive.

Discovery

- **SQL syntax issues detected** — Download the issues report. Objects in error-free sections are still cataloged. Choose to continue with the parsed objects, choose specific databases, or upload corrected files.

- **Objects grouped under a generic name** — The parser could not find a `USE [Database]` statement. When prompted, provide the correct database name.
- **No objects found** — The files contained no `CREATE` statements, or all failed to parse. Re-upload valid DDL.

Assessment

- **LOE estimates look off** — Choose to customize LOE scores, edit the template, and re-upload to regenerate the assessment.
- **Missed objects** — Choose to upload additional SQL files; discovery and assessment re-run automatically.

Schema conversion

- **A database conversion fails** — AWS Transform reports the reason. Reply asking to retry; conversion restarts for that database.
- **A response goes to the wrong database** — When multiple databases are awaiting input, specify which database your message is for. You can address a specific database with `@DatabaseName:`.

Target provisioning and deployment

- **Cannot create the Aurora cluster** — Complete the guided credential and networking prompts, and ensure the account has permission to create Aurora PostgreSQL.
- **Some objects did not deploy** — Use the post-deployment options to fix remaining objects in the chat or in an IDE, then redeploy.

Best practices and troubleshooting

Best practices, and common issues and their resolutions during the modernization process.

ECS application logs

CloudWatch logs

- All ECS container logs are automatically sent to CloudWatch Logs
- Access logs in the CloudWatch console under Log Groups

- Log group naming format: /aws/ecs/{application-name}
- Each container instance creates a new log stream within the group

Viewing logs

Through AWS Console:

- Navigate to CloudWatch > Log Groups
- Select your application's log group
- Choose the relevant log stream to view container logs

Using AWS CLI:

```
aws logs get-log-events --log-group-name /aws/ecs/your-app-name --log-stream-name your-stream-name
```

Common log locations

- Application logs: CloudWatch Logs
- ECS Service Events: ECS Console > Cluster > Service > Events tab
- Container health/status: ECS Console > Cluster > Service > Tasks tab

Database connection management

Applications use environment variables for database connection settings

If you experience connectivity issues:

- Verify the current connection settings in your environment variables
- Update environment variables to modify database connection strings as needed
- Connection string changes can be made through environment variable updates without application redeployment

Database connection issues

Problem: Cannot connect AWS Transform to SQL Server

Solutions:

- Verify network connectivity between AWS Transform and SQL Server
- Check security group rules for proper port access (1433)
- Confirm database credentials in Secrets Manager
- Test database permissions with the created user
- Ensure SQL Server is configured for mixed mode authentication
- Verify secret has required tags (Project: atx-db-modernization, Owner: database-connector)

Firewall and security group issues

Problem: Connection timeout or "cannot reach database" errors

Root Cause: Security groups or network ACLs blocking traffic

Solutions:

1. Verify Security Group Configuration:
   - Confirm your SQL Server security group has an inbound rule allowing port 1433 from the DMS Schema Conversion security group
   - Check that the source is the security group ID (e.g., sg-0123456789abcdef0), not an IP address
   - Verify the DMS Schema Conversion security group is correctly specified in the Instance Profile
   - Ensure there are no conflicting deny rules
2. Check Network ACLs:
   - Verify subnet-level Network ACLs allow inbound traffic on port 1433
   - Ensure Network ACLs allow outbound ephemeral ports for return traffic
   - Check both the database subnet and DMS subnet Network ACLs
3. Verify VPC Configuration:
   - Confirm the DMS Schema Conversion instance and SQL Server are in the same VPC or have proper VPC peering
   - Check route tables allow traffic between subnets
   - Verify no firewall appliances are blocking traffic
4. Test Connectivity:
   - Launch a test EC2 instance in the same subnet as DMS Schema Conversion

- Attach the same security group as DMS Schema Conversion
- Test connection to SQL Server using telnet or SQL Server Management Studio
- If test succeeds, the issue is with AWS Transform configuration; if it fails, the issue is network/firewall

Common Mistake: Opening port 1433 to 0.0.0.0/0 (all sources) is a security risk. Always use security group-based access control to limit access to only the DMS Schema Conversion security group.

Schema conversion issues

Problem: Schema conversion shows many action items

Solutions:

- Review action items in conversion report
- Prioritize based on impact
- Use Amazon Q Developer for complex SQL conversions
- Consult AWS Support for guidance
- Consider phased approach for complex databases

Application transformation issues

Problem: Application transformation fails to build

Solutions:

- Review build errors in transformation report
- Configure private NuGet feeds if needed
- Update package references if required
- Check for Windows-specific dependencies
- Review transformation logs for detailed errors

Data migration issues

Problem: Data migration validation fails

Solutions:

- Review validation report for specific failures
- Check data type mappings
- Verify identity column configuration (GENERATED BY DEFAULT vs GENERATED ALWAYS)
- Review computed column expressions
- Contact AWS Support for complex data issues

Resource cleanup issues

Problem: Transformation job fails with resource errors

Solutions:

- Check for existing DMS resources (migration projects, data providers, instance profiles)
- Clean up failed or incomplete resources from previous attempts
- Verify secrets are not scheduled for deletion
- Check service quotas for DMS and Aurora PostgreSQL
- Contact AWS Support if cleanup doesn't resolve the issue

Deployment issues

Problem: Transformed application cannot connect to Aurora PostgreSQL

Solutions:

- Verify connection string format for PostgreSQL
- Check security group rules
- Verify database credentials in Secrets Manager
- Ensure SSL/TLS is properly configured
- Test connection using psql or pgAdmin

Getting additional help

When contacting AWS Support, please provide:

- Transformation job ID
- AWS account ID
- Region
- Error messages and screenshots
- Transformation logs (available in AWS Transform console)

Security in AWS Transform

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from data centers and network architectures that are built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The shared responsibility model describes this as security *of* the cloud and security *in* the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. Third-party auditors regularly test and verify the effectiveness of our security as part of the AWS Compliance Programs. To learn about the compliance programs that apply to AWS Transform, see AWS Services in Scope by Compliance Program.
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your company's requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using AWS Transform. The following topics show you how to configure AWS Transform to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your AWS Transform resources.

Topics

- Data protection in AWS Transform
- Identity and access management for AWS Transform
- Compliance validation for AWS Transform
- Resilience in AWS Transform
- AWS Transform and interface endpoints (AWS PrivateLink)
- Accessing the AWS Transform web application from a VPC

Data protection in AWS Transform

The AWS shared responsibility model applies to data protection in AWS Transform. As described in this model, AWS is responsible for protecting the global infrastructure that runs all of the

AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. You are also responsible for the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the Data Privacy FAQ. For information about data protection in Europe, see the AWS Shared Responsibility Model and GDPR blog post on the *AWS Security Blog*.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual users with AWS Identity and Access Management (IAM). That way each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We recommend TLS 1.2 or later.
- Set up API and user activity logging with AWS CloudTrail.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing sensitive data that is stored in Amazon S3.
- If you require FIPS 140-2 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-2.

We strongly recommend that you never put confidential or sensitive information, such as your customers' email addresses, into tags or free-form text fields such as a **Name** field. This includes when you work with AWS Transform or other AWS services using the AWS Management Console, API, AWS Command Line Interface (AWS CLI), or AWS SDKs. Any data that you enter into tags or free-form text fields used for names may be used for billing or diagnostic logs. For more information about how AWS Transform uses content, see AWS Transform service improvement.

AWS Transform stores your questions, its responses, and additional context, such as console metadata and code in your IDE, to generate responses to your questions. For information about how data is encrypted, see Data encryption in AWS Transform. For information about how AWS may use some questions that you ask AWS Transform and its responses to improve our services, see AWS Transform service improvement.

In AWS Transform, your data is stored in the AWS Region where your AWS Transform profile was created.

With cross-inferencing, your requests to AWS Transform may be processed in a different Region within the geography where your data is stored. For more information, see Cross-Region inference.

Topics

- Cross-region processing in AWS Transform
- Data encryption in AWS Transform
- AWS Transform service improvement

Cross-region processing in AWS Transform

The following sections describe how cross-region inference and cross-region calls are used to provide the AWS Transform service.

Cross-region inference

AWS Transform is powered by Amazon Bedrock, and uses cross-region inference to distribute traffic across different AWS Regions to enhance large language model (LLM) inference performance and reliability. With cross-region inference, you get:

- Increased throughput and resilience during high demand periods
- Improved performance
- Access to newly launched AWS Transform capabilities and features that rely on the most powerful LLMs hosted on Amazon Bedrock

Cross-region inference requests are kept within the AWS Regions that are part of the geography where the data originally resides. For example, a request made from a AWS Transform configuration in the US is kept within the AWS Regions in the US. Although cross-region inferencing doesn't change where your data is stored, your requests and output results may move outside of the Region where the data originally resides. All data will be encrypted while transmitted across Amazon's secure network. There's no additional cost for using cross-region inference.

Cross region inference doesn't affect where your data is stored. For information on where data is stored when you use AWS Transform, see Data protection in AWS Transform.

Supported regions for AWS Transform cross-region inference

Certain requests you make to AWS Transform might require cross-region calls. The following table describes what Regions your requests may be routed to depending on the geography where the request originated.

> **Important**
>
> Beginning October 7, 2026, AWS Transform Chat is updating how it processes inference requests that originate in the Asia Pacific (Mumbai) (ap-south-1) and Asia Pacific (Seoul) (ap-northeast-2) Regions. Inference requests originating in these Regions will be securely routed to available compute resources across all commercial AWS Regions, rather than being processed only within the Asia Pacific geography.
> Your data continues to be stored only in the Region where the request originated.
> Only input prompts and output results may be processed in another Region. All data is transmitted encrypted across Amazon's secure network.

Source Region	Destination Regions
US East (N. Virginia) (us-east-1)	US East (N. Virginia) (us-east-1) US East (Ohio) (us-east-2) US West (Oregon) (us-west-2)
Europe (Frankfurt) (eu-central-1)	Europe (Frankfurt) (eu-central-1) Europe (Stockholm) (eu-north-1) Europe (Milan) (eu-south-1) Europe (Spain) (eu-south-2) Europe (Ireland) (eu-west-1) Europe (Paris) (eu-west-3)
Asia Pacific (Mumbai) (ap-south-1)	Asia Pacific (Tokyo) (ap-northeast-1) Asia Pacific (Seoul) (ap-northeast-2)

Source Region	Destination Regions
	Asia Pacific (Osaka) (ap-northeast-3) Asia Pacific (Mumbai) (ap-south-1) Asia Pacific (Hyderabad) (ap-south-2) Asia Pacific (Singapore) (ap-southeast-1) Asia Pacific (Sydney) (ap-southeast-2) Asia Pacific (Melbourne) (ap-southeast-4)
Asia Pacific (Tokyo) (ap-northeast-1)	Asia Pacific (Tokyo) (ap-northeast-1) Asia Pacific (Osaka) (ap-northeast-3)
Asia Pacific (Seoul) (ap-northeast-2)	Asia Pacific (Tokyo) (ap-northeast-1) Asia Pacific (Seoul) (ap-northeast-2) Asia Pacific (Osaka) (ap-northeast-3) Asia Pacific (Mumbai) (ap-south-1) Asia Pacific (Hyderabad) (ap-south-2) Asia Pacific (Singapore) (ap-southeast-1) Asia Pacific (Sydney) (ap-southeast-2) Asia Pacific (Melbourne) (ap-southeast-4)
Asia Pacific (Sydney) (ap-southeast-2)	Asia Pacific (Sydney) (ap-southeast-2) Asia Pacific (Melbourne) (ap-southeast-4)

Source Region	Destination Regions
Europe (London) (eu-west-2)	Europe (Frankfurt) (eu-central-1) Europe (Stockholm) (eu-north-1) Europe (Milan) (eu-south-1) Europe (Spain) (eu-south-2) Europe (Ireland) (eu-west-1) Europe (London) (eu-west-2) Europe (Paris) (eu-west-3)
Canada (Central) (ca-central-1)	All commercial regions
South America (São Paulo) (sa-east-1)	All commercial regions
Asia Pacific (Singapore) (ap-southeast-1)	All commercial regions

For a complete list of Regions where you can use AWS Transform, see Supported Regions for AWS Transform.

Cross-Region knowledge

When you ask a general question about AWS Transform services, transformation workflows, or related AWS documentation, AWS Transform might make cross-region requests to US East (Virginia) (us-east-1) for US regions or Europe (Frankfurt) (eu-central-1) for all other regions to retrieve documentation and fulfill your request. For example, when you ask questions about how to use other AWS services such as Lambda, AWS Transform might make a cross-region call to retrieve relevant AWS documentation to respond to your question. The following table describes what Regions your requests may be routed to depending on the geography where the request originated.

Source Region	Destination Regions
US East (N. Virginia) (us-east-1)	US East (N. Virginia) (us-east-1)

Source Region	Destination Regions
Europe (Frankfurt) (eu-central-1)	Europe (Frankfurt) (eu-central-1)
Europe (London) (eu-west-2)	Europe (Frankfurt) (eu-central-1)
Asia Pacific (Tokyo) (ap-northeast-1)	Europe (Frankfurt) (eu-central-1)
Asia Pacific (Sydney) (ap-southeast-2)	Europe (Frankfurt) (eu-central-1)
Asia Pacific (Seoul) (ap-northeast-2)	Europe (Frankfurt) (eu-central-1)
Asia Pacific (Mumbai) (ap-south-1)	Europe (Frankfurt) (eu-central-1)
Canada (Central) (ca-central-1)	Europe (Frankfurt) (eu-central-1)
South America (São Paulo) (sa-east-1)	Europe (Frankfurt) (eu-central-1)
Asia Pacific (Singapore) (ap-southeast-1)	Europe (Frankfurt) (eu-central-1)

This setting is enabled by default. An account administrator can modify this setting. Disabling this feature results in the loss of access to features that require AWS Transform to retrieve knowledge from other regions. This might result in less accurate responses.

To disable cross-region calls made by AWS Transform:

1. When first setting up AWS Transform, navigate to the **Get Started** page and complete the configuration. For for an existing AWS Transform configuration, navigate to the **Settings** page.
2. Toggle **Enable cross-region calls to answer general AWS related questions** to the *off* position.

AWS Transform for Windows cross-region inference

The following table shows the source Regions from which you can call the inference profile and the destination Regions to which the requests can be routed:

Source Region	Inference Destination Regions
US East (N. Virginia) (us-east-1)	US East (N. Virginia) (us-east-1)

Source Region	Inference Destination Regions
	US East (Ohio) (us-east-2) US West (Oregon) (us-west-2)
Europe (Frankfurt) (eu-central-1)	Europe (Frankfurt) (eu-central-1) Europe (Stockholm) (eu-north-1) Europe (Milan) (eu-south-1) Europe (Spain) (eu-south-2) Europe (Ireland) (eu-west-1) Europe (Paris) (eu-west-3)
Asia Pacific (Tokyo) (ap-northeast-1)	Asia Pacific (Tokyo) (ap-northeast-1) Asia Pacific (Osaka) (ap-northeast-3)
Asia Pacific (Sydney) (ap-southeast-2)	Asia Pacific (Sydney) (ap-southeast-2) Asia Pacific (Melbourne) (ap-southeast-4)
Asia Pacific (Seoul) (ap-northeast-2)	All commercial regions
Asia Pacific (Mumbai) (ap-south-1)	All commercial regions
Canada (Central) (ca-central-1)	All commercial regions

AWS Transform for Migrations cross-region inference

The following table shows the source Regions from which you can call the inference profile and the destination Regions to which the requests can be routed:

Source Region	Inference Destination Regions
US East (N. Virginia) (us-east-1)	All commercial regions

Source Region	Inference Destination Regions
Europe (Frankfurt) (eu-central-1)	All commercial regions
Europe (London) (eu-west-2)	All commercial regions
Asia Pacific (Tokyo) (ap-northeast-1)	Asia Pacific (Tokyo) (ap-northeast-1) Asia Pacific (Osaka) (ap-northeast-3)
Asia Pacific (Sydney) (ap-southeast-2)	Asia Pacific (Sydney) (ap-southeast-2) Asia Pacific (Melbourne) (ap-southeast-4)
Asia Pacific (Seoul) (ap-northeast-2)	All commercial regions
Asia Pacific (Mumbai) (ap-south-1)	All commercial regions
Canada (Central) (ca-central-1)	All commercial regions

AWS Transform for mainframe cross-region inference

AWS Transform for mainframe uses Amazon Bedrock geographic cross-region inference. This means that some of your calls might be routed to AWS Regions outside of your source region. The following table shows the source Regions from which you can call the inference profile and the destination Regions to which the requests can be routed:

Source Region	Inference Destination Regions
US East (N. Virginia) (us-east-1)	US East (N. Virginia) (us-east-1) US East (Ohio) (us-east-2) US West (Oregon) (us-west-2)
Europe (Frankfurt) (eu-central-1)	Europe (Frankfurt) (eu-central-1) Europe (Stockholm) (eu-north-1) Europe (Milan) (eu-south-1)

Source Region	Inference Destination Regions
	Europe (Spain) (eu-south-2) Europe (Ireland) (eu-west-1) Europe (Paris) (eu-west-3)
Europe (London) (eu-west-2)	Europe (Frankfurt) (eu-central-1) Europe (Stockholm) (eu-north-1) Europe (Milan) (eu-south-1) Europe (Spain) (eu-south-2) Europe (Ireland) (eu-west-1) Europe (London) (eu-west-2) Europe (Paris) (eu-west-3)
Asia Pacific (Tokyo) (ap-northeast-1)	Asia Pacific (Tokyo) (ap-northeast-1) Asia Pacific (Osaka) (ap-northeast-3)
Asia Pacific (Sydney) (ap-southeast-2)	Asia Pacific (Sydney) (ap-southeast-2) Asia Pacific (Melbourne) (ap-southeast-4)
Asia Pacific (Seoul) (ap-northeast-2)	All commercial regions
Asia Pacific (Mumbai) (ap-south-1)	All commercial regions
Canada (Central) (ca-central-1)	Canada (Central) (ca-central-1) US East (N. Virginia) (us-east-1) US East (Ohio) (us-east-2) US West (Oregon) (us-west-2)
South America (São Paulo) (sa-east-1)	All commercial regions

AWS Transform Custom

AWS Transform custom uses Amazon Bedrock geographic cross-region inference. This means that some of your calls might be routed to AWS Regions outside of your source region. The following table shows the source Regions from which you can call the inference profile and the destination Regions to which the requests can be routed:

Source Region	Inference Destination Regions
US East (N. Virginia) (us-east-1)	US East (N. Virginia) (us-east-1) US East (Ohio) (us-east-2) US West (Oregon) (us-west-2)
Europe (Frankfurt) (eu-central-1)	Europe (Frankfurt) (eu-central-1) Europe (Stockholm) (eu-north-1) Europe (Milan) (eu-south-1) Europe (Spain) (eu-south-2) Europe (Ireland) (eu-west-1) Europe (Paris) (eu-west-3)
Europe (London) (eu-west-2)	Europe (Frankfurt) (eu-central-1) Europe (Stockholm) (eu-north-1) Europe (Milan) (eu-south-1) Europe (Spain) (eu-south-2) Europe (Ireland) (eu-west-1) Europe (London) (eu-west-2) Europe (Paris) (eu-west-3)
Asia Pacific (Tokyo) (ap-northeast-1)	All commercial regions

Source Region	Inference Destination Regions
Asia Pacific (Sydney) (ap-southeast-2)	Asia Pacific (Sydney) (ap-southeast-2) Asia Pacific (Melbourne) (ap-southeast-4)
Asia Pacific (Seoul) (ap-northeast-2)	All commercial regions
Asia Pacific (Mumbai) (ap-south-1)	All commercial regions
Canada (Central) (ca-central-1)	Canada (Central) (ca-central-1) US East (N. Virginia) (us-east-1) US East (Ohio) (us-east-2) US West (Oregon) (us-west-2)

Data encryption in AWS Transform

This topic provides information specific to AWS Transform about encryption in transit and encryption at rest.

AWS Transform provides encryption by default to protect sensitive customer data at rest with encryption using AWS owned keys.

Encryption types

Encryption in transit

Communication between customers and AWS Transform and between AWS Transform and its downstream dependencies is protected using TLS 1.2 or higher connections.

> **Important**
>
> When you Connect a discovery account, AWS Transform creates an Amazon S3 bucket on your behalf in that account. That bucket does not have `SecureTransport` enabled by default. If you want the bucket policy to include secure transport you must update the policy. For more information, see Security best practices for Amazon S3.

Encryption at rest

AWS Transform stores data at rest using Amazon DynamoDB and Amazon Simple Storage Service (Amazon S3). The data at rest is encrypted using AWS encryption solutions by default. AWS Transform encrypts your data with encryption using AWS owned keys from AWS Key Management Service (AWS KMS). You don't have to take any action to protect the AWS managed keys that encrypt your data. For more information, see AWS owned keys in the *AWS Key Management Service Developer Guide*.

Data type	AWS-owned key encryption	Customer managed key encryption (Optional)
Customer bucket data Customer inputs and outputs such as code and documentation stored in an Amazon S3 bucket	Enabled	Enabled
Artifact Store Intermediate artifacts as part of code transformation stored in an S3 bucket	Enabled	Enabled
Job Objective The customer's intent for the job stored in an Amazon S3 bucket	Enabled	Enabled
Chat messages Messages between the customer and AWS Transform stored in an Amazon S3 bucket	Enabled	Enabled
Chat Knowledge Base	Enabled	Enabled

Data type	AWS-owned key encryption	Customer managed key encryption (Optional)
Indexed data relevant to AWS Transform and customer chat stored in Amazon OpenSearch and processed via AWS Bedrock		

Note: The customer can register their own Customer Managed Key (CMK) to be used for encrypting all of the above data types.

Customer managed keys are KMS keys in your AWS account that you create, own, and manage to directly control access to your data by controlling access to the KMS key. Only symmetric keys are supported. For information on creating your own KMS key, see Creating keys in the *AWS Key Management Service Developer Guide*.

When you use a customer managed key, AWS Transform makes use of KMS grants, allowing authorized users, roles, or applications to use a KMS key. When an AWS Transform administrator chooses to use a customer managed key for encryption during configuration, a grant is created for them. This grant is what allows the end user to use the encryption key for data encryption at rest. For more information on grants, see Grants in AWS KMS.

AWS Owned Keys (Default)

> **Note**
>
> **AWS owned keys** — AWS Transform uses these keys by default to automatically encrypt personally identifiable data. You can't view, manage, or use AWS owned keys, or audit their use. However, you don't have to take any action or change any programs to protect the keys that encrypt your data. For more information, see AWS owned keys in the AWSKey Management Service Developer Guide.

Encryption of data at rest by default helps reduce the operational overhead and complexity involved in protecting sensitive data. At the same time, it enables you to build secure applications that meet strict encryption compliance and regulatory requirements.

While you can't disable this layer of encryption or select an alternate encryption type, you can add a second layer of encryption over the existing AWS owned encryption keys by choosing a customer managed key when you create your transformation:

Customer managed keys (Optional)

Customer managed keys — AWS Transform supports the use of a symmetric customer managed key that you create, own, and manage to add a second layer of encryption over the existing encryption using AWS owned keys. Because you have full control of this layer of encryption, you can perform such tasks as:

- Establishing and maintaining key policies
- Establishing and maintaining IAM policies and grants
- Enabling and disabling key policies
- Rotating key cryptographic material
- Adding tags
- Creating key aliases
- Rotating key cryptographic material
- Adding tags
- Creating key aliases
- Scheduling keys for deletion
- For more information, see customer managed key in the *AWS Key Management Service Developer Guide*.

> **Note**
>
> AWS Transform automatically enables encryption at rest using AWS owned keys to protect personally identifiable data at no charge. However, AWS KMS charges apply for using a customer managed key. For more information about pricing, see the AWS Key Management Service pricing.

For more information on AWS KMS, see What is AWS Key Management Service?

How AWS Transform uses grants in AWS Key Management Service

AWS Transform requires a grant to use your customer managed key.

When you create a [Resource Name] encrypted with a customer managed key, AWS Transform creates a grant on your behalf by sending a CreateGrant request to AWS Key Management Service. Grants in AWS Key Management Service are used to give AWS Transform access to a KMS key in a customer account.

AWS Transform requires the grant to use your customer managed key for the following internal operations:

- Send [KMS API] requests to AWS KMS to verify that the symmetric customer managed KMS key ID entered when creating [Resource Name] is valid.
- Send [KMS API] requests to AWS KMS to generate data keys encrypted by your customer managed key.
- Send [KMS API] requests to AWS KMS to decrypt the encrypted data keys so that they can be used to encrypt your data.

You can revoke access to the grant, or remove the service's access to the customer managed key at any time. If you do, AWS Transform won't be able to access any of the data encrypted by the customer managed key, which affects operations that are dependent on that data. For example, if you attempt to get [Resource Name] from an encrypted [Resource Name]that AWS Transform can't access, then the operation would return an AccessDeniedException error.

Create a customer managed key

You can create a symmetric customer managed key by using the AWS Management Console, or the AWS Key Management Service APIs.

To create a symmetric customer managed key, follow the steps for Creating symmetric customer managed key in the *AWS Key Management Service Developer Guide*.

To use your customer managed key with your AWS Transform resources, the following API operations must be permitted in the key policy:

kms:CreateGrant – Adds a grant to a customer managed key. Grants control access to a specified KMS key, which allows access to grant operations AWS Transform requires. For more information about Using Grants, see the *AWS Key Management Service Developer Guide*.

This allows AWS Transform to do the following:

- Call [KMS API (GenerateDataKeyWithoutPlainText/GenerateDatakey) to generate an encrypted data key and store it, because the data key isn't immediately used to encrypt.
- Call [KMS API (Decrypt)] to use the stored encrypted data key to access encrypted data.
- Set up a retiring principal to allow the service to RetireGrant.
- kms:DescribeKey – Provides the customer managed key details to allow [Service Name] to validate the key.

Encryption for SQL modernization

During the SQL Server modernization process AWS Transform creates resources in your account, such as EC2 instances, ECS clusters, ECR images and S3 objects. You have the option of providing a customer managed key to encrypt data in your account. You will need to tag your KMS key with key **CreatedFor** and value **AWSTransform** . This example policy lists the minimum permissions required in your key policy if you don't have a default key policy. If you have a default key policy, you still need to manually update the KMS key in addition to your default key policy to allow ECS to use your key.

```
"Statement": [
   {
     "Effect": "Allow",
     "Principal": {
       "AWS": "arn:aws:iam::111122223333:root"
     },
     "Action": [
       "kms:Decrypt",
       "kms:GenerateDataKey"
     ],
     "Resource": "*",
     "Condition": {
       "StringLike": {
         "kms:ViaService": "s3.*.amazonaws.com",
         "kms:EncryptionContext:aws-transform": "*"
       },
       "ArnLike": {
         "aws:PrincipalArn": "arn:aws:iam::*:role/*AWSTransform*"
       }
     }
   },
```

```
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": [
        "kms:Decrypt",
        "kms:GenerateDataKey",
        "kms:GenerateDataKeyWithoutPlaintext",
        "kms:ReEncrypt*"
      ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "kms:ViaService": "ec2.*.amazonaws.com"
        },
        "ArnLike": {
          "aws:PrincipalArn": "arn:aws:iam::*:role/*AWSTransform*"
        }
      }
    },
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": "kms:CreateGrant",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "kms:ViaService": [
            "ec2.*.amazonaws.com",
            "ecr.*.amazonaws.com"
          ]
        },
        "Bool": {
          "kms:GrantIsForAWSResource": "true"
        },
        "ArnLike": {
          "aws:PrincipalArn": "arn:aws:iam::*:role/*AWSTransform*"
        }
      }
    },
    {
```

```
      "Effect": "Allow",
      "Principal": {
        "Service": "fargate.amazonaws.com"
      },
      "Action": "kms:GenerateDataKeyWithoutPlaintext",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "kms:EncryptionContext:aws:ecs:clusterName": "AWSTransform*"
        }
      }
    },
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "fargate.amazonaws.com"
      },
      "Action": "kms:CreateGrant",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "kms:EncryptionContext:aws:ecs:clusterName": "AWSTransform*"
        },
        "ForAllValues:StringEquals": {
          "kms:GrantOperations": "Decrypt"
        }
      }
    }
  ]
```

Encryption for database migration

During the database migration process, AWS Transform uses AWS Database Migration Service (DMS) to migrate your database data. You can optionally provide a customer managed key to encrypt your database data during migration.

The following policy contains three statements. The first statement grants root account permissions for the AWS KMS key. The second statement allows the connector role to validate the key during discovery. The third statement allows the connector role to use the key for DMS operations.

```
  "Statement": [
```

```
    {
      "Sid": "Enable IAM User Permissions",
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::111122223333:root" },
      "Action": "kms:*",
      "Resource": "*"
    },
    {
      "Sid": "Allow connector role to validate key during discovery",
      "Effect": "Allow",
      "Principal": { "AWS": "connector-role-arn" },
      "Action": "kms:DescribeKey",
      "Resource": "*"
    },
    {
      "Sid": "Allow connector role to use key via DMS",
      "Effect": "Allow",
      "Principal": { "AWS": "connector-role-arn" },
      "Action": [
        "kms:CreateGrant",
        "kms:Decrypt",
        "kms:DescribeKey",
        "kms:Encrypt",
        "kms:GenerateDataKey*",
        "kms:ReEncrypt*"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:ViaService": "dms.us-east-1.amazonaws.com"
        }
      }
    }
  ]
```

Using customer managed KMS keys

After creating a customer managed KMS key, an AWS Transform administrator must provide the key in the AWS Transform console to use it to encrypt data.

To set up a customer managed key to encrypt data in AWS Transform, administrators need permissions to use AWS KMS.

To use functionality that is encrypted with a customer managed key, users need permissions to allow AWS Transform to access the customer managed key.

If you see an error related to KMS grants while using AWS Transform, you likely need to update your permissions to allow AWS Transform to create grants. To automatically configure the needed permissions, go to the AWS Transform console and choose **Update permissions** in the banner at the top of the page. In order to allow AWS Transform to create grants, you need to update the permissions on the AWS Transform console page.

Allow AWS Transform access to customer managed keys

The following example policy statement grants users permissions to access features encrypted with a customer managed key by allowing AWS Transform access to the key. This policy is required to use AWS Transform if an administrator has set up a customer managed key for encryption.

> **Note**
>
> This information does not apply to SQL Server modernization deployment workflow using CMK.

```
"Statement": [
   {
     "Sid": "QKMSDecryptGenerateDataKeyPermissions",
     "Effect": "Allow",
     "Action": [
       "kms:Decrypt",
       "kms:GenerateDataKey",
       "kms:GenerateDataKeyWithoutPlaintext",
       "kms:ReEncryptFrom",
       "kms:ReEncryptTo"
     ],
     "Resource": [
       "arn:aws:kms:arn:aws::111122223333:key/[[key_id]]"
     ],
     "Condition": {
       "StringLike": {
         "kms:ViaService": [
           "q.us-east-1.amazonaws.com"
         ]
       }
```

```
      }
    }
  ]
```

AWS Transform service improvement

To help AWS Transform provide the most relevant information, we may use certain content from AWS Transform, such as questions that you ask AWS Transform and its responses, for service improvement. This page explains what content we use and how to opt out.

AWS Transform content used for service improvement

We may use certain content from AWS Transform for service improvement. AWS Transform may use this content, for example, to provide better responses to common questions, fix AWS Transform operational issues, for de-bugging, or for model training.

Content that AWS may use for service improvement includes, for example, your questions to AWS Transform and the responses and code that AWS Transform generates.

How to opt out

The way you opt out of AWS Transform using content for service improvement depends on the environment where you use AWS Transform.

For AWS Transform configure an AI services opt-out policy in AWS Organizations. For more information, see AI services opt-out policies in the *AWS Organizations User Guide*.

To opt out of sharing your content in Visual Studio IDE, use the following procedure.

Go to **Tools** -> **Options** -> **AWS Toolkit** -> **Amazon Q**

Toggle **Share Amazon Q Content with AWS** to **True** or **False**.

To opt out of sharing your telemetry data in the AWS Toolkit for Visual Studio, use this procedure:

1. Under **Tools**, choose **Options**.
2. In the **Options** pane, choose **AWS Toolkit**, and then choose **General**.
3. Deselect **Allow AWS Toolkit to collect usage information**.

Identity and access management for AWS Transform

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be *authenticated* (signed in) and *authorized* (have permissions) to use AWS Transform resources. IAM is an AWS service that you can use with no additional charge.

Topics

- Audience
- Authenticating with identities
- Managing access using policies
- How AWS Transform works with IAM
- Identity-based policy examples for AWS Transform
- Troubleshooting AWS Transform identity and access
- Using service-linked roles for AWS Transform
- AWS managed policies for AWS Transform
- AWS Transform permissions reference

Audience

How you use AWS Identity and Access Management (IAM) differs based on your role:

- **Service user** - request permissions from your administrator if you cannot access features (see Troubleshooting AWS Transform identity and access)
- **Service administrator** - determine user access and submit permission requests (see How AWS Transform works with IAM)
- **IAM administrator** - write policies to manage access (see Identity-based policy examples for AWS Transform)

Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. You must be authenticated as the AWS account root user, an IAM user, or by assuming an IAM role.

You can sign in as a federated identity using credentials from an identity source like AWS IAM Identity Center (IAM Identity Center), single sign-on authentication, or Google/Facebook credentials. For more information about signing in, see How to sign in to your AWS account in the *AWS Sign-In User Guide*.

For programmatic access, AWS provides an SDK and CLI to cryptographically sign requests. For more information, see AWS Signature Version 4 for API requests in the *IAM User Guide*.

AWS account root user

When you create an AWS account, you begin with one sign-in identity called the AWS account *root user* that has complete access to all AWS services and resources. We strongly recommend that you don't use the root user for everyday tasks. For tasks that require root user credentials, see Tasks that require root user credentials in the *IAM User Guide*.

Federated identity

As a best practice, require human users to use federation with an identity provider to access AWS services using temporary credentials.

A *federated identity* is a user from your enterprise directory, web identity provider, or Directory Service that accesses AWS services using credentials from an identity source. Federated identities assume roles that provide temporary credentials.

For centralized access management, we recommend AWS IAM Identity Center. For more information, see What is IAM Identity Center? in the *AWS IAM Identity Center User Guide*.

IAM users and groups

An *IAM user* is an identity with specific permissions for a single person or application. We recommend using temporary credentials instead of IAM users with long-term credentials. For more information, see Require human users to use federation with an identity provider to access AWS using temporary credentials in the *IAM User Guide*.

An *IAM group* specifies a collection of IAM users and makes permissions easier to manage for large sets of users. For more information, see Use cases for IAM users in the *IAM User Guide*.

IAM roles

An *IAM role* is an identity with specific permissions that provides temporary credentials. You can assume a role by switching from a user to an IAM role (console) or by calling an AWS CLI or AWS API operation. For more information, see Methods to assume a role in the *IAM User Guide*.

IAM roles are useful for federated user access, temporary IAM user permissions, cross-account access, cross-service access, and applications running on Amazon EC2. For more information, see Cross account resource access in IAM in the *IAM User Guide*.

Managing access using policies

You control access in AWS by creating policies and attaching them to AWS identities or resources. A policy defines permissions when associated with an identity or resource. AWS evaluates these policies when a principal makes a request. Most policies are stored in AWS as JSON documents. For more information about JSON policy documents, see Overview of JSON policies in the *IAM User Guide*.

Using policies, administrators specify who has access to what by defining which **principal** can perform **actions** on what **resources**, and under what **conditions**.

By default, users and roles have no permissions. An IAM administrator creates IAM policies and adds them to roles, which users can then assume. IAM policies define permissions regardless of the method used to perform the operation.

Identity-based policies

Identity-based policies are JSON permissions policy documents that you attach to an identity (user, group, or role). These policies control what actions identities can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Define custom IAM permissions with customer managed policies in the *IAM User Guide*.

Identity-based policies can be *inline policies* (embedded directly into a single identity) or *managed policies* (standalone policies attached to multiple identities). To learn how to choose between managed and inline policies, see Choose between managed policies and inline policies in the *IAM User Guide*.

Resource-based policies

Resource-based policies are JSON policy documents that you attach to a resource. Examples include IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-

based policies, service administrators can use them to control access to a specific resource. You must specify a principal in a resource-based policy.

Resource-based policies are inline policies that are located in that service. You can't use AWS managed policies from IAM in a resource-based policy.

Other policy types

AWS supports additional policy types that can set the maximum permissions granted by more common policy types:

- **Permissions boundaries** – Set the maximum permissions that an identity-based policy can grant to an IAM entity. For more information, see Permissions boundaries for IAM entities in the *IAM User Guide*.
- **Service control policies (SCPs)** – Specify the maximum permissions for an organization or organizational unit in AWS Organizations. For more information, see Service control policies in the *AWS Organizations User Guide*.
- **Resource control policies (RCPs)** – Set the maximum available permissions for resources in your accounts. For more information, see Resource control policies (RCPs) in the *AWS Organizations User Guide*.
- **Session policies** – Advanced policies passed as a parameter when creating a temporary session for a role or federated user. For more information, see Session policies in the *IAM User Guide*.

Multiple policy types

When multiple types of policies apply to a request, the resulting permissions are more complicated to understand. To learn how AWS determines whether to allow a request when multiple policy types are involved, see Policy evaluation logic in the *IAM User Guide*.

How AWS Transform works with IAM

Before you use IAM to manage access to AWS Transform, learn what IAM features are available to use with AWS Transform.

IAM feature	AWS Transform support
Identity-based policies	Yes

IAM feature	AWS Transform support
Resource-based policies	No
Policy actions	Yes
Policy resources	Yes
Policy condition keys	Yes
ACLs	No
ABAC (tags in policies)	Partial
Temporary credentials	Yes
Principal permissions	Yes
Service roles	Yes
Service-linked roles	Yes

To get a high-level view of how AWS Transform and other AWS services work with most IAM features, see AWS services that work with IAM in the *IAM User Guide*.

Identity-based policies for AWS Transform

Supports identity-based policies: Yes

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Define custom IAM permissions with customer managed policies in the *IAM User Guide*.

With IAM identity-based policies, you can specify allowed or denied actions and resources as well as the conditions under which actions are allowed or denied. To learn about all of the elements that you can use in a JSON policy, see IAM JSON policy elements reference in the *IAM User Guide*.

Identity-based policy examples for AWS Transform

To view examples of AWS Transform identity-based policies, see Identity-based policy examples for AWS Transform.

Resource-based policies within AWS Transform

Supports resource-based policies: No

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must specify a principal in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

To enable cross-account access, you can specify an entire account or IAM entities in another account as the principal in a resource-based policy. For more information, see Cross account resource access in IAM in the *IAM User Guide*.

Policy actions for AWS Transform

Supports policy actions: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Action` element of a JSON policy describes the actions that you can use to allow or deny access in a policy. Include actions in a policy to grant permissions to perform the associated operation.

To see a list of AWS Transform actions, see Actions, resources, and condition keys for AWS Transform in the *Service Authorization Reference*.

To see a list of actions for AWS Transform *custom*, see Actions, resources, and condition keys for AWS Transform custom in the *Service Authorization Reference*.

Policy actions in AWS Transform use the following prefix before the action:

```
transform
```

To specify multiple actions in a single statement, separate them with commas.

```
"Action": [
      "transform:action1",
      "transform:action2"
         ]
```

To view examples of AWS Transform identity-based policies, see Identity-based policy examples for AWS Transform.

Policy resources for AWS Transform

Supports policy resources: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Resource` JSON policy element specifies the object or objects to which the action applies. As a best practice, specify a resource using its Amazon Resource Name (ARN). For actions that don't support resource-level permissions, use a wildcard (*) to indicate that the statement applies to all resources.

```
"Resource": "*"
```

To see a list of AWS Transform resource types and their ARNs, and with which actions you can specify the ARN of each resource, see Actions, resources, and condition keys for AWS Transform in the *Service Authorization Reference*.

To see a list of resource types and their ARNs, and with which actions you can specify the ARN of each resource for AWS Transform *custom*, see Actions, resources, and condition keys for AWS Transform custom in the *Service Authorization Reference*.

To view examples of AWS Transform identity-based policies, see Identity-based policy examples for AWS Transform.

Policy condition keys for AWS Transform

Supports service-specific policy condition keys: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Condition` element specifies when statements execute based on defined criteria. You can create conditional expressions that use condition operators, such as equals or less than, to match the condition in the policy with values in the request. To see all AWS global condition keys, see AWS global condition context keys in the *IAM User Guide*.

To see a list of AWS Transform condition keys, and with which actions and resources you can use a condition key, see Actions, resources, and condition keys for AWS Transform in the *Service Authorization Reference*.

To see a list of condition keys, and with which actions and resources you can use a condition key for AWS Transform *custom*, see Actions, resources, and condition keys for AWS Transform custom in the *Service Authorization Reference*.

To view examples of AWS Transform identity-based policies, see Identity-based policy examples for AWS Transform.

ACLs in AWS Transform

Supports ACLs: No

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

ABAC with AWS Transform

Supports ABAC (tags in policies): Partial

Attribute-based access control (ABAC) is an authorization strategy that defines permissions based on attributes called tags. You can attach tags to IAM entities and AWS resources, then design ABAC policies to allow operations when the principal's tag matches the tag on the resource.

To control access based on tags, you provide tag information in the condition element of a policy using the `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, or `aws:TagKeys` condition keys.

If a service supports all three condition keys for every resource type, then the value is **Yes** for the service. If a service supports all three condition keys for only some resource types, then the value is **Partial**.

For more information about ABAC, see Define permissions with ABAC authorization in the *IAM User Guide*. To view a tutorial with steps for setting up ABAC, see Use attribute-based access control (ABAC) in the *IAM User Guide*.

Using temporary credentials with AWS Transform

Supports temporary credentials: Yes

Temporary credentials provide short-term access to AWS resources and are automatically created when you use federation or switch roles. AWS recommends that you dynamically generate temporary credentials instead of using long-term access keys. For more information, see Temporary security credentials in IAM and AWS services that work with IAM in the *IAM User Guide*.

Cross-service principal permissions for AWS Transform

Supports forward access sessions (FAS): Yes

Forward access sessions (FAS) use the permissions of the principal calling an AWS service, combined with the requesting AWS service to make requests to downstream services. For policy details when making FAS requests, see Forward access sessions.

Service roles for AWS Transform

Supports service roles: Yes

A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Create a role to delegate permissions to an AWS service in the *IAM User Guide*.

> **Warning**
>
> Changing the permissions for a service role might break AWS Transform functionality. Edit service roles only when AWS Transform provides guidance to do so.

Service-linked roles for AWS Transform

Supports service-linked roles: Yes

A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your AWS

account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.

For details about creating or managing AWS Transform service-linked roles, see Using service-linked roles for AWS Transform.

Identity-based policy examples for AWS Transform

By default, users and roles don't have permission to create or modify AWS Transform resources. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies.

To learn how to create an IAM identity-based policy by using these example JSON policy documents, see Create IAM policies (console) in the *IAM User Guide*.

For details about actions and resource types defined by AWS Transform, including the format of the ARNs for each of the resource types, see Actions, Resources, and Condition Keys for AWS Transform in the *Service Authorization Reference*.

Topics

- Policy best practices
- Using the AWS Transform console
- Allow users to view their own permissions
- Allow administrators to accept a connector request from the account with AWS Transform
- Allow administrators to assign existing IAM Identity Center users and create new IAM Identity Center users to assign to AWS Transform
- Allow administrators to enable AWS Transform
- Allow users to access AWS Transform with IAM credentials

Policy best practices

Identity-based policies determine whether someone can create, access, or delete AWS Transform resources in your account. These actions can incur costs for your AWS account. When you create or edit identity-based policies, follow these guidelines and recommendations:

- **Get started with AWS managed policies and move toward least-privilege permissions** – To get started granting permissions to your users and workloads, use the *AWS managed policies*

that grant permissions for many common use cases. They are available in your AWS account. We recommend that you reduce permissions further by defining AWS customer managed policies that are specific to your use cases. For more information, see AWS managed policies or AWS managed policies for job functions in the *IAM User Guide*.

- **Apply least-privilege permissions** – When you set permissions with IAM policies, grant only the permissions required to perform a task. You do this by defining the actions that can be taken on specific resources under specific conditions, also known as *least-privilege permissions*. For more information about using IAM to apply permissions, see Policies and permissions in IAM in the *IAM User Guide*.
- **Use conditions in IAM policies to further restrict access** – You can add a condition to your policies to limit access to actions and resources. For example, you can write a policy condition to specify that all requests must be sent using SSL. You can also use conditions to grant access to service actions if they are used through a specific AWS service, such as CloudFormation. For more information, see IAM JSON policy elements: Condition in the *IAM User Guide*.
- **Use IAM Access Analyzer to validate your IAM policies to ensure secure and functional permissions** – IAM Access Analyzer validates new and existing policies so that the policies adhere to the IAM policy language (JSON) and IAM best practices. IAM Access Analyzer provides more than 100 policy checks and actionable recommendations to help you author secure and functional policies. For more information, see Validate policies with IAM Access Analyzer in the *IAM User Guide*.
- **Require multi-factor authentication (MFA)** – If you have a scenario that requires IAM users or a root user in your AWS account, turn on MFA for additional security. To require MFA when API operations are called, add MFA conditions to your policies. For more information, see Secure API access with MFA in the *IAM User Guide*.

For more information about best practices in IAM, see Security best practices in IAM in the *IAM User Guide*.

Using the AWS Transform console

To access the AWS Transform console, you must have a minimum set of permissions. These permissions must allow you to list and view details about the AWS Transform resources in your AWS account. If you create an identity-based policy that is more restrictive than the minimum required permissions, the console won't function as intended for entities (users or roles) with that policy.

You don't need to allow minimum console permissions for users that are making calls only to the AWS CLI or the AWS API. Instead, allow access to only the actions that match the API operation that they're trying to perform.

Allow users to view their own permissions

This example shows how you might create a policy that allows IAM users to view the inline and managed policies that are attached to their user identity. This policy includes permissions to complete this action on the console or programmatically using the AWS CLI or AWS API.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Allow administrators to accept a connector request from the account with AWS Transform

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "transform:GetConnector",
                "transform:AssociateConnectorResource",
                "transform:RejectConnector"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketPublicAccessBlock",
                "s3:GetAccountPublicAccessBlock"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:CreateRole",
                "iam:AttachRolePolicy",
                "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::111122223333:role/service-role/
AWSTransform-*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:CreatePolicy"
            ],
            "Resource": "arn:aws:iam::111122223333:policy/service-role/
AWSTransform-*"
```

```
        }
    ]
}
```

Allow administrators to assign existing IAM Identity Center users and create new IAM Identity Center users to assign to AWS Transform

The following policy grants the permissions that an administrator needs to manage IAM Identity Center users and groups in the AWS Transform console. With this policy, an administrator can assign and remove users and groups, view display names, create new IAM Identity Center users, manage application assignments, and configure application session settings.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "SSOApplicationAssignments",
      "Effect": "Allow",
      "Action": [
        "sso:ListApplicationAssignments",
        "sso:CreateApplicationAssignment",
        "sso:DeleteApplicationAssignment",
        "sso:ListInstances",
        "sso:DescribeInstance",
        "sso:ListDirectoryAssociations",
        "sso:GetApplicationGrant",
        "sso:GetApplicationAccessScope",
        "sso:GetApplicationSessionConfiguration",
        "sso:PutApplicationSessionConfiguration"
      ],
      "Resource": "*"
    },
    {
      "Sid": "SSODirectoryCreateUser",
      "Effect": "Allow",
      "Action": [
        "sso-directory:CreateUser"
      ],
      "Resource": "*"
    },
    {
```

```
      "Sid": "IdentityStoreAccess",
      "Effect": "Allow",
      "Action": [
        "identitystore:DescribeUser",
        "identitystore:DescribeGroup",
        "identitystore:ListUsers",
        "identitystore:ListGroups"
      ],
      "Resource": "*"
    },
    {
      "Sid": "AllowKmsAccessViaIdentityCenter",
      "Effect": "Allow",
      "Action": [
        "kms:Decrypt"
      ],
      "Resource": "*",
      "Condition": {
        "ArnLike": {
          "kms:EncryptionContext:aws:sso:instance-arn": "arn:*:sso:::instance/*"
        },
        "StringLike": {
          "kms:ViaService": "sso.*.amazonaws.com"
        }
      }
    },
    {
      "Sid": "AllowKmsAccessViaIdentityStore",
      "Effect": "Allow",
      "Action": [
        "kms:Decrypt"
      ],
      "Resource": "*",
      "Condition": {
        "ArnLike": {
          "kms:EncryptionContext:aws:identitystore:identitystore-arn":
"arn:*:identitystore::*:identitystore/*"
        },
        "StringLike": {
          "kms:ViaService": "identitystore.*.amazonaws.com"
        }
      }
    }
  ]
```

```
}
```

Allow administrators to enable AWS Transform

The following policy grants the permissions that an administrator needs to enable and administer AWS Transform through the AWS console. This includes managing profiles, configuring agents, and managing connectors. This policy is intended for administrators only and should be scoped to trusted principals in your account.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "SSOEnableTransform",
      "Effect": "Allow",
      "Action": [
        "sso:ListInstances",
        "sso:CreateInstance",
        "sso:CreateApplication",
        "sso:PutApplicationAuthenticationMethod",
        "sso:PutApplicationGrant",
        "sso:PutApplicationAssignmentConfiguration",
        "sso:ListApplications",
        "sso:GetSharedSsoConfiguration",
        "sso:DescribeInstance",
        "sso:PutApplicationAccessScope",
        "sso:DescribeApplication",
        "sso:DeleteApplication",
        "sso:UpdateApplication",
        "sso:DescribeRegisteredRegions",
        "sso:GetSSOStatus"
      ],
      "Resource": "*"
    },
    {
      "Sid": "SSODirectoryActions",
      "Effect": "Allow",
      "Action": [
        "sso-directory:GetUserPoolInfo",
        "sso-directory:DescribeUsers",
        "sso-directory:DescribeGroups",
        "sso-directory:SearchGroups",
        "sso-directory:SearchUsers",
```

```
        "sso-directory:DescribeDirectory"
      ],
      "Resource": "*"
    },
    {
      "Sid": "KMSListAliases",
      "Effect": "Allow",
      "Action": [
        "kms:ListAliases"
      ],
      "Resource": "*"
    },
    {
      "Sid": "KMSActions",
      "Effect": "Allow",
      "Action": [
        "kms:CreateGrant",
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:GenerateDataKey*",
        "kms:RetireGrant",
        "kms:DescribeKey"
      ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "kms:ViaService": "transform.*.amazonaws.com"
        }
      }
    },
    {
      "Sid": "IAMServiceLinkedRole",
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": [
        "arn:*:iam::*:role/aws-service-role/transform.amazonaws.com/
AWSServiceRoleForAWSTransform"
      ]
    },
    {
      "Sid": "TransformConsoleActions",
      "Effect": "Allow",
```

```
      "Action": [
        "transform:GetAccountSettings",
        "transform:UpdateAccountSettings",
        "transform:CreateProfile",
        "transform:UpdateProfile",
        "transform:DeleteProfile",
        "transform:ListProfiles",
        "transform:GetConnector",
        "transform:ListConnectors",
        "transform:DeleteConnector",
        "transform:AssociateConnectorResource",
        "transform:RejectConnector",
        "transform:ListAgents",
        "transform:GetAgent",
        "transform:GetAgentRuntimeConfiguration",
        "transform:PutAgentRuntimeConfiguration",
        "transform:UpdateAgentAccess",
        "transform:TagResource",
        "transform:UntagResource",
        "transform:ListTagsForResource",
        "transform:GetWebAppUrl"
      ],
      "Resource": "*"
    }
  ]
}
```

Allow users to access AWS Transform with IAM credentials

The following policy grants an IAM principal access to the AWS Transform web application and APIs using IAM credentials. This policy is required for IAM-only access and for IAM access enabled alongside an identity provider.

Replace *`region`*, *`account-id`*, and *`profile-id`* with your values. You can find the profile ID in the AWS Transform console under **Settings**.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowTransformAccess",
      "Effect": "Allow",
      "Action": "transform:AccessTransformProfile",
```

```
      "Resource": "arn:aws:transform:region:account-id:profile/profile-id"
    }
  ]
}
```

To grant access to all profiles in an account, use a wildcard for the profile ID:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowTransformAccessAllProfiles",
      "Effect": "Allow",
      "Action": "transform:AccessTransformProfile",
      "Resource": "arn:aws:transform:region:account-id:profile/*"
    }
  ]
}
```

Note

The `transform:AccessTransformProfile` action grants access to AWS Transform only. Actions within workspaces are controlled by AWS Transform workspace roles (Admin, Contributor, Approver, Read-only), not by IAM policies. Workspace roles determine what a user can do, such as creating jobs, managing collaborators, or approving tasks.

Troubleshooting AWS Transform identity and access

Use the following information to help you diagnose and fix common issues that you might encounter when working with AWS Transform and IAM.

Topics

- I am not authorized to perform an action in AWS Transform
- I am not authorized to perform iam:PassRole
- I want to allow people outside of my AWS account to access my AWS Transform resources

I am not authorized to perform an action in AWS Transform

If you receive an error that you're not authorized to perform an action, your policies must be updated to allow you to perform the action.

The following example error occurs when the `mateojackson` IAM user tries to use the console to view details about a fictional *`my-example-widget`* resource but doesn't have the fictional `AWS Transform:`*`GetWidget`* permissions.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform: AWS
 Transform:GetWidget on resource: my-example-widget
```

In this case, the policy for the `mateojackson` user must be updated to allow access to the *`my-example-widget`* resource by using the `AWS Transform:`*`GetWidget`* action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

I am not authorized to perform iam:PassRole

If you receive an error that you're not authorized to perform the `iam:PassRole` action, your policies must be updated to allow you to pass a role to AWS Transform.

Some AWS services allow you to pass an existing role to that service instead of creating a new service role or service-linked role. To do this, you must have permissions to pass the role to the service.

The following example error occurs when an IAM user named `marymajor` tries to use the console to perform an action in AWS Transform. However, the action requires the service to have permissions that are granted by a service role. Mary does not have permissions to pass the role to the service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In this case, Mary's policies must be updated to allow her to perform the `iam:PassRole` action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

I want to allow people outside of my AWS account to access my AWS Transform resources

You can create a role that users in other accounts or people outside of your organization can use to access your resources. You can specify who is trusted to assume the role. For services that support resource-based policies or access control lists (ACLs), you can use those policies to grant people access to your resources.

To learn more, consult the following:

- To learn whether AWS Transform supports these features, see How AWS Transform works with IAM.
- To learn how to provide access to your resources across AWS accounts that you own, see Providing access to an IAM user in another AWS account that you own in the *IAM User Guide*.
- To learn how to provide access to your resources to third-party AWS accounts, see Providing access to AWS accounts owned by third parties in the *IAM User Guide*.
- To learn how to provide access through identity federation, see Providing access to externally authenticated users (identity federation) in the *IAM User Guide*.
- To learn the difference between using roles and resource-based policies for cross-account access, see Cross account resource access in IAM in the *IAM User Guide*.

Using service-linked roles for AWS Transform

AWS Transform uses AWS Identity and Access Management (IAM) service-linked roles. A service-linked role is a unique type of IAM role that is linked directly to AWS Transform. Service-linked roles are predefined by AWS Transform and include all the permissions that the service requires to call other AWS services on your behalf.

Using service-linked roles for AWS Transform

AWS Transform uses AWS Identity and Access Management (IAM) service-linked roles. A service-linked role is a unique type of IAM role that is linked directly to AWS Transform. Service-linked roles are predefined by AWS Transform and include all the permissions that the service requires to call other AWS services on your behalf.

A service-linked role makes setting up AWS Transform easier because you don't have to manually add the necessary permissions. AWS Transform defines the permissions of its service-linked roles,

and unless defined otherwise, only AWS Transform can assume its roles. The defined permissions include the trust policy and the permissions policy, and that permissions policy cannot be attached to any other IAM entity.

You can delete a service-linked role only after first deleting their related resources. This protects your AWS Transform resources because you can't inadvertently remove permission to access the resources.

For information about other services that support service-linked roles, see AWS services that work with IAM and look for the services that have **Yes** in the **Service-linked roles** column. Choose a **Yes** with a link to view the service-linked role documentation for that service.

Service-linked role permissions for AWS Transform

AWS Transform uses the service-linked role named AWSServiceRoleForAWSTransform – This Service-Linked Role provides AWS Transform with the ability to provide usage information.

The AWSServiceRoleForAWSTransform service-linked role trusts the following services to assume the role:

- `transform.amazonaws.com`

The role permissions policy named AWSServiceRoleForAWSTransform allows AWS Transform to complete the following actions on the specified resources:

- cloudwatch:PutMetricData
  - Send custom metrics to CloudWatch for AWS Transform operations
  - Track transformation progress, success rates, and performance metrics
  - Enable monitoring and alerting on transformation workflows
- sso:DescribeApplication
  - View details about a specific application in Identity Center
  - Get application metadata like name, description, status, and configuration
- sso:GetApplicationAssignmentConfiguration
  - Retrieve assignment configuration settings for an application
  - See how users/groups are configured to be assigned to the application
- sso:ListApplicationAssignmentsForPrincipal
  - List all applications assigned to a specific user or group (principal)

  - View which applications a particular identity has access to
- Enables decryption of KMS-encrypted data when accessed through IAM Identity Center. Only works when the encryption context contains a valid IAM Identity Center instance ARN and must be accessed via IAM Identity Center service endpoints.
- Allows decryption of KMS-encrypted data when accessed through Identity Store. Only works when the encryption context contains a valid Identity Store ARN and must be accessed via Identity Store service endpoints.
- secretsmanager:GetSecretValue
  - Access AWS Transform service-linked secrets used to store client secrets for external identity providers
  - Resource: arn:aws:secretsmanager:*:*:secret:transform-preprod!*
  - Condition: Must be owned by transform-preprod service and accessed from same account
- support:CreateCase, support:DescribeCases, support:DescribeCommunications, support:AddCommunicationToCase, support:ResolveCase
  - Create and manage premium support cases from the AWS Transform web application
  - View case details and communications
  - Add communications and resolve support cases

You must configure permissions to allow your users, groups, or roles to create, edit, or delete a service-linked role. For more information, see Service-linked role permissions in the *IAM User Guide*.

Creating a service-linked role for AWS Transform

You don't need to manually create a service-linked role. When you create a profile for AWS Transform in the AWS Management Console, AWS Transform creates the service-linked role for you.

If you delete this service-linked role, and then need to create it again, you can use the same process to recreate the role in your account. When you update the settings, AWS Transform creates the service-linked role for you again.

You can also use the IAM console or AWS CLI to create a service-linked role with the `transform.amazonaws.com` service name. For more information, see Creating a service-linked role in the *IAM User Guide*. If you delete this service-linked role, you can use this same process to create the role again.

Editing a service-linked role for AWS Transform

AWS Transform does not allow you to edit the AWSServiceRoleForAWSTransform service-linked role. After you create a service-linked role, you cannot change the name of the role because various entities might reference the role. However, you can edit the description of the role using IAM. For more information, see Editing a service-linked role in the *IAM User Guide*.

Deleting a service-linked role for AWS Transform

If you no longer need to use a feature or service that requires a service-linked role, we recommend that you delete that role. That way you don't have an unused entity that is not actively monitored or maintained. However, you must clean up the resources for your service-linked role before you can manually delete it.

> **Note**
>
> If the AWS Transform service is using the role when you try to delete the resources, then the deletion might fail. If that happens, wait for a few minutes and try the operation again.

To manually delete the service-linked role using IAM

Use the IAM console, the AWS CLI, or the AWS API to delete the AWSServiceRoleForAWSTransform service-linked role. For more information, see Deleting a service-linked role in the *IAM User Guide*.

Supported Regions for AWS Transform service-linked roles

AWS Transform does not support using service-linked roles in every Region where the service is available. You can use the AWSServiceRoleForAWSTransform role in the following Regions. For more information, see AWS Regions and endpoints.

Region name	Region identity	Support in AWS Transform
US East (N. Virginia)	us-east-1	Yes
Europe (Frankfurt)	eu-central-1	Yes

Using service-linked roles for AWS Transform Custom

AWS Transform Custom uses the service-linked role named **AWSServiceRoleForAWSTransformCustom** to publish metrics and logs to your account on your behalf. These metrics let you monitor transformation counts, latencies, and status codes directly in your dashboards. Logs provide visibility into issues encountered during transformations.

This service-linked role is predefined by AWS Transform Custom and includes only the permissions the service needs. You don't have to manually add any permissions, and the role can only be assumed by AWS Transform Custom. For general information about service-linked roles, see Using service-linked roles in the *IAM User Guide*.

Service-linked role permissions for AWS Transform Custom

The AWSServiceRoleForAWSTransformCustom service-linked role trusts the following services to assume the role:

- `transform-custom.amazonaws.com`

The role permissions policy named AWSServiceRoleForAWSTransformCustom allows AWS Transform Custom to complete the following actions on the specified resources:

- `cloudwatch:PutMetricData` on all AWS resources
  - Publish operational metrics to under the `AWS/TransformCustom` namespace
  - Track transformation counts, latencies, and status codes
  - Scoped to the `AWS/TransformCustom` namespace via the `cloudwatch:namespace` condition key
- `logs:CreateLogGroup` and `logs:PutRetentionPolicy` on the `/aws/TransformCustom` log group
  - Create the CloudWatch Logs log group for publishing transformation logs
  - Set log retention policies on the log group
  - Scoped to `arn:aws:logs:*:*:log-group:/aws/TransformCustom` with an `aws:ResourceAccount` condition ensuring access only within your account
- `logs:CreateLogStream` and `logs:PutLogEvents` on log streams within the `/aws/TransformCustom` log group
  - Create log streams and publish log events for transformation operations

- Scoped to `arn:aws:logs:*:*:log-group:/aws/TransformCustom:log-stream:*` with an `aws:ResourceAccount` condition ensuring access only within your account

You must configure permissions to allow your users, groups, or roles to create, edit, or delete a service-linked role. For more information, see Service-linked role permissions in the *IAM User Guide*.

Creating a service-linked role for AWS Transform Custom

You don't need to manually create a service-linked role. When you run a transformation using the AWS Transform Custom CLI, AWS Transform Custom creates the service-linked role for you.

If you delete this service-linked role, and then need to create it again, you can use the same process to recreate the role in your account. When you run a transformation using the AWS Transform Custom CLI, AWS Transform Custom creates the service-linked role for you again.

You can also use the IAM console to create a service-linked role with the **AWSServiceRoleForAWSTransformCustom** use case. In the AWS CLI or the AWS API, create a service-linked role with the `transform-custom.amazonaws.com` service name. For more information, see Creating a service-linked role in the *IAM User Guide*. If you delete this service-linked role, you can use this same process to create the role again.

Editing a service-linked role for AWS Transform Custom

AWS Transform Custom does not allow you to edit the AWSServiceRoleForAWSTransformCustom service-linked role. After you create a service-linked role, you cannot change the name of the role because various entities might reference the role. However, you can edit the description of the role using IAM. For more information, see Editing a service-linked role in the *IAM User Guide*.

Deleting a service-linked role for AWS Transform Custom

If you no longer need to use a feature or service that requires a service-linked role, we recommend that you delete that role. That way you don't have an unused entity that is not actively monitored or maintained.

The AWSServiceRoleForAWSTransformCustom service-linked role does not create persistent resources in your account. You can delete it directly without any resource cleanup.

If you delete this role and later run a transformation using the AWS Transform Custom CLI, the service-linked role is automatically recreated.

To manually delete the service-linked role using IAM

Use the IAM console, the AWS CLI, or the AWS API to delete the AWSServiceRoleForAWSTransformCustom service-linked role. For more information, see Deleting a service-linked role in the *IAM User Guide*.

Supported Regions for AWS Transform Custom service-linked roles

AWS Transform Custom supports using service-linked roles in all Regions where AWS Transform Custom is available. For a list of supported Regions, see Supported Regions.

AWS managed policies for AWS Transform

An AWS managed policy is a standalone policy that is created and administered by AWS. AWS managed policies are designed to provide permissions for many common use cases so that you can start assigning permissions to users, groups, and roles.

Keep in mind that AWS managed policies might not grant least-privilege permissions for your specific use cases because they're available for all AWS customers to use. We recommend that you reduce permissions further by defining customer managed policies that are specific to your use cases.

You cannot change the permissions defined in AWS managed policies. If AWS updates the permissions defined in an AWS managed policy, the update affects all principal identities (users, groups, and roles) that the policy is attached to. AWS is most likely to update an AWS managed policy when a new AWS service is launched or new API operations become available for existing services.

For more information, see AWS managed policies in the *IAM User Guide*.

AWS Transform updates for AWS managed policies

View details about updates to AWS managed policies for AWS Transform since March 1, 2021.

Change	Description	Date
AWSTransformServerMigrationAgentPolicy – Updated policy	Added permission for AWS Transform MGN to assume the `AWSTransformRehostSharingRole_`*`account-id`* role in other accounts	October 11, 2026

Change	Description	Date
	in your organization on your behalf. This lets MGN run multi-account server migration operations in member accounts.	
AWSTransformLandingZoneAgentPolicy – Updated policy	Updated the read-only AWS Organizations permissions that describe accounts, OUs, and policies, list the OU hierarchy, and list tags. These permissions now apply to resources in your organization (`aws:ResourceOrgID`) instead of only to resources in the calling account (`aws:ResourceAccount`). With this change, you can use the policy in a delegated administrator account to read your organization structure.	October 6, 2026
AWSTransformApplicationDeploymentPolicy – Updated policy	Added permissions to add and remove CloudFormation stack tags during stack create and update operations.	September 11, 2026
AWSTransformApplicationECSDeploymentPolicy – Updated policy	Added permissions to add and remove CloudFormation stack tags during stack create and update operations.	September 11, 2026

Change	Description	Date
AWSTransformLandingZoneAgentPolicy – New policy	Added a new AWS managed policy that grants the permissions needed to set up AWS landing zones, including account provisioning, organizational governance, and AWS Control Tower configuration.	September 3, 2026
AWSTransformNetworkMigrationAgentPolicy – New policy	Added a new AWS managed policy that grants the permissions needed to deploy and configure network infrastructure, including VPCs, transit gateways, and route tables, for network migration.	September 3, 2026
AWSTransformServerMigrationAgentPolicy – New policy	Added a new AWS managed policy that grants the permissions needed to perform server migration operations, including replication, testing, and cutover, using AWS Transform MGN.	September 3, 2026
AWSTransformInfrastructureExecutorAccessBatch – Updated policy	Added Amazon SQS `SendMessage` permission to submit jobs to the `atx-dispatcher-queue`, and added CloudFormation visibility for the `AtxDispatcherStack`.	September 2, 2026

Change	Description	Date
AWSTransformInfrastructureExecutorAccessEC2 – Updated policy	Added Amazon SQS `SendMessage` permission to submit jobs to the `atx-dispatcher-queue` , and added CloudFormation visibility for the `AtxDispatcherStack` .	September 2, 2026
AWSTransformInfrastructureExecutorAccessBatch – Updated policy	Narrowed AWS Lambda permissions to an explicit function allowlist, added Amazon S3 write access for batch sidecar output, widened CloudFormation and Amazon EventBridge Scheduler resource scoping for multi-stack deployments, and added support for regional AWS Key Management Service key alias variants.	July 27, 2026
AWSTransformInfrastructureExecutorAccessEC2 – Updated policy	Widened CloudFormation, Amazon EventBridge Scheduler, and AWS Identity and Access Management PassRole resource scoping for multi-stack deployments, added Security Agent infrastructure visibility, and added support for regional AWS Key Management Service key alias variants.	July 27, 2026

Change	Description	Date
AWSTransformSecurityAgentExecutorAccess – Updated policy	Added CloudFormation permissions to discover and describe the Security Agent infrastructure stack, and migrated the Amazon S3 bucket prefix from `kct-security-agent-*` to `atx-security-agent-*` .	July 27, 2026
AWSTransformInfrastructureExecutorAccessBatch – New policy	Added a new AWS managed policy that grants the permissions needed to execute AWS Transform Continuous Modernization assessments and transformations using AWS Batch, including uploading source code, retrieving results, monitoring Batch job status, reading logs, and managing transformation schedules.	July 20, 2026
AWSTransformInfrastructureExecutorAccessEC2 – New policy	Added a new AWS managed policy that grants the permissions needed to execute AWS Transform Continuous Modernization assessments and transformations using Amazon EC2, including uploading source code, retrieving results, monitoring job status, and managing transformation schedules.	July 20, 2026

Change	Description	Date
AWSTransformSecurityAgentExecutorAccess – New policy	Added a new AWS managed policy that grants AWS Transform Continuous Modernization the permissions needed to invoke the AWS Security Agent service for automated code security reviews and remediation, including uploading scan artifacts and retrieving findings.	June 30, 2026
AWSServiceRoleForAWSTransformCustom – Updated policy	Added CloudWatch Logs permissions to allow AWS Transform custom to publish logs to the `/aws/TransformCustom` log group in your account.	May 5, 2026
AWSTransformCustomExecuteTransformations – Updated policy	Added permission to create the AWS Transform custom service-linked role (`AWSServiceRoleForAWSTransformCustom`) to enable CloudWatch metrics emission to customer accounts.	April 27, 2026

Change	Description	Date
AWSTransformCustomManageTransformations – Updated policy	Added permission to create the AWS Transform custom service-linked role (`AWSServiceRoleForAWSTransformCustom`) to enable CloudWatch metrics emission to customer accounts.	April 27, 2026
AWSTransformCustomFullAccess – Updated policy	Added permission to create the AWS Transform custom service-linked role (`AWSServiceRoleForAWSTransformCustom`) to enable CloudWatch metrics emission to customer accounts.	April 7, 2026
AWSServiceRoleForAWSTransformCustom – New policy	Added a new AWS managed policy for the AWS Transform custom service-linked role. This policy allows AWS Transform custom to publish CloudWatch metrics to your account.	March 23, 2026
DBModProvisioningAndMigration – New policy	This policy grants database provisioning and migration capabilities.	March 24, 2026
DBModDiscoveryAndAssessment – New policy	Added a new AWS managed policy that provides comprehensive database modernization discovery and assessment capabilities.	March 24, 2026

Change	Description	Date
AWSTransformCustomFullAccess – New policy	Added a new AWS managed policy that provides full access to AWS Transform custom.	December 5, 2025
AWSTransformCustomExecuteTransformations – New policy	Added a new AWS managed policy that provides access to execute transformations in AWS Transform custom.	December 5, 2025
AWSTransformCustomManageTransformations – New policy	Added a new AWS managed policy that provides access to create, update, read, and delete transformation resources in AWS Transform custom, as well as execute transformations.	December 5, 2025
AWSServiceRoleForAWSTransform – Updated policy	Added permissions to access the AWS Transform service-linked secret used to store the client secret for external identity providers. Added permissions to create a premium support case from the AWS Transform web app.	December 1, 2025
AWSTransformApplicationECSDeploymentPolicy – Updated policy	Added IAM role inspection permissions, ECS service-linked role creation, and KMS permissions for ECR encryption support.	November 22, 2025

Change	Description	Date
AWSTransformApplicationDeploymentPolicy – Updated policy	Added EC2 networking permissions, IAM role inspection permissions, S3 bucket listing permissions, and KMS encryption support for enhanced deployment capabilities.	November 22, 2025
AWSServiceRoleForAWSTransform – Updated policy	Added support for customer managed keys in IAM Identity Center.	September 17, 2025
AWSTransformApplicationDeploymentPolicy – New policy	Added a new AWS managed policy that enables AWS Transform to deploy transformed .NET applications by creating and managing Amazon EC2 instances, CloudFormation stacks, and associated resources.	August 28, 2025
AWSServiceRoleForAWSTransform – Updated policy	Added a new policy.	May 15, 2025

AWS managed policy: AWSServiceRoleForAWSTransform

This policy is attached to the AWSServiceRoleForAWSTransform service-linked role (SLR).

Permissions details

To view the policy permission details see AWSServiceRoleForAWSTransform in the AWS Managed Policy Reference Guide.

AWS managed policy: AWSServiceRoleForAWSTransformCustom

This policy is attached to the AWSServiceRoleForAWSTransformCustom service-linked role (SLR). This role allows AWS Transform custom to publish CloudWatch metrics and logs to your account on your behalf.

Description

This policy includes the following permissions:

- **Amazon CloudWatch** – Allows publishing metrics to CloudWatch under the `AWS/TransformCustom` namespace. This enables monitoring of transformation counts, latencies, and status codes in your CloudWatch dashboards.
- **Amazon CloudWatch Logs** – Allows creating log groups, log streams, setting retention policies, and publishing log events to the `/aws/TransformCustom` log group. This provides visibility into issues encountered during transformations.

AWS managed policy: AWSTransformApplicationDeploymentPolicy

This policy enables AWS Transform to deploy transformed .NET applications by creating and managing Amazon EC2 instances, CloudFormation stacks, and associated resources.

Description

This policy includes the following permissions:

- **CloudFormation** – Allows creating, updating, deleting, and describing CloudFormation stacks with names that start with AWSTransform, and adding or removing stack tags as part of stack create and update operations. Stack operations are restricted to resources tagged with CreatedBy: AWSTransform and limited to the same AWS account.
- **Amazon EC2** – Allows describing VPCs, subnets, security groups, images, instances, route tables, and internet gateways. Permits running, starting, stopping, terminating, and modifying EC2 instances, but only when called through CloudFormation. Tag creation is restricted to specific allowed tag keys and only during CloudFormation operations.
- **AWS Identity and Access Management (IAM)** – Allows getting and passing specific IAM roles for AWSTransform deployment instances. Includes permissions to inspect role policies and attachments. Access is restricted to the same AWS account.

- **Amazon EC2 Systems Manager (SSM)** – Allows retrieving Amazon Linux AMI parameters from the AWS-managed parameter store and sending commands to AWSTransform-tagged instances.
- **Amazon S3** – Allows managing objects in AWSTransform deployment buckets, including listing buckets and getting bucket locations within the same AWS account.
- **AWS Key Management Service (KMS)** – Allows encryption and decryption operations using KMS keys tagged for AWSTransform, with restrictions to S3 and EC2 service usage.

The policy implements least-privilege access through resource-level permissions, tag-based conditions, service control restrictions using `aws:CalledVia`, account-level restrictions, and explicit deny statements to prevent unauthorized tag modifications outside of CloudFormation operations.

Permissions details

To view the policy permission details see AWSTransformApplicationDeploymentPolicy in the AWS Managed Policy Reference Guide.

AWS managed policy: AWSTransformApplicationECSDeploymentPolicy

This policy enables AWS Transform to deploy transformed applications to Amazon ECS by creating and managing ECS clusters, services, tasks, and associated resources.

Description

This policy includes the following permissions:

- **CloudFormation** – Allows creating, updating, deleting, and describing CloudFormation stacks with names that start with AWSTransform, and adding or removing stack tags as part of stack create and update operations. Stack operations are restricted to resources tagged with CreatedBy: AWSTransform and limited to the same AWS account.
- **Amazon ECS** – Allows creating, updating, and deleting ECS clusters, services, and task definitions. Permits running tasks, listing tasks, and describing task status. All operations are restricted to resources with names starting with AWSTransform and tagged with CreatedBy: AWSTransform.
- **AWS Identity and Access Management (IAM)** – Allows getting and passing specific IAM roles for ECS tasks (AWSTransform-Deploy-ECS-Task-Role and AWSTransform-Deploy-ECS-Execution-Role). Includes permissions to inspect role policies and create the ECS service-linked role when needed.

- **Amazon CloudWatch Logs** – Allows creating, deleting, and managing log groups with names starting with /aws/ecs/AWSTransform. Permits retrieving log events for troubleshooting deployed applications.
- **Amazon ECR** – Allows creating container repositories with names starting with awstransform for storing application container images.
- **AWS Key Management Service (KMS)** – Allows creating grants and generating data keys for ECR encryption when using customer-managed KMS keys.

The policy implements least-privilege access through resource-level permissions, tag-based conditions, and account-level restrictions to ensure operations are limited to AWSTransform-managed resources within the same AWS account.

Permissions details

To view the policy permission details see AWSTransformApplicationECSDeploymentPolicy in the AWS Managed Policy Reference Guide.

AWS managed policy: AWSTransformCustomFullAccess

This policy provides full access to AWS Transform custom.

Description

This policy includes the following permissions:

- **AWS Transform Custom** – Allows all actions on all AWS Transform custom resources. This provides complete administrative access to the service.
- **AWS Identity and Access Management (IAM)** – Allows creating the AWS Transform custom service-linked role (`AWSServiceRoleForAWSTransformCustom`). This role is required for AWS Transform custom to emit CloudWatch metrics and logs to your account. The permission is scoped to only allow creating this specific service-linked role.

Permissions details

To view the policy permission details see AWSTransformCustomFullAccess in the AWS Managed Policy Reference Guide.

AWS managed policy: AWSTransformCustomExecuteTransformations

This policy provides access to execute transformations in AWS Transform custom.

Description

This policy includes the following permissions:

- **AWS Transform Custom** – Allows streaming conversations, executing transformations, and managing campaigns. Includes permissions to get campaign details, update campaign repository status, and update campaigns.
- **AWS Identity and Access Management (IAM)** – Allows creating the AWS Transform custom service-linked role (`AWSServiceRoleForAWSTransformCustom`). This role is required for AWS Transform custom to emit CloudWatch metrics and logs to your account. The permission is scoped to only allow creating this specific service-linked role.

Permissions details

To view the policy permission details see AWSTransformCustomExecuteTransformations in the AWS Managed Policy Reference Guide.

AWS managed policy: AWSTransformCustomManageTransformations

This policy provides access to create, update, read, and delete transformation resources in AWS Transform custom, as well as execute transformations.

Description

This policy includes the following permissions:

- **AWS Transform Custom** – Allows comprehensive management of transformation resources including streaming conversations, executing transformations, and managing transformation packages. Permits creating, getting, and deleting transformation package URLs and completing package uploads.
- **Knowledge Management** – Allows listing, getting, deleting, and updating knowledge items and their configurations and status.
- **Campaign Management** – Allows getting campaign details, updating campaign repository status, and updating campaigns.

- **Resource Tagging** – Allows listing, adding, and removing tags for AWS Transform custom resources.
- **AWS Identity and Access Management (IAM)** – Allows creating the AWS Transform custom service-linked role (`AWSServiceRoleForAWSTransformCustom`). This role is required for AWS Transform custom to emit CloudWatch metrics and logs to your account. The permission is scoped to only allow creating this specific service-linked role.

Permissions details

To view the policy permission details see AWSTransformCustomManageTransformations in the AWS Managed Policy Reference Guide.

AWS managed policy: AWSTransformInfrastructureExecutorAccessBatch

This policy grants the permissions needed to execute AWS Transform Continuous Modernization assessments and transformations using AWS Batch. Attach this policy to an IAM role in your account. The AWS Transform Continuous Modernization CLI and agent assume this role to upload source code, retrieve results, monitor Batch job status, read logs, and manage transformation schedules.

Description

This policy includes the following permissions:

- **AWS Lambda** – Allows invoking and retrieving the configuration of specific `atx`-prefixed Lambda functions that orchestrate the transformation workflow, including job triggering, status retrieval, and job listing functions.
- **Amazon S3** – Allows uploading source code to `atx-source-code-*` buckets, downloading transformation results from `atx-custom-output-*` and `atx-ct-output-*` buckets, and writing batch sidecar output to `atx-custom-output-*` buckets.
- **AWS Key Management Service (KMS)** – Allows encryption and decryption of data using the AWS Transform encryption key (`alias/atx-encryption-key`).
- **Amazon CloudWatch Logs** – Allows reading Batch job and Lambda execution log events for transformation monitoring and debugging.
- **Amazon CloudWatch** – Allows getting and listing the AWS Transform CLI dashboard for operational visibility.

- **AWS Batch** – Allows describing and listing compute environments, job queues, job definitions, and job status.
- **CloudFormation** – Allows describing and listing AWS Transform infrastructure stacks, including the Security Agent stack, to check deployment status.
- **Amazon SQS** – Allows sending messages to the `atx-dispatcher-queue` to enqueue transformation jobs through the dispatcher.
- **Resource Groups Tagging API** – Allows retrieving tagged resources to discover AWS Transform infrastructure in your account.
- **AWS Secrets Manager** – Allows retrieving and describing AWS Transform secrets stored under the `atx/` prefix.
- **Amazon EventBridge Scheduler** – Allows managing scheduled transformation jobs within the `atx-ct` schedule group.
- **Amazon EC2** – Allows describing VPC, subnet, security group, route table, and NAT gateway configuration for Batch compute environment networking.
- **AWS Identity and Access Management (IAM)** – Allows reading `ATX`-prefixed and `Atx`-prefixed role configurations to validate infrastructure setup, and passing the `AtxSchedulerInvocationRole` to Amazon EventBridge Scheduler so that it can invoke scheduled jobs.

The policy implements least-privilege access through resource-level permissions, tag-based conditions, the `aws:ResourceAccount` condition, and service-scoped `iam:PassedToService` conditions to ensure operations are limited to AWS Transform resources within the same AWS account.

Permissions details

To view the policy permission details see AWSTransformInfrastructureExecutorAccessBatch in the AWS Managed Policy Reference Guide.

AWS managed policy: AWSTransformInfrastructureExecutorAccessEC2

This policy grants the permissions needed to execute AWS Transform Continuous Modernization assessments and transformations using Amazon EC2. Attach this policy to an IAM role in your account. The AWS Transform Continuous Modernization CLI and agent assume this role to upload source code, retrieve results, monitor job status, and manage transformation schedules.

Description

This policy includes the following permissions:

- **CloudFormation** – Allows describing AWS Transform infrastructure stacks (including the Security Agent stack), stack events, stack resources, and drift detection status, listing stacks, and validating CloudFormation templates before deployment.
- **Amazon SQS** – Allows sending messages to the `atx-dispatcher-queue` to enqueue transformation jobs through the dispatcher.
- **Amazon EC2** – Allows describing instances, AMIs, VPCs, subnets, security groups, key pairs, route tables, NAT gateways, and internet gateways. Permits starting and stopping EC2 instances tagged with `atx-remote-infra: true`.
- **Amazon EC2 Systems Manager (SSM)** – Allows monitoring command execution status, and running commands and sessions on AWS Transform-tagged instances, including running the `AWS-RunShellScript` document.
- **Amazon S3** – Allows managing source code and output artifacts in AWS Transform buckets, including getting, putting, and deleting objects and listing buckets.
- **AWS Key Management Service (KMS)** – Allows encryption and decryption of data using the AWS Transform encryption key (`alias/atx-encryption-key`).
- **AWS Secrets Manager** – Allows retrieving and describing transformation secrets, such as repository credentials, stored under the `atx/` prefix.
- **Amazon EventBridge Scheduler** – Allows managing scheduled transformation jobs within the `atx-ct` schedule group.
- **Resource Groups Tagging API** – Allows retrieving tagged resources to discover AWS Transform infrastructure in your account.
- **AWS Identity and Access Management (IAM)** – Allows passing the `atx-transform-role*` role to Amazon EC2 and the `AtxSchedulerInvocationRole` to Amazon EventBridge Scheduler, and reading role and instance profile configuration for EC2 instances.

The policy implements least-privilege access through resource-level permissions, tag-based conditions, the `aws:ResourceAccount` condition, and service-scoped `iam:PassedToService` conditions to ensure operations are limited to AWS Transform resources within the same AWS account.

Permissions details

To view the policy permission details see AWSTransformInfrastructureExecutorAccessEC2 in the AWS Managed Policy Reference Guide.

AWS managed policy: AWSTransformLandingZoneAgentPolicy

This policy grants the permissions needed to set up AWS landing zones, including account provisioning, organizational governance, and AWS Control Tower configuration. Attach this policy to the IAM role that AWS Transform assumes in your management account when it builds a landing zone.

Description

This policy includes the following permissions:

- **AWS Control Tower** – Allows listing and describing landing zones, baselines, enabled baselines, enabled controls, and the status of baseline and control operations. Permits enabling controls and baselines and tagging Control Tower resources.
- **AWS Organizations** – Allows describing the organization, accounts, organizational units (OUs), and policies, and listing roots, parents, and the OU hierarchy. These read permissions apply to all resources in your organization, so they also work from a delegated administrator account. Permits creating accounts and OUs, moving accounts between OUs, and tagging organization resources. Creating, updating, and attaching service control policies is allowed only when the call is made through AWS Control Tower.
- **CloudFormation** – Allows creating, updating, tagging, and describing landing zone stacks with names that start with `AtxLz`, reading stack events and templates, and creating, describing, listing, and executing change sets. Deleting stacks is not permitted.
- **AWS Service Catalog** – Allows creating, updating, deleting, and listing provisioning artifacts for products in your account, which AWS Control Tower uses to provision accounts.
- **Amazon S3** – Allows creating and managing the `transform-vmware-landing-zone-*` bucket and its objects, including bucket policy, tagging, encryption configuration, and multipart uploads.

The policy implements least-privilege access through resource name prefixes (`AtxLz*` and `transform-vmware-landing-zone-*`), tag-based conditions on `CreatedBy: AWSTransform` and the workspace ID, the `aws:ResourceAccount` condition, and an `aws:RequestedRegion` condition that restricts operations to your target AWS Region. Certain AWS Organizations read-only operations, including describing accounts, organizational units, and policies, listing the

organizational hierarchy, and listing resource tags, are permitted for resources within the same AWS Organization using the `aws:ResourceOrgID` condition instead of restricting access to the principal's account. The `aws:CalledVia` condition limits service control policy changes to calls made by AWS Control Tower.

Permissions details

To view the policy permission details see AWSTransformLandingZoneAgentPolicy in the AWS Managed Policy Reference Guide.

AWS managed policy: AWSTransformNetworkMigrationAgentPolicy

This policy grants the permissions needed to deploy and configure network infrastructure, including VPCs, transit gateways, and route tables, for network migration. Attach this policy to the IAM role that AWS Transform assumes in your target account when it migrates a network topology.

Description

This policy includes the following permissions:

- **AWS Transform MGN (MGN)** – Allows creating, reading, updating, deleting, and tagging network migration definitions and their mapper segments. Permits starting and listing network mappings, analyses, code generations, deployments, and deployed stack deletions.
- **CloudFormation** – Allows creating, updating, deleting, and describing network stacks with names that start with `Nmd`, listing stacks, and managing termination protection.
- **Amazon EC2** – Allows describing network topology, including VPCs, subnets, security groups, route tables, network interfaces, NAT gateways, internet gateways, transit gateways and their attachments and route tables, VPN connections, and prefix lists. Permits creating VPCs, internet gateways, transit gateways, and Elastic IP addresses, and tagging them on creation. Through CloudFormation, permits creating, modifying, and deleting these and other network resources, including subnets, route tables and routes, NAT gateways, network interfaces, security groups and rules, and transit gateway attachments and route tables. Attachments to VPCs in other accounts are limited to accounts in your organization.
- **Reachability Analyzer** – Allows creating and tagging network insights paths, starting analyses, and reading results to validate connectivity between source and target networks. To place analysis probes, permits creating network interfaces and security groups, authorizing security group rules, and deleting those resources, paths, and analyses during cleanup. Includes read-only access to AWS Direct Connect, Elastic Load Balancing, AWS Global Accelerator, and AWS Network Firewall configuration used during path analysis.

- **AWS Resource Access Manager (RAM)** – Allows creating, updating, tagging, and deleting resource shares and their associations to share transit gateways and Systems Manager parameters with other accounts in your organization. Shares that allow external principals are not permitted.
- **AWS Lambda** – Allows creating, tagging, invoking, and deleting `network-migration*` functions that CloudFormation uses as custom resources for transit gateway modifications.
- **AWS Identity and Access Management (IAM)** – Allows creating, tagging, reading, and deleting the `Nmd*modifyTransitGateway*` Lambda execution role, attaching only the `AWSApplicationMigrationNetworkMigrationCustomResource` and `AWSLambdaBasicExecutionRole` policies to it, and passing it to AWS Lambda. Also allows creating the transit gateway service-linked role.
- **Amazon EC2 Systems Manager (SSM)** – Allows reading and writing `/network-migration/*` parameters and their resource policies for cross-account sharing.
- **Amazon CloudWatch Logs** – Allows creating log groups and log streams and publishing log events for `/aws/lambda/network-migration*` functions.
- **Amazon S3** – Allows creating and managing the `transform-vmware-target-*` bucket and the objects in it that store network migration artifacts.
- **AWS Key Management Service (KMS)** – Allows decrypting, describing, and generating data keys with your workspace encryption key, restricted to Amazon S3 usage.
- **AWS Organizations** – Allows listing accounts to discover the accounts in your organization.
- **AWS STS** – Allows assuming the `AWSTransformNetworkMigrationAgentSharingRole_`*`management-or-delegated-admin-account-id`* role in other accounts in your organization for multi-account deployments.

The policy implements least-privilege access through resource name prefixes (`Nmd*`, `network-migration*`, and `transform-vmware-target-*`), tag-based conditions on `CreatedBy` and the workspace ID, the `aws:ResourceAccount` and `aws:ResourceOrgID` conditions, an `aws:RequestedRegion` condition that restricts operations to your target AWS Region, `aws:CalledVia` conditions that limit most network changes to calls made by CloudFormation, and an external ID that includes your workspace ID for cross-account role assumption.

Permissions details

To view the policy permission details see AWSTransformNetworkMigrationAgentPolicy in the AWS Managed Policy Reference Guide.

AWS managed policy: AWSTransformSecurityAgentExecutorAccess

This policy grants AWS Transform Continuous Modernization the permissions needed to invoke the AWS Security Agent service for automated code security reviews and remediation. Attach this policy to an IAM role in your account that the AWS Transform Continuous Modernization CLI and agent assume to upload scan artifacts and retrieve findings.

Description

This policy includes the following permissions:

- **AWS Security Agent** – Allows listing agent spaces, creating code reviews, starting code review jobs, listing and retrieving findings, and starting code remediation. These actions are scoped to agent spaces in your account.
- **Amazon S3** – Allows downloading scan results and findings from the `atx-security-agent-*` bucket, and uploading source code for security scanning to the `security-scans/*` prefix within that bucket.
- **AWS Identity and Access Management (IAM)** – Allows passing the `security-agent-*` role to the AWS Security Agent service so that it can perform the scan.
- **CloudFormation** – Allows listing stacks in the account and describing the `AtxSecurityAgentStack-*` stack to discover and monitor the Security Agent infrastructure deployment status.

The policy implements least-privilege access through resource-level permissions, S3 bucket and key prefixes, the `aws:ResourceAccount` and `s3:ResourceAccount` conditions, and a service-scoped `iam:PassedToService` condition to ensure operations are limited to AWS Security Agent resources within the same AWS account.

Permissions details

To view the policy permission details see AWSTransformSecurityAgentExecutorAccess in the AWS Managed Policy Reference Guide.

AWS managed policy: AWSTransformServerMigrationAgentPolicy

This policy grants the permissions needed to perform server migration operations, including replication, testing, and cutover, using AWS Transform MGN. Attach this policy to the IAM role that AWS Transform assumes in your target account when it migrates servers.

Description

This policy includes the following permissions:

- **AWS Transform MGN (MGN)** – Allows describing and managing source servers, applications, waves, jobs, connectors, and launch and replication configuration templates, including creating, updating, deleting, associating, and tagging them. Permits reading and updating account settings, registering agents, starting tests and cutovers, controlling replication (start, stop, pause, and resume), finalizing cutover, archiving source servers, terminating target instances, configuring source server and template post-launch actions, and importing and exporting server inventory.
- **Amazon EC2** – Allows describing instances, images, volumes, snapshots, VPCs, subnets, security groups, instance types, and launch templates. Permits creating launch template versions and modifying launch templates, restricted to launch templates that carry the `AWSApplicationMigrationServiceManaged` tag, and tagging launch templates and security groups on creation. Through MGN, permits running, starting, stopping, and terminating target instances, modifying instance attributes, creating volumes and snapshots, attaching, detaching, and deleting volumes, deleting launch template versions, and creating security groups and authorizing and revoking their egress rules. Instances can launch into VPCs in your account or into shared VPCs in other accounts in your organization.
- **Amazon FSx** – Allows describing file systems, storage virtual machines, volumes, and snapshots. Through MGN, permits creating volumes and snapshots, and deleting and tagging volumes, when you use FSx for ONTAP as target storage.
- **Amazon EC2 Systems Manager (SSM)** – Allows reading AWS-managed and account documents, sending commands to migrated instances, and starting and monitoring post-launch automation executions. Permits managing `ManagedByAWSApplicationMigrationService-*` parameters.
- **AWS Identity and Access Management (IAM)** – Allows passing the MGN service roles (`AWSApplicationMigrationConversionServerRole`, `AWSApplicationMigrationLaunchInstanceWithDrsRole`, and `AWSApplicationMigrationLaunchInstanceWithSsmRole`) to Amazon EC2.
- **Amazon S3** – Allows creating and managing the `transform-vmware-target-*` bucket and the objects in it that store migration artifacts, and reading the MGN source automation client signature file.
- **AWS Key Management Service (KMS)** – Allows decrypting, describing, and generating data keys with your workspace encryption key, restricted to Amazon S3 usage.

- **Service Quotas** – Allows reading the VPC quota for security groups per network interface through MGN.
- **AWS Secrets Manager** – Allows listing secrets to discover the credentials used for source server discovery.
- **AWS Organizations** – Allows listing accounts to discover the accounts in your organization.
- **AWS STS** – Allows assuming the `AWSTransformRehostSharingRole_`*`management-or-delegated-admin-account-id`* role in other accounts in your organization for multi-account migrations. Through MGN, also allows MGN to assume the same role in member accounts of your organization, using an external ID of *`account-id`*`:AWSTransform`.

The policy implements least-privilege access through resource-level permissions, tag-based conditions on `CreatedBy`, `AWSApplicationMigrationServiceManaged`, and the workspace ID, the `aws:ResourceAccount` and `aws:ResourceOrgID` conditions, an `aws:RequestedRegion` condition that restricts operations to your target AWS Region, `aws:CalledVia` conditions that limit most Amazon EC2, Amazon FSx, Systems Manager, and AWS STS actions to calls made by MGN, and external IDs that include your account ID or workspace ID for cross-account role assumption.

Permissions details

To view the policy permission details see AWSTransformServerMigrationAgentPolicy in the AWS Managed Policy Reference Guide.

AWS managed policy: DBModDiscoveryAndAssessment

This policy provides comprehensive database modernization discovery and assessment capabilities for AWS Transform.

Description

This policy includes the following permissions:

- **Amazon EC2** – Allows describing infrastructure components including instances, VPCs, subnets, security groups, availability zones, VPC endpoints, and internet gateways.
- **Amazon RDS** – Allows describing database instances, clusters, and subnet groups. Allows modifying DB subnet groups within the same AWS account for migration preparation.
- **Amazon RDS Data API** – Allows enabling and disabling HTTP endpoints and executing SQL statements on database clusters tagged for the database modernization project.

- **AWS DMS** – Allows describing endpoints, replication instances, tasks, subnet groups, and orderable instances. Allows listing data providers, instance profiles, and migration projects. Allows describing table statistics, assessment runs, and metadata model operations. Detailed describe and metadata operations are restricted to resources tagged for the database modernization project.
- – Allows listing secrets for discovering database credentials.
- **AWS Identity and Access Management (IAM)** – Allows inspecting specific AWS DMS service roles and their attached policies. Includes access to read AWS-managed AWS DMS policies.
- **AWS Key Management Service (KMS)** – Allows listing key aliases and describing keys. Allows decryption of secrets through integration, restricted to the same AWS account.

The policy implements least-privilege access through resource-level permissions, tag-based conditions, and account-level restrictions to ensure operations are limited to database modernization project resources within the same AWS account.

Permissions details

To view the policy permission details see DBModDiscoveryAndAssessment in the AWS Managed Policy Reference Guide.

AWS managed policy: DBModProvisioningAndMigration

This policy provides database provisioning and migration capabilities for AWS Transform database modernization projects. It includes permissions to create and manage migration infrastructure, provision target databases, and store migration data.

Description

This policy includes the following permissions:

- **AWS DMS** – Allows creating and managing replication subnet groups, instance profiles, data providers, migration projects, endpoints, replication instances, and replication tasks. Includes schema conversion operations such as metadata model import, conversion, export, and assessment. Allows lifecycle management of replication instances (create, delete, modify, reboot) and replication tasks (delete, start, stop, assess). All write operations are restricted to resources tagged for the database modernization project.
- **Amazon RDS** – Allows creating database subnet groups, database clusters, and database instances. All resources must be tagged for the database modernization project.

- – Allows creating, updating, and tagging secrets with the database modernization naming prefix. Allows retrieving secret values and describing secrets for tagged resources.
- **Amazon S3** – Allows creating and managing S3 buckets and objects for migration data storage. Includes bucket tagging, versioning, and object lifecycle operations. Restricted to buckets with the database modernization naming prefix.
- **AWS Identity and Access Management (IAM)** – Allows passing specific AWS DMS service roles to AWS DMS and schema conversion services. Allows creating the Amazon RDS service-linked role required for database operations.

The policy implements least-privilege access through resource-level permissions, tag-based conditions, and account-level restrictions to ensure operations are limited to database modernization project resources within the same AWS account.

Permissions details

To view the policy permission details see DBModProvisioningAndMigration in the AWS Managed Policy Reference Guide.

AWS Transform permissions reference

This section provides information about the APIs used by AWS Transform, and what they do.

Topics

- AWS Transform APIs

AWS Transform APIs

- transform:BatchGetMessage
- transform:BatchGetUserDetails
- transform:CompleteArtifactUpload
- transform:CreateArtifactDownloadUrl
- transform:CreateArtifactUploadUrl
- transform:CreateAssetDownloadUrl
- transform:CreateConnector
- transform:CreateFeedback

- transform:CreateJob
- transform:CreateSession
- transform:CreateWorkspace
- transform:DeleteConnector
- transform:DeleteJob
- transform:DeleteSelfRoleMappings
- transform:DeleteUserRoleMappings
- transform:DeleteWorkspace
- transform:DetectIsAllowedForOperation
- transform:GetConnector
- transform:GetFeedbackQuestions
- transform:GetHitlTask
- transform:GetJob
- transform:GetLoginRedirectUri
- transform:GetUserDetails
- transform:GetUserPreferences
- transform:GetWebAppUrl
- transform:GetWorkspace
- transform:ListArtifacts
- transform:ListConnectors
- transform:ListHitlTasks
- transform:ListJobPlanSteps
- transform:ListJobs
- transform:ListMessages
- transform:ListPlanUpdates
- transform:ListUserRoleMappings
- transform:ListWorklogs
- transform:ListWorkspaces
- transform:PutUserPreferences

- transform:PutUserRoleMappings
- transform:RevokeSession
- transform:SearchUsers
- transform:SearchUsersTypeahead
- transform:SendMessage
- transform:StartJob
- transform:StopJob
- transform:SubmitCriticalHitlTask
- transform:SubmitStandardHitlTask
- transform:TestReleaseTraitPreProd
- transform:TestReleaseTraitProd
- transform:UpdateHitlTask
- transform:UpdateJob
- transform:UpdateWorkspace
- transform:VerifySession

Compliance validation for AWS Transform

Our new AWS sign-up experience is not designed for regulated workloads. If you're using our new AWS sign-up experience, but you want to use AWS for regulated workloads, you can sign up for AWS (advanced) or activate advanced features for your AWS environment.

To learn whether an AWS service is within the scope of specific compliance programs, see AWS services in Scope by Compliance Program and choose the compliance program that you are interested in. For general information, see AWS Compliance Programs.

You can download third-party audit reports using AWS Artifact. For more information, see Downloading Reports in AWS Artifact.

Your compliance responsibility when using AWS services is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. For more information about your compliance responsibility when using AWS services, see AWS Security Documentation.

Resilience in AWS Transform

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

For more information about AWS Regions and Availability Zones, see AWS Global Infrastructure.

In addition to the AWS global infrastructure, AWS Transform offers several features to help support your data resiliency and backup needs.

AWS Transform and interface endpoints (AWS PrivateLink)

You can establish a private connection between your VPC and AWS Transform by creating an *interface VPC endpoint*. Interface endpoints are powered by AWS PrivateLink, a technology that enables you to privately access the AWS Transform console without an internet gateway, NAT device, VPN connection, or Direct Connect connection. Traffic between your VPC and AWS Transform does not leave the Amazon network.

Each interface endpoint is represented by one or more Elastic Network Interfaces in your subnets.

For more information, see Interface VPC endpoints (AWS PrivateLink) in the *Amazon VPC User Guide*.

> **Note**
>
> For AWS Transform custom PrivateLink documentation, see AWS Transform custom and interface endpoints (AWS PrivateLink).

Considerations for AWS Transform VPC endpoints

Before you set up an interface VPC endpoint for AWS Transform, ensure that you review Interface endpoint properties and limitations in the *Amazon VPC User Guide*.

Prerequisites

Before you begin any of the procedures below, ensure that you have the following:

- An AWS account with appropriate permissions to create and configure resources.
- A VPC already created in your AWS account.
- Familiarity with AWS services, especially Amazon VPC and AWS Transform.

Creating an interface VPC endpoint for AWS Transform

You can create a VPC endpoint for the AWS Transform service using either the Amazon VPC console or the AWS Command Line Interface (AWS CLI). For more information, see Creating an interface endpoint in the *Amazon VPC User Guide*.

AWS Transform supports the following VPC endpoint services:

Service name	Endpoint	Note
Control plane	`com.amazonaws.`*`region`*`.transform`	
Agentic API	`com.amazonaws.`*`region`*`.transform-agents`	
Web application	`com.amazonaws.`*`region`*`.api.transform`	Required for the AWS Transform web application. This endpoint **must** have private DNS enabled (the *Enable DNS name* option) so that `api.transform.`*`region`*`.on.aws` resolves to a private IP address in your VPC.

Replace *region* with the AWS Region where your AWS Transform profile is installed, for example, *com.amazonaws.us-east-1.transform*.

> **Note**
>
> If you use the AWS Transform web application, the `api.transform` endpoint is required. For the full setup guide, see Accessing the AWS Transform web application from a VPC.

For more information, see Supported Regions for AWS Transform and Accessing a service through an interface endpoint in the *Amazon VPC User Guide*.

Using an on-premises computer to connect to a AWS Transform endpoint

This section describes the process of using an on-premises computer to connect to AWS Transform through a AWS PrivateLink endpoint in your AWS VPC.

1. Create a VPN connection between your on-premises device and your VPC.
2. Create an interface VPC endpoint for AWS Transform.
3. Set up an inbound Amazon Route 53 endpoint. This will enable you to use the DNS name of your AWS Transform endpoint from your on-premises device.

Accessing the AWS Transform web application from a VPC

When you use AWS PrivateLink to access the AWS Transform API privately from your VPC, the web application requires additional network configuration. It serves static content (HTML, JavaScript, CSS) through CloudFront, which requires internet connectivity. API calls from the web application go through your VPC endpoint and remain fully private.

This guide shows you how to configure controlled internet egress from your VPC so the web application can load while keeping your VPC locked down to only the required domains.

How it works

The AWS Transform web application uses two network paths:

- **API calls** – When you interact with the web application (starting jobs, viewing workspaces, and so on), the browser sends API requests to `api.transform.region.on.aws`. With the `com.amazonaws.region.api.transform` VPC endpoint and private DNS enabled, these requests resolve to a private IP address in your VPC and never leave the AWS network.
- **Static content** – The web application's HTML, JavaScript, and CSS files are served through CloudFront via `tenant-id.transform.region.on.aws`. Loading these files requires internet connectivity because CloudFront content delivery is only available over the public internet.
- **Authentication** – AWS IAM Identity Center sign-in flows use `region.signin.aws`, which also requires internet connectivity.

To enable the web application while maintaining security, you create a controlled egress path using AWS Network Firewall with domain-based filtering. This allows your VPC to reach *only* the specific domains required by the web application while blocking all other internet traffic.

Architecture

The following diagram shows the network path for web application traffic:

```
EC2 Instance / Workspace (Private Subnet)
    |
    | Route: 0.0.0.0/0 # Network Firewall Endpoint
    v
AWS Network Firewall (Firewall Subnet)
    | Allows: *.cloudfront.net, *.transform.<region>.on.aws,
    |         <region>.signin.aws, SSO domains, S3 presigned URLs
    | Blocks: everything else (TLS SNI inspection)
    |
    | Route: 0.0.0.0/0 # NAT Gateway
    v
NAT Gateway (Public Subnet)
    |
    | Route: 0.0.0.0/0 # Internet Gateway
    v
Internet Gateway # CloudFront Edge Locations
```

> **Important**
>
> The Network Firewall must see both directions of traffic (symmetric routing) to perform TLS SNI inspection. You must configure a return route in the NAT Gateway subnet that sends traffic destined for the private subnet back through the firewall. Without this, domain-based filtering rules will not work.

Prerequisites

Before you begin, ensure you have:

- An AWS account with permissions to create VPC resources, Network Firewall, and NAT Gateways.
- A VPC with a private subnet where your instances or workloads run.
- An internet gateway attached to the VPC (or permissions to create one).
- A AWS Transform VPC endpoint for `com.amazonaws.region.api.transform` with private DNS enabled. This is the endpoint used by the web application browser client. For instructions on creating endpoints, see AWS Transform and interface endpoints (AWS PrivateLink).

Configuring the VPC endpoint policy

If your `com.amazonaws.region.api.transform` VPC endpoint has a custom endpoint policy, you must add a statement that allows AWS Transform operations. Without this, the web application will be unable to make API calls through the endpoint.

Add the following statement to your VPC endpoint policy. Replace *AWS_TRANSFORM_PROFILE_ARN* with your AWS Transform service profile ARN, which you can find in the AWS Transform console on the *Settings* page.

```
{
    "Statement": [
        {
            "Sid": "AllowAWSTransform",
            "Effect": "Allow",
            "Principal": "*",
            "Action": "*",
            "Resource": "*",
            "Condition": {
```

```
                "StringEquals": {
                    "aws:PrincipalArn": [
                        "AWS_TRANSFORM_PROFILE_ARN"
                    ]
                }
            }
        }
    ]
}
```

> **Note**
>
> If your VPC endpoint uses the default policy (full access), no changes are needed and you can skip this step.

Setting up controlled internet egress

Complete the following steps to configure Network Firewall with domain-based filtering for the AWS Transform web application.

Step 1: Create the firewall subnet

Create a small /28 subnet dedicated to the Network Firewall endpoint. This subnet must be in the same Availability Zone as your private subnet.

```
aws ec2 create-subnet \
  --vpc-id your-vpc-id \
  --cidr-block firewall-subnet-cidr \
  --availability-zone your-az \
  --region region
```

Step 2: Create a public subnet for the NAT Gateway

```
aws ec2 create-subnet \
  --vpc-id your-vpc-id \
  --cidr-block public-subnet-cidr \
  --availability-zone your-az \
```

```
  --region region
```

Step 3: Configure the public subnet route table

Create a route table for the public subnet that routes internet traffic to the internet gateway.

```
# Create route table
PUB_RTB=$(aws ec2 create-route-table \
  --vpc-id your-vpc-id \
  --region region \
  --query 'RouteTable.RouteTableId' --output text)

# Add default route to internet gateway
aws ec2 create-route \
  --route-table-id $PUB_RTB \
  --destination-cidr-block 0.0.0.0/0 \
  --gateway-id your-igw-id \
  --region region

# Associate with public subnet
aws ec2 associate-route-table \
  --route-table-id $PUB_RTB \
  --subnet-id public-subnet-id \
  --region region
```

Step 4: Create the NAT Gateway

```
# Allocate an Elastic IP
EIP=$(aws ec2 allocate-address --domain vpc \
  --region region --query 'AllocationId' --output text)

# Create NAT Gateway in the public subnet
NAT_ID=$(aws ec2 create-nat-gateway \
  --subnet-id public-subnet-id \
  --allocation-id $EIP \
  --region region \
  --query 'NatGateway.NatGatewayId' --output text)
```

```
# Wait for NAT Gateway to become available (~2 minutes)
aws ec2 wait nat-gateway-available \
  --nat-gateway-ids $NAT_ID --region region
```

Step 5: Create the Network Firewall rule group

Create a stateful rule group that allows traffic only to the domains required by the AWS Transform web application.

```
aws network-firewall create-rule-group \
  --rule-group-name transform-webapp-domains \
  --type STATEFUL \
  --capacity 100 \
  --rule-group '{
    "StatefulRuleOptions": {
      "RuleOrder": "STRICT_ORDER"
    },
    "RulesSource": {
      "RulesSourceList": {
        "Targets": [
          ".cloudfront.net",
          ".transform.region.on.aws",
          "region.signin.aws",
          ".s3.region.amazonaws.com",
          "oidc.region.amazonaws.com",
          "portal.sso.region.amazonaws.com",
          "assets.sso-portal.region.amazonaws.com",
          "directory-id.awsapps.com"
        ],
        "TargetTypes": ["TLS_SNI", "HTTP_HOST"],
        "GeneratedRulesType": "ALLOWLIST"
      }
    }
  }' \
  --region region
```

Replace *region* with the AWS Region where your AWS Transform profile is installed (for example, us-east-1). Replace *directory-id* with your IAM Identity Center directory ID (for example, d-1234567890). You can find your directory ID in the IAM Identity Center console.

The following table explains the allowed domains.

Domain	Purpose
`.cloudfront.net`	CloudFront CDN – serves web application static assets (JavaScript, CSS, images)
`.transform.` *`region`*`.on.aws`	Web application tenant URL – the browser loads the initial page from this domain via CloudFront
`region``.signin.aws`	SSO sign-in redirect page
`.s3.`*`region`*`.amazonaws.com`	S3 presigned URLs – artifact uploads and downloads
`oidc.`*`region`*`.amazonaws.com`	OIDC token exchange for SSO authentication
`portal.sso.` *`region`*`.amazonaws.com`	SSO portal login page
`assets.sso-portal.` *`region`*`.amazonaws.com`	SSO portal static assets (CSS, JavaScript)
`directory-id` `.awsapps.com`	IAM Identity Center portal for your organization

Note

The `.cloudfront.net` wildcard allows traffic to any CloudFront distribution, not only the AWS Transform web application's. A narrower domain filter is not possible because CloudFront edge IPs are shared across distributions and TLS SNI inspection cannot distinguish individual distributions behind the same domain.

Note

API calls to `api.transform.`*`region`*`.on.aws` go through AWS PrivateLink and do not require internet egress. They are not affected by the firewall.

Step 6: Create the firewall policy

The policy must use `STRICT_ORDER` rule evaluation with `drop_established` as the default action. This ensures that any traffic not matching the allowlist is dropped.

```
# Get the rule group ARN
RG_ARN=$(aws network-firewall describe-rule-group \
  --rule-group-name transform-webapp-domains \
  --type STATEFUL --region region \
  --query 'RuleGroupResponse.RuleGroupArn' --output text)

# Create the firewall policy
aws network-firewall create-firewall-policy \
  --firewall-policy-name transform-webapp-policy \
  --firewall-policy "{
    \"StatelessDefaultActions\": [\"aws:forward_to_sfe\"],
    \"StatelessFragmentDefaultActions\": [\"aws:forward_to_sfe\"],
    \"StatefulRuleGroupReferences\": [
      {
        \"ResourceArn\": \"$RG_ARN\",
        \"Priority\": 1
      }
    ],
    \"StatefulEngineOptions\": {
      \"RuleOrder\": \"STRICT_ORDER\"
    },
    \"StatefulDefaultActions\": [\"aws:drop_established\", \"aws:alert_established\"]
  }" \
  --region region
```

Step 7: Create the Network Firewall

```
# Get the policy ARN
FW_POLICY_ARN=$(aws network-firewall describe-firewall-policy \
  --firewall-policy-name transform-webapp-policy \
  --region region \
  --query 'FirewallPolicyResponse.FirewallPolicyArn' --output text)

# Create the firewall
aws network-firewall create-firewall \
```

```
  --firewall-name transform-webapp-firewall \
  --firewall-policy-arn $FW_POLICY_ARN \
  --vpc-id your-vpc-id \
  --subnet-mappings SubnetId=firewall-subnet-id \
  --region region

# Wait for the firewall to become READY (3-5 minutes)
while true; do
  STATUS=$(aws network-firewall describe-firewall \
    --firewall-name transform-webapp-firewall \
    --region region \
    --query 'FirewallStatus.Status' --output text)
  echo "Status: $STATUS"
  if [ "$STATUS" = "READY" ]; then break; fi
  sleep 15
done
```

Step 8: Get the firewall endpoint ID

```
FW_ENDPOINT=$(aws network-firewall describe-firewall \
  --firewall-name transform-webapp-firewall \
  --region region \
  --query "FirewallStatus.SyncStates.\"your-az\".Attachment.EndpointId" \
  --output text)
echo "Firewall endpoint: $FW_ENDPOINT"
```

Step 9: Configure the firewall subnet route table

Route internet-bound traffic from the firewall subnet to the NAT Gateway.

```
FW_RTB=$(aws ec2 create-route-table \
  --vpc-id your-vpc-id \
  --region region \
  --query 'RouteTable.RouteTableId' --output text)

aws ec2 create-route \
  --route-table-id $FW_RTB \
  --destination-cidr-block 0.0.0.0/0 \
  --nat-gateway-id $NAT_ID \
```

```
  --region region

aws ec2 associate-route-table \
  --route-table-id $FW_RTB \
  --subnet-id firewall-subnet-id \
  --region region
```

Step 10: Update the private subnet route table

Route all internet-bound traffic from the private subnet through the firewall.

```
aws ec2 create-route \
  --route-table-id private-subnet-route-table-id \
  --destination-cidr-block 0.0.0.0/0 \
  --vpc-endpoint-id $FW_ENDPOINT \
  --region region
```

If a default route already exists, use `replace-route` instead of `create-route`.

Step 11: Add the symmetric return route (required)

> **Important**
>
> This step is critical. Network Firewall uses TLS SNI inspection and must see both directions of a TCP connection. Add a route in the NAT Gateway subnet's route table that sends return traffic destined for the private subnet back through the firewall.

```
aws ec2 create-route \
  --route-table-id $PUB_RTB \
  --destination-cidr-block private-subnet-cidr \
  --vpc-endpoint-id $FW_ENDPOINT \
  --region region
```

Replace *private-subnet-cidr* with the CIDR block of your private subnet (for example, `10.0.144.0/20`).

Without symmetric routing, the firewall only sees one direction of traffic. The TLS inspection engine cannot extract the Server Name Indication (SNI) from the TLS handshake, and all domain-based rules will fail silently.

Step 12: Remove the IPv6 default route (if present)

If the private subnet route table has an IPv6 default route (`::/0`) pointing directly to an internet gateway, it will bypass the firewall for IPv6-capable destinations. Remove it:

```
aws ec2 delete-route \
  --route-table-id private-subnet-route-table-id \
  --destination-ipv6-cidr-block ::/0 \
  --region region
```

Verification

From an instance in the private subnet, verify the configuration:

```
# Should SUCCEED - web application content via CloudFront (allowed)
curl -vL --connect-timeout 15 \
  'https://tenant-id.transform.region.on.aws'

# Should SUCCEED - API via PrivateLink (does not use firewall)
curl -v --connect-timeout 15 \
  'https://api.transform.region.on.aws/'

# Should FAIL - non-allowlisted domain (blocked by firewall)
curl -v --connect-timeout 15 'https://www.example.com'
# Expected: TLS connection error (firewall drops after SNI inspection)
```

Troubleshooting

API calls to api.transform are blocked by your firewall

The domain `api.transform.region.on.aws` should resolve to a private IP address via the VPC endpoint and should not reach your internet firewall.

- Verify you have created the `com.amazonaws.region.api.transform` endpoint.

- Verify private DNS is enabled on the endpoint:

```
aws ec2 describe-vpc-endpoints \
  --filters "Name=service-name,Values=com.amazonaws.region.api.transform" \
  --query 'VpcEndpoints[*].[State,PrivateDnsEnabled]' \
  --output table
```

  Expected output: `available | True`

The web application does not load (connection timeout)

- Verify the private subnet route table has a `0.0.0.0/0` route pointing to the firewall endpoint.
- Verify the firewall subnet route table has a `0.0.0.0/0` route pointing to the NAT Gateway.
- Verify the NAT Gateway is in an `available` state.

Non-allowlisted domains are not blocked

- Check for an IPv6 `::/0` route pointing to the internet gateway. This bypasses the firewall. Remove it (Step 12).
- Verify symmetric routing is configured. The NAT Gateway subnet route table must have a return route through the firewall for the private subnet CIDR (Step 11).
- Verify the firewall policy uses `STRICT_ORDER` with `drop_established` and `alert_established` as default actions.

Cost considerations

Resource	Approximate cost
Network Firewall	~$0.395/hr (~$288/month) per Availability Zone
NAT Gateway	~$0.045/hr (~$33/month) + data processing fees
Elastic IP (public IPv4)	~$0.005/hr (~$3.60/month)
Network Firewall data processing	$0.065/GB

Resource	Approximate cost
NAT Gateway data processing	$0.045/GB

Estimated base cost is approximately $325/month for a single Availability Zone deployment. For production deployments, deploy the firewall, NAT Gateway, and associated subnets in each Availability Zone where private subnets exist.

Cleanup

To remove all resources created by this guide, run the following commands in reverse order. Replace the placeholders with the resource IDs from your deployment. You can find these values in the AWS Management Console or by using the `describe` commands from the setup steps.

```
# Remove return route from NAT Gateway subnet
aws ec2 delete-route --route-table-id pub-rtb-id \
  --destination-cidr-block private-subnet-cidr \
  --region region

# Remove route from private subnet
aws ec2 delete-route \
  --route-table-id private-subnet-route-table-id \
  --destination-cidr-block 0.0.0.0/0 --region region

# Delete Network Firewall (takes ~5 minutes)
aws network-firewall delete-firewall \
  --firewall-name transform-webapp-firewall \
  --region region

# Delete firewall policy and rule group
aws network-firewall delete-firewall-policy \
  --firewall-policy-name transform-webapp-policy \
  --region region
aws network-firewall delete-rule-group \
  --rule-group-name transform-webapp-domains \
  --type STATEFUL --region region

# Delete NAT Gateway (wait ~5 minutes for full deletion)
aws ec2 delete-nat-gateway --nat-gateway-id nat-gateway-id \
  --region region
```

```
# Release Elastic IP
aws ec2 release-address --allocation-id eip-allocation-id \
  --region region

# Delete subnets
aws ec2 delete-subnet --subnet-id firewall-subnet-id \
  --region region
aws ec2 delete-subnet --subnet-id public-subnet-id \
  --region region
```

Monitoring AWS Transform

Monitoring is an important part of maintaining the reliability, availability, and performance of AWS Transform and your other AWS solutions. AWS provides the following monitoring tools to watch AWS Transform, report when something is wrong, and take automatic actions when appropriate:

- *Amazon CloudWatch* monitors your AWS resources and and the applications you run on AWS in real time. You can collect and track metrics, create customized dashboards, and set alarms that notify you or take actions when a specified metric reaches a threshold that you specify. For example, you can have CloudWatch track CPU usage or other metrics of your Amazon EC2 instances and automatically launch new instances when needed. For more information, see the Amazon CloudWatch User Guide.
- *Amazon CloudWatch Logs* enables you to monitor, store, and access your log files from Amazon EC2 instances, CloudTrail, and other sources. CloudWatch Logs can monitor information in the log files and notify you when certain thresholds are met. You can also archive your log data in highly durable storage. For more information, see the Amazon CloudWatch Logs User Guide.
- *Amazon EventBridge* can be used to automate your AWS services and respond automatically to system events, such as application availability issues or resource changes. Events from AWS services are delivered to EventBridge in near real time. You can write simple rules to indicate which events are of interest to you and which automated actions to take when an event matches a rule. For more information, see Amazon EventBridge User Guide.
- *AWS CloudTrail* captures API calls and related events made by or on behalf of your AWS account and delivers the log files to an Amazon S3 bucket that you specify. You can identify which users and accounts called AWS, the source IP address from which the calls were made, and when the calls occurred. For more information, see the AWS CloudTrail User Guide.

Logging AWS Transform API calls using AWS CloudTrail

AWS Transform is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service in AWS Transform. CloudTrail captures all API calls for AWS Transform as events. The calls captured include calls from the AWS Transform console and code calls to the AWS Transform API operations. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for AWS Transform. If you don't configure a trail, you can still view the most recent events in the CloudTrail console in **Event history**. Using the information collected by CloudTrail, you can determine the request that was

made to AWS Transform, the IP address from which the request was made, who made the request, when it was made, and additional details.

To learn more about CloudTrail, see the AWS CloudTrail User Guide.

AWS Transform information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in AWS Transform, that activity is recorded in a CloudTrail event along with other AWS service events in **Event history**. You can view, search, and download recent events in your AWS account. For more information, see Viewing events with CloudTrail Event history.

For an ongoing record of events in your AWS account, including events for AWS Transform, create a trail. A *trail* enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create a trail in the console, the trail applies to all AWS Regions. The trail logs events from all Regions in the AWS partition and delivers the log files to the Amazon S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see the following:

- Overview for creating a trail
- CloudTrail supported services and integrations
- Configuring Amazon SNS notifications for CloudTrail
- Receiving CloudTrail log files from multiple regions and Receiving CloudTrail log files from multiple accounts

Every event or log entry contains information about who generated the request. The identity information helps you determine the following:

- Whether the request was made with root or AWS Identity and Access Management (IAM) user credentials.
- Whether the request was made with temporary security credentials for a role or federated user.
- Whether the request was made by another AWS service.

For more information, see the CloudTrail userIdentity element.

Understanding AWS Transform log file entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An event represents a single request from any source and includes information about the requested action, the date and time of the action, request parameters, and so on. CloudTrail log files aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.

Notifications

AWS Transform provides email notifications to keep you informed about workspace access changes, job status updates, and pending collaborator requests.

Email notifications

Email notifications are enabled by default, and are sent to users who have read-only or higher permissions on a job. Users can modify their notification preferences. There are three categories of email notifications:

- **Workspace access updates**

  You receive a notification when you are added to a new workspace, and when your role is changed within a workspace.
- **Daily digest emails**

  You receive a daily digest email at 9:00 AM Pacific time summarizing your ongoing jobs if you have active jobs with one or more open required collaborator requests. The email includes a table showing workspace name, job name, job status, collaborator request details, and required actions (limited to 10 rows)

  > **Note**
  >
  > Daily digest emails are only sent for jobs that have required collaborator requests. Optional collaborator requests do not trigger digest emails.
- **Job status updates**

  You receive an email when a job you have access to completes or fails.
- **Collaborator request updates**

You receive an email whenever a task requires your input, review, or approval.

Managing email notifications

You can modify your email notification preferences in the web application by clicking the **Settings** cog in the top right corner of the application, and choosing **Notification settings**.

The notification settings page provides the following controls:

Notification Type	Description	Default Setting
Workspace Access Updates	Receive emails when added to a workspace, or when your role is changed within a workspace	Enabled
Daily Collaborator Request Digest	Receive a daily digest of open collaborator requests for jobs in accessible workspaces	Enabled
Job Completion Updates	Receive emails when a job is completed or fails	Enabled
Collaborator Request Updates	Receive emails when an input or approval is required from you	Disabled

You can use a general setting to enable or disable all notifications, or configure individual notification types separately.

Quotas for AWS Transform

Your AWS account has default quotas, formerly referred to as limits, for each AWS service. Unless otherwise noted, each quota is Region-specific. You can request increases for some quotas, and other quotas cannot be increased.

To view the quotas for AWS Transform, open the Service Quotas console. In the navigation pane, choose **AWS services** and select **AWS Transform**.

To request a quota increase, see Requesting a Quota Increase in the *Service Quotas User Guide*. If the quota is not yet available in Service Quotas use the limit increase form.

Your AWS account has the following quotas related to AWS Transform.

Workload	Dimension	Quota	Adjustable	Description
Platform	Workspaces per AWS Region per AWS Transform-enabled account	100 workspaces for each supported AWS Region	Yes	The maximum number of workspaces per AWS Transform-enabled account per AWS Region
VMWare Migration	Server count for each wave	Each wave: 150 servers	No	The maximum number of servers in a wave
Mainframe	Concurrent jobs for each IAM Identity Center account	10 jobs for each supported Region	Yes	The maximum number of concurrent Mainframe jobs in an IAM Identity Center account
	Lines of code for each job	30,000,000 lines of code for	No	The maximum number of lines

Workload	Dimension	Quota	Adjustable	Description
		each supported region		of code for each job.
	Analysis monthly lines of code	100,000,000 lines of code for each supported region	Yes	The maximum number of lines of code for Analyze Code job objective in a calendar month
	Business document generation monthly lines of code	50,000,000 lines of code for each supported region	Yes	The maximum number of lines of code for Business Documentation Generation job objective in a calendar month
	Technical document generation monthly lines of code	50,000,000 lines of code for each supported region	Yes	The maximum number of lines of code for Technical Documentation Generation job objective in a calendar month
	Transform of lines of code	10,000,000 lines of code for each supported region	Yes	The maximum number of lines of code transformed in a calendar month

Workload	Dimension	Quota	Adjustable	Description
	Decomposition monthly lines of code	50,000,000 lines of code for each supported region	Yes	The maximum number of lines of code for Decomposition job objective in a calendar month
	Reforge monthly lines of code	50,000,000 lines of code for each supported region	Yes	The maximum number of lines of code for Reforge job objective in a calendar month
	Extracted business logic monthly lines of code	50,000,000 lines of code for each supported region	Yes	The maximum number of lines of code for Extracted Business Logic job objective in a calendar month
.NET	.NET monthly lines of code (web & IDE)	1,000,000 lines of code for each supported Region	Yes	The maximum lines of .NET application code that can be transform ed in a calendar month in each supported AWS Region of an IAM Identity Center account

Workload	Dimension	Quota	Adjustable	Description
	Concurrent .NET jobs with the Microsoft Visual Studio Extension for each IAM Identity Center account	10 jobs for each supported Region	Yes	The maximum number of concurrent .NET transformation jobs in each supported AWS Region of an IAM Identity Center account when using the Microsoft Visual Studio Extension .
	Concurren t .NET web jobs for each IAM Identity Center account	5 jobs for each supported Region	Yes	The maximum number of concurrent .NET transformation jobs in each supported AWS Region of an IAM Identity Center account when using the AWS Transform web app.
Custom transformations	Custom transformation definitions	1,000	Yes	The maximum number of custom transformation definitions

Workload	Dimension	Quota	Adjustable	Description
	Custom transformation definition size	10 MB	No	The maximum total size of all files included in the transformation definition, including document references.
	Custom transformation monthly CLI conversation requests	1,000,000	Yes	The maximum number of custom transformation conversation requests made in the CLI allowed in a calendar month
Assessment	Server count limit for each assessment	Each assessment: 30,000 servers	No	The maximum number of servers in a single assessment job

Supported Regions for AWS Transform

> **Note**
>
> If you make a request that requires AWS Transform to retrieve information from an opt-in Region not listed on this page, AWS Transform can make calls to that Region. To manage access to Regions AWS Transform can make calls to, see Security in AWS Transform.

This topic describes the AWS Regions where you can use AWS Transform. For more information about AWS Regions, see Specify which AWS Regions your account can use in the *AWS Account Management Reference Guide*.

Your data might be processed in a different Region from the Region where you use AWS Transform. For information on cross-region processing in AWS Transform, see Cross-region processing. For information on where data is stored during processing, see Data protection.

Supported AWS Regions (enabled by default)

You can create AWS Transform workspaces in the following AWS Regions. These Regions are enabled by default - you don't need to enable them before use. For more information, see Regions that are enabled by default.

- US East (N. Virginia)
- Asia Pacific (Mumbai)
- Asia Pacific (Seoul)
- Asia Pacific (Singapore) - Migration capabilities only
- Asia Pacific (Sydney)
- Asia Pacific (Tokyo)
- Canada (Central)
- Europe (Frankfurt)
- Europe (London)
- South America (São Paulo) - Mainframe modernization capabilities only
- AWS GovCloud (US-West) - Migration capabilities only

> **Note**
>
> AWS GovCloud (US-West) is available only when you use AWS Transform in the AWS GovCloud (US) partition.

The workspace in which you create a job determines the AWS Region of the job. To create a job in a different Region, you must use a different workspace that is in your desired Region.

For VMware projects, the Region of the workspace is used for discovery. However, you can specify a different Region as the migration target. That target Region is where your workloads are hosted when the migration is complete. For more information about AWS Region considerations for VMware migrations, including the list of possible target Regions, see the section called "Supported target Regions".

AWS support for AWS Transform

AWS Transform enables you to create and manage AWS Support cases through the chat if you encounter issues during your transformation jobs. Your cases are automatically enriched with job context and routed to the appropriate support team. If you are an active Countdown Premium customer, your designated engineer will reach out to manage issue resolution. Learn more in AWS Countdown Premium. Otherwise, your case will be investigated by the AWS Transform support team in their support queue.

Prerequisites

Before you can use support case management, you need:

- An active AWS Support plan - Business, Enterprise On-Ramp, or Enterprise
- An existing AWS Transform application

Enabling Support Case Management

1. Open the AWS Transform console.
2. In the navigation pane, choose **Settings**.
3. Under **Support**, choose **Enable Support case management**.
4. Choose **Save changes**.

AWS Transform uses a service-linked role (SLR) to create and manage support cases on your behalf. No additional IAM permissions are required.

Creating a Support Case

To create a support case, use the chat interface:

1. Ask to create a support case. For example:
   - "Create a support case for my failed transformation job"
   - "I need help with job errors, create a case"
2. The chat prompts you for required information:

- A brief description of your issue
- A detailed description

3. Confirm the case creation when prompted.

The case is automatically populated with context from your transformation job, including:

- Job ID and status
- Job metadata
- Job execution logs (work logs)
- Job plan

The case is automatically assigned based on your job type and issue category.

Viewing Support Cases

To view your support cases, prompt the chat:

- "List my support cases"
- "Show me open support cases"
- "What's the status of my cases?"

The chat will display:

- Case ID
- Subject
- Status, such as Open, Pending, Resolved, or Closed
- Last updated date
- Associated job ID

To view details of a specific case:

- "Show me details for case [case-id]"
- "What's the latest update on case [case-id]?"

Adding Communications to a Case

To add a message to an existing case:

1. Ask the chat: "Add a message to case [case-id]"
2. Provide your message when prompted.
3. The chat will confirm the message was added.

You'll receive notifications when AWS Support responds to your case.

Resolving a Case

To resolve a support case:

1. Ask the chat: "Resolve case [case-id]"
2. Confirm the resolution when prompted.

> **Note**
>
> You can only resolve cases that have been addressed by AWS Support.

Disabling Support Case Management

1. Open the AWS Transform console.
2. In the navigation pane, choose **Settings**.
3. Under **Support**, choose **Disable Support case management**.
4. Choose **Save changes**.

> **Note**
>
> Disabling Support case management removes the chat-based case management interface but does not close existing cases. You can still access your cases through the AWS Support Center.

Usage Telemetry

The AWS Transform IDE plugin, AWS Transform Agent Skill, and AWS Transform Kiro Power collect usage telemetry by default during transformation execution. The telemetry consists of different data points, such as, the IDE name (for example, VS Code or Kiro), the AI agent name (for example, Claude Code or OpenAI Codex), and the execution mode (local or remote). This data is used by AWS Transform to prioritize compatibility testing, as well as latency and reliability.

To disable telemetry, tell the agent during your chat session that you do not want telemetry emitted. This opt-out applies only to the current chat session and must be repeated for each new session. If you are running the AWS Transform CLI directly, set the `ATX_DISABLE_TELEMETRY=true` environment variable.

Operational Data

We may collect operational signals regarding your use of AWS Transform, such as which features you use, error logs, and metrics based on your natural language feedback (such as whether you approved proceeding with a transformation plan or provided instructions to iterate on a proposed plan) to provide the service and assess service performance.

Change log

The following table describes notable releases and updates for AWS Transform.

Date	Title	Description	Links
2026-10-16	AWS Transform for migrations (includin g VMware) now available in Asia Pacific (Singapore)	AWS Transform migration capabilit ies are now available in the Asia Pacific (Singapore) Region (ap-southeast-1). You can now create workspaces and run discovery and migration jobs in this Region. Modernization, custom transform ation, and mainframe capabilities remain available only in other supported Regions.	What's New post
2026-10-15	AWS Transform landing zone agent adds IaC-only flow documentation	A new Generate landing zone IaC without deploymen t page describes how to use the AWS Transform landing zone agent for organization analysis, recommend ations, and Infrastru cture as Code	Generate landing zone IaC without deployment

Date	Title	Description	Links
		generation without direct deployment. The page covers when IaC-only mode applies (Delegated Administrator connector or Management Account with IaC-only preference), the workflow, required permissions, limitations, supported IaC formats (AWS CDK, Terraform, and Landing Zone Accelerator), and common use cases.	

Date	Title	Description	Links
2026-10-15	AWS Transform discovery tool adds SQL Server direct database collection and self-update	The discovery tool now connects directly to SQL Server instances to collect database count, Always On availability group membership, cluster role, and read-only replica status. You can also update the discovery tool in place using the new self-update feature, without redeploying the appliance. Additional improvements include VM skip lists for VMware and Hyper-V, and improved handling of non-routable guest addresses.	Documentation

Date	Title	Description	Links
2026-09-30	AWS Transform landing zone agent adds support for additional brownfield scenarios	AWS Transform landing zone agent now supports additional brownfiel d scenarios, including existing AWS Control Tower landing zones and landing zones built with AWS Organizations. You can run a foundatio n gap analysis and expand your landing zone on top of the existing landing zone in your target account.	Foundation discovery and planning

Date	Title	Description	Links
2026-09-29	AWS Transform assessments add Amazon MSK Express for Apache Kafka	AWS Transform migration assessments now assess self-managed Apache Kafka clusters for migration to Amazon MSK Express. AWS Transform recommends a broker configuration and estimates the total cost of ownership across broker hours, storage, data-in, and cross-Availability Zone transfer, along with a compatibility review of the source cluster.	Documentation

Date	Title	Description	Links
2026-09-29	AWS Transform assessments add Savings Plans pricing for Amazon EC2	AWS Transform migration assessments now include Compute Savings Plans and EC2 Instance Savings Plans as pricing options for Amazon EC2 recommendations, alongside On-Demand and Reserved Instances, so you can compare committed-use discounts in your migration business case.	Documentation

Date	Title	Description	Links
2026-09-15	AWS Transform landing zone agent adds Delegated Administrator connector and IaC-only support	AWS Transform landing zone agent now supports connecting through a Delegated Administrator account, allowing organizations with strict governance policies to use AWS Transform without granting direct access to the managemen t account. When using a Delegated Administrator connector, the agent operates in IaC-only mode, generating Infrastru cture as Code artifacts (AWS CDK, Terraform, or Landing Zone Accelerator format) for your own deployment pipeline instead of directly provision ing resources. This provides flexibili ty for organizat ions that prefer to manage landing zone changes through	Connector setup and IaC-only mode

Date	Title	Description	Links
		existing deployment processes.	
2026-09-08	AWS Transform now available in AWS GovCloud (US-West)	AWS Transform migration capabilities are now available in the AWS GovCloud (US-West) Region, so government agencies and regulated organizations can run large-scale migrations in a compliant environment. You can migrate VMware, bare metal, Hyper-V, and database workloads to both the AWS GovCloud (US-East) and AWS GovCloud (US-West) destination Regions. Modernization, custom transformation, and assessment capabilities remain available only in supported commercial Regions.	What's New post

Date	Title	Description	Links
2026-08-31	AWS Transform for migrations adds security group modernization	AWS Transform for migrations now lets you create, configure , and delete security groups and their rules while reviewing your target network. You can associate a rule with a CIDR range or a source VM. You can also enable security group referencing for a specific rule.	Documentation

Date	Title	Description	Links
2026-08-30	AWS Transform for migrations adds Amazon FSx for NetApp ONTAP support	AWS Transform for migrations now supports Amazon FSx for NetApp ONTAP as a generally available block storage target, alongside Amazon Elastic Block Store (Amazon EBS). You can migrate block storage directly to FSx for ONTAP within the same migration wave that handles compute and networks. This removes the need for intermediate storage platforms or separate migration tools. This capability is available in all AWS Transform supported target Regions and where Amazon FSx for NetApp ONTAP is supported.	Configuring FSx for ONTAP as a target storage type, Supported target Regions, FSx for ONTAP supported Regions

Date	Title	Description	Links
2026-08-28	AWS Transform achieves FedRAMP Class C authorization	The AWS Transform service is now in scope for FedRAMP Class C (formerly Moderate baseline) in the US East (N. Virginia) Region. You can use AWS Transform from this region to migrate and modernize resources that are or will be located in any of the US East/US West Regions in order to satisfy FedRAMP Class C complianc e requirements. Additionally, AWS Transform MGN is in scope for FedRAMP Class D, and can be used independently to migrate resources that require FedRAMP Class D compliance.	What's New post, Compliance in Scope

Date	Title	Description	Links
2026-08-24	AWS Transform for migrations adds security posture migration to existing VPCs	AWS Transform for migrations now offers a second network migration option: apply your source security posture to VPCs you have already provision ed. You upload a source network file that contains firewall rules, tag the existing VPCs you want in scope, and AWS Transform matches your source subnets to those VPCs by CIDR and generates the corresponding security groups. You can deploy the security groups directly with AWS Transform or generate Infrastru cture as Code (CloudFormation, AWS CDK, Terraform , or Landing Zone Accelerator).	Documentation

Date	Title	Description	Links
2026-08-17	AWS Transform for migrations adds application alignment recommendation	Guided network recommendations can now identify VPCs that host applications from multiple environments, such as development and production workloads sharing a single VPC. When the environments can be cleanly separated along subnet boundaries, AWS Transform suggests splitting the VPC along that boundary to follow AWS best practices for workload isolation. Subnets and server IP addresses remain unchanged by the split. This recommendation uses application and environment data from your migration plan and is available only after migration planning.	Documentation

Date	Title	Description	Links
2026-08-16	AWS Transform for migrations adds stale security group rule removal	AWS Transform for migrations now identifies unused inbound firewall rules migrated from your on-premises environment and suggests removing them as part of guided network recommendations. This helps you avoid carrying forward security exposure, such as open inbound access, that no longer serves a purpose. Removal is limited to unused ingress rules, where the absence of observed traffic is a reliable signal that a rule is unused.	Documentation

Date	Title	Description	Links
2026-08-15	AWS Transform for VMware rehost enables inventory modifications through chat interface	AWS Transform for VMware rehost now allows you to modify server configurations through the chat interface during the inventory validation step. In addition to editing the inventory file directly, you can specify changes for individual columns on specific source servers without downloading and editing the entire inventory file. This provides a more convenient way to make targeted updates to your server configurations.	Documentation

Date	Title	Description	Links
2026-08-06	AWS Transform for migrations automates post-launch actions	AWS Transform for migrations now automates post-launch actions in the migration workflow. Define actions once at the account level. AWS Transform applies them to each source server across your target accounts through AWS Systems Manager (SSM), using predefined or custom SSM documents . This feature is available in all AWS Regions where AWS Transform is available .	What's New post, Documentation

Date	Title	Description	Links
2026-08-03	AWS Transform continuous modernization is now generally available	AWS Transform continuous modernization is now generally available. You can analyze and remediate technical debt across source code repositories at scale by connectin g GitHub organizat ions, GitLab groups, and Bitbucket workspaces. Run analyses on demand or on a schedule, prioritize findings, and open pull or merge requests with the fixes. This feature is generally available in all AWS Regions where AWS Transform is available .	What's New post, Documentation

Date	Title	Description	Links
2026-08-03	AWS Transform for Windows adds offline SQL Server-to-Aurora conversion	AWS Transform for full-stack Windows modernization now supports offline source transform ation. You can modernize Microsoft SQL Server databases to Amazon Aurora PostgreSQL without a live database connection. Upload SQL Server Data Definition Language (DDL) files to generate a transform ation plan. AWS Transform then converts schemas and code objects, validates functiona l equivalence, and deploys to Aurora PostgreSQL. This feature is available in the US East (N. Virginia) Region.	What's New post, Documentation

Date	Title	Description	Links
2026-07-29	AWS Transform for migrations expands beyond VMware	AWS Transform for migration s now supports migrating virtual and bare metal server environme nts from virtually any source, including VMware, Hyper-V, and other platforms . Migration capabilit ies are available regardless of your source infrastructure: discovery, migration planning, landing zone creation, network migration , and server rehost. The web applicati on workflow has been renamed from 'VMware Migration s' to 'Migrations (including VMware)'.	Documentation

Date	Title	Description	Links
2026-07-29	AWS Transform for VMware rehost adds post-launch actions	AWS Transform for VMware rehost now enables post-launch actions that automate modernization and validation tasks on each source server immediately after it launches as a test or cutover instance. The agent recommends, configures, and runs actions on your behalf using AWS Systems Manager. Actions can be predefined (available in AWS Transform MGN) or custom (created from existing SSM documents), defined at the account level or customized per server.	Documentation

Date	Title	Description	Links
2026-07-10	AWS Transform for migrations adds expert mode	AWS Transform for migrations now offers expert mode, which lets you provide your inputs up front so the agent completes migration steps automatically with minimal back-and-forth.	Documentation

Date	Title	Description	Links
2026-07-03	AWS Transform discovery tool adds Oracle Database support	The AWS Transform discovery tool now collects detailed Oracle metadata directly from your databases across VMware, Hyper-V, and bare metal servers. You can discover Oracle instances through direct SQL connections. The tool collects Container Database/Pluggable Database (CDB/PDB) enumeration, component inventory, and datafile sizing. It also captures topology flags for Real Application Clusters (RAC), Data Guard, and multitenant architecture. If you haven't configured database credentials, the tool falls back to OS-level detection. All Oracle versions are supported.	Oracle Database discovery

Date	Title	Description	Links
2026-06-22	South America (São Paulo) Region support for mainframe modernization agents	AWS Transform for mainframe modernization agents is now available in the South America (São Paulo) Region (sa-east-1). Added sa-east-1 to supported regions and cross-reg ion inference tables.	Supported Regions, Cross-region inference
2026-06-22	AWS Transform for migrations supports localization	AWS Transform for migrations now supports localization. You can change the display language of the web application when working with VMware migration workflows by choosing the Settings icon and selecting your preferred language.	Localization settings

Date	Title	Description	Links
2026-06-18	AWS Transform for Migrations supports all AWS commercial regions as migration targets	AWS Transform for migrations supports all AWS commercial regions as migration targets, excluding Middle East (Bahrain) and Middle East (UAE). A migration target region is the AWS region where migrated resources are deployed, including landing zones, network infrastructure, and server rehosting.	Supported migration target regions
2026-06-18	AWS Transform for Migrations allows you to attach existing ENIs to your launch template	AWS Transform allows you to attach existing Elastic Network Interfaces (ENIs) to your launch template for your migration. AWS Transform displays all existing ENIs found in your target account, and you can tag them so that they are available for use through the launch configuration when instances are launched.	ENI resource tagging for migration

Date	Title	Description	Links
2026-06-18	AWS Transform for Migrations allows you to configure replication and launch settings	AWS Transform allows you to configure your replication settings and launch settings for your migration . You can set the configuration per your target account or across specific source servers.	Replication and launch settings

Date	Title	Description	Links
2026-06-17	Model-to-model migration assessment for generative AI workloads	AWS Transform now offers a model-to-model migration custom transformation. The transformation assesses your generative AI workloads. It produces a migration plan for moving from third-party providers to Amazon Bedrock. The AI-powered agent scans your codebase to identify every AI SDK and model in use. The agent gathers migration requirements through interactive questions. Then it maps models to Amazon Bedrock equivalents with transparent cost comparisons and production-ready code changes. Supports migrations from OpenAI, Google Gemini, direct Anthropic SDK usage, and open-source models	What's New post, Custom transformations

Date	Title	Description	Links
		through LiteLLM or Ollama. Handles direct SDK integrations and framework-wrapped patterns including LangChain, LlamaIndex, CrewAI, LangGraph, and multi-provider routing layers. Available at no additional charge beyond standard AWS Transform pricing.	

Date	Title	Description	Links
2026-06-16	AWS Transform for mainframe delivers traceable reimagine workflow	AWS Transform for mainframe now delivers a connected , traceable reimagine experience from assessment through code generation. Enterprises running z/OS COBOL and PL/I workloads can assess their portfolio to identify discrete business functions, extract business rules, generate developme nt-ready requireme nts, and produce traceable cloud-nat ive code in a single connected workflow. Every requirement traces back to the source code for full auditability.	What's New post, Documentation

Date	Title	Description	Links
2026-06-16	AWS Transform now supports Amazon FSx for NetApp ONTAP as target storage (Public Preview)	You can now migrate block storage workloads to Amazon FSx for NetApp ONTAP in addition to Amazon EBS as part of compute rehosting . AWS Transform replicates block storage data directly to FSx for ONTAP volumes within the same migration wave that handles compute and network.	What's New post, Documentation

Date	Title	Description	Links
2026-06-08	AWS Application Migration Service is now AWS Transform MGN	AWS Application Migration Service (MGN) is now available as AWS Transform MGN, the replication engine powering AWS Transform. You can choose between two rehosting experiences. Use the AWS Transform MGN console for direct control over replication and cutover. Or use the AWS Transform agentic workflow to handle discovery , wave planning, network creation, and rehosting for you. AWS Transform MGN keeps all existing compliance certifica tions and is available in all commercial AWS Regions and both AWS GovCloud (US) Regions.	What's New post, Documentation

Date	Title	Description	Links
2026-06-05	Amazon RDS for SQL Server TCO assessment support	You can now estimate costs when migrating on-premises SQL Server databases to Amazon RDS for SQL Server. Use the AI-powered TCO assessment in AWS Transform to analyze your environment and get optimal database instance recommendations. The assessment supports Bring Your Own Media (BYOM) and License Included (LI) options with Database Savings Plans for up to 20% savings.	What's New post, Documentation
2026-06-04	AWS Transform discovery tool now supported as network migration input source	You can now use the AWS Transform discovery tool as a source for network migration alongside modelizeI T, enabling hybrid network migration s for environments running both VMware and non-VMware workloads.	User Guide

Date	Title	Description	Links
2026-05-27	AWS Transform custom documentation adds client-side skill examples and best practices	The client-side skills documentation now includes practical examples for Dockerfile compliance, database migration safety, Terraform policy enforcement, and API deprecation helpers, along with best practices for building and sharing skills.	Documentation
2026-05-22	New agentic migration assessment capabilities now available with AWS Transform	AWS Transform now offers advanced migration assessment capabilities including what-if scenarios, customizable assumptions, flexible file format support, and multiple new total cost of ownership (TCO) assessment features.	What's New post, Documentation

Date	Title	Description	Links
2026-05-20	AWS Transform detects existing VPCs during network migration review	AWS Transform now detects existing VPCs in your target account during network migration review. You can see existing VPCs alongside your mapped VPCs, identify CIDR conflicts, and resolve them before deployment.	Documentation
2026-05-14	AWS Transform agents now available in Kiro, Claude, Cursor, and Codex	AWS Transform agents are now accessible through a Kiro power, agent plugins, and the AWS Transform MCP server, letting developers consume transformation capabilities directly from their preferred development environment.	What's New post

Date	Title	Description	Links
2026-05-14	AWS Transform now supports IAM role authentication	AWS Transform now supports IAM role authentication. Customers who start using AWS Transform in their IDE or the web app can use their existing AWS credentials to create a Transform environment, workspace, and transformation job.	What's New post
2026-05-14	AWS Transform introduces the agent builder toolkit for building customize d transformation agents	Partners and customers can now build, share, and register specializ ed transformation agents using the agent builder toolkit Kiro power, integrati ng custom tools and workflows with AWS Transform.	What's New post

Date	Title	Description	Links
2026-05-14	AWS Transform adds agentic AI assistant to the AWS Toolkit for Visual Studio	.NET developers can now modernize applications through a conversational, step-by-step guided experience directly in Visual Studio, with full context preserved across IDE and web console.	What's New post
2026-05-14	AWS Transform now supports customer-owned artifact stores	You can configure your own Amazon S3 bucket and optionall y encrypt artifacts with your own AWS KMS key, giving full control over where transformation artifacts are stored.	What's New post, User Guide

Date	Title	Description	Links
2026-05-14	AWS Transform adds migration strategy (7Rs) recommend ations	With AWS Transform , you can now analyze your discovere d inventory and get a migration strategy recommend ation for every server and applicati on. Recommend ations follow the industry-standard seven migration strategies (7Rs). Each recommendation includes a suggested AWS target service, a confidence score, and the reasoning behind it. You can generate an interactive HTML dashboard, PDF, or Microsoft PowerPoint output.	7Rs recommend ations

Date	Title	Description	Links
2026-05-11	AWS Transform adds containerization capability during migrations	AWS Transform now supports replatforming applications to containers during migration, automating Dockerfile generation, container image builds with CVE scanning, and deployment to Amazon ECS or Amazon EKS.	What's New post, User Guide
2026-05-01	AWS Transform now offers BI migration agents for Power BI and Tableau to QuickSight	Partner-built agents from Wavicle Data Solutions convert Power BI and Tableau dashboards to Quick assets within the AWS Transform workflow.	What's New post
2026-04-21	AWS Transform custom is now available in six additional AWS Regions	AWS Transform custom expands to Asia Pacific, Canada, and Europe (London), reaching a total of eight AWS Regions.	What's New post

Date	Title	Description	Links
2026-04-15	AWS Transform adds landing zone creation within migration workflows	Added landing zone creation directly within migration workflows. AWS Transform checks for an existing foundation, recommends Organizational Unit (OU) structures and target accounts, and offers fully automated deployment or Infrastructure as Code (IaC) output (CloudFormation, AWS CDK, or Landing Zone Accelerator (LZA) formats).	What's New post, Documentation

Date	Title	Description	Links
2026-04-15	AWS Transform adds migration-planning diagrams and reports	You can now generate interacti ve diagrams and analytical reports during migration planning. Diagram types include network topology maps, application dependency graphs, wave Gantt charts, and general charts. Report types include risk assessments, 7Rs recommendations, and custom analytica l reports. Outputs are available as interacti ve HTML, PDF, or Microsoft PowerPoint (.pptx) files.	Diagramming and reporting
2026-04-14	AWS Transform is now available in Kiro and VS Code	AWS Transform custom transform ations are now accessible from Kiro and VS Code, with job state shared across IDE, CLI, and web console surfaces.	What's New post

Date	Title	Description	Links
2026-04-10	AWS Transform adds DHCP with security group mapping for network migration	Added support for DHCP with security group mapping during network migration. If you have DHCP-enabled migration workloads , you can now ensure your security rules remain valid after migration. This applies to both in-VPC security group referencing and cross-VPC and cross-account outbound rules via Transit Gateway.	Documentation
2026-03-31	AWS Transform custom announces general availability of automated codebase analysis	The comprehensive codebase analysis transformation reaches general availability, generatin g architecture docs, technical debt reports, and modernization recommendations across any language.	What's New post

Date	Title	Description	Links
2026-03-31	AWS Transform custom introduces new managed transformations to modernize code at scale	Seven new managed transformations added, including Node.js upgrades (general availability), Java optimization, Log4j migration, Angular-to-React, Angular upgrades, and Vue upgrades (early access).	What's New post
2026-03-20	AWS Transform adds public APIs for network migration	Added public APIs for network migration, enabling partner and programmatic access to network migration capabilities.	API Reference
2026-01-14	AWS Transform custom adds AWS PrivateLink support and expands to Europe (Frankfurt)	AWS Transform custom now supports private VPC access via AWS PrivateLink and is available in Europe (Frankfurt) in addition to US East (N. Virginia).	What's New post

Date	Title	Description	Links
2025-12-23	AWS Transform enables network conversion for hybrid data center migrations	AWS Transform for VMware automatically converts hybrid data center network configs (VLANs, IP ranges) to AWS VPCs, subnets, and security groups without manual mapping.	What's New post
2025-12-01	AWS launches AWS Transform custom to accelerate organization-wide modernization	AWS Transform custom reaches general availability, enabling organizations to automate repeatable code transformations at scale via a single CLI command.	What's New post
2025-12-01	AWS Transform adds new agentic AI capabilities for enterprise VMware migrations	AWS Transform's VMware agent can now discover on-premises environments, map dependencies, generate migration wave plans, and migrate across HyperV, Nutanix, KVM, and bare-metal targets.	What's New post

Date	Title	Description	Links
2025-12-01	AWS Transform launches an AI agent for full-stack Windows modernization	A new full-stack Windows agent automates transformation of .NET apps and SQL Server databases to Aurora PostgreSQL, deploying to containers on Amazon ECS or EC2 Linux.	What's New post
2025-12-01	AWS Transform expands .NET transformation capabilities and enhances developer experience	AWS Transform adds .NET 10 support, ASP.NET Web Forms to Blazor porting, and an enhanced interactive IDE experience via AWS Toolkit for Visual Studio.	What's New post
2025-12-01	AWS Transform for mainframe delivers new testing automation capabilities	AWS Transform for mainframe introduces automated test plan generation, test data collection scripts, and test case automation to accelerate mainframe modernization testing.	What's New post

Date	Title	Description	Links
2025-12-01	AWS Transform for mainframe now supports application reimagining	New data and activity analysis plus business logic extraction capabilities enable decomposition of legacy mainframe applications into cloud-native architectures.	What's New post
2025-11-13	AWS Transform automates Landing Zone Accelerator network configuration	AWS Transform for migrations auto-generates LZA-compatible network YAML files, streamlining multi-account cloud infrastructure setup from VMware environments.	What's New post
2025-07-16	AWS Transform for mainframe introduces enhanced code refactoring and business logic capabilities	AWS Transform for mainframe adds general availability of reforge (code restructuring) and expanded application-level business logic extraction to improve code quality.	What's New post

Date	Title	Description	Links
2025-05-15	AWS Transform for VMware is now generally available	AWS Transform for VMware, an agentic AI service automating the full VMware moderniza tion lifecycle from discovery through deployment, reaches general availability.	What's New post
2025-05-15	AWS Transform for mainframe is now generally available	AWS Transform for mainframe, the first agentic AI service for modernizing IBM z/OS applicati ons at scale, reaches general availability with new analysis, decomposition, and job management capabilities.	What's New post

Download this data.

Document history for the AWS Transform User Guide

The following table describes the document history for the AWS Transform User Guide.

Change	Description	Date
AWS Transform availability in AWS GovCloud (US-West)	Added AWS GovCloud (US-West) to the list of supported Regions as a migration-only Region, available when you use AWS Transform in the AWS GovCloud (US) partition . For more information, see Supported Regions.	October 18, 2026
Migration capabilities available in Asia Pacific (Singapore)	Added the Asia Pacific (Singapore) Region (ap-southeast-1), which supports AWS Transform migration capabilities, to the supported Regions and cross-region inference tables. For more information, see Supported Regions and Cross-region processing.	October 16, 2026
Updated discovery tool documentation for this release	Added SQL Server direct database collection, self-update, and VM skip list documentation. SQL Server database credentials now enable collection of database count, Always On availability group, cluster role, and read-only replica information. Self-update lets you update the discovery tool in place without redeployi	October 15, 2026

	ng the appliance. The VM skip list lets you exclude specific virtual machines from VMware and Hyper-V inventory. For more informati on, see Set up the discovery tool and Data collection.	
Added .NET cloud-native code conversion feature	Added the Convert cloud-native code to AWS services page to the .NET section. Also removed the preview designation for VB.NET, WinForms, WPF, and Xamarin projects which are now supported project types.	October 15, 2026
Added PowerShell Transcrip tion logging note to the discovery tool setup guide	Added a note to the Windows prerequisites and WinRM setup sections describing how Windows PowerShel l Transcription logging can cause rapid disk accumulat ion on target servers when agentless collection is enabled, and documenti ng the recommended GPO service account exemption as the mitigation. For more information, see Set up WinRM and WMI.	September 25, 2026

Cross-region inference for AWS Transform Chat in Asia Pacific (Mumbai) and Asia Pacific (Seoul)	Added a notice indicating that, beginning October 7, 2026, AWS Transform Chat processes inference requests originating in the Asia Pacific (Mumbai) (ap-south-1) and Asia Pacific (Seoul) (ap-northeast-2) Regions across all commercial AWS Regions. Also updated the destination Regions in the cross-region inference table. For more information, see Cross-region processing.	September 16, 2026
Added Kerberos trace-logging aliases to the discovery tool release notes	Added a release note for release 1.0.2325 describing the `enablekrb5trace` and `disablekrb5trace` aliases. For more information, see Release notes.	September 16, 2026
.NET ATX CLI and unit test generation	Updated the .NET section with a description of the ATX CLI experience. Added a unit test page describing unit test porting and automatic unit test generation features.	September 10, 2026

AWS GovCloud (US) Regions added as migration target Regions	Added AWS GovCloud (US-East) and AWS GovCloud (US-West) to the list of supported migration target Regions for VMware and other server migrations. These target Regions are available only when you use AWS Transform in the AWS GovCloud (US) partition. For more informati on, see Supported target Regions.	September 8, 2026
Run migrations with the AWS Transform MCP server	Added a topic describing how to plan and run AWS Transform migrations from an MCP-compatible AI assistant , such as Kiro, Claude Code, Cursor, or Cline, by using the AWS Transform Model Context Protocol (MCP) server. The topic covers setup, authentication, the migration workflow, best practices, and example prompts. For more information, see Run migrations with the AWS Transform MCP server.	September 4, 2026

Updated discovery tool documentation for release 1.0.2325	The discovery tool guide now covers disk monitoring and storage management. It also covers using Kerberos with IP addresses. The guide documents the new `unmapped_storage_devices.csv` export file. It also lists the renamed disk throughput columns and the new 95th percentile columns. For more information, see Managing discovery tool storage.	September 3, 2026
Updated .NET agent web application experience	Restructured the .NET agent documentation to reflect an updated web application experience, connecting to repository by Personal Access Token, and transformation mode, checkpoints, and beaming features. Added guidance for choosing between the web application and IDE experiences.	September 3, 2026

Updated AWS managed policies for CloudFormation stack tagging	AWS Transform updated two existing AWS managed policies, `AWSTransformApplicationDeploymentPolicy` and `AWSTransformApplicationECSDeploymentPolicy`, to add permissions for adding and removing CloudFormation stack tags during stack create and update operations. For more information, see AWS Transform updates for AWS managed policies.	September 3, 2026
New AWS managed policies	AWS Transform added three new AWS managed policies: `AWSTransformLandingZoneAgentPolicy`, `AWSTransformNetworkMigrationAgentPolicy`, and `AWSTransformServerMigrationAgentPolicy`. These policies grant the permissions needed to set up AWS landing zones, deploy network infrastructure for network migration, and perform server migration operations. For more information, see AWS Transform updates for AWS managed policies.	September 3, 2026

Network migration adds security group modernization	While reviewing your target network, you can now create, configure, and delete security groups and rules. You can also associate a rule with a CIDR range or source VM, and enable security group referencing for a specific rule. See Review and optimize your network.	August 31, 2026
Migrations adds Amazon FSx for NetApp ONTAP support	AWS Transform for migrations now supports Amazon FSx for NetApp ONTAP as a generally available block storage target, alongside Amazon Elastic Block Store (Amazon EBS). This capability is available in all AWS Transform supported target Regions and where Amazon FSx for NetApp ONTAP is supported. See Configuring FSx for ONTAP as a target storage type.	August 30, 2026
Network migration adds security posture migration to existing VPCs	You can now apply your source security posture to VPCs you have already provisioned in AWS, instead of creating new VPCs. See Apply security posture to existing VPCs.	August 24, 2026

Guided network recommendations add application alignment	Guided network recommendations can now identify VPCs that host applications from multiple environments and suggest splitting them along subnet boundaries to isolate environments such as development and production. See Guided network recommendations.	August 17, 2026
Network migration adds stale security group rule removal	Guided network recommendations can now identify unused inbound firewall rules migrated from your on-premises environment and suggest removing them. See Guided network recommendations.	August 16, 2026
Connectors topic moved to Getting started	The Connectors topic has moved from the top-level navigation to the Getting started section. See Connectors.	August 8, 2026

Continuous modernization adds AWS Transform-managed remote execution	Continuous modernization can now run analyses on AWS Transform-managed infrastructure with nothing to provision in your account, using `atx ct remote analysis --mode aws-managed` . You can also schedule recurring analyses that run on AWS Transform -managed infrastructure without Amazon EventBridge. See Compute options.	August 7, 2026
Continuous modernization adds remote execution, scheduling, and tagging	Continuous modernization now includes `atx ct remote` commands to provision and run analyses and remediations on Amazon EC2 or AWS Batch, `atx ct schedule` commands to run recurring analyses with Amazon EventBridge Scheduler, and resource tagging for attribute-based access control (ABAC). New AWS managed policies support least-privilege remote and security agent execution. The `atx ct server` command is deprecated. See Working with continuous modernization.	July 28, 2026

Continuous modernization portfolio analysis and remediation	Added new chapter documenting the AWS Transform continuous modernization feature (`atx ct` subcommands) for automated code analysis and remediation. See continuous modernization.	June 3, 2026
Brownfield network: existing VPCs visibility	Added existing VPC detection during network migration review. AWS Transform automatically detects VPCs already deployed in your target account and flags CIDR conflicts with your mapped VPCs. See Existing VPCs in your target account.	May 18, 2026
Added skills format, client-side skills, and subagent logs for AWS Transform custom	Transformation definitions now use the skills format (`SKILL.md` with YAML frontmatter and optional `references/` folder). Added client-side skills for extending the agent with additional capabilities, and subagent logs for improved debugging. Requires CLI version 2.0 or later. See Transformation Definitions and Client-Side Skills.	May 11, 2026

Added source code containerization	AWS Transform can now containerize your applications using generative AI. Provide your source code from Git repositories or zip uploads, and AWS Transform analyzes it, generates Docker artifacts, publishes container images to Amazon ECR, generates infrastructure as code, and deploys to Amazon Elastic Container Service or Amazon Elastic Kubernetes Service. See Source code containerization.	May 7, 2026
Added network optimization operations and guided recommendations	Added Step 5: Review and optimize network, with VPC and subnet operations (delete, exclude, merge, split, resize, rename, change IP address) and guided recommendations for naming standardization, scope review, capacity right-sizing, security review, and VPC consolidation. See Review and optimize network.	April 27, 2026
AWS Transform custom available in 6 new regions	Added support for ca-central-1, eu-west-2, ap-northeast-1, ap-northeast-2, ap-southeast-2, and ap-south-1 in AWS Transform custom.	April 21, 2026

Updated connector permissions documentation	Added detailed permissions breakdown for the target AWS account connector, covering S3, MGN, Migration Hub permissions, workload migration, and landing zone infrastructure provisioning. See Connect target AWS accounts.	April 19, 2026
Updated VMware migration job types	Added Discovery and migration planning, Landing zone, and Landing zone, network, and server migration job types. Renamed Network migration only to Network migration and Migration planning to Build migration plan. See VMware migration job types.	April 19, 2026
Restructured Connect target AWS accounts	Restructured the connector setup into three steps: Migration type selection, MAP agreement, and Connector configuration, including supported target regions, account roles, delegated administrator guidance, and IAM roles. See Connect target AWS accounts.	April 19, 2026

Added additional server-migration-only target regions	Documented 14 additional commercial regions available for server migration jobs without network migration , with allow-list requirement until Q3 2026. See Supported target regions.	April 19, 2026
Added Migrate servers chapter	Added new Migrate servers chapter covering the full rehost workflow including account structure, IAM roles, delegated administrator setup, replication agent deployment, testing, and cutover.	April 19, 2026
Added Build landing zone chapter	Added new Build landing zone chapter with connector permissions details for landing zone provisioning.	April 19, 2026
Security groups mapping and referencing	Added security groups mapping strategies (MAP, MAP_DHCP, SKIP) and security group referencing documentation to Migrate network.	April 13, 2026
Auto-conversion for unsupported network file types	Added support for automatic conversion of unsupported network configuration file types in Migrate network.	April 6, 2026
Added KMS key policy for database migration	Added Encryption for database migration section.	March 26, 2026
Documented new AWS managed policy	Added DBModProvisioningAndMigration section.	March 24, 2026

Documented new AWS managed policy	Added DBModDiscoveryAndAssessment section.	March 24, 2026
Network diagram generation	Added optional network diagram generation (Mermaid and PNG formats) to Migrate network.	March 18, 2026
Firewall and SDN configuration file support	Added support for Palo Alto Networks, Fortinet FortiGate, and Cisco ACI configuration files for network migration in Migrate network.	February 11, 2026
Custom tagging for network resources	Added job-level and VPC-level custom tagging support for network migration in Migrate network.	January 28, 2026
Added support documentation	Added AWS support for section.	December 23, 2025
Documented new AWS managed policies	Added AWSTransformCustomFullAccess, AWSTransformCustomExecuteTransformations, and AWSTransformCustomManageTransformations sections.	December 7, 2025
VMware migration	Updated the VMware migration section.	December 1, 2025
Updated AWS managed policy	Updated the AWSServiceRoleForAWSTransform documentation to reflect added permissions.	December 1, 2025

Supported regions	Updated the Supported regions topic.	December 1, 2025
Mainframe modernization	Updated the Mainframe modernization section.	December 1, 2025
Getting Startedy	Updated the Getting Started topic.	December 1, 2025
Full-stack Windows modernization	Added the Full-stack Windows modernization topic, which includes the updated .NETsection and the new SQL Server modernization section.	December 1, 2025
Discovery tool	Added the Discovery tool section.	December 1, 2025
Updated AWS managed policy	Updated the AWSTransformApplicationECSDeploymentPolicy with expanded IAM role inspection permissions, ECS service-linked role creation, and KMS permissions for ECR encryption support.	November 22, 2025
Updated AWS managed policy	Updated the AWSTransformApplicationDeploymentPolicy with additional EC2 networking, IAM role inspection, S3 bucket management, and KMS encryption permissions.	November 22, 2025
Discovery collector	Added the AWS Transform discovery tool> documentation.	October 31, 2025

VMware migration architecture diagram added	Added a VMware migration architecture diagram in Migrate network.	October 23, 2025
VMware migration results update	Updated migration results information in Migrate network.	October 12, 2025
VMware migration results update	Updated migration results information in Migrate network.	October 12, 2025
S3 connector documentation	Added S3 connector documentation for .NET web app.	October 8, 2025
New user connectors topic	Added user connectors topic.	September 21, 2025
Updated AWS managed policy	Updated the AWSServiceRoleForAWSTransform documentation.	September 17, 2025
Storage assessment support	Added NetApp Data Infrastructure Insights (DII) file support for storage assessments.	September 2, 2025
Target regions update	Added three supported target regions.	August 31, 2025
Managed policy update	Added IAM Identity Center read-only permisions to be used by AWSServiceRoleForAWSTransform.	August 29, 2025
Documented new AWS managed policy	Added AWSTransformApplicationDeploymentPolicy section.	August 28, 2025

interface endpoints	Added interface endpoints topic.	July 9, 2025
Initial release	Initial release of the AWS Transform User Guide.	May 15, 2025