Table of Contents

What Is Amazon AppStream 2.0? ........................................................................................................... 1
  Features ........................................................................................................................................ 1
  Key Concepts ............................................................................................................................ 2
  How to Get Started .................................................................................................................. 2
  Accessing AppStream 2.0 ........................................................................................................... 3
Setting Up ......................................................................................................................................... 4
  Sign Up for AWS ....................................................................................................................... 4
Get Started: Set Up With Sample Applications ............................................................................... 5
  Step 1: Set Up a Sample Stack, Choose an Image, and Configure a Fleet ................................... 5
  Step 2: Provide Access to Users .............................................................................................. 8
Networking and Access .................................................................................................................... 10
  Internet Access ......................................................................................................................... 10
  VPC Requirements ................................................................................................................... 11
    VPC Setup Recommendations ............................................................................................... 11
    Configure a VPC with Private Subnets and a NAT Gateway ............................................... 12
    Configure a VPC with a Public Subnet ................................................................................... 18
    Use the Default VPC and Public Subnet .................................................................................. 20
Amazon S3 VPC Endpoints ............................................................................................................... 21
AppStream 2.0 Connections to Your VPC ...................................................................................... 21
  Network Interfaces .................................................................................................................... 22
  Management Network Interface IP Address Range and Ports ................................................... 22
  Customer Network Interface Ports ........................................................................................... 22
User Connections to AppStream 2.0 ............................................................................................... 23
  Bandwidth Recommendations .................................................................................................. 23
  IP Address and Port Requirements .......................................................................................... 24
  Allowed Domains ..................................................................................................................... 24
Image Builders ............................................................................................................................... 26
  Launch an Image Builder ......................................................................................................... 26
  Image Builder Actions .............................................................................................................. 28
  Instance Metadata for Image Builders ...................................................................................... 29
  Base Image Versions ................................................................................................................. 29
Images ............................................................................................................................................... 35
  Default Settings and Application Launch Performance ........................................................... 35
    Creating Default Application and Windows Settings ......................................................... 35
    Optimizing the Launch Performance of Your Applications .................................................. 36
Manage AppStream 2.0 Agent Versions ........................................................................................ 36
  Create an Image That Always Uses the Latest Version of the AppStream 2.0 Agent .......... 37
  Create an Image That Uses a Specific Version of the AppStream 2.0 Agent ......................... 37
  Create an Image That Uses a Newer Version of the AppStream 2.0 Agent .............................. 38
AppStream 2.0 Agent Versions ...................................................................................................... 39
Tutorial: Create a Custom Image by Using the Console ................................................................. 47
  Step 1: Install Applications on the Image Builder ................................................................. 47
  Step 2: Create an AppStream 2.0 Application Catalog ......................................................... 48
  Step 3: Create Default Application and Windows Settings ................................................... 48
  Step 4: Test Applications ......................................................................................................... 49
  Step 5: Optimize Applications ................................................................................................. 50
  Step 6: Finish Creating Your Image ........................................................................................ 50
  Step 7 (Optional): Tag and Copy an Image ............................................................................... 51
  Step 8: Clean Up ...................................................................................................................... 52
Administer Your Images .................................................................................................................. 52
  Delete a Private Image .............................................................................................................. 53
  Copy an Image That You Own to Another AWS Region ....................................................... 53
  Share an Image That You Own With Another AWS Account ............................................... 53
Specify a Default Display Language ................................................................. 155
Specify a Default System Locale ................................................................. 157
Specify a Default User Locale ................................................................. 157
Specify a Default Input Method ................................................................. 158
Special Considerations for Application Settings Persistence ................. 159
Special Considerations for Japanese Language Settings ......................... 160
Enable Your Users to Configure Their Regional Settings ....................... 160
Supported Locales ..................................................................................... 161
Enable Regional Settings for Your AppStream 2.0 Users ......................... 161
Manage App Entitlement ............................................................................. 163
Example API Operations WorkFlow ........................................................ 163
Use the Dynamic Application Framework .................................................. 165
About the Dynamic Application Framework ............................................. 165
Dynamic Application Framework Thrift Definitions and Named Pipe name .... 165
API Actions for Managing App Entitlement ............................................. 167
Enable and Test Dynamic App Providers .................................................. 170
Enable Dynamic App Providers .............................................................. 170
Test Dynamic App Providers ................................................................ 171
Third-Party Dynamic App Providers ....................................................... 171
Provide User Access .................................................................................. 173
Provide Access Through a Web Browser ................................................... 173
Dual-Monitor Support ........................................................................... 173
Touchscreen Device Support ................................................................ 174
Provide Access Through the Client .......................................................... 174
Requirements and Features ................................................................ 175
Install and Configure the AppStream 2.0 Client ..................................... 176
AppStream 2.0 Client Versions ................................................................. 197
Tagging Your Resources ........................................................................... 201
Tagging Basics ......................................................................................... 201
Tag Restrictions ..................................................................................... 202
Adding Tags during Resource Creation in the AppStream 2.0 Console ...... 202
Adding, Editing, and Deleting Tags for Existing Resources in the AppStream 2.0 Console 202
Working with Tags by Using the AppStream 2.0 API, an AWS SDK, or AWS CLI 203
Monitoring and Reporting ........................................................................ 205
Monitoring Resources ............................................................................. 205
Viewing Fleet Usage Using the Console ..................................................... 205
AppStream 2.0 Metrics and Dimensions .................................................... 205
Usage Reports ........................................................................................ 207
Enable Usage Reports ........................................................................... 207
Usage Reports Fields ............................................................................. 209
Create Custom Reports ........................................................................ 213
Logging AppStream 2.0 API Calls ............................................................ 218
AppStream 2.0 Information in CloudTrail ................................................ 219
Example: AppStream 2.0 Log File Entries .............................................. 220
Security .................................................................................................... 222
Data Protection ......................................................................................... 222
Encryption at Rest .................................................................................. 223
Encryption in Transit .............................................................................. 223
Identity and Access Management ............................................................ 224
Network Access ..................................................................................... 224
Access to AppStream 2.0 Resources ....................................................... 224
Access to Application Auto Scaling ....................................................... 229
Access to the S3 Bucket for Home Folders and Application Settings Persistence 230
Access to Applications and Scripts on Streaming Instances ................. 232
Logging and Monitoring ......................................................................... 235
Compliance Validation ............................................................................ 236
Resilience ............................................................................................... 237
Infrastructure Security ............................................................................................................................. 237
Network Isolation ................................................................................................................................. 237
Isolation on Physical Hosts .................................................................................................................. 238
Controlling Network Traffic .............................................................................................................. 238
Authentication of Corporate Users .................................................................................................. 239
Interface VPC Endpoints .................................................................................................................. 249
FIPS Endpoints .................................................................................................................................... 252
Security Groups ..................................................................................................................................... 253
Update Management .......................................................................................................................... 254
Troubleshooting .................................................................................................................................... 256
General Troubleshooting ..................................................................................................................... 256
SAML federation is not working. The user is not authorized to view AppStream 2.0 applications. 256
After federating from an ADFS portal, my streaming session doesn't start. I am getting the error "Sorry connection went down". 257
I get an invalid redirect URI error. ...................................................................................................... 257
My image builders and fleets never reach the running state. My DNS servers are in a Simple AD directory. 257
My stack's home folders aren't working correctly. .............................................................................. 257
My users can't access their home folder directory from one of our applications. .......................... 257
I've enabled application settings persistence for my users, but their persistent application settings aren't being saved or loaded. ................................................................. 258
I've enabled app settings persistence for my users, but for certain streaming apps, my users' passwords aren't persisting across sessions. ................................................................. 258
Google Chrome data is filling the VHD file that contains my users' persistent applications. This is preventing their settings from persisting. How can I manage the Chrome profile? ......................................................... 259
My users can't copy and paste between their local device and their streaming session. .................. 259
Some keyboard shortcuts aren't working for users during their streaming sessions. ....................... 260
Troubleshooting Image Builders ......................................................................................................... 260
I cannot connect to the internet from my image builder. .................................................................. 261
When I tried installing my application, I see an error that the operating system version is not supported. ................................................................................................................................................. 261
I want to use a Windows PowerShell script to open my applications. .............................................. 261
I want to make ClickOnce applications available to users. .............................................................. 261
When I connect to my image builder, I see a login screen asking me to enter Ctrl+Alt+Delete to log in. However, my local machine intercepts the keystrokes. ............................................................ 262
When I switched between admin and test modes, I saw a request for a password. I don't know how to get a password. ................................................................................................................................................. 262
I get an error when I add my installed application. ........................................................................... 262
I accidentally quit a background service on the image builder and got disconnected. I am now unable to connect to that image builder. .............................................................................................................. 263
The application fails to launch in test mode. ....................................................................................... 263
The application could not connect to a network resource in my VPC. ................................................ 263
I customized my image builder desktop, but my changes are not available when connecting to a session after launching a fleet from the image I created. ......................................................... 263
My application is missing a command line parameter when launching. ........................................... 263
I am unable to use my image with a fleet after installing an antivirus application. ......................... 264
My image creation failed. .................................................................................................................... 264
The Image Assistant create-image operation failed with an error message that access to the PrewarmManifest.txt is denied ................................................................................................................. 264
Troubleshooting Fleets ......................................................................................................................... 264
My applications won't work correctly unless I use the Internet Explorer defaults. How do I restore the Internet Explorer default settings? .............................................................................................. 264
I need to persist environment variables across my fleet instances. ................................................... 264
I want to change the default Internet Explorer home page for my users. ......................................... 265
When my users end a streaming session and then start a new one, they see a message that says no streaming resources are available. ......................................................................................... 266
Troubleshooting Active Directory Domain Join ................................................................. 266
  My image builders and fleet instances are stuck in the PENDING state. ......................... 266
  My users aren't able to log in with the SAML application. ........................................... 266
  My fleet instances work for one user but don't cycle correctly. ..................................... 267
  My user Group Policy objects aren't applying successfully. ....................................... 267
  My AppStream 2.0 streaming instances aren't joining the Active Directory domain. .......... 267
User login is taking a long time to complete on a domain-joined streaming session. ........... 268
  My users can't access a domain resource in a domain-joined streaming session but they can access the resource from a domain-joined image builder. ........................................... 268
Troubleshooting Notification Codes ................................................................................. 268
  Active Directory Internal Service ................................................................................. 269
  Active Directory Domain Join ...................................................................................... 269
Quotas .............................................................................................................................. 271
Guidance for AppStream 2.0 Users ................................................................................... 272
  Access Methods and Clients ......................................................................................... 272
    Web Browser Access .................................................................................................. 272
    Client for Windows .................................................................................................... 275
File Storage Options ......................................................................................................... 278
  Use Home Folders ........................................................................................................ 278
  Use Google Drive ........................................................................................................ 279
  Use OneDrive for Business ......................................................................................... 281
Regional Settings ................................................................................................................ 284
Document History ............................................................................................................. 285
  Earlier Updates ............................................................................................................. 291
What Is Amazon AppStream 2.0?

Amazon AppStream 2.0 is a fully managed application streaming service that provides users with instant access to their desktop applications from anywhere. AppStream 2.0 manages the AWS resources required to host and run your applications, scales automatically, and provides access to your users on demand. AppStream 2.0 provides users access to the applications they need on the device of their choice, with a responsive, fluid user experience that is indistinguishable from natively installed applications.

With AppStream 2.0, you can easily add your existing desktop applications to AWS and enable your users to instantly stream them. Windows users can use either the AppStream 2.0 client or an HTML5-capable web browser for application streaming. You can maintain a single version of each of your applications, which makes application management easier. Your users always access the latest versions of their applications. Your applications run on AWS compute resources, and data is never stored on users’ devices, which means they always get a high performance, secure experience.

Unlike traditional on-premises solutions for desktop application streaming, AppStream 2.0 offers pay-as-you-go pricing, with no upfront investment and no infrastructure to maintain. You can scale instantly and globally, ensuring that your users always have the best possible experience.

For more information, see AppStream 2.0.

Features

Using Amazon AppStream 2.0 provides the following advantages:

Access desktop applications securely from any supported device

Your desktop applications can be accessed securely through an HTML5-capable web browser on Windows and Linux PCs, Macs, Chromebooks, iPads, and Android tablets. Or, for supported versions of Windows, the AppStream 2.0 client can be used for application streaming.

Secure applications and data

Applications and data remain on AWS — only encrypted pixels are streamed to users. Applications run on an AppStream 2.0 instance dedicated to each user so that compute resources are not shared. Applications can run inside your own virtual private cloud (VPC), and you can use Amazon VPC security features to control access. This enables you to isolate your applications and deliver them in a secure way.

Consistent, scalable performance

AppStream 2.0 runs on AWS with access to compute capabilities not available on local devices, which means that your applications run with consistently high performance. You can instantly scale locally and globally, and ensure that your users always get a low-latency experience. Unlike on-premises solutions, you can quickly deploy your applications to the AWS region that is closest to your users, and start streaming with no incremental capital investment.

Integrate with your IT environment

Integrate with your existing AWS services and your on-premises environments. By running applications inside your VPCs, your users can access data and other resources that you have in AWS. This reduces the movement of data between AWS and your environment and provides a faster user experience.

Integrate with your existing Microsoft Active Directory environment. This enables you to use existing Active Directory governance, user experience, and security policies with your streaming applications.
Configure identity federation, which allows your users to access their applications using their corporate credentials. You can also allow authenticated access to your IT resources from applications running on AppStream 2.0.

**Choose the fleet type that meets your needs**

There are two types of fleets:

- **Always-On** — Your instances run all the time, even when no users are streaming applications. Use an Always-On fleet to provide your users with instant access to their applications.
- **On-Demand** — Your instances run only when users are streaming applications. Idle instances that are available for streaming are in a stopped state. Use an On-Demand fleet to optimize your streaming charges and provide your users with access to their applications after a 1-2 minute wait.

For more information, see Amazon AppStream 2.0 Pricing.

---

**Key Concepts**

To get the most out of AppStream 2.0, be familiar with the following concepts:

**image builder**

An *image builder* is a virtual machine that you use to create an image. You can launch and connect to an image builder by using the AppStream 2.0 console. After you connect to an image builder, you can install, add, and test your applications, and then use the image builder to create an image. You can launch new image builders by using private images that you own.

**image**

An *image* contains applications that you can stream to your users, and default Windows and application settings to enable your users to get started with their applications quickly. AWS provides base images that you can use to create images that include your own applications. After you create an image, you can't change it. To add other applications, update existing applications, or change image settings, you must create a new image. You can copy your images to other AWS Regions or share them with other AWS accounts in the same Region.

**fleet**

A *fleet* consists of fleet instances (also known as streaming instances) that run the image that you specify. You can set the desired number of streaming instances for your fleet and configure policies to scale your fleet automatically based on demand. Note that one user requires one instance.

**stack**

A *stack* consists of an associated fleet, user access policies, and storage configurations. You set up a stack to start streaming applications to users.

**streaming instance**

A *streaming instance* (also known as a fleet instance) is an EC2 instance that is made available to a single user for application streaming.

**user pool**

Use the *user pool* to manage users and their assigned stacks.

---

**How to Get Started**

If you are using AppStream 2.0 for the first time, you can use the Try it Now feature or follow the Get Started with Amazon AppStream 2.0: Set Up With Sample Applications (p. 5) tutorial (both are available in the AppStream 2.0 console).
Try It Now provides you with a free trial experience that allows you to easily start desktop applications from your desktop browser. The Getting Started tutorial enables you to set up application streaming by using sample applications or your own applications. If you decide to start by using sample applications, you can always add your own applications later.

For more information about these two options, see Amazon AppStream 2.0 FAQs.

When you use the service for the first time, AppStream 2.0 creates an AWS Identity and Access Management (IAM) role to create and manage AppStream 2.0 resources on your behalf.

To use the Try It Now feature

2. Choose Try it now.
3. Sign in using your AWS account credentials, if requested.
4. Read the terms and conditions and choose Agree and Continue.
5. From the list of applications shown, select one to try.

To run the Getting Started tutorial

2. Choose Get Started.
3. Select the option to learn more about AppStream 2.0 resources.

Accessing AppStream 2.0

You can work with AppStream 2.0 using any of the following interfaces:

AWS Management Console

The console is a browser-based interface to manage AppStream 2.0 resources. For more information, see Get Started with Amazon AppStream 2.0: Set Up With Sample Applications (p. 5).

AWS command line tools

AWS provides two sets of command line tools: the AWS Command Line Interface (AWS CLI) and the AWS Tools for Windows PowerShell. To use the AWS CLI to run AppStream 2.0 commands, see Amazon AppStream 2.0 Command Line Reference.

AWS SDKs

You can access AppStream 2.0 from a variety of programming languages. The SDKs automatically take care of tasks such as the following:

• Setting up an AppStream 2.0 stack or fleet
• Getting an application streaming URL to your stack
• Describing your resources

For more information, see Tools for Amazon Web Services.
Setting Up for Amazon AppStream 2.0

Complete the following tasks to get set up for Amazon AppStream 2.0.

Sign Up for AWS

When you sign up for AWS, your AWS account is automatically signed up for all services, including AppStream 2.0. You are charged only for the services that you use.

If you have an AWS account already, skip to the next task. If you don't have an AWS account, use the following procedure to create one.

To create an AWS account

2. Follow the online instructions.
   
   Part of the sign-up procedure involves receiving a phone call and entering a verification code on the phone keypad.
Get Started with Amazon AppStream 2.0: Set Up With Sample Applications

To stream your applications, Amazon AppStream 2.0 requires an environment that includes a fleet that is associated with a stack, and at least one application image. This tutorial describes how to configure a sample AppStream 2.0 environment for application streaming and give users access to that stream.

**Note**
For additional guidance in learning how to get started with AppStream 2.0, see the Amazon AppStream 2.0 Getting Started Guide. This guide describes how to install and configure two applications, perform foundational administrative tasks using the AppStream 2.0 console, and provision an Amazon Virtual Private Cloud by using a provided AWS CloudFormation template.

**Tasks**
- Step 1: Set Up a Sample Stack, Choose an Image, and Configure a Fleet (p. 5)
- Step 2: Provide Access to Users (p. 8)
- Resources (p. 9)

**Step 1: Set Up a Sample Stack, Choose an Image, and Configure a Fleet**

Before you can stream your applications, you need to set up a stack, choose an image that has applications installed, and configure a fleet. In this step, you use a template to help simplify these tasks.

**To set up a sample stack, choose an image, and configure a fleet**

2. Choose **Get Started** if you are new to the console, or **Quick Links** from the left navigation menu. Choose **Set up with sample apps**.
3. For **Step 1: Stack Details**, keep the default stack name or enter your own. Optionally, you can specify the following:
   - **Display name** — Type a name to display for the stack (maximum of 100 characters).
   - **Description** — Keep the default description or enter your own (maximum of 256 characters).
   - **Redirect URL** — Specify a URL to which users are redirected after their streaming sessions end.
   - **Feedback URL** — Specify a URL to which users are redirected after they click the Send Feedback link to submit feedback about their application streaming experience. If you do not specify a URL, this link is not displayed.
   - **Tags** — Choose **Add Tag**, and type the key and value for the tag. To add more tags, repeat this step as needed. For more information, see **Tagging Your Amazon AppStream 2.0 Resources** (p. 201).
4. For **Step 1: Stack Details**, under **Network Access Endpoints (Optional)**, you can create a private link, which is an interface VPC endpoint (interface endpoint), in your virtual private cloud (VPC). To
start creating the interface endpoint, select **Create PrivateLink**. Selecting this link opens the VPC console. To finish creating the endpoint, follow steps 3 through 6 in *To create an interface endpoint*, in Creating and Streaming from Interface VPC Endpoints (p. 250).

After you create the interface endpoint, you can use it to keep streaming traffic within your VPC.

5. Choose **Next**.

6. For **Step 2: Choose Image**, a sample image is already selected. The image contains pre-installed open-source applications for evaluation purposes. Choose **Next**.

7. For **Step 3: Configure Fleet**, we recommend that you keep the default values and choose **Next**. You can change most of these values after fleet creation.

   - **Choose instance type** — Choose the instance type that matches the performance requirements of your applications. All streaming instances in your fleet launch with the instance type that you select. For more information, see AppStream 2.0 Instance Families (p. 80).
   - **Fleet type** — Choose the fleet type that suits your use case. The fleet type determines its immediate availability and how you pay for it.
   - **Maximum session duration in minutes** — Choose the maximum amount of time that a streaming session can remain active. If users are still connected to a streaming instance five minutes before this limit is reached, they are prompted to save any open documents before being disconnected. After this time elapses, the instance is terminated and replaced by a new instance.
   - **Disconnect timeout in minutes** — Choose the amount of time that a streaming session should remain active after users disconnect. If users try to reconnect to the streaming instance after a disconnection or network interruption within this time interval, they are connected to the previous session. Otherwise, they are connected to a new session with a new instance. If you associate a stack with a fleet for which a redirect URL is specified, after users’ streaming sessions end, the users are redirected to that URL.

   If a user ends the session by choosing **End Session** on the streaming session toolbar, the disconnect timeout does not apply. Instead, the user is prompted to save any open documents, and then immediately disconnected from the streaming instance.

   - **Idle disconnect timeout in minutes** — Choose the amount of time that users can be idle (inactive) before they are disconnected from their streaming session and the **Disconnect timeout in minutes** time interval begins. Users are notified before they are disconnected due to inactivity. If they try to reconnect to the streaming session before the time interval specified in **Disconnect timeout in minutes** has elapsed, they are connected to their previous session. Otherwise, they are connected to a new session with a new streaming instance. Setting this value to 0 disables it. When this value is disabled, users are not disconnected due to inactivity.

   **Note**

   Users are considered idle when they stop providing keyboard or mouse input during their streaming session. File uploads and downloads, audio in, audio out, and pixels changing do not qualify as user activity. If users continue to be idle after the time interval in **Idle disconnect timeout in minutes** elapses, they are disconnected.

   - **Minimum capacity** — Choose a minimum number of instances for your fleet based on the minimum number of expected concurrent users. Every unique user session is served by an instance. For example, to have your stack support 100 concurrent users during low demand, specify a minimum capacity of 100. This ensures that 100 instances are running even if there are fewer than 100 users.
   - **Maximum capacity** — Choose a maximum number of instances for your fleet based on the maximum number of expected concurrent users. Every unique user session is served by an instance. For example, to have your stack support 500 concurrent users during high demand, specify a maximum capacity of 500. This ensures that up to 500 instances can be created on demand.

8. For **Step 4: Configure Network**, choose an Amazon VPC and two subnets with access to the network resources that your application needs, and then choose **Next**. If you don’t have a VPC or subnets,
you can create them using the links provided and then click the refresh icons. For Security groups, you can select up to five security groups. Otherwise, the default security group is used. For more information, see Networking and Access for Amazon AppStream 2.0 (p. 10).

9. For Step 5: Enable Storage, choose one or more of the following, then choose Next.

- **Enable Home Folders** — By default, this setting is enabled. Keep the default setting. For information about requirements for enabling home folders, see Enable Home Folders for Your AppStream 2.0 Users (p. 137).

- **Enable Google Drive** — Optionally, you can enable users to link their Google Drive for G Suite account to AppStream 2.0. You can enable Google Drive for accounts in G Suite domains only, not for personal Gmail accounts. For information about requirements for enabling Google Drive, see Enable Google Drive for Your AppStream 2.0 Users (p. 141).

- **Enable OneDrive** — Optionally, you can enable users to link their OneDrive for Business account to AppStream 2.0. You can enable OneDrive for accounts in OneDrive domains only, not for personal accounts. For information about requirements for enabling OneDrive, see Enable OneDrive for Your AppStream 2.0 Users (p. 143).

10. For Step 6: User Settings, select the ways in which your users can transfer data between their streaming session and their local device. Then, choose whether to enable application settings persistence. When you're done, choose Review:

**Clipboard, file transfer, and print to local device permissions options:**

- **Clipboard** — By default, users can copy and paste data between their local device and streaming applications. You can limit Clipboard options so that users can paste data to their remote streaming session only or copy data to their local device only. You can also disable Clipboard options entirely. Users can still copy and paste between applications in their streaming session.

- **File transfer** — By default, users can upload and download files between their local device and streaming session. You can limit file transfer options so that users can upload files to their streaming session only or download files to their local device only. You can also disable file transfer entirely.

**Important**

If your users require AppStream 2.0 file system redirection to access local drives and folders during their streaming sessions, you must enable both file upload and download. To use file system redirection, your users must have AppStream 2.0 client version 1.0.480 or later installed. For more information, see Enable File System Redirection for Your AppStream 2.0 Users (p. 187).

- **Print to local device** — By default, users can print to their local device from within a streaming application. When they choose Print in the application, they can download a .pdf file that they can print to a local printer. You can disable this option to prevent users from printing to a local device.

**Note**

These settings affect only whether users can use AppStream 2.0 data transfer features. If your image provides access to a browser, network printer, or other remote resource, your users might be able to transfer data to or from their streaming session in other ways.

**Application settings persistence options:**

- **Enable Application Settings Persistence** — Users' application customizations and Windows settings are automatically saved after each streaming session and applied during the next session. These settings are saved to an Amazon Simple Storage Service (Amazon S3) bucket in your account, within the AWS Region in which application settings persistence is enabled.

- **Settings Group** — The settings group determines which saved application settings are used for a streaming session from this stack. If the same settings group is applied to another stack, both stacks use the same application settings. By default, the settings group value is the name of the stack.
Step 2: Provide Access to Users

After you create a stack with an associated fleet, you can provide access to users through the AppStream 2.0 user pool, SAML 2.0 [single sign-on (SSO)], or the AppStream 2.0 API. For more information, see User Pool Administration (p. 240) and Single Sign-on Access (SAML 2.0) (p. 243).

Note
Users in the AppStream 2.0 user pool can't be assigned to stacks with fleets that are joined to an Active Directory domain.

For this getting started exercise, you can use the AppStream 2.0 user pool. This access method enables you to create and manage users by using a permanent login portal URL. To quickly test application streaming without setting up users, complete the following steps to create a temporary URL, also known as a streaming URL.

To provide access to users with a temporary URL

1. In the navigation pane, choose Fleets.
2. In the list of fleets, choose the fleet that is associated with the stack for which you want to create a streaming URL. Verify that the status of the fleet is Running.
3. In the navigation pane, choose Stacks. Select the stack, and then choose Actions, Create Streaming URL.
4. For User id, type the user ID. Choose an expiration time, which determines how long the generated URL is valid.
5. To view the user ID and URL, choose Get URL.
6. To copy the link to the clipboard, choose Copy Link.

After you provide your users with access to AppStream 2.0, they can start AppStream 2.0 streaming sessions. If you provide access through the AppStream 2.0 user pool, they must use a web browser for streaming sessions.

If you plan to use SAML 2.0 [single sign-on (SSO)] or the AppStream 2.0 API to provide access to your users, you can make the AppStream 2.0 client available to them. The AppStream 2.0 client is a native application that is designed for users who require additional functionality during their AppStream 2.0 streaming sessions. For more information, see Provide Access Through the AppStream 2.0 Client for Windows (p. 174).
Resources

For more information, see the following:

- Learn how to use the AppStream 2.0 image builder to add your own applications and create images that you can stream to your users. For more information, see Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47).
- Provide persistent storage for your session users by using AppStream 2.0 home folders and Google Drive. For more information, see Enable and Administer Persistent Storage for Your AppStream 2.0 Users (p. 136).
- Integrate your AppStream 2.0 streaming resources with your Microsoft Active Directory environment. For more information, see Using Active Directory with AppStream 2.0 (p. 106).
- Control who has access to your AppStream 2.0 streaming instances. For more information, see Identity and Access Management for Amazon AppStream 2.0 (p. 224), AppStream 2.0 User Pools (p. 239) and Single Sign-on Access (SAML 2.0) (p. 243).
- Monitor your AppStream 2.0 resources by using Amazon CloudWatch. For more information, see AppStream 2.0 Metrics and Dimensions (p. 205).
- Troubleshoot your AppStream 2.0 streaming experience. For more information, see Troubleshooting (p. 256).
Networking and Access for Amazon AppStream 2.0

The following topics provide information about enabling users to connect to AppStream 2.0 streaming instances (fleet instances) and enabling your AppStream 2.0 fleets and image builders to access network resources and the internet.

Contents
- Internet Access (p. 10)
- Configure a VPC for AppStream 2.0 (p. 11)
- Using Amazon S3 VPC Endpoints for Home Folders and Application Settings Persistence (p. 21)
- Amazon AppStream 2.0 Connections to Your VPC (p. 21)
- User Connections to Amazon AppStream 2.0 (p. 23)

Internet Access

If your fleets and image builders require internet access, enable internet access by doing one of the following:

- **Configure a VPC with Private Subnets and a NAT Gateway (p. 12) (recommended)** — With this configuration, you launch your fleets and image builders in a private subnet and configure a NAT gateway in a public subnet in your VPC. Your streaming instances are assigned a private IP address that is not directly accessible from the internet.

  In addition, unlike configurations that use the Default Internet Access option for enabling internet access, the NAT configuration is not limited to 100 fleet instances. If your deployment must support more than 100 concurrent users, use this configuration.

  You can create and configure a new VPC to use with a NAT gateway, or add a NAT gateway to an existing VPC.

- **Configure a New or Existing VPC with a Public Subnet (p. 18)** — With this configuration, you launch your fleets and image builders in a public subnet and enable Default Internet Access. When you enable this option, AppStream 2.0 uses the internet gateway in your Amazon VPC public subnet to provide the internet connection. Your streaming instances are assigned a public IP address that is directly accessible from the internet. You can create a new VPC or configure an existing one for this purpose.

  **Note**
  When Default Internet Access is enabled, a maximum of 100 fleet instances is supported. If your deployment must support more than 100 concurrent users, use the NAT gateway configuration (p. 12) instead.

- **Use the Default VPC, Public Subnet, and Security Group (p. 20)** — If you are new to AppStream 2.0 and want to get started using the service, you can launch your fleets and image builders in a default public subnet and enable Default Internet Access. When you enable this option, AppStream 2.0 uses the internet gateway in your Amazon VPC public subnet to provide the internet connection. Your streaming instances are assigned a public IP address that is directly accessible from the internet.
Default VPCs are available in AWS accounts created after 2013-12-04.

**Note**  
When Default Internet Access is enabled, a maximum of 100 fleet instances is supported. If your deployment must support more than 100 concurrent users, use the NAT gateway configuration (p. 12) instead.

**Configure a VPC for AppStream 2.0**

When you set up AppStream 2.0, you must specify the virtual private cloud (VPC) and at least one subnet in which to launch your fleet instances and image builders. A VPC is a virtual network in your own logically isolated area within the AWS cloud. A subnet is a range of IP addresses in your VPC.

When you configure your VPC for AppStream 2.0, you can specify either public or private subnets, or a mix of both types of subnets. A public subnet has direct access to the internet through an internet gateway. A private subnet, which doesn't have a route to an internet gateway, requires a Network Address Translation (NAT) gateway or NAT instance to provide access to the internet.

**Contents**
- VPC Setup Recommendations (p. 11)
- Configure a VPC with Private Subnets and a NAT Gateway (p. 12)
- Configure a New or Existing VPC with a Public Subnet (p. 18)
- Use the Default VPC, Public Subnet, and Security Group (p. 20)

**VPC Setup Recommendations**

When you create a fleet or launch an image builder, you specify the VPC and one or more subnets to use. You can provide additional access control to your VPC by specifying security groups.

The following recommendations can help you configure your VPC more effectively and securely. In addition, they can help you configure an environment that supports effective fleet scaling. With effective fleet scaling, you can meet current and anticipated AppStream 2.0 user demand, while avoiding unnecessary resource usage and associated costs.

**Overall VPC Configuration**

- Make sure that your VPC configuration can support your fleet scaling needs.

  As you develop your plan for fleet scaling, keep in mind that one user requires one fleet instance. Therefore, the size of your fleet determines the number of users who can stream concurrently. For this reason, for each instance type (p. 80) that you plan to use, make sure that the number of fleet instances that your VPC can support is greater than the number of anticipated concurrent users for the same instance type.

- Make sure that your AppStream 2.0 account quotas (also referred to as limits) are sufficient to support your anticipated demand. To request a quota increase, use the AppStream 2.0 Limits form. For more information about AppStream 2.0 limits, see Amazon AppStream 2.0 Service Quotas (p. 271).

- If you plan to provide your streaming instances (fleet instances or image builders) with access to the internet, we recommend that you configure a VPC with two private subnets for your streaming instances and a NAT gateway in a public subnet.

  The NAT gateway lets the streaming instances in your private subnets connect to the internet or other AWS services. However, it prevents the internet from initiating a connection with those instances. In addition, unlike configurations that use the Default Internet Access option for enabling internet
access, the NAT configuration supports more than 100 fleet instances. For more information, see Configure a VPC with Private Subnets and a NAT Gateway (p. 12).

Elastic Network Interfaces

- AppStream 2.0 creates as many elastic network interfaces (network interfaces) as the maximum desired capacity of your fleet. By default, the limit for network interfaces per Region is 5000.

When planning capacity for very large deployments, for example, thousands of streaming instances, consider the number of EC2 instances that are also used in the same Region.

Subnets

- If you are configuring more than one private subnet for your VPC, configure each in a different Availability Zone. Doing so increases fault tolerance and can help prevent insufficient capacity errors.
- Make sure that the network resources required for your applications are accessible through both of your private subnets.
- Configure each of your private subnets with a subnet mask that allows for enough client IP addresses to account for the maximum number of expected concurrent users. In addition, allow for additional IP addresses to account for anticipated growth. For more information, see VPC and Subnet Sizing for IPv4.
- If you are using a VPC with NAT, configure at least one public subnet with a NAT Gateway for internet access, preferably two. Configure the public subnets in the same Availability Zones where your private subnets reside.

To enhance fault tolerance and reduce the chance of insufficient capacity errors for large AppStream 2.0 fleet deployments, consider extending your VPC configuration into a third Availability Zone. Include a private subnet, public subnet, and NAT gateway in this additional Availability Zone.

**Note**

To configure more than two Availability Zones, you can use the UpdateFleet API action or the update-fleet AWS CLI command.

Security Groups

- Use security groups to provide additional access control to your VPC.

Security groups that belong to your VPC let you control the network traffic between AppStream 2.0 streaming instances and network resources required by applications. These resources may include other AWS services such as Amazon RDS or Amazon FSx, license servers, database servers, file servers, and application servers.
- Make sure that the security groups provide access to the network resources that your applications require.

For more information about configuring security groups for AppStream 2.0, see Security Groups in Amazon AppStream 2.0 (p. 253). For general information about security groups, see Security Groups for Your VPC in the Amazon VPC User Guide.

**Configure a VPC with Private Subnets and a NAT Gateway**

If you plan to provide your streaming instances (fleet instances and image builders) with access to the internet, we recommend that you configure a VPC with two private subnets for your streaming instances.
and a NAT gateway in a public subnet. You can create and configure a new VPC to use with a NAT gateway, or add a NAT gateway to an existing VPC. For additional VPC configuration recommendations, see VPC Setup Recommendations (p. 11).

The NAT gateway lets the streaming instances in your private subnets connect to the internet or other AWS services, but prevents the internet from initiating a connection with those instances. In addition, unlike configurations that use the Default Internet Access option for enabling internet access for AppStream 2.0 streaming instances, this configuration is not limited to 100 fleet instances.

For information about using NAT Gateways and this configuration, see NAT Gateways and VPC with Public and Private Subnets (NAT) in the Amazon VPC User Guide.

Contents

• Create and Configure a New VPC (p. 13)
• Add a NAT Gateway to an Existing VPC (p. 16)
• Enable Internet Access for Your Fleet and Image Builder (p. 17)

Create and Configure a New VPC

This topic describes how to use the VPC wizard to create a VPC with a public subnet and one private subnet. As part of this process, the wizard creates an internet gateway and a NAT gateway. It also creates a custom route table associated with the public subnet and updates the main route table associated with the private subnet. The NAT gateway is automatically created in the public subnet of your VPC.

After you use the wizard to create the initial VPC configuration, you'll add a second private subnet. For more information about this configuration, see VPC with Public and Private Subnets (NAT) in the Amazon VPC User Guide.

Note

If you already have a VPC, complete the steps in Add a NAT Gateway to an Existing VPC (p. 16) instead.

Contents

• Step 1: Allocate an Elastic IP Address (p. 13)
• Step 2: Create a New VPC (p. 14)
• Step 3: Add a Second Private Subnet (p. 15)
• Step 4: Verify and Name Your Subnet Route Tables (p. 15)

Step 1: Allocate an Elastic IP Address

Before you create your VPC, you must allocate an Elastic IP address in your AppStream 2.0 Region. You must first allocate an Elastic IP address for use in your VPC, and then associate it with your NAT gateway. With an Elastic IP address, you can mask the failure of streaming instance by rapidly remapping the address to another streaming instance in your VPC. For more information, see Elastic IP Addresses in the Amazon VPC User Guide.

Note

Charges may apply to Elastic IP addresses that you use. For more information, see Elastic IP Addresses on the Amazon EC2 pricing page.

Complete the following steps if you don't already have an Elastic IP address. If you want to use an existing Elastic IP address, verify that it's not currently associated with another instance or network interface.

To allocate an Elastic IP address

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, under **Network & Security**, choose **Elastic IPs**.
3. Choose **Allocate New Address**, and then choose **Allocate**.
4. Note the Elastic IP address.
5. In the upper right of the **Elastic IPs** pane, click the X icon to close the pane.

**Step 2: Create a New VPC**

Complete the following steps to create a new VPC with a public subnet and one private subnet.

**To create a new VPC**

1. Open the Amazon VPC console at [https://console.aws.amazon.com/vpc/](https://console.aws.amazon.com/vpc/).
2. In the navigation pane, choose **VPC Dashboard**.
3. Choose **Launch VPC Wizard**.
4. In **Step 1: Select a VPC Configuration**, choose **VPC with Public and Private Subnets**, and then choose **Select**.
5. In **Step 2: VPC with Public and Private Subnets**, configure the VPC as follows:
   - For **IPv4 CIDR block**, specify an IPv4 CIDR block for the VPC.
   - For **IPv6 CIDR block**, keep the default value, **No IPv6 CIDR Block**.
   - For **VPC name**, type a unique name for the VPC.
6. Configure the public subnet as follows:
   - For **Public subnet's IPv4 CIDR**, specify the CIDR block for the subnet.
   - For **Availability Zone**, keep the default value, **No Preference**.
   - For **Public subnet name**, type a name for the subnet; for example, **AppStream2 Public Subnet**.
7. Configure the first private subnet as follows:
   - For **Private subnet's IPv4 CIDR**, specify the CIDR block for the subnet. Make a note of the value that you specify.
   - For **Availability Zone**, select a specific zone and make a note of the zone that you select.
   - For **Private subnet name**, type a name for the subnet; for example, **AppStream2 Private Subnet1**.
   - For the remaining fields, where applicable, keep the default values.
8. For **Elastic IP Allocation ID**, click in the text box and select the value that corresponds to the Elastic IP address that you created. This address is assigned to the NAT gateway. If you don't have an Elastic IP address, create one by using the Amazon VPC console at [https://console.aws.amazon.com/vpc/](https://console.aws.amazon.com/vpc/).
9. For **Service endpoints**, if an Amazon S3 endpoint is required for your environment, specify one. An S3 endpoint is required to provide users with access to **home folders** (p. 136) or to enable **application settings persistence** (p. 146) for your users in a private network.
   - To specify an Amazon S3 endpoint, do the following:
     a. Choose **Add Endpoint**.
     b. For **Service**, select the entry in the list that ends with "s3" (the **com.amazonaws.region.s3** entry that corresponds to the Region in which the VPC is being created).
     c. For **Subnet**, choose **Private subnet**.
     d. For **Policy**, keep the default value, **Full Access**.
10. For **Enable DNS hostnames**, keep the default value, **Yes**.
11. For **Hardware tenancy**, keep the default value, **Default**.
12. Choose **Create VPC**.
13. Note that it takes several minutes to set up your VPC. After the VPC is created, choose **OK**.

**Step 3: Add a Second Private Subnet**

In the previous step (Step 2: Create a New VPC (p. 14)), you created a VPC with one public subnet and one private subnet. Perform the following steps to add a second private subnet. We recommend that you add a second private subnet in a different Availability Zone than your first private subnet.

1. In the navigation pane, choose **Subnets**.
2. Select the first private subnet that you created in the previous step. On the **Description** tab, below the list of subnets, make a note of the Availability Zone for this subnet.
3. On the upper left of the subnets pane, choose **Create Subnet**.
4. For **Name tag**, type a name for the private subnet; for example, **AppStream2 Private Subnet2**.
5. For **VPC**, select the VPC that you created in the previous step.
6. For **Availability Zone**, select an Availability Zone other than the one you are using for your first private subnet. Selecting a different Availability Zone increases fault tolerance and helps prevent insufficient capacity errors.
7. For **IPv4 CIDR block**, specify a unique CIDR block range for the new subnet. For example, if your first private subnet has an IPv4 CIDR block range of **10.0.1.0/24**, you could specify a CIDR block range of **10.0.2.0/24** for the new private subnet.
8. Choose **Create**.
9. After your subnet is created, choose **Close**.

**Step 4: Verify and Name Your Subnet Route Tables**

After you've created and configured your VPC, complete the following steps to specify a name for your route tables, and to verify that:

- The route table associated with the subnet in which your NAT gateway resides includes a route that points internet traffic to an internet gateway. This ensures that your NAT gateway can access the internet.
- The route tables associated with your private subnets are configured to point internet traffic to the NAT gateway. This enables the streaming instances in your private subnets to communicate with the internet.

1. In the navigation pane, choose **Subnets**, and select the public subnet that you created; for example, **AppStream 2.0 Public Subnet**.
   a. On the **Route Table** tab, choose the ID of the route table; for example, **rtb-12345678**.
   b. Select the route table. Under **Name**, choose the edit icon (the pencil), and type a name (for example, **appstream2-public-routetable**), and then select the check mark to save the name.
   c. With the public route table still selected, on the **Routes** tab, verify that there is one route for local traffic and another route that sends all other traffic to the internet gateway for the VPC. The following table describes these two routes:

<table>
<thead>
<tr>
<th>Destination</th>
<th>Target</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public subnet IPv4 CIDR Block</td>
<td>Local</td>
<td>All traffic from the resources destined for IPv4 addresses within the public</td>
</tr>
<tr>
<td>(for example, 10.0.0/20)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Configure a VPC with Private Subnets and a NAT Gateway

<table>
<thead>
<tr>
<th>Destination</th>
<th>Target</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traffic destined to all other IPv4 addresses (for example, 0.0.0.0/0)</td>
<td>Outbound (igw-\textit{ID})</td>
<td>Traffic destined for all other IPv4 addresses is routed to the internet gateway (identified by \textit{igw-\textit{ID}}) that was created by the VPC Wizard.</td>
</tr>
</tbody>
</table>

2. In the navigation pane, choose **Subnets**, and select the first private subnet that you created (for example, AppStream2 Private Subnet1).
   a. On the **Route Table** tab, choose the ID of the route table.
   b. Select the route table. Under **Name**, choose the edit icon (the pencil), and enter a name (for example, \textit{appstream2-private-routetable}), and then choose the check mark to save the name.
   c. On the **Routes** tab, verify that the route table includes the following routes:

<table>
<thead>
<tr>
<th>Destination</th>
<th>Target</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public subnet IPv4 CIDR Block (for example, 10.0.0/20)</td>
<td>Local</td>
<td>All traffic from the resources destined for IPv4 addresses within the public subnet IPv4 CIDR block is routed locally within the VPC.</td>
</tr>
<tr>
<td>Traffic destined to all other IPv4 addresses (for example, 0.0.0.0/0)</td>
<td>Outbound (nat-\textit{ID})</td>
<td>Traffic destined for all other IPv4 addresses is routed to the NAT gateway (identified by \textit{nat-\textit{ID}}).</td>
</tr>
<tr>
<td>Traffic destined for S3 buckets (applicable if you specified an S3 endpoint)</td>
<td>Storage (vpce-\textit{ID})</td>
<td>Traffic destined for S3 buckets is routed to the S3 endpoint (identified by \textit{vpce-\textit{ID}}).</td>
</tr>
</tbody>
</table>

3. In the navigation pane, choose **Subnets**, and select the second private subnet that you created (for example, AppStream2 Private Subnet2).
4. On the **Route Table** tab, verify that the route table is the private route table (for example, \textit{appstream2-private-routetable}). If the route table is different, choose **Edit** and select this route table.

### Next Steps

To enable your fleet instances and image builders to access the internet, complete the steps in **Enable Internet Access for Your Fleet and Image Builder** (p. 17).

### Add a NAT Gateway to an Existing VPC

If you have already configured a VPC, complete the following steps to add a NAT gateway to your VPC. If you need to create a new VPC, see **Create and Configure a New VPC** (p. 13).

#### To add a NAT gateway to an existing VPC

1. To create your NAT gateway, complete the steps in **Creating a NAT Gateway** in the **Amazon VPC User Guide**.
2. Verify that your VPC has at least one private subnet. We recommend that you specify two private subnets from different Availability Zones for high availability and fault tolerance. For information about how to create a second private subnet, see Step 3: Add a Second Private Subnet (p. 15).

3. Update the route table associated with one or more of your private subnets to point internet-bound traffic to the NAT gateway. This enables the streaming instances in your private subnets to communicate with the internet. To do so, complete the steps in Updating Your Route Table in the Amazon VPC User Guide.

Next Steps

To enable your fleet instances and image builders to access the internet, complete the steps in Enable Internet Access for Your Fleet and Image Builder (p. 17).

Enable Internet Access for Your Fleet and Image Builder

After your NAT gateway is available on a VPC, you can enable internet access for your fleet and image builder.

Enable Internet Access for Your Fleet

You can enable internet access either when you create the fleet or later.

To enable internet access at fleet creation

1. Complete the steps in Create a Fleet (p. 82) up to Step 4: Configure Network.
2. Choose a VPC with a NAT gateway.
3. If the subnet fields are empty, select a private subnet for Subnet 1 and, optionally, another private subnet for Subnet 2. If you don't already have a private subnet in your VPC, you may need to create a second private subnet.
4. Continue with the steps in Create a Fleet (p. 82).

To enable internet access after fleet creation by using a NAT gateway

1. In the navigation pane, choose Fleets.
2. Select a fleet and verify that the state is Stopped.
3. Choose Fleet Details, Edit, and choose a VPC with a NAT gateway.
4. Choose a private subnet for Subnet 1 and, optionally, another private subnet for Subnet 2. If you don't already have a private subnet in your VPC, you may need to create a second private subnet (p. 15).
5. Choose Update.

You can test your internet connectivity by starting your fleet, and then connecting to your streaming instance and browsing to the internet.

Enable Internet Access for Your Image Builder

If you plan to enable internet access for your image builder, you must do so when you create the image builder.

To enable internet access for an image builder

1. Complete the steps in Launch an Image Builder to Install and Configure Streaming Applications (p. 26), up to Step 3: Configure Network.
2. Choose the VPC with a NAT gateway.
3. If Subnet is empty, select a subnet.
4. Continue with the steps in Launch an Image Builder to Install and Configure Streaming Applications (p. 26).

Configure a New or Existing VPC with a Public Subnet

If you created your AWS account after 2013-12-04, you have a default VPC (p. 20) in each AWS Region that includes default public subnets. However, you may want to create your own nondefault VPC or configure an existing VPC to use with AppStream 2.0. This topic describes how to configure a nondefault VPC and public subnet to use with AppStream 2.0.

After you configure your VPC and public subnet, you can provide your streaming instances (fleet instances and image builders) with access to the internet by enabling the Default Internet Access option. When you enable this option, AppStream 2.0 enables internet connectivity by associating an Elastic IP address to the network interface that is attached from the streaming instance to your public subnet. An Elastic IP address is a public IPv4 address that is reachable from the internet. For this reason, we recommend that you instead use a NAT gateway to provide internet access to your AppStream 2.0 instances. In addition, when Default Internet Access is enabled, a maximum of 100 fleet instances is supported. If your deployment must support more than 100 concurrent users, use the NAT gateway configuration (p. 12) instead.

For more information, see the steps in Configure a VPC with Private Subnets and a NAT Gateway (p. 12). For additional VPC configuration recommendations, see VPC Setup Recommendations (p. 11).

Contents
- Step 1: Configure a VPC with a Public Subnet (p. 18)
- Step 2: Enable Default Internet Access for Your Fleet and Image Builder (p. 19)

Step 1: Configure a VPC with a Public Subnet

You can configure your own non-default VPC with a public subnet by using either of the following methods:

- Create a New VPC with a Single Public Subnet (p. 18)
- Configure an Existing VPC (p. 18)

Create a New VPC with a Single Public Subnet

When you use the VPC wizard to create a new VPC, the wizard creates an internet gateway and a custom route table that is associated with the public subnet. The route table routes all traffic destined for an address outside the VPC to the internet gateway. For more information about this configuration, see VPC with a Single Public Subnet in the Amazon VPC User Guide.

1. Complete the steps in Step 1: Create the VPC in the Amazon VPC User Guide to create your VPC.
2. To enable your fleet instances and image builders to access the internet, complete the steps in Step 2: Enable Default Internet Access for Your Fleet and Image Builder (p. 19).

Configure an Existing VPC

If you want to use an existing VPC that does not have a public subnet, you can add a new public subnet. In addition to a public subnet, you must also have an internet gateway attached to your VPC and a route
table that routes all traffic destined for an address outside the VPC to the internet gateway. To configure these components, complete the following steps.

1. To add a public subnet, complete the steps in Creating a Subnet in Your VPC. Use the existing VPC that you plan to use with AppStream 2.0.

   If your VPC is configured to support IPv6 addressing, the IPv6 CIDR block list displays. Select Don’t assign IPv6.

2. To create and attach an internet gateway to your VPC, complete the steps in Creating and Attaching an Internet Gateway.

3. To configure your subnet to route internet traffic through the internet gateway, complete the steps in Creating a Custom Route Table. In step 5, for Destination, use IPv4 format (0.0.0.0/0).

4. To enable your fleet instances and image builders to access the internet, complete the steps in Step 2: Enable Default Internet Access for Your Fleet and Image Builder (p. 19).

**Step 2: Enable Default Internet Access for Your Fleet and Image Builder**

After you configure a VPC that has a public subnet, you can enable the Default Internet Access option for your fleet and image builder.

**Enable Default Internet Access for a Fleet**

You can enable the Default Internet Access option when you create the fleet, or later.

**Note**

For fleet instances that have the Default Internet Access option enabled, the limit is 100.

**To enable internet access at fleet creation**

1. Complete the steps in Create a Fleet (p. 82) up to Step 4: Configure Network.
2. Select the Default Internet Access check box.
3. If the subnet fields are empty, select a subnet for Subnet 1 and, optionally, Subnet 2.
4. Continue with the steps in Create a Fleet (p. 82).

**To enable internet access after fleet creation**

1. In the navigation pane, choose Fleets.
2. Select a fleet and verify that its state is Stopped.
3. Choose Fleet Details, Edit, then select the Default Internet Access check box.

You can test internet connectivity by starting your fleet, creating a stack, associating the fleet with a stack, and browsing the internet within a streaming session for stack. For more information, see Create an AppStream 2.0 Fleet and Stack (p. 81).

**Enable Default Internet Access for an Image Builder**

After you configure a VPC that has a public subnet, you can enable the Default Internet Access option for your image builder. You can do so when you create the image builder.
To enable internet access for an image builder

1. Complete the steps in Launch an Image Builder to Install and Configure Streaming Applications (p. 26) up to Step 3: Configure Network.
2. Select the Default Internet Access check box.
3. If Subnet 1 is empty, select a subnet.
4. Continue with the steps in Launch an Image Builder to Install and Configure Streaming Applications (p. 26).

Use the Default VPC, Public Subnet, and Security Group

Your AWS account, if it was created after 2013-12-04, has a default VPC in each AWS Region. The default VPC includes a default public subnet in each Availability Zone and an internet gateway that is attached to your VPC. The VPC also includes a default security group. If you are new to AppStream 2.0 and want to get started using the service, you can keep the default VPC and security group selected when you create a fleet or launch an image builder. Then, you can select at least one default subnet.

**Note**
If your AWS account was created before 2013-12-04, you must create a new VPC or configure an existing one to use with AppStream 2.0. We recommend that you manually configure a VPC with two private subnets for your fleets and image builders and a NAT gateway in a public subnet. For more information, see Configure a VPC with Private Subnets and a NAT Gateway (p. 12). Alternatively, you can configure a non-default VPC with a public subnet. For more information, see Configure a New or Existing VPC with a Public Subnet (p. 18).

To use the default VPC, subnet, and security group for a fleet

1. Complete the steps in Create a Fleet (p. 82) up to Step 4: Configure Network.
2. In Step 4: Configure Network, do the following:
   - To enable your fleet instances to access the internet, select the Default Internet Access check box.
   
   **Note**
   For fleet instances that have the Default Internet Access option enabled, the limit is 100.
   - For VPC, choose the default VPC for your AWS Region.
   - The default VPC name uses the following format: vpc-vpc-id (No_default_value_Name).
   - For Subnet 1, choose a default public subnet and make a note of the Availability Zone.
   - The default subnet names use the following format: subnet-subnet-id | (IPv4 CIDR block) | Default in availability-zone.
   - Optionally, for Subnet 2, choose a default subnet in a different Availability Zone.
   - For Security groups, select the default security group.
   - The default security group name uses the following format: sg-security-group-id-default
3. Continue with the steps in Create a Fleet (p. 82).

Complete the following steps to use the default VPC, subnet, and security group for an image builder.

To use the default VPC, subnet, and security group for an image builder

1. Follow the steps in Launch an Image Builder to Install and Configure Streaming Applications (p. 26) up to Step 3: Configure Network.
2. In **Step 4: Configure Network**, do the following:

- To enable your image builder to access the internet, select the **Default Internet Access** check box.
- For **VPC**, choose the default VPC for your AWS Region.
  
  The default VPC name uses the following format: `vpc-vpc-id` *(No_default_value_Name)*.
- For **Subnet 1**, choose a default public subnet.
  
  The default subnet names use the following format: `subnet-subnet-id | (IPv4 CIDR block) | Default in availability-zone`.
- For **Security groups**, select the default security group.
  
  The default security group name uses the following format: `sg-security-group-id-default`.

3. Continue with the steps in **Launch an Image Builder to Install and Configure Streaming Applications** *(p. 26)*.

## Using Amazon S3 VPC Endpoints for Home Folders and Application Settings Persistence

To support home folders and application settings persistence on a private network, AppStream 2.0 needs access permissions to the Amazon S3 VPC endpoint. To enable AppStream 2.0 access to your private S3 endpoint, attach the following custom policy to your VPC endpoint for Amazon S3. For more information about private Amazon S3 endpoints, see VPC Endpoints and Endpoints for Amazon S3 in the Amazon VPC User Guide.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Allow-AppStream-to-access-home-folder-and-application-settings",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:sts::account-id-without-hyphens:assumed-role/AmazonAppStreamServiceAccess/AppStream2.0"
            },
            "Action": [
                "s3:ListBucket",
                "s3:GetObject",
                "s3:PutObject",
                "s3:DeleteObject",
                "s3:GetObjectVersion",
                "s3:DeleteObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::appstream2-36fb080bb8-*",
                "arn:aws:s3:::appstream-app-settings-*"
            ]
        }
    ]
}
```

## Amazon AppStream 2.0 Connections to Your VPC

To enable AppStream 2.0 connectivity to network resources and the internet, configure your streaming instances as follows.
Network Interfaces

Each AppStream 2.0 streaming instance has the following network interfaces:

- The customer network interface provides connectivity to the resources within your VPC, as well as the internet, and is used to join the streaming instance to your directory.
- The management network interface is connected to a secure AppStream 2.0 management network. It is used for interactive streaming of the streaming instance to a user’s device, and to allow AppStream 2.0 to manage the streaming instance.

AppStream 2.0 selects the IP address for the management network interface from the following private IP address range: 198.19.0.0/16. Do not use this range for your VPC CIDR or peer your VPC with another VPC with this range, as this might create a conflict and cause streaming instances to be unreachable. Also, do not modify or delete any of the network interfaces attached to a streaming instance, as this might also cause the streaming instance to become unreachable.

Management Network Interface IP Address Range and Ports

The management network interface IP address range is 198.19.0.0/16. The following ports must be open on the management network interface of all streaming instances:

- Inbound TCP on port 8300. This is used for establishment of the streaming connection.
- Inbound TCP on ports 8000 and 8443. These are used for management of the streaming instance by AppStream 2.0.

Limit the inbound range on the management network interface to 198.19.0.0/16.

Under normal circumstances, AppStream 2.0 correctly configures these ports for your streaming instances. If any security or firewall software is installed on a streaming instance that blocks any of these ports, the streaming instance may not function correctly or may be unreachable.

Do not disable IPv6. If you disable IPv6, AppStream 2.0 will not function correctly. For information about configuring IPv6 for Windows, see Guidance for configuring IPv6 in Windows for advanced users.

Note
AppStream 2.0 relies on the DNS servers within your VPC to return a non-existent domain (NXDOMAIN) response for local domain names that don’t exist. This enables the AppStream 2.0-managed network interface to communicate with the management servers.

When you create a directory with Simple AD, AWS Directory Service creates two domain controllers that also function as DNS servers on your behalf. Because the domain controllers don't provide the NXDOMAIN response, they can't be used with AppStream 2.0.

Customer Network Interface Ports

- For internet connectivity, the following ports must be open to all destinations. If you are using a modified or custom security group, you need to add the required rules manually. For more information, see Security Group Rules in the Amazon VPC User Guide.
  - TCP 80 (HTTP)
  - TCP 443 (HTTPS)
- If you join your streaming instances to a directory, the following ports must be open between your AppStream 2.0 VPC and your directory controllers.
  - TCP/UDP 53 - DNS
User Connections to Amazon AppStream 2.0

Users can connect to AppStream 2.0 streaming instances through the default public internet endpoint, or by using an interface VPC endpoint (interface endpoint) that you create in your virtual private cloud (VPC). For more information, see Creating and Streaming from Interface VPC Endpoints (p. 250).

By default, AppStream 2.0 is configured to route streaming connections over the public internet. Internet connectivity is required to authenticate users and deliver the web assets that AppStream 2.0 requires to function. To allow this traffic, you must allow the domains listed in Allowed Domains (p. 24).

Note
For user authentication, AppStream 2.0 supports user pools, Security Assertion Markup Language 2.0 (SAML 2.0), and the CreateStreamingURL API action. For more information, see Authentication of Corporate Users (p. 239).

The following topics provide information about how to enable user connections to AppStream 2.0.

Contents
- Bandwidth Recommendations (p. 23)
- IP Address and Port Requirements for AppStream 2.0 User Devices (p. 24)
- Allowed Domains (p. 24)

Bandwidth Recommendations

To optimize the performance of AppStream 2.0, make sure that your network bandwidth and latency can sustain your users’ needs.

AppStream 2.0 uses NICE Desktop Cloud Visualization (DCV) to enable your users to securely access and stream your applications over varying network conditions. To help reduce bandwidth consumption, NICE DCV uses H.264-based video compression and encoding. During streaming sessions, the visual output of applications is compressed and streamed to your users as an AES-256 encrypted pixel stream over HTTPS. After the stream is received, it is decrypted and output to your users’ local screen. When your users interact with their streaming applications, the NICE DCV protocol captures their input and sends it back to their streaming applications over HTTPS.

Network conditions are constantly measured during this process and information is sent back to AppStream 2.0. AppStream 2.0 dynamically responds to changing network conditions by changing the...
video and audio encoding in real time to produce a high-quality stream for a wide variety of applications and network conditions.

The recommended bandwidth and latency for AppStream 2.0 streaming sessions depends on the workload. For example, a user who works with graphic-intensive applications to perform computer-aided design tasks will require more bandwidth and lower latency than a user who works with business productivity applications to write documents.

The following table provides guidance on the recommended network bandwidth and latency for AppStream 2.0 streaming sessions based on common workloads.

For each workload, the bandwidth recommendation is based on what an individual user might require at a specific point in time. The recommendation does not reflect the bandwidth required for sustained throughput. When only a few pixels change on the screen during a streaming session, the sustained throughput is much lower. Although users who have less bandwidth available can still stream their applications, the frame rate or image quality may not be optimal.

<table>
<thead>
<tr>
<th>Workload</th>
<th>Description</th>
<th>Bandwidth recommended per user</th>
<th>Recommended maximum roundtrip latency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Line of business applications</td>
<td>Document writing applications, database analysis utilities</td>
<td>2 Mbps</td>
<td>&lt; 150 ms</td>
</tr>
<tr>
<td>Graphics applications</td>
<td>Computer-aided design and modeling applications, photo and video editing</td>
<td>5 Mbps</td>
<td>&lt; 100 ms</td>
</tr>
<tr>
<td>High fidelity</td>
<td>High-fidelity datasets or maps across multiple monitors</td>
<td>10 Mbps</td>
<td>&lt; 50 ms</td>
</tr>
</tbody>
</table>

**IP Address and Port Requirements for AppStream 2.0 User Devices**

AppStream 2.0 users' devices require outbound access on port 443 (TCP), and if you are using DNS servers for domain name resolution, port 53 (UDP).

- Port 443 is used for HTTPS communication between AppStream 2.0 users' devices and streaming instances. Typically, when end users browse the web during streaming sessions, the web browser randomly selects a source port in the high range for streaming traffic. You must ensure that return traffic to this port is allowed.
- Port 53 is used for communication between AppStream 2.0 users' devices and your DNS servers. The port must be open to the IP addresses for your DNS servers so that public domain names can be resolved. This port is optional if you are not using DNS servers for domain name resolution.

**Allowed Domains**

For AppStream 2.0 users to access streaming instances, you must allow the following domain on the network from which users initiate access to the streaming instances.
• Session Gateway: *.amazonappstream.com

One or more of the following domains must be allowed to enable user authentication. You must allow the domains that correspond to the Regions where AppStream 2.0 is deployed.

<table>
<thead>
<tr>
<th>Region</th>
<th>Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>US East (N. Virginia)</td>
<td>appstream2.us-east-1.aws.amazon.com</td>
</tr>
<tr>
<td>US West (Oregon)</td>
<td>appstream2.us-west-2.aws.amazon.com</td>
</tr>
<tr>
<td>Asia Pacific (Seoul)</td>
<td>appstream2.ap-northeast-2.aws.amazon.com</td>
</tr>
<tr>
<td>Asia Pacific (Singapore)</td>
<td>appstream2.ap-southeast-1.aws.amazon.com</td>
</tr>
<tr>
<td>Asia Pacific (Sydney)</td>
<td>appstream2.ap-southeast-2.aws.amazon.com</td>
</tr>
<tr>
<td>Asia Pacific (Tokyo)</td>
<td>appstream2.ap-northeast-1.aws.amazon.com</td>
</tr>
<tr>
<td>Europe (Frankfurt)</td>
<td>appstream2.eu-central-1.aws.amazon.com</td>
</tr>
<tr>
<td>Europe (Ireland)</td>
<td>appstream2.eu-west-1.aws.amazon.com</td>
</tr>
</tbody>
</table>

Amazon Web Services (AWS) publishes its current IP address ranges, including the ranges that the Session Gateway and CloudFront domains may resolve to, in JSON format. For information about how to download the .json file and view the current ranges, see AWS IP Address Ranges in the Amazon Web Services General Reference. Or, if you are using AWS Tools for Windows PowerShell, you can access the same information by using the Get-AWSPublicIpAddressRange cmdlet. For more information, see Querying the Public IP Address Ranges for AWS.
Image Builders

Amazon AppStream 2.0 uses EC2 instances to stream applications. You launch instances from base images, called image builders, which AppStream 2.0 provides. To create your own custom image, you connect to an image builder instance, install and configure your applications for streaming, and then create your image by creating a snapshot of the image builder instance.

When you launch an image builder, you choose:

- An instance type — AppStream 2.0 provides different instance types with various compute, memory, and graphics configurations. The instance type must align with the instance family you need. For more information, see AppStream 2.0 Instance Families (p. 80).
- An operating system — AppStream 2.0 provides the following Microsoft Windows operating systems:
  - Windows Server 2012 R2
  - Windows Server 2016 Base
  - Windows Server 2019 Base
- The subnet and security groups to use — Make sure that the subnet and security groups provide access to the network resources that your applications require. Typical network resources required by applications may include licensing servers, database servers, file servers, and application servers.

Contents

- Launch an Image Builder to Install and Configure Streaming Applications (p. 26)
- Image Builder Actions (p. 28)
- Instance Metadata for AppStream 2.0 Image Builders (p. 29)
- AppStream 2.0 Base Image Version History (p. 29)

Launch an Image Builder to Install and Configure Streaming Applications

To install and configure applications to stream to your users, you start by launching an image builder instance as described in the following procedure.

Important
After you launch an image builder and it is running, your account may incur nominal charges. For more information, see AppStream 2.0 Pricing.

To launch an image builder

2. You can launch the image builder in the following ways:
   - If a welcome screen appears displaying two options (Try it now and Get started), choose Get started, Custom set up.
     For information about these two options, see Amazon AppStream 2.0 FAQs.
• If a welcome screen does not appear, choose **Quick links** in the left navigation pane, then **Custom set up**.

• Alternatively, choose **Images** in the left navigation pane, then the **Image Builder tab**, **Launch Image Builder**.

3. For **Step 1: Choose Image**, choose a base image. If you are launching the image builder for the first time, you can use one of the latest base images released by AWS (selected by default). For a list of the latest versions of base images released by AWS, see **AppStream 2.0 Base Image Version History** (p. 29). If you have already created images, or you want to update applications in an existing image, you can select one of your existing images. Be sure to select an image that aligns with the instance family that you need. For more information, see **AppStream 2.0 Instance Families** (p. 80).

Choose **Next**.

4. For **Step 2: Configure Image Builder**, configure the image builder by doing the following:

   • **Name**: Type a unique name identifier for the image builder.

   • **Display name (optional)**: Type a name to display for the image builder (maximum of 100 characters).

   • **Tags (optional)**: Choose **Add Tag**, and type the key and value for the tag. To add more tags, repeat this step. For more information, see **Tagging Your Amazon AppStream 2.0 Resources** (p. 201).

   • **Instance Type**: Select the instance type for the image builder. Choose a type that matches the performance requirements of the applications that you plan to install. For more information, see **AppStream 2.0 Instance Families** (p. 80).

   • **Network Access Points (Optional)**: You can create a private link, which is an interface VPC endpoint (interface endpoint), in your virtual private cloud (VPC). To start creating the interface endpoint, select **Create PrivateLink**. Selecting this link opens the VPC console. To finish creating the endpoint, follow steps 3 through 6 in **To create an interface endpoint**, in **Creating and Streaming from Interface VPC Endpoints** (p. 250).

After you create the interface endpoint, you can use it to keep streaming traffic within your VPC.

   • **AppStream 2.0 Agent**: This section displays only if you are not using the latest base image from AWS or a custom image that uses the latest version of the agent.

The AppStream 2.0 agent software runs on your streaming instances, enabling your users to connect to and stream their applications. Starting December 7, 2017, your streaming instances can be automatically updated with the latest AppStream 2.0 agent software. This capability helps to ensure that your image builder includes the latest features, performance improvements, and security updates that are available from AWS.

You can enable automatic updates of the AppStream 2.0 agent by creating a new image from any base image published by AWS on or after December 7, 2017. If the image that you are launching your image builder from doesn't use the latest version of the AppStream 2.0 agent, we recommend that you select the option to launch your image builder with the latest agent.

   • **IAM role (Advanced)**: When you apply an IAM role from your account to an AppStream 2.0 image builder, you can make AWS API requests from the image builder instance without manually managing AWS credentials. To apply an IAM role to the image builder, do either of the following:

     • To use an existing IAM role in your AWS account, choose the role that you want to use from the **IAM role** list. The role must must be accessible from the image builder. For more information, see **Configuring an Existing IAM Role to Use With AppStream 2.0 Streaming Instances** (p. 232).

     • To create a new IAM role, choose **Create new IAM role** and follow the steps in **How to Create an IAM Role to Use With AppStream 2.0 Streaming Instances** (p. 233).

5. Choose **Next**.

6. For **Step 3: Configure Network**, do the following:
• To add internet access for the image builder in a VPC with a public subnet, choose **Default Internet Access**. If you are providing internet access by using a NAT gateway, leave **Default Internet Access** unselected. For more information, see [Internet Access](p. 10).

• For **VPC** and **Subnet 1**, choose a VPC and at least one subnet. For increased fault tolerance, we recommend that you choose two subnets in different Availability Zones. For more information, see [Configure a VPC with Private Subnets and a NAT Gateway](p. 12).

If you don't have your own VPC and subnet, you can use the **default VPC** (p. 20) or create your own. To create your own, choose the **Create a new VPC** and **Create new subnet** links to create them. Choosing these links opens the Amazon VPC console. After you create your VPC and subnets, return to the AppStream 2.0 console and choose the refresh icon to the left of the **Create a new VPC** and **Create new subnet** links to display them in the list. For more information, see [Configure a VPC for AppStream 2.0](p. 11).

• For **Security group(s)**, choose up to five security groups to associate with this image builder. If you don't have your own security group and you don't want to use the default security group, choose the **Create new security group** link to create one. After you create your subnets in the Amazon VPC console, return to the AppStream 2.0 console and choose the refresh icon to the left of the **Create new security group** link to display them in the list. For more information, see [Security Groups in Amazon AppStream 2.0](p. 253).

7. For **Active Directory Domain** *(Optional)*, expand this section to choose the Active Directory configuration and organizational unit in which to place your streaming instance computer objects. Ensure that the selected network access settings enable DNS resolvability and communication with your directory. For more information, see [Using Active Directory with AppStream 2.0](p. 106).

8. Choose **Review** and confirm the details for the image builder. To change the configuration for any section, choose **Edit** and make the needed changes.

9. After you finish reviewing the configuration details, choose **Launch**. If an error message notifies you that you don't have sufficient quotas (also referred to as limits) to create the image builder, submit a quota increase request through the AWS Support Center. For more information, see [Amazon AppStream 2.0 Service Quotas](p. 271).

10. During the image builder creation process, the status of the image builder displays as **Pending** while AppStream 2.0 prepares the necessary resources. Click the **Refresh** icon periodically to update the image builder status. After the status changes to **Running**, the image builder is ready to use and you can create a custom image.

**Next Steps**

Next, install and configure your applications for streaming, and then create an image by creating a snapshot of the image builder instance. For more information, see [Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console](p. 47).

**Image Builder Actions**

You can perform the following actions on an image builder, depending on the current state (status) of the image builder instance.

**Delete**

Permanently delete an image builder.

The instance must be in a **Stopped** state.

**Connect**

Connect to a running image builder. This action starts a desktop streaming session with the image builder to install and add applications to the image, and create an image.
The instance must be in a **Running** state.

**Start**

Start a stopped image builder. A running instance is billed to your account.

The instance must be in a **Stopped** state.

**Stop**

Stop a running image builder. A stopped instance is not billed to your account.

The instance must be in a **Running** state.

None of these actions can be performed on an instance in any of the following intermediate states:

- Pending
- Snapshottting
- Stopping
- Starting
- Deleting

## Instance Metadata for AppStream 2.0 Image Builders

AppStream 2.0 image builder instances have instance metadata available through Windows environment variables. You can use the following environment variables in your applications and scripts to modify your environment based on the image builder instance details.

<table>
<thead>
<tr>
<th>Environment Variable</th>
<th>Context</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AppStream_Image_Arn</td>
<td>Machine</td>
<td>The ARN of the image that was used to create the streaming instance.</td>
</tr>
<tr>
<td>AppStream_Instance_Type</td>
<td>Machine</td>
<td>The instance type of the streaming instance. For example, <code>stream.standard.medium</code>.</td>
</tr>
<tr>
<td>AppStream_Resource_Type</td>
<td>Machine</td>
<td>The type of AppStream 2.0 resource. The value is either <code>fleet</code> or <code>imagebuilder</code>.</td>
</tr>
<tr>
<td>AppStream_Resource_Name</td>
<td>Machine</td>
<td>The name of the image builder.</td>
</tr>
</tbody>
</table>

## AppStream 2.0 Base Image Version History

Amazon AppStream 2.0 provides base images to help you create images that include your own applications. Base images are Amazon Machine Images (AMIs) that contain software configurations specific to the Windows platform. For AppStream 2.0, each base image includes the AppStream 2.0 agent and the latest version of one of the following Microsoft Windows operating systems:

• Windows Server 2019 Base — Available on the following image types: Base, Graphics G4, and Graphics Pro

You are responsible for installing and maintaining the updates for the Windows operating system, your applications, and their dependencies. For more information, see "Keep Your AppStream 2.0 Image Up-to-Date" in Administer Your Amazon AppStream 2.0 Images (p. 52). For information about the latest AppStream 2.0 software, see AppStream 2.0 Agent Version History (p. 39).

The following table lists the latest released images.

<table>
<thead>
<tr>
<th>Image type</th>
<th>Image name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base</td>
<td>• AppStream-WinServer2012R2-03-18-2020</td>
</tr>
<tr>
<td></td>
<td>• AppStream-WinServer2016-03-18-2020</td>
</tr>
<tr>
<td></td>
<td>• AppStream-WinServer2019-03-18-2020</td>
</tr>
<tr>
<td></td>
<td>• AppStream-Graphics-Design-WinServer2016-03-18-2020</td>
</tr>
<tr>
<td></td>
<td>• AppStream-Graphics-G4dn-WinServer2016-03-05-2020</td>
</tr>
<tr>
<td>Graphics Pro</td>
<td>• AppStream-Graphics-Pro-WinServer2012R2-03-18-2020</td>
</tr>
<tr>
<td></td>
<td>• AppStream-Graphics-Pro-WinServer2016-03-18-2020</td>
</tr>
<tr>
<td></td>
<td>• AppStream-Graphics-Pro-WinServer2019-03-18-2020</td>
</tr>
<tr>
<td>Sample apps</td>
<td>Amazon-AppStream2-Sample-Image-02-04-2019</td>
</tr>
</tbody>
</table>

For information about how to access this base image, see Get Started with Amazon AppStream 2.0: Set Up With Sample Applications (p. 5).

The latest Base, Graphics Design, and Graphics Pro images released on March 18, 2020, and the Graphics G4 images released on March 5, 2020 and March 16, 2020, include the following software components:

- **Amazon SSM Agent** — v2.3.842.0
- **Amazon CloudWatch Agent** — 1.3.14217
- **Amazon WDDM Hook Driver** — 1.0.0.56
- **EC2Config service (Windows Server 2012 R2 only)** — v4.9.3865.0
- **AWS Command Line Interface (AWS CLI)** — 1.17.5
- **AMD Driver for Graphics Design instances** — 24.20.13028.3002
- **NVIDIA Graphics Driver for Graphics Pro instances** — 441.66

**Important**

The following public images are deprecated and therefore no longer available from AWS:

- Images released on or before June 2019
- Images for the Graphics Desktop instance family
The following table describes all released images.

<table>
<thead>
<tr>
<th>Release</th>
<th>Image</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>03-18-2020</td>
<td>• Base</td>
<td>• Includes Microsoft Windows updates up to February 11, 2020</td>
</tr>
<tr>
<td></td>
<td>• Graphics Design</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Graphics Pro</td>
<td>• Includes AWS CLI version 1.17.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Includes NVIDIA Graphics Driver version 441.66 for Graphics Pro instances</td>
</tr>
<tr>
<td>03-16-2020</td>
<td>• Graphics G4</td>
<td>• Adds support for Graphics g4dn instances based on the EC2 G4 family (Windows Server 2012 R2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Includes Microsoft Windows updates up to February 11, 2020</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Includes AWS CLI version 1.17.5</td>
</tr>
<tr>
<td>03-05-2020</td>
<td>• Graphics G4</td>
<td>• Adds support for Graphics g4dn instances based on the EC2 G4 family (Windows Server 2016 and Windows Server 2019)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Includes Microsoft Windows updates up to February 11, 2020</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Includes AWS CLI version 1.17.5</td>
</tr>
<tr>
<td>01-13-2020</td>
<td>• Graphics Design</td>
<td>• Adds support for Windows Server 2019, with Microsoft Windows updates up to November 12, 2019</td>
</tr>
<tr>
<td>12-12-2019</td>
<td>• Base</td>
<td>• Includes Microsoft Windows updates up to November 12, 2019</td>
</tr>
<tr>
<td></td>
<td>• Graphics Design</td>
<td>• Includes AWS CLI version 1.16.284</td>
</tr>
<tr>
<td></td>
<td>• Graphics Pro</td>
<td>• Includes a new version of the SSM Agent (v2.3.760.0), which resolves an issue that prevented streaming instances from being provisioned.</td>
</tr>
<tr>
<td></td>
<td>• Graphics Pro</td>
<td>• Includes a fix to prevent Windows Defender from being enabled by default on Windows Server 2016 and Windows Server 2019 image builder instances. For more information, see Windows Update and Antivirus Software on AppStream 2.0 (p. 55).</td>
</tr>
<tr>
<td>09-05-2019</td>
<td>• Graphics Design</td>
<td>• Adds support for Windows Server 2016</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Includes Microsoft Windows updates up to August 13, 2019</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Includes AWS CLI version 1.16.222</td>
</tr>
</tbody>
</table>
## Base Image Versions

<table>
<thead>
<tr>
<th>Release</th>
<th>Image</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>06-24-2019</td>
<td>Base</td>
<td>• Includes AMD Driver version 24.20.13028.3002 for Graphics Design instances (compatible with Windows Server 2016)</td>
</tr>
<tr>
<td></td>
<td>Graphics Pro</td>
<td>• Adds support for Windows Server 2016 and Windows Server 2019</td>
</tr>
<tr>
<td>05-28-2019</td>
<td>Base</td>
<td>• Includes Microsoft Windows updates up to May 14, 2019</td>
</tr>
<tr>
<td></td>
<td>Graphics Design</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Graphics Pro</td>
<td></td>
</tr>
<tr>
<td>04-29-2019</td>
<td>Base</td>
<td>• Includes Microsoft Windows updates up to April 20, 2019</td>
</tr>
<tr>
<td></td>
<td>Graphics Design</td>
<td>• Includes AWS CLI version 1.16.126</td>
</tr>
<tr>
<td></td>
<td>Graphics Pro</td>
<td>• Includes NVIDIA Graphics Driver 412.16 for Graphics Pro instances</td>
</tr>
<tr>
<td>01-22-2019</td>
<td>Base</td>
<td>• Includes Microsoft Windows updates up to December 10, 2018</td>
</tr>
<tr>
<td></td>
<td>Graphics Design</td>
<td>• Includes AWS CLI version 1.16.84</td>
</tr>
<tr>
<td></td>
<td>Graphics Pro</td>
<td>• Includes NVIDIA Graphics Driver version 391.58 for Graphics Pro instances</td>
</tr>
<tr>
<td>06-12-2018</td>
<td>Base</td>
<td>• Includes Microsoft Windows updates up to May 9, 2018</td>
</tr>
<tr>
<td></td>
<td>Graphics Design</td>
<td>• Includes Windows PowerShell 5.1</td>
</tr>
<tr>
<td></td>
<td>Graphics Desktop</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Graphics Pro</td>
<td></td>
</tr>
<tr>
<td>05-02-2018</td>
<td>Base</td>
<td>• Includes Microsoft Windows updates up to April 10, 2018</td>
</tr>
<tr>
<td></td>
<td>Graphics Design</td>
<td>• Adds the following language packs: Japanese, Korean, Portuguese (Brazil), Thai, Chinese (Simplified), Chinese (Traditional)</td>
</tr>
<tr>
<td></td>
<td>Graphics Desktop</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Graphics Pro</td>
<td></td>
</tr>
<tr>
<td>03-19-2018</td>
<td>Base</td>
<td>• Includes Microsoft Windows updates up to February 23, 2018</td>
</tr>
<tr>
<td></td>
<td>Graphics Design</td>
<td>• Includes the following language packs: German, French, Italian, Spanish, Dutch</td>
</tr>
<tr>
<td></td>
<td>Graphics Desktop</td>
<td>• Resolves intermittent issues with using Microsoft Visio and Microsoft Project applications during streaming sessions</td>
</tr>
<tr>
<td></td>
<td>Graphics Pro</td>
<td></td>
</tr>
<tr>
<td>01-24-2018</td>
<td>Base</td>
<td>• Includes Microsoft Windows updates up to January 5, 2018</td>
</tr>
<tr>
<td></td>
<td>Graphics Design</td>
<td>• Includes Microsoft Windows updates for the Spectre and Meltdown vulnerabilities</td>
</tr>
<tr>
<td></td>
<td>Graphics Desktop</td>
<td>• Enables a default profile to be created on image builders and used for the AWS Command Line Interface (CLI) during streaming sessions</td>
</tr>
<tr>
<td></td>
<td>Graphics Pro</td>
<td></td>
</tr>
<tr>
<td>Release</td>
<td>Image</td>
<td>Changes</td>
</tr>
<tr>
<td>--------------</td>
<td>------------------------</td>
<td>----------------------------------------------------------------</td>
</tr>
<tr>
<td>01-01-2018</td>
<td>• Base</td>
<td>• Resolves an issue with connectivity to AppStream 2.0 instances</td>
</tr>
<tr>
<td></td>
<td>• Graphics Design</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Graphics Desktop</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Graphics Pro</td>
<td></td>
</tr>
<tr>
<td>12-07-2017</td>
<td>• Base</td>
<td>• Includes Microsoft Windows updates up to November 19, 2017</td>
</tr>
<tr>
<td></td>
<td>• Graphics Design</td>
<td>• Adds support for managed AppStream 2.0 agent updates</td>
</tr>
<tr>
<td></td>
<td>• Graphics Desktop</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Graphics Pro</td>
<td></td>
</tr>
<tr>
<td>11-13-2017</td>
<td>• Base</td>
<td>• Resolves an issue with Microsoft Office 365 applications not working during streaming sessions</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Includes Microsoft Windows updates up to October 11, 2017</td>
</tr>
<tr>
<td>09-05-2017</td>
<td>• Base</td>
<td>• New Graphics Design instance family</td>
</tr>
<tr>
<td></td>
<td>• Graphics Design</td>
<td>• Support for On-Demand fleets</td>
</tr>
<tr>
<td></td>
<td>• Graphics Desktop</td>
<td>• Updated approach for session context</td>
</tr>
<tr>
<td></td>
<td>• Graphics Pro</td>
<td>• Includes Microsoft Windows updates up to August 9, 2017</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Resolves an intermittent issue with applications not coming to the foreground</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Resolves an intermittent issue with applications not appearing in tile view</td>
</tr>
<tr>
<td>07-25-2017</td>
<td>• Graphics Desktop</td>
<td>• New Graphics Desktop and Graphics Pro instance families</td>
</tr>
<tr>
<td></td>
<td>• Graphics Pro</td>
<td>• Adds support for 2 K resolution</td>
</tr>
<tr>
<td>07-24-2017</td>
<td>• Base</td>
<td>• Includes Microsoft Windows updates up to July 13, 2017</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Adds support for Microsoft Active Directory domains</td>
</tr>
<tr>
<td>06-20-2017</td>
<td>• Base</td>
<td>• Optimizes application launch performance</td>
</tr>
<tr>
<td></td>
<td>• Sample apps</td>
<td>• Resolves an issue with applications not displaying in tile view</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Resolves an issue with applications displaying in tile view only</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Resolves an issue with applications displaying multiple times in tile view</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Resolves an issue with recently launched application windows not appearing in the foreground</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Resolves an issue with page margins when printing</td>
</tr>
</tbody>
</table>
### Base Image Versions

<table>
<thead>
<tr>
<th>Release</th>
<th>Image</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>05-18-2017</td>
<td>• Base</td>
<td>• Adds support for Amazon AppStream 2.0 home folders</td>
</tr>
<tr>
<td></td>
<td>• Sample apps</td>
<td>• Includes Microsoft Windows updates up to May 16, 2017</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Resolves an intermittent network issue that affects internet connections from streaming instances</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Resolves an issue with application tiles not functioning correctly</td>
</tr>
</tbody>
</table>
Images

You can create Amazon AppStream 2.0 images that contain applications you can stream to your users and default Windows and application settings to enable your users to get started with those applications quickly. However, after you create an image, you can’t change it. To add other applications, update existing applications, or change image settings, you must start and reconnect to the image builder that you used to create the image. If you deleted that image builder, launch a new image builder that is based on your image. Then make your changes and create a new image. For more information, see Launch an Image Builder to Install and Configure Streaming Applications (p. 26) and Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47).

Images that are available to you are listed in the Image Registry in the AppStream 2.0 console. They are categorized as public, private, or shared. You can use any of these image types to launch an image builder and set up an AppStream 2.0 fleet. Shared images are owned by other AWS accounts and shared with you. Permissions set on images that are shared with you may limit what you can do with those images. For more information, see Administer Your Amazon AppStream 2.0 Images (p. 52).

Default Application and Windows Settings and Application Launch Performance

You can create default application and Windows settings to enable your users to get started with their applications quickly, so that they won’t need to create or configure the settings themselves.

AppStream 2.0 optimizes the launch performance of your applications for your users' streaming sessions. To ensure that all of the required files are included in this process, you may need to manually add certain files and folders to the optimization manifest.

Contents

- Default Application and Windows Settings and Application Launch Performance (p. 35)
- Manage AppStream 2.0 Agent Versions (p. 36)
- AppStream 2.0 Agent Version History (p. 39)
- Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47)
- Administer Your Amazon AppStream 2.0 Images (p. 52)
- Create Your AppStream 2.0 Image Programmatically by Using the Image Assistant CLI Operations (p. 57)
- Use Session Scripts to Manage Your AppStream 2.0 Users' Streaming Experience (p. 66)

Creating Default Application and Windows Settings for Your AppStream 2.0 Users

Application customizations and Windows settings that are saved to the Windows user profile folder or the user registry hive can be set as defaults. When you save the default settings by using the Template User in Image Assistant, AppStream 2.0 replaces the Windows default user profile with the profile that you configure. The Windows default user profile is then used to create the initial settings for users in
the fleet instance. If the application or Windows settings that you configure don't work in the fleet, confirm that they are saved in the Windows user profile. For more information, see Step 3: Create Default Application and Windows Settings in Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47).

Default settings that you can create and configure include:

- Application preferences, including a browser home page, toolbar customizations, and security settings.
- Application data settings, including browser bookmarks and connection profiles.
- Windows experience settings, including displaying file name extensions and hidden folders.

Additionally, you can modify or disable Internet Explorer security settings, such as Enhanced Security Configuration (ESC). For more information, see Disable Internet Explorer Enhanced Security Configuration (p. 91).

### Optimizing the Launch Performance of Your Applications

When you create an image, AppStream 2.0 requires that you optimize the launch performance of your applications for your users' streaming sessions. When your applications are opened during this process, make sure that they use the initial components required by your users. Doing so ensures that these components are captured by the optimization process. In some cases, not all of the files required for the optimizations are detected. Examples of such files would be plug-ins or components that aren't opened in the image builder. To ensure that all of the files needed for your application are captured, you can include them in the optimization manifest. Adding files to the optimization manifest may increase the time it takes for fleet instances to be created and made available for users. Doing so, however, reduces the time it takes for the application to be launched the first time on the fleet instance.

To optimize all the files in a folder, open PowerShell as an administrator and use the following PowerShell command:

```bash
dir -path "C:\Path\To\Folder\To\Optimize" -Recurse -ErrorAction SilentlyContinue | %{$_.FullName} | Out-File "C:\ProgramData\Amazon\Photon\Prewarm\PrewarmManifest.txt" -encoding UTF8 -append
```

Alternatively, you can specify the optimization manifest on a per-application basis by using the Image Assistant command line interface (CLI) operations. For more information, see Create Your AppStream 2.0 Image Programmatically by Using the Image Assistant CLI Operations (p. 57).

### Manage AppStream 2.0 Agent Versions

The AppStream 2.0 agent is software that runs on your streaming instances and enables users to stream applications. When you create a new image, the Always use latest agent version option is selected by default. When this option is selected, new image builders or fleet instances that are launched from your image always use the latest AppStream 2.0 agent version. We recommend that you leave this option selected. In some cases, however, you may want to control agent updates to ensure compatibility.

The following procedures describe how to manage AppStream 2.0 agent versions.

**Contents**

- Create an Image That Always Uses the Latest Version of the AppStream 2.0 Agent (p. 37)
- Create an Image That Uses a Specific Version of the AppStream 2.0 Agent (p. 37)
- Create an Image That Uses a Newer Version of the AppStream 2.0 Agent (p. 38)
Create an Image That Always Uses the Latest Version of the AppStream 2.0 Agent

When your images are configured to always use the latest AppStream 2.0 agent version, your streaming instances are automatically updated with the latest features, performance improvements, and security updates that are available from AWS.

To create an image that always uses the latest version of the AppStream 2.0 agent

2. Do either of the following:
   - If you have an image builder that you want to use to create the image, start the image builder and then connect to it. If the image builder is not running the latest version of the AppStream 2.0 agent, you are prompted to choose whether to start the image builder with the latest agent. Make sure that this option is selected, choose Start, and then connect to the image builder.
   - If you do not have an image builder that you want to use to create the image, launch a new image builder. In Step 1: Choose Image, choose an AWS base image or a custom image. In Step 2: Configure Image Builder, if the image that you choose is not running the latest version of the AppStream 2.0 agent, the AppStream 2.0 section displays. In the Agent version list, select the latest agent version. Complete the remaining steps to create the image builder, and then connect to it. For more information, see Launch an Image Builder to Install and Configure Streaming Applications (p. 26).
3. On the image builder desktop, open Image Assistant and follow the steps to create your new image. For the Configure Image step, make sure that Always use the latest agent version is selected. For more information, see Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47).
   - If you decide later to not always use the latest version of the AppStream 2.0 agent, you must create a new image and clear that option.
4. Create a new fleet or modify an existing one. When you configure the fleet, select the new image that you created. For more information, see Create an AppStream 2.0 Fleet and Stack (p. 81).
5. Create a new stack or modify an existing one and associate it with your fleet.

Create an Image That Uses a Specific Version of the AppStream 2.0 Agent

You may want to control AppStream 2.0 agent updates rather than always using the latest version so that you can test for compatibility first. To ensure that the version of the AppStream 2.0 agent you use is compatible with your streaming applications, you can create an image that uses a specific version of the agent software. Then perform your qualification tests in a separate fleet before deploying to your production fleet.

When you create the image, make sure that the Always use latest agent version option is not selected. Doing so pins your image to the version of the AppStream 2.0 agent that you selected when you launched the image builder, rather than always using the latest version. After you finish your qualification tests, you can update your production fleet with the image.

To create an image that uses a specific version of the AppStream 2.0 agent

2. Do either of the following:
Create an Image That Uses a Newer Version of the AppStream 2.0 Agent

If you pin your image to a specific AppStream 2.0 agent version, you must update to a newer version by creating a new image. This approach lets you test each agent update for compatibility first, and then update your fleet incrementally.

When you create the image, make sure that the Always use latest agent version option is not selected. After you create your image, perform your qualification tests in a separate fleet before deploying to your production fleet. After you finish your qualification tests, you can update your production fleet with the image.

To create an image that uses a newer version of the AppStream 2.0 agent

2. Do either of the following:
   - If you have an image builder that you want to use to create the image, start the image builder and then connect to it. If the image builder is not running the latest version of the AppStream 2.0 agent, you are prompted to choose whether to start the image builder with the latest agent. Make sure that this option is selected, choose Start, and then connect to the image builder.
   - If you do not have an image builder that you want to use to create the image, launch a new image builder. In Step 1: Choose Image, choose an AWS base image or a custom image. In Step 2: Configure Image Builder, if the image that you choose is not running the latest version of the AppStream 2.0 agent, the AppStream 2.0 section displays. In the Agent version list, select the latest agent version. Complete the remaining steps to create the image builder, and then connect to it. For more information, see Launch an Image Builder to Install and Configure Streaming Applications (p. 26).
3. On the image builder desktop, open Image Assistant and follow the steps to create your new image. For the Configure Image step in Image Assistant, make sure that Always use the latest agent version is not selected. For more information, see Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47).
If you decide later to always use the latest version of the AppStream 2.0 agent, you must create a new image and select that option.

4. Create a new fleet or modify an existing one. When you configure the fleet, select the new image that you created. For more information, see Create an AppStream 2.0 Fleet and Stack (p. 81).

5. Create a new stack or modify an existing one and associate it with your fleet.

6. Connect to your fleet and test your applications for compatibility.

AppStream 2.0 Agent Version History

The Amazon AppStream 2.0 agent software runs on your streaming instances, enabling end users to connect to and start their streaming applications. Starting December 7, 2017, your streaming instances can be automatically updated with the latest features, performance improvements, and security updates that are available from AWS. Before December 7, 2017, agent updates were included with new base image releases.

To use the latest AppStream 2.0 agent software, you need to rebuild your images by using new base images published by AWS on or after December 7, 2017. When you do this, the option to enable automatic updates of the agent is selected by default in the Image Assistant. We recommend that you leave this option selected so that any new image builder or fleet instance that is launched from your image always uses the latest version of the agent. For more information, see Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47).

The following table describes the latest updates that are available in released versions of the AppStream 2.0 agent.

<table>
<thead>
<tr>
<th>Amazon AppStream 2.0 agent version</th>
<th>Changes</th>
</tr>
</thead>
</table>
| 02-19-2020                         | • Adds support for native application mode. For more information, see Native Application Mode (p. 175)  
• Improves interprocess communication between AppStream 2.0 components  
• Resolves an issue that caused streaming instances to fail to be provisioned  
• Works with these software components:  
  • Amazon SSM Agent — 2.3.701.0  
  • Amazon WDDM Hook Driver — 1.0.0.56  
  • EC2Config service (Windows Server 2012R2 only) — 4.9.3519.0 |
| 01-13-2020                         | • For persistent storage with Google Drive for G Suite, Team Drives have been renamed to Shared Drives  
• Resolves an issue that causes slow provisioning for streaming instances in Active Directory environments that have many users  
• Resolves an issue with accessing applications from the application switcher when the fleet user is an administrator  
• Works with these software components:  
  • Amazon SSM Agent — 2.3.701.0  
  • Amazon WDDM Hook Driver — 1.0.0.56 |
<table>
<thead>
<tr>
<th>Amazon AppStream 2.0 agent version</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• EC2Config service (Windows Server 2012R2 only) — 4.9.3519.0</td>
</tr>
</tbody>
</table>
| 11-13-2019                       | • AppStream 2.0 assemblies are now signed, including executables and installer packages  
|                                   | • Works with these software components:  
|                                   |   • Amazon SSM Agent — 2.3.701.0  
|                                   |   • Amazon WDDM Hook Driver — 1.0.0.56  
|                                   |   • EC2Config service — 4.9.3519.0 |
| 10-08-2019                       | • Modifies the AppStream 2.0 storage connector to no longer bypass the system proxy server  
|                                   | • Works with these software components:  
|                                   |   • Amazon SSM Agent — 2.3.701.0  
|                                   |   • Amazon WDDM Hook Driver — 1.0.0.56  
|                                   |   • EC2Config service — 4.9.3519.0 |
| 09-23-2019                       | • Resolves an issue that occurs when launching applications that start child processes  
|                                   | • Resolves an issue with directory traversal  
|                                   | • Resolves an issue that causes the AppStream 2.0 agent to stop functioning, which prevents interaction with applications  
|                                   | • Works with these software components:  
|                                   |   • Amazon SSM Agent — 2.3.701.0  
|                                   |   • Amazon WDDM Hook Driver — 1.0.0.56  
|                                   |   • EC2Config service — 4.9.3519.0 |
| 09-03-2019                       | • Adds support for applying IAM roles to AppStream 2.0 streaming instances. For more information, see Using an IAM Role to Grant Permissions to Applications and Scripts Running on AppStream 2.0 Streaming Instances (p. 232)  
|                                   | • Adds support for specifying tags when creating AppStream 2.0 images programmatically with a command line interface  
|                                   | • Modifies the AppStream 2.0 storage connector to bypass the system proxy server when mounting storage  
|                                   | • Resolves an issue that prevented .lnk files from being specified in Image Assistant  
|                                   | • Works with these software components:  
|                                   |   • Amazon SSM Agent — 2.3.612.0  
|                                   |   • Amazon WDDM Hook Driver — 1.0.0.56  
<p>|                                   |   • EC2Config service — 4.9.3429 |</p>
<table>
<thead>
<tr>
<th>Amazon AppStream 2.0 agent version</th>
<th>Changes</th>
</tr>
</thead>
</table>
| 08-08-2019                        | • Adds support for AppStream 2.0 file system redirection. For more information, see [Enable File System Redirection for Your AppStream 2.0 Users](p. 187)  
• Adds support for three new locales: English-United Kingdom (en-GB), English-Canada (en-CA), and English-Australia (en-AU)  
• Works with these software components:  
  • Amazon SSM Agent — 2.3.612.0  
  • Amazon WDDM Hook Driver — 1.0.0.56  
  • EC2Config service — 4.9.3429 |
| 07-26-2019                        | • Adds support for creating and managing AppStream 2.0 images programmatically with a command line interface. For more information, see [Create Your AppStream 2.0 Image Programmatically by Using the Image Assistant CLI Operations](p. 57).  
• Image creation is no longer blocked when automatic Windows updates are enabled on an image builder. However, a message notifies administrators that automatic Windows updates will be disabled on the fleet in this case (that is, automatic Windows updates won't be enabled on fleet instances).  
• Disables Windows updates when a fleet instance starts  
• User accounts in the Administrators group are no longer disabled when an image builder instance starts  
• User accounts in the Administrators group are now disabled rather than deleted when an image builder instance starts  
• Resolves an issue that prevents the streaming resolution from resizing when network connections change  
• Resolves a race condition that prevents the streaming resolution from resizing when application settings persistence is enabled  
• Works with these software components:  
  • Amazon SSM Agent — 2.3.612.0  
  • Amazon WDDM Hook Driver — 1.0.0.56  
  • EC2Config service — 4.9.3429 |
## Amazon AppStream 2.0 Agent Versions

<table>
<thead>
<tr>
<th>Amazon AppStream 2.0 agent version</th>
<th>Changes</th>
</tr>
</thead>
</table>
- AppStream 2.0 session scripts are now terminated after the configured timeout is exceeded  
- Resolves an issue where streaming instances may not be provisioned if the locale is changed  
- Includes a change to block image creation when automatic Windows updates are enabled on an image builder  
- Resolves an issue where streaming instances may take a long time to stop if the storage connector mount fails  
- Works with these software components:  
  - Amazon SSM Agent — 2.3.612.0  
  - Amazon WDDM Hook Driver — 1.0.0.56  
  - EC2Config service — 4.9.3429 |
| 05-07-2019                        | - Adds support for subscribing to AppStream 2.0 usage reports. For more information, see AppStream 2.0 Usage Reports (p. 207).  
- Adds support for configuring the amount of time that users can be idle (inactive) before they are disconnected from their streaming session. For more information, see "Create a Fleet" in Create an AppStream 2.0 Fleet and Stack (p. 81).  
- Resolves an issue with using Amazon S3 buckets for home folder and application settings persistence with an Amazon S3 virtual private gateway  
- Includes a change to block image creation when automatic Windows updates are enabled on an image builder  
- Resolves an issue with persistent storage drives (home folders, OneDrive, and Google Drive) intermittently disappearing from the My Files dialog box  
- Works with these software components:  
  - Amazon SSM Agent — 2.3.542.0  
  - Amazon WDDM Hook Driver — 1.0.0.56  
  - EC2Config service — 4.9.3289 |
| 04-02-2019                        | - Resolves an issue with session scripts and storage connector mounting  
- Resolves a minor issue with instance provisioning  
- Works with these software components:  
  - Amazon SSM Agent — 2.3.344.0  
  - Amazon WDDM Hook Driver — 1.0.0.56  
  - EC2Config service — 4.9.3067 |
<table>
<thead>
<tr>
<th>Amazon AppStream 2.0 agent version</th>
<th>Changes</th>
</tr>
</thead>
</table>
| 03-07-2019                        | • Adds support for gestures on touch-enabled iPads, Android tablets, and Windows devices  
• Resolves an issue with switching users in an image builder instance  
• Resolves an intermittent issue with instance reservations  
• Works with these software components:  
  • Amazon SSM Agent — 2.3.344.0  
  • Amazon WDDM Hook Driver — 1.0.0.56  
  • EC2Config service — 4.9.3067 |
| 01-22-2019                        | • Adds support for using on-instance session scripts to run your own custom scripts when specific events occur in users' streaming sessions  
• Adds support for adding tags to the following AppStream 2.0 resource types during resource creation: image builders, images, fleets, and stacks  
• Includes a fix removing storage connector log files from the application settings persistence Virtual Hard Disk (VHD) file  
• Prevents image creation when the display language is changed from English and the AWS Command Line Interface (AWS CLI) version is earlier than 1.16.36. For more information, see "Special Considerations for Japanese Language Settings" in Configure Default Regional Settings for Your AppStream 2.0 Users (p. 154).  
• Works with these software components:  
  • Amazon SSM Agent — 2.3.344.0  
  • Amazon WDDM Hook Driver — 1.0.0.56  
  • EC2Config service — 4.9.3067 |
| 01-08-2019                        | • Improves the instance provisioning time for base images dated 01-08-2019  
• Works with these software components:  
  • Amazon SSM Agent — 2.3.344.0  
  • Amazon WDDM Hook Driver — 1.0.0.56  
  • EC2Config service — 4.9.3067 |
| 12-19-2018                        | • Resolves an issue with dynamic applications not being added to the application catalog  
• Works with these software components:  
  • Amazon SSM Agent — 2.2.619.0  
  • Amazon WDDM Hook Driver — 1.0.0.56  
  • EC2Config service — 4.9.2644 |
<table>
<thead>
<tr>
<th>Amazon AppStream 2.0 agent version</th>
<th>Changes</th>
</tr>
</thead>
</table>
| **12-17-2018**                   | • The AppStream 2.0 client now supports a multiple-monitor experience for streaming instances that use a Graphics Design instance type  
• Resolves an issue with the temporary drive being visible on fleet instances that use a Graphics Desktop or Memory Optimized instance type  
• Works with these software components:  
  • Amazon SSM Agent — 2.2.619.0  
  • Amazon WDDM Hook Driver — 1.0.0.56  
  • EC2Config service — 4.9.2644 |
| **12-04-2018**                   | • Adds support for using a Japanese keyboard with web clients that run on Windows  
• Adds support for using the AppStream 2.0 dynamic application framework APIs to build a dynamic app provider  
• Resolves an issue with streaming the same session concurrently on multiple tabs or browsers  
• Includes a fix to make home folders, Google Drive, and OneDrive read-only until mounting is completed  
• Improves the mount time for home folders that are stored on fleet instances connected to an Amazon S3 VPC endpoint  
• Works with these software components:  
  • Amazon SSM Agent — 2.2.619.0  
  • Amazon WDDM Hook Driver — 1.0.0.56  
  • EC2Config service — 4.9.2644 |
| **11-14-2018**                   | • Adds support for launching streaming sessions using the AppStream 2.0 Windows client  
• Resolves an issue with opening applications that use environment variables for the fleet user name  
• Works with these software components:  
  • Amazon SSM Agent — 2.2.619.0  
  • Amazon WDDM Hook Driver — 1.0.0.56  
  • EC2Config service — 4.9.2644 |
| **10-30-2018**                   | • Resolves an issue with mounting home folders that are larger than 1 GB when application settings persistence is enabled  
• Resolves an issue with image creation when IPv6 is disabled  
• Session information is now provided as environment variables within streaming instances  
• Works with these software components:  
  • Amazon SSM Agent — 2.2.619.0  
  • Amazon WDDM Hook Driver — 1.0.0.56  
  • EC2Config service — 4.9.2644 |
<table>
<thead>
<tr>
<th>Amazon AppStream 2.0 agent version</th>
<th>Changes</th>
</tr>
</thead>
</table>
| 10-24-2018                        | • Includes a fix to display more than 1,000 files in the Amazon S3 home folders directory  
• Works with these software components:  
  • Amazon SSM Agent — 2.2.619.0  
  • Amazon WDDM Hook Driver — 1.0.0.56  
  • EC2Config service — 4.9.2644 |
| 10-01-2018                        | • Improves the performance of application settings persistence  
• Includes a fix to unhide all drives on a fleet instance, except Drive C and Drive D, during user streaming sessions that are launched from the instance  
• Resolves an issue with accessing minimized application subwindows from the application switcher  
• Works with these software components:  
  • Amazon SSM Agent — 2.2.619.0  
  • Amazon WDDM Hook Driver — 1.0.0.56  
  • EC2Config service — 4.9.2644 |
| 08-29-2018                        | • Adds support for application settings persistence  
• Resolves an issue with copying and pasting large amounts of data between applications within an AppStream 2.0 streaming session  
• Resolves an issue with accessing unresponsive applications from the application switcher  
• Works with these software components:  
  • Amazon SSM Agent — 2.2.619.0  
  • Amazon WDDM Hook Driver — 1.0.0.56  
  • EC2Config service — 4.9.2644 |
| 07-26-2018                        | • Adds support for OneDrive persistent storage  
• Resolves an issue with saving Visio files to home folders and Google Drive  
• Works with these software components:  
  • Amazon SSM Agent — 2.2.619.0  
  • Amazon WDDM Hook Driver — 1.0.0.56  
  • EC2Config service — 4.9.2644 |
| 06-19-2018                        | • Resolves an issue with optimizing images for application launch  
• Works with these software components:  
  • Amazon SSM Agent — 2.2.619.0  
  • Amazon WDDM Hook Driver — 1.0.0.56  
  • EC2Config service — 4.9.2644 |
<table>
<thead>
<tr>
<th>Amazon AppStream 2.0 agent version</th>
<th>Changes</th>
</tr>
</thead>
</table>
| 06-06-2018                        | • Adds support for regional settings and default application and Windows settings  
• Works with these software components:  
  • Amazon SSM Agent — 2.2.619.0  
  • Amazon WDDM Hook Driver — 1.0.0.56  
  • EC2Config service — 4.9.2644 |
| 05-31-2018                        | • Adds support for Google Drive persistent storage  
• Works with these software components:  
  • Amazon SSM Agent — 2.2.392.0  
  • Amazon WDDM Hook Driver — 1.0.0.56  
  • EC2Config service — 4.9.2586 |
| 05-21-2018                        | • Adds support for administrative controls for data transfer  
• Adds support for the Safari browser on macOS X  
• Works with these software components:  
  • Amazon SSM Agent — 2.2.392.0  
  • Amazon WDDM Hook Driver — 1.0.0.56  
  • EC2Config service — 4.9.2586 |
| 03-19-2018                        | • Resolves an issue with minimizing the application window in certain environments  
• Works with these software components:  
  • Amazon SSM Agent — 2.2.160.0  
  • Amazon WDDM Hook Driver — 1.0.0.56  
  • EC2Config service — 4.9.2400.0 |
| 01-24-2018                        | • Resolves an issue with the Alt Graph key not working on certain keyboard layouts  
• Works with these software components:  
  • Amazon SSM Agent — 2.2.93.0  
  • Amazon WDDM Hook Driver — 1.0.0.50  
  • EC2Config service — 4.9.2262.0 |
| 12-07-2017                        | • Resolves issues with using ALT key combinations  
• Resolves an issue with file uploads from local computers to streaming sessions  
• Works with these software components:  
  • Amazon SSM Agent — 2.2.93.0  
  • Amazon WDDM Hook Driver — 1.0.0.21  
  • EC2Config service — 4.9.2218.0 |
Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console

This tutorial describes how to create a custom Amazon AppStream 2.0 image that contains applications you can stream to your users, and default application and Windows settings to enable your users to get started with their applications quickly. To complete this tutorial, you must already have an image builder. If you don’t have an image builder, see Launch an Image Builder to Install and Configure Streaming Applications (p. 26).

Important
This tutorial includes information that applies to the latest base image release. For more information, see AppStream 2.0 Base Image Version History (p. 29).

Contents
• Step 1: Install Applications on the Image Builder (p. 47)
• Step 2: Create an AppStream 2.0 Application Catalog (p. 48)
• Step 3: Create Default Application and Windows Settings (p. 48)
• Step 4: Test Applications (p. 49)
• Step 5: Optimize Applications (p. 50)
• Step 6: Finish Creating Your Image (p. 50)
• Step 7 (Optional): Tag and Copy an Image (p. 51)
• Step 8: Clean Up (p. 52)

Step 1: Install Applications on the Image Builder

In this step, you connect an image builder and install your applications on the image builder.

Important
To complete this step, you must log into the image builder with the local Administrator account or a domain user account that has local administrator permissions.

To install applications on the image builder

1. In the left navigation pane, choose Images, Image Builder.
2. Choose the image builder to use, verify that it is in the Running state, and choose Connect. For this step to work, you may need to configure your browser to allow pop-ups from https://stream.<aws-region>.amazonappstream.com/.
3. Log in to the image builder by doing either of the following:
   • If your image builder is not joined to an Active Directory domain, on the Local User tab, choose Administrator.
   • If your image builder is joined to an Active Directory domain and you require access to resources that are managed by Active Directory to install your applications, choose the Directory User tab, type the credentials for a domain user account that has local administrator permissions on the image builder, then choose Log in.
4. Install applications from an application website or other download source. Install the applications you want before proceeding to the next step.

Note
Download and install applications only from sites that you trust.
If an application requires the Windows operating system to restart, let it do so. Before the operating system restarts, you are disconnected from your image builder. After the restart is complete, connect to the image builder again, then finish installing the application.

**Step 2: Create an AppStream 2.0 Application Catalog**

In this step, create an AppStream 2.0 application catalog by specifying applications (.exe), batch scripts (.bat), and application shortcuts (.lnk) for your image. For each application that you plan to stream, you can specify the name, display name, executable file to launch, and icon to display. If you choose an application shortcut, these values are prepopulated for you.

**Important**
To complete this step, you must be logged into the image builder with the local Administrator account or a domain user account that has local administrator permissions.

**To create an AppStream 2.0 application catalog**

1. From the image builder desktop, open Image Assistant. Image Assistant guides you through the image creation process.
2. In **Add Apps**, choose **Add App**, and navigate to the location of the application, script, or shortcut to add. Choose **Open**.
3. In the **App Launch Settings** dialog box, keep or change the default settings for **Name**, **Display Name**, and **Icon Path**. Optionally, you can specify launch parameters (additional arguments passed to the application when it is launched) and a working directory for the application. When you're done, choose **Save**.

The **Display Name** and **Icon Path** settings determine how your application name and icon appear in the application catalog. The catalog displays to users when they sign in to an AppStream 2.0 streaming session.
4. Repeat steps 2 and 3 for each application in Image Assistant and confirm that the applications appear on the **Add Apps** tab. When you're done, choose **Next** to continue using Image Assistant to create your image.

**Step 3: Create Default Application and Windows Settings**

In this step, you create default application and Windows settings for your AppStream 2.0 users. Doing this enables your users to get started with applications quickly during their AppStream 2.0 streaming sessions, without the need to create or configure these settings themselves.

**Important**
To complete this step, you must be logged into the image builder with the local Template User account or a domain user account that does not have local administrator permissions.

**To create default application and Windows settings for your users**

1. In Image Assistant, in **Configure Apps**, choose **Switch user**. This disconnects you from the current session and displays the login menu.
2. Do either of the following:
   - If your image builder is not joined to an Active Directory domain, on the **Local User** tab, choose **Template User**. This account enables you to create your default application and Windows settings.
• If your image builder is joined to an Active Directory domain, choose **Directory User**, and log in as a domain user that does not have local administrator permissions.

3. From the image builder desktop, open Image Assistant, which displays the applications that you added when you created the application catalog.

4. Choose the application for which you want to create default application settings.

5. After the application opens, create these settings as needed.

6. When you're done, close the application, and return to Image Assistant.

7. If you specified more than one application in Image Assistant, repeat steps 4 through 6 for each application as needed.

8. If you want default Windows settings, create them now. When you're done, return to Image Assistant.

9. Choose **Switch user** and log in with the same account that you used to create the application catalog (an account that has local administrator permissions).

10. In Image Assistant, in **2. Configure Apps**, do either of the following:

    • If your image builder is not joined to an Active Directory domain, choose **Save settings**.

    • If your image builder is joined to an Active Directory domain, in the **Choose which user settings to copy** list, choose the same account that you used to log into the image builder when you created the default application and Windows settings, then choose **Save settings**.

    The **Choose which settings to copy** list displays any user account that currently has settings saved on the image builder.

11. When you're done, choose **Next** to continue creating your image.

### Step 4: Test Applications

In this step, verify that the applications you've added open correctly and perform as expected. To do so, start a new Windows session as a user who has the same permissions as your users.

**Important**
To complete this step, you must log in to the image builder with the **Test User** account or a domain user account that does not have local administrator permissions.

**To test your applications**

1. In Image Assistant, in **3. Test**, do either of the following:

    • If your image builder is not joined to an Active Directory domain, choose **Switch user**.

    • If your image builder is joined to an Active Directory domain, you require a domain user account to test your applications, and the user already has settings on the image builder, you must reset the application settings for that user. To do so, select the user from the **User to reset** list, and choose **Reset**. When you're done, choose **Switch user**.

    **Note**
If your image builder is new and no users have settings on the image builder, the list does not display any users.

2. Choose the user account to use for testing by doing either of the following:

    • If your image builder is not joined to an Active Directory domain, choose **Test User**. This account enables you to test your applications by using the same policies and permissions as your users.

    • If your image builder is joined to an Active Directory domain, choose **Directory User**, specify the credentials for a domain user account that does not have local administrator permissions, then choose **Log in**.
3. From the image builder desktop, open Image Assistant, which displays the applications that you specified when you created the application catalog.

4. Choose the application that you want to test, to confirm that it opens correctly and that any default application settings you created are applied.

5. After the application opens, test it as needed. When you're done, close the application and return to Image Assistant.

6. If you specified more than one application in Image Assistant, repeat steps 4 and 5 to test each application as needed.

7. When you're done, choose **Switch user**, then do either of the following:
   - If your image builder is not joined to an Active Directory domain, on the **Local User** tab, choose **Administrator**.
   - If your image builder is joined to an Active Directory domain and you logged in as a domain user with local administrator permissions to specify applications in Image Assistant, log in as that user.

8. Choose **Next** to continue creating your image.

**Step 5: Optimize Applications**

In this step, Image Assistant opens your applications one after another, identifies their launch dependencies, and performs optimizations to ensure that applications launch quickly. These are required steps that are performed on all applications in the list.

**To optimize your applications**

1. In Image Assistant, in **4. Optimize**, choose **Launch**.

2. AppStream 2.0 automatically launches the first application in your list. When the application completely starts, provide any required input to perform the first run experience for the application. For example, a web browser may prompt you to import settings before it is completely up and running.

3. After you complete the first run experience and verify that the application performs as expected, choose **Continue**. If you added more than one application to your image, each application opens automatically. Repeat this step for each application as needed, leaving all applications running.

4. When you're done, the next tab in Image Assistant, **5. Configure Image**, automatically displays.

**Step 6: Finish Creating Your Image**

In this step, choose an image name and finish creating your image.

**To create the image**

1. Type a unique image name, and an optional image display name and description. The image name cannot begin with "Amazon," "AWS," or "AppStream."

   You can also add one or more tags to the image. To do so, choose **Add Tag**, and type the key and value for the tag. To add more tags, repeat this step. For more information, see [Tagging Your Amazon AppStream 2.0 Resources](p. 201). When you're done, choose **Next**.

   **Note**

   If you choose a base image that is published by AWS on or after December 7, 2017, the option **Always use the latest agent version** appears, selected by default. We recommend that you leave this option selected so that streaming instances that are launched from the image always use the latest version of the agent. If you disable this option, you cannot
enable it after you finish creating the image. For information about the latest release of the AppStream 2.0 agent, see AppStream 2.0 Agent Version History (p. 39).

2. In 6. Review, verify the image details. To make changes, choose Previous to navigate to the appropriate Image Assistant tab, make your changes, and then proceed through the steps in Image Assistant as needed.

3. After you finish reviewing the image details, choose Disconnect and Create Image.

4. The remote session disconnects within a few moments. When the Lost Connectivity message appears, close the browser tab. While the image is created, the image builder status appears as Snapshotting. You cannot connect to the image builder until this process finishes.

5. Return to the console and navigate to Images, Image Registry. Verify that your new image appears in the list.

While your image is being created, the image status in the image registry of the console appears as Pending and you cannot connect to it.

6. Choose the Refresh icon periodically to update the status. After your image is created, the image status changes to Available and the image builder is automatically stopped.

To continue creating images, start the image builder and connect to it from the console, or create a new image builder.

Note
After you create an image, you can’t change it. To add other applications, update existing applications, or change image settings, you must start and reconnect to the image builder that you used to create the image, or, if you deleted that image builder, launch a new image builder that is based on your image. Then, make your changes and create a new image.

Step 7 (Optional): Tag and Copy an Image

You can add one or more tags to an image during image creation or after you create an image. You can also copy the image within the same Region or to a new Region within the same AWS account. Copying a source image results in an identical but distinct destination image. AWS does not copy any user-defined tags, however. Also, you can only copy custom images that you create, not the base images that are provided by AWS.

Note
You can copy up to two images at the same time to a destination. If the destination to which you are copying an image is at the image limit, you receive an error. To copy the image in this case, you must first remove images from the destination. After the destination is below the image quota (also referred to as limit), initiate the image copy from the source Region. For more information, see Amazon AppStream 2.0 Service Quotas (p. 271).

To add tags to an existing image

1. In the navigation pane, choose Images, Image Registry.

2. In the image list, select the image to which you want to add tags.

3. Choose Tags, choose Add/Edit Tags, choose Add Tag, specify the key and value for the tag, and then choose Save.

For more information, see Tagging Your Amazon AppStream 2.0 Resources (p. 201).

To copy an image

Copying an image across geographically diverse regions enables you to stream applications from multiple regions based on the same image. By streaming your applications in closer proximity to your users, you can improve your users’ experience streaming applications with AppStream 2.0.
1. In the navigation pane, choose Images, Image Registry.
2. In the image list, select the image that you want to copy.
3. Choose Actions, Copy.
4. In the Copy Image dialog box, specify the following information, and then choose Copy Image:
   - For Destination region, choose the region to which to copy the new image.
   - For Name, specify a name that the image will have when it is copied to the destination.
   - For Description (optional), specify a description that the image will have when it is copied to the destination.
5. To check on the progress of the copy operation, return to the console and navigate to Images, Image Registry. Use the navigation bar to switch to the destination region (if applicable), and confirm that your new image appears in the list of images.

The new image first appears with a status of Copying in the image registry of your console. After the image is successfully created, the status of the image changes to Available, which means that you can use the image to launch a stack and stream your applications.

Step 8: Clean Up

Finally, stop your running image builders to free up resources and avoid unintended charges to your account. We recommend stopping any unused, running image builders. For more information, see AppStream 2.0 Pricing.

To stop a running image builder

1. In the navigation pane, choose Images, Image Builders, and select the running image builder instance.
2. Choose Actions, Stop.

Administer Your Amazon AppStream 2.0 Images

Available images are listed in the Image Registry in the AppStream 2.0 console, and categorized by visibility as follows:

- **Public** — Base images that are owned and made available by AWS. Base images include the latest Windows operating system and the AppStream 2.0 agent software. You can use these base images to create new images that include your own applications. For information about the base images released by AWS, see AppStream 2.0 Base Image Version History (p. 29).
- **Private** — Images that you create and own, and that you have not shared with other AWS accounts.
- **Shared with others** — Images that you create and own, and that you have shared with one or more AWS accounts in the same AWS Region. When you share an image with another AWS account, you can specify whether the image can be used for an image builder (to create a new image), for a fleet, or both.
- **Shared with me** — Images that are created and owned by another AWS account in the same AWS Region, and that are shared with your AWS account. Depending on the permissions that the owner provided when sharing the image with your account, you can use this image for image builders, for fleets, or both.

Contents

- Delete a Private Image (p. 53)
- Copy an Image That You Own to Another AWS Region (p. 53)
Delete a Private Image

You can delete your private images when you no longer need them. You can't delete an image that is used by fleets or shared with other AWS accounts. To delete an image that is used by fleets or shared, you must first remove the image from any fleets and remove all image sharing permissions. After you delete an image, you can't recover it.

To delete a private image

2. In the navigation pane, choose **Images, Image Registry**.
3. In the image list, select the private image you want to delete.
4. Choose **Actions, Delete**, then choose **Delete** again.

The image is removed from the image registry and deleted.

Copy an Image That You Own to Another AWS Region

You can copy images that you own to another AWS Region. Using the same image across different AWS Regions can help simplify global deployments of your applications on AppStream 2.0. By deploying your applications in the AWS Regions that are geographically closest to your users, you can help provide your users with a more responsive experience.

To copy an image that you own to another AWS Region

2. In the navigation pane, choose **Images, Image Registry**.
3. In the image list, select the image that you want to copy to another AWS Region.
4. Choose **Actions, Copy**.
5. In the **Copy image** dialog box, in **Destination region**, select the AWS Region that you want to copy the image to.
6. Type a unique name and optionally, a description for the image in **Destination region**.
7. Choose **Copy Image**.

Share an Image That You Own With Another AWS Account

AppStream 2.0 images are a regional resource, so you can share an image that you own with other AWS accounts within the same AWS Region. Doing so can be helpful in several different scenarios. For example, if you separate your development and production resources by using different AWS accounts, you can create an image by using your development account. Then you can share the image with your production account. If your organization is an independent software vendor (ISV), you can share...
optimized images with your customers. Optimized images that have the required applications already installed and configured let your customers get started with your applications quickly, so that they won’t need to install and configure those applications themselves.

When you share an image with another AWS account, you specify whether the destination account can use the image in a fleet or create new images by creating an image builder. You continue to own images that you share. This way, you can add, change, or remove permissions as needed for your shared images.

If you share an image with an account and grant the account fleet permissions, the shared image can be used to create or update fleets in that account. If you remove these permissions later, the account can no longer use the image. For fleets in the account that use the shared image, the desired capacity is set to 0, which prevents new fleet instances from being created. Existing sessions continue until the streaming session ends. For new fleet instances to be created, the fleet in that account must be updated with a valid image.

If you share an image with an account and grant the account image builder permissions, the shared image can be used to create image builders and images in that account. If you remove these permissions later, image builders and images that were created from your image are not affected.

**Important**
After you share an image with an account, you can't control image builders or images in the account that are created from your image. For this reason, grant image builder permissions to an account only if you want to enable the account to make a copy of your image, and retain access to the copy after you stop sharing your image.

**To share an image that you own with another AWS account**

2. In the navigation pane, choose *Images, Image Registry*.
3. In the image list, select the image that you want to share.
4. Choose *Actions, Share*.
5. In the *Share image* dialog box, choose *Add account*.
6. Type the 12-digit AWS account ID of the account that you want to share the image with, and then select whether the account can do one or both of the following:
   - Use the image to launch an image builder, if you want to create a new image.
   - Use the image with a fleet.

   To remove an account from the list of accounts that the image is shared with, in the row for the account you want to remove, choose the X icon to the right of the *Use for fleet* option.

7. To share the image with more AWS accounts, repeat step 6 for each account that you want to share the image with.
8. Choose *Share Image*.

**To add or update image sharing permissions for an image that you own**

2. In the navigation pane, choose *Images, Image Registry*.
3. In the image list, select the image that you want to change the permissions for.
4. Below the image list, choose the *Permissions* tab for the image you selected, then choose *Edit*.
5. In the *Edit image permissions* dialog box, select or clear one or both of the following image sharing options as needed for one or more AWS accounts. If you clear both options for an account, the image is no longer shared with that account:
   - Use the image to launch an image builder, if you want to create a new image.
Stop Sharing an Image That You Own

Follow these steps to stop sharing an image that you own with any other AWS account.

To stop sharing an image that you own with any other AWS account

2. In the navigation pane, choose Images, Image Registry.
3. In the image list, select the image that you want to change the permissions for.
4. Below the image list, choose the Permissions tab for the image you selected, then choose Edit.
5. In the Edit image permissions dialog box, in the row for all AWS accounts that the image is shared with, choose the X icon to the right of the Use for fleet option.
6. Choose Update image sharing permissions.

Keep Your AppStream 2.0 Image Up-to-Date

AppStream 2.0 provides an automated way to update your image builder with newer AppStream 2.0 agent software. Doing so enables you to create a new image whenever a new version of the agent is released. You can then test the image before updating your production fleets. For more information about how to manage the AppStream 2.0 agent software, see Manage AppStream 2.0 Agent Versions (p. 36).

You are responsible for installing and maintaining the updates for the Windows operating system, your applications, and their dependencies. To keep your AppStream 2.0 image updated with the latest Windows operating system updates, do one of the following:

- Install your applications on the latest base image each time a new image is released.
- Install the updates for the Windows operating system, your applications, and their dependencies on an existing image builder.
- Install the updates for the Windows operating system, your applications, and their dependencies on a new image builder from an existing image.

After you create a new image with the latest Windows operating system, applications and their dependencies, and the AppStream 2.0 agent software, test the image on a development fleet. After you verify that your applications work as expected, update your production fleet with the new image.

Windows Update and Antivirus Software on AppStream 2.0

AppStream 2.0 streaming instances are non-persistent. When a user streaming session ends, AppStream 2.0 terminates the instance used by the session and, depending on your scaling policies, provisions a
new instance to replace it in your fleet. All fleet instances are provisioned from the same image. Because images cannot be changed once created, all fleet instances used in user streaming sessions have only the Windows and application updates that were installed on the underlying image when the image was created. In addition, because a fleet instance used for a streaming session terminates at the end of the session, any updates made to Windows or to applications on the instance during the streaming session will not persist to future sessions by the same user or other users.

Note
If you enabled application settings persistence for your stack, AppStream 2.0 persists Windows and application configuration changes made by a user to future sessions for the same user if those configuration changes are stored in the user’s Windows profile. However, the application settings persistence feature persists only Windows and application configuration settings. It does not persist software updates to Windows or applications on the streaming instance.

For these reasons, AppStream 2.0 takes the following approach to Windows Update and antivirus software on AppStream 2.0 instances.

Windows Update

Windows Update is not enabled by default on AppStream 2.0 base images. If you enable Windows Update on an image builder and then try to create an image, Image Assistant displays a warning and disables Windows Update during the image creation process. To ensure that your fleet instances have the latest Windows updates installed, we recommend that you install Windows updates on your image builder, create a new image, and update your fleet with the new image on a regular basis.

Antivirus Software

If you choose to install antivirus software on your image, we recommend that you do not enable automatic updates for the antivirus software. Otherwise, the antivirus software may attempt to update itself with the latest definition files or other updates during user sessions. This may affect performance. In addition, any updates made to the antivirus software will not persist beyond the current user session. To ensure that your fleet instances always have the latest antivirus updates, we recommend that you do either of the following:

- Update your image builder and create a new image on a regular basis (for example, by using the Image Assistant CLI operations).
- Use an antivirus application that delegates scanning or other operations to an always-up-to-date external server.

Note
Even if you do not enable automatic updates for your antivirus software, the antivirus software may perform hard drive scans or other operations that may impact the performance of your fleet instances during user sessions.


To enable Windows Defender manually

If Windows Defender is not enabled on your base image, you can enable it manually. To do so, complete the following steps.

2. In the left navigation pane, choose Images, Image Builder.
3. Choose the image builder on which to enable Windows Defender, verify that it is in the Running state, and choose Connect.
4. Log in to the image builder with the local Administrator account or with a domain user account that has local administrator permissions.
5. Open Registry Editor.
6. Navigate to the following location in the registry: HKLM\SOFTWARE\Policies\Microsoft\Windows Defender\DisableAntiSpyware.
7. To edit this registry key, double-click it, or right-click the registry key, and choose Modify.
8. In the Edit DWORD (32-bit) Value dialog box, in Value data, change 1 to 0.
9. Choose OK.
10. Close Registry Editor.
11. Open the Microsoft Management Console (MMC) Services snap-in (services.msc).
12. In the list of services, do either of the following:
   • Right-click Windows Defender Antivirus Service, and choose Start.
   • Double-click Windows Defender Antivirus Service, choose Start in the properties dialog box, and then choose OK.
13. Close the Services snap-in.

Programmatically Create a New Image

You can create AppStream 2.0 images programmatically by connecting to an image builder and using the Image Assistant command line interface (CLI) operations. For more information, see Create Your AppStream 2.0 Image Programmatically by Using the Image Assistant CLI Operations (p. 57).

Create Your AppStream 2.0 Image Programmatically by Using the Image Assistant CLI Operations

You can create Amazon AppStream 2.0 images by connecting to an image builder and using the Image Assistant graphical user interface (GUI) or command line interface (CLI) operations. The Image Assistant CLI operations provide functionality that is similar to the Image Assistant GUI. With these operations, you can programmatically do the following:

- Manage the applications that are included in an image.
- Save, update, and reset default application settings.
- Enable or disable the AppStream 2.0 dynamic application framework.
- Specify tags.
- Create an image.

You can use these operations to integrate AppStream 2.0 image creation with your continuous integration or deployment software development process.

To work with the Image Assistant CLI operations, use the command line shell of your choice on an image builder. For example, you can use the Windows command prompt or PowerShell.
Creating Default Application and Windows Settings with the Image Assistant CLI operations

You can create default application and Windows settings so that your users can get started with their applications quickly. When you create these settings, AppStream 2.0 replaces the Windows default user profile with the profile that you configure. The Windows default user profile is then used to create the initial settings for users in the fleet instance. If you create these settings by using the Image Assistant CLI operations, your application installer, or the automation, should modify the Windows default user profile directly.

To overwrite the Windows default user profile with that of another Windows user, you can also use the Image Assistant `update-default-profile` CLI operation.

For more information about configuring default application and Windows settings, see Creating Default Application and Windows Settings for Your AppStream 2.0 Users in Default Application and Windows Settings and Application Launch Performance (p. 35).

Optimizing the Launch Performance of Your Applications with the Image Assistant CLI Operations

AppStream 2.0 lets you optimize the launch performance of your applications for your users’ streaming sessions. When you do so by using the Image Assistant CLI operations, you can specify the files to optimize for your application launch. Adding files to the application optimization manifest reduces the time that it takes for the application to launch for the first time on a new fleet instance. However, this also increases the time that it takes for the fleet instances to be made available to users. The optimization manifest is a line-delimited text file that is per application.

For more information about creating application optimization manifests, see Creating Default Application and Windows Settings with the Image Assistant CLI operations (p. 58).

Note

If you onboard application optimization manifests by using both the Image Assistant CLI operations and the Image Assistant GUI, the manifests are merged.

Following is an example of an application optimization manifest file named C:\path\to\application-optimization-manifest.txt:

```
C:\Program Files (x86)\Notepad++\autoCompletion
C:\Program Files (x86)\Notepad++\localization
C:\Program Files (x86)\Notepad++\plugins
C:\Program Files (x86)\Notepad++\themes
C:\Program Files (x86)\Notepad++\updater
C:\Program Files (x86)\Notepad++\userDefineLangs
C:\Program Files (x86)\Notepad++\change.log
C:\Program Files (x86)\Notepad++\config.xml
```
For more information about optimizing the launch performance of your applications, see *Optimizing the Launch Performance of Your Applications* in Default Application and Windows Settings and Application Launch Performance (p. 35).

**Process Overview for Programmatically Creating an AppStream 2.0 Image**

You can use the Image Assistant CLI operations with your application installation automation to create a fully programmatic AppStream 2.0 image creation workflow. After your application installation automation completes, but before the image is created, use the Image Assistant CLI operations to specify the following:

- The executable files that your users can launch
- The optimization manifests for your applications
- Other AppStream 2.0 image metadata

The following high-level overview describes the process for programmatically creating an AppStream 2.0 image.

1. Use your application installation automation to install the required applications on your image builder. This installation may include applications that your users will launch, any dependencies, and background applications.
2. Determine the files and folders to optimize.
3. If applicable, use the Image Assistant `add-application` CLI operation to specify the application metadata and optimization manifest for the AppStream 2.0 image.
4. To specify additional applications for the AppStream 2.0 image, repeat steps 1 through 3 for each application as needed.
5. If applicable, use the Image Assistant `update-default-profile` CLI operation to overwrite the default Windows profile and create default application and Windows settings for your users.
6. Use the Image Assistant `create-image` CLI operation to create the image.

**Image Assistant CLI Operations for Creating and Managing Your AppStream 2.0 Image**

This section describes the Image Assistant CLI operations that you can use to create and manage your AppStream 2.0 image.

The executable file that includes the command line interface is located at: C:\Program Files\Amazon\Photon\ConsoleImageBuilder\Image-Assistant.exe. For your convenience, this executable file is included in the Windows PATH variable. This lets you call the Image Assistant CLI operations without specifying the absolute path to the executable file. To call these operations, type the image-assistant.exe command.
help operation

Retrieves a list of all Image Assistant CLI operations. For each operation in the list, a description and usage syntax is provided. To display help for a specific operation, type the name of the operation and specify the --help parameter. For example:

```
add-application --help
```

Synopsis

```bash
help
```

Output

Prints to standard out the list of available operations with a description of their function.

add-application operation

Adds the application to the application list for AppStream 2.0 users. Applications in this list are included in the application catalog. The application catalog displays to users when they sign in to an AppStream 2.0 streaming session.

Synopsis

```
add-application
  --name <value>
  --absolute-app-path <value>
  [--display-name <value>]
  [--absolute-icon-path <value>]
  [--working-directory <value>]
  [--launch-parameters ""-escaped value"]
  [--absolute-manifest-path <value>]
```

Options

|--name (string)
  A unique name for the application. The maximum length is 256 characters. You can add up to 50 applications. You cannot use whitespace characters.

|--absolute-app-path (string)
  The absolute path to the executable file, batch file, or script for the application. The path must point to a valid file.

|--display-name (string)
  The name to display for the application in the application catalog. If you don't specify a display name, AppStream 2.0 creates a name that is derived from the executable file name. The name is created without the file extension and with underscores in place of spaces. The maximum length is 256 characters.

|--absolute-icon-path (string)
  The absolute path to the icon for the application. The path must point to a valid icon file that is one of the following types: .jpg, .png, or .bmp. The maximum dimensions are: 256 px x 256 px. If you don't specify a path, the default icon for the executable file is used, if available. If a default icon is not available for the executable file, a default AppStream 2.0 application icon is used.

|--working-directory (string)
  The initial working directory for the application when the application is launched.
--optimization-manifest (string)

The path to a new line-delimited text file. The file specifies the absolute paths of the files to optimize before the fleet instance is made available to a user for streaming. The path must point to a valid text file.

Message output

<table>
<thead>
<tr>
<th>Exit code</th>
<th>Message printed to standard out</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>&quot;status&quot;: 0, &quot;message&quot;: &quot;Success&quot;</td>
<td>The application was added successfully.</td>
</tr>
<tr>
<td>1</td>
<td>&quot;status&quot;: 1, &quot;message&quot;: &quot;Administrator privileges are required to perform this operation.&quot;</td>
<td>Administrator privileges are required to complete the operation.</td>
</tr>
<tr>
<td>1</td>
<td>&quot;status&quot;: 1, &quot;message&quot;: &quot;Unable to add more than 50 apps to the catalog.&quot;</td>
<td>The application could not be added because the maximum number of applications that can be added to the AppStream 2.0 application catalog is 50.</td>
</tr>
<tr>
<td>1</td>
<td>&quot;status&quot;: 1, &quot;message&quot;: &quot;Name is not unique&quot;</td>
<td>An application with that name already exists in the AppStream 2.0 application catalog.</td>
</tr>
<tr>
<td>1</td>
<td>&quot;status&quot;: 1, &quot;message&quot;: &quot;File not found (absolute-app-path)&quot;</td>
<td>The file that was specified for absolute-app-path could not be found.</td>
</tr>
<tr>
<td>1</td>
<td>&quot;status&quot;: 1, &quot;message&quot;: &quot;Unsupported file extension&quot;</td>
<td>The Absolute-app-path parameter only supports the following file types: .exe and .bat.</td>
</tr>
<tr>
<td>1</td>
<td>&quot;status&quot;: 1, &quot;message&quot;: &quot;Directory not found (working-directory)&quot;</td>
<td>The directory that was specified for working-directory could not be found.</td>
</tr>
<tr>
<td>1</td>
<td>&quot;status&quot;: 1, &quot;message&quot;: &quot;Optimization-manifest not found: &lt;filename&gt;&quot;</td>
<td>The file that was specified for optimization-manifest could not be found.</td>
</tr>
<tr>
<td>1</td>
<td>&quot;status&quot;: 1, &quot;message&quot;: &quot;File not found: &lt;filename&gt;&quot;</td>
<td>A file that was specified within the optimization manifest could not be found.</td>
</tr>
<tr>
<td>255</td>
<td>&quot;status&quot;: 255, &quot;message&quot;: &lt;error message&gt;</td>
<td>An unexpected error occurred. Try the request again. If the error persists, contact AWS Support for assistance. For more information, see AWS Support Center.</td>
</tr>
</tbody>
</table>

remove-application operation

Removes an application from the application list for the AppStream 2.0 image. The application is not uninstalled or modified, but users will not be able to launch it from the AppStream 2.0 application catalog.

Synopsis
remove-application
--name <value>

Options

--name (string)
The unique identifier of the application to remove.

Message output

<table>
<thead>
<tr>
<th>Exit code</th>
<th>Message printed to standard out</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>{&quot;status&quot;: 0, &quot;message&quot;: &quot;Success&quot;}</td>
<td>The application was removed successfully.</td>
</tr>
<tr>
<td>1</td>
<td>{&quot;status&quot;: 1, &quot;message&quot;: &quot;Administrator privileges are required to perform this operation&quot;}</td>
<td>Administrator privileges are required to complete the operation.</td>
</tr>
<tr>
<td>1</td>
<td>{&quot;status&quot;: 1, &quot;message&quot;: &quot;App not found&quot;}</td>
<td>The application that was specified could not be found in the AppStream 2.0 application catalog.</td>
</tr>
<tr>
<td>255</td>
<td>{&quot;status&quot;: 255, &quot;message&quot;: &quot;&lt;error message&gt;&quot;}</td>
<td>An unexpected error occurred. Try the request again. If the error persists, contact AWS Support for assistance. For more information, see AWS Support Center.</td>
</tr>
</tbody>
</table>

list-applications operation

Lists all of the applications that are specified in the application catalog.

Synopsis

```
list-applications
```

Message output

<table>
<thead>
<tr>
<th>Exit code</th>
<th>Message printed to standard out</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>{&quot;status&quot;: 0, &quot;message&quot;: &quot;Success&quot;, &quot;applications&quot;: [ {..app1.. }, { ..app2.. }]}</td>
<td>List of applications in the AppStream 2.0 application catalog.</td>
</tr>
<tr>
<td>255</td>
<td>{&quot;status&quot;: 255, &quot;message&quot;: &quot;&lt;error message&gt;&quot;}</td>
<td>An unexpected error occurred. Try the request again. If the error persists, contact AWS Support for assistance. For more information, see AWS Support Center.</td>
</tr>
</tbody>
</table>

update-default-profile operation

Copies the specified Windows user’s profile to the Windows default user profile. New users who stream inherit the settings stored in the specified profile.
Amazon AppStream 2.0 Administration Guide
Image Assistant CLI Operations for Creating
and Managing Your AppStream 2.0 Image

Synopsis

update-default-profile
[--profile <value>]

Options

--profile (string)

The name of the user whose Windows profile will be copied to the Windows default user profile. Use the following format for the name:

"<domain>\<username>"

If your image builder isn’t joined to a Microsoft Active Directory domain, enter a period "." for the domain. If you don’t specify a user, the AppStream 2.0 Template User account is used.

Message output

<table>
<thead>
<tr>
<th>Exit code</th>
<th>Message printed to standard out</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>{&quot;status&quot;: 0, &quot;message&quot;: &quot;Success&quot;}</td>
<td>The user settings were successfully copied to the default Windows profile.</td>
</tr>
<tr>
<td>1</td>
<td>{&quot;status&quot;: 1, &quot;message&quot;: &quot;Administrator privileges are required to perform this operation&quot;}</td>
<td>Administrator privileges are required to complete the operation.</td>
</tr>
<tr>
<td>1</td>
<td>{&quot;status&quot;: 1, &quot;message&quot;: &quot;Unable to copy file or folder: &lt;path&gt;. &lt;reason&gt;&quot;}</td>
<td>The user settings could not be copied because a file or folder was unavailable.</td>
</tr>
<tr>
<td>1</td>
<td>{&quot;status&quot;: 1, &quot;message&quot;: &quot;Cannot copy a domain user when not joined to a domain&quot;}</td>
<td>A Microsoft Active Directory domain user was specified, but the image builder is not joined to an Active Directory domain.</td>
</tr>
<tr>
<td>255</td>
<td>{&quot;status&quot;: 255, &quot;message&quot;: &lt;error message&gt;}</td>
<td>An unexpected error occurred. Try the request again. If the error persists, contact AWS Support for assistance. For more information, see AWS Support Center.</td>
</tr>
</tbody>
</table>

reset-user-profile operation

Deletes the Windows user profile for the specified user.

Synopsis

reset-user-profile
[--profile <value>]

Options
The name of the Windows user whose Windows profile will be deleted. Use the following format for the name:

"<domain><username>"

If your image builder isn’t joined to a Microsoft Active Directory domain, enter a period “.” for the domain.

Message output

<table>
<thead>
<tr>
<th>Exit code</th>
<th>Message printed to standard out</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>{&quot;status&quot;: 0, &quot;message&quot;: &quot;Success&quot;}</td>
<td>The specified user settings were deleted successfully.</td>
</tr>
<tr>
<td>1</td>
<td>{&quot;status&quot;: 1, &quot;message&quot;: &quot;Administrator privileges are required to perform this operation&quot;}</td>
<td>Administrator privileges are required to complete the operation.</td>
</tr>
<tr>
<td>1</td>
<td>{&quot;status&quot;: 1, &quot;message&quot;: &quot;Unable to copy file or folder: &lt;path&gt;. &lt;reason&gt;&quot;}</td>
<td>The user settings could not be reset because a file or folder was unavailable.</td>
</tr>
<tr>
<td>1</td>
<td>{&quot;status&quot;: 1, &quot;message&quot;: &quot;Cannot copy a domain user when not joined to a domain&quot;}</td>
<td>A Microsoft Active Directory domain user was specified, but the image builder is not joined to an Active Directory domain.</td>
</tr>
<tr>
<td>255</td>
<td>{&quot;status&quot;: 255, &quot;message&quot;: &quot;&lt;error message&gt;&quot;}</td>
<td>An unexpected error occurred. Try the request again. If the error persists, contact AWS Support for assistance. For more information, see AWS Support Center.</td>
</tr>
</tbody>
</table>

create-image operation

Starts the image creation workflow, resulting in an AppStream 2.0 image that can be used for AppStream 2.0 fleets.

Synopsis

```
create-image
  --name <value>  
  [--description <value>] 
  [--display-name <value>] 
  [--enable-dynamic-app-catalog] | [--no-enable-dynamic-app-catalog] 
  [--use-latest-agent-version] | [--no-use-latest-agent-version] 
  [--tags <value>] 
  [--dry-run] 
```

Options

--name (string)

The name for the AppStream 2.0 image. The name must be unique within the AWS account and AWS Region. The maximum length is 100 characters. Allowed characters are:
a-z, A-Z, 0-9, and underscores (_)

The image name cannot start with any of the following prefixes: 'aws', 'appstream', and 'amazon'. These prefixes are reserved for AWS use.

--description (string)

The description to display for the image. The maximum length is 256 characters.

--display-name (string)

The name to display for the image. The maximum length is 256 characters.

--enable-dynamic-app-catalog | --no-enable-dynamic-app-catalog

Enables or disables support for the AppStream 2.0 dynamic application framework. If you don't specify either parameter, support for the dynamic application framework is not enabled.

The dynamic application framework provides operations within an AppStream 2.0 streaming instance that you can use to build a dynamic app provider. Dynamic app providers can use these operations to modify the catalog of applications that your users can access in real time. For more information, see Use the AppStream 2.0 Dynamic Application Framework to Build a Dynamic App Provider (p. 165).

--use-latest-agent-version | --no-use-latest-agent-version

Specifies whether to pin the image to the version of the AppStream 2.0 agent that is currently installed, or to always use the latest agent version. If you don't specify either parameter, the image is pinned to the version of the AppStream 2.0 agent that is currently installed. For more information, see Manage AppStream 2.0 Agent Versions (p. 36).

--tags (string)

The tags to associate with the image. A tag is a key-value pair. Use the following format:

--tags "mykey" "myval" "mykey2" "myval2"

For more information about tags, see Tagging Your Amazon AppStream 2.0 Resources (p. 201).

--dry-run (string)

Performs validation without creating the image. Use this command to identify whether your image has any issues before you create it.

Message output

<table>
<thead>
<tr>
<th>Exit code</th>
<th>Message printed to standard out</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>{&quot;status&quot;: 0, &quot;message&quot;: &quot;Success&quot;}</td>
<td>The workflow to create the image was initiated successfully.</td>
</tr>
<tr>
<td>1</td>
<td>{&quot;status&quot;: 1, &quot;message&quot;: &quot;Administrator privileges are required to perform this operation&quot;}</td>
<td>Administrator privileges are required to complete the operation.</td>
</tr>
<tr>
<td>1</td>
<td>{&quot;status&quot;: 1, &quot;message&quot;: &quot;An image with the given name already exists&quot;}</td>
<td>An image with the specified name already exists in the AWS account.</td>
</tr>
<tr>
<td>1</td>
<td>{&quot;status&quot;: 1, &quot;message&quot;: &quot;Invalid value (tags)&quot;}</td>
<td>The specified tags are not valid.</td>
</tr>
</tbody>
</table>
Use Session Scripts to Manage Your AppStream 2.0 Users' Streaming Experience

AppStream 2.0 provides on-instance session scripts. You can use these scripts to run your own custom scripts when specific events occur in users' streaming sessions. For example, you can use custom scripts to prepare your AppStream 2.0 environment before your users' streaming sessions begin. You can also use custom scripts to clean up streaming instances after users complete their streaming sessions.

Session scripts are specified within an AppStream 2.0 image. These scripts are run within the user context or the system context. If your session scripts use the standard out to write information, error, or debugging messaging, these can be optionally saved to an Amazon S3 bucket within your AWS account.

Contents
- Run Scripts Before Streaming Sessions Begin (p. 66)
- Run Scripts After Streaming Sessions End (p. 70)
- Create and Specify Session Scripts (p. 72)
- Session Scripts Configuration File (p. 72)
- Using Windows PowerShell Files (p. 74)
- Logging Session Script Output (p. 74)
- Use Storage Connectors with Session Scripts (p. 75)
- Enable Amazon S3 Bucket Storage for Session Script Logs (p. 75)

Run Scripts Before Streaming Sessions Begin

You can configure your scripts to run for a maximum of 60 seconds before your users' applications launch and their streaming sessions begin. Doing so enables you to customize the AppStream 2.0 environment before users start streaming their applications. When the session scripts run, a loading spinner displays for your users. When your scripts complete successfully or the maximum waiting time elapses, your users' streaming session will begin. If your scripts don't complete successfully, an error message displays for your users. However, your users are not prevented from using their streaming session.

When you specify a file name, you must use a double backslash. For example:

C:\\Scripts\\Myscript.bat

If you don't use a double backslash, an error displays to notify you that the .json file is incorrectly formatted.

**Note**

When your scripts complete successfully, they must return a value of 0. If your scripts return a value other than 0, AppStream 2.0 displays the error message to the user.

When you run scripts before streaming sessions begin and the AppStream 2.0 dynamic application framework is not enabled, the following process occurs:

<table>
<thead>
<tr>
<th>Exit code</th>
<th>Message printed to standard out</th>
<th>Description</th>
</tr>
</thead>
</table>
| 255       | {"status": 255, "message": "<error message>"} | An unexpected error occurred. Try the request again. If the error persists, contact AWS Support for assistance. For more information, see [AWS Support Center](https://aws.amazon.com/support/).


1. Your users connect to an AppStream 2.0 fleet instance that is not domain-joined. They connect by using one of the following access methods:
   - AppStream 2.0 user pool
Run Scripts Before Streaming Sessions Begin

- SAML 2.0
- AppStream 2.0 API

2. The application catalog displays in the AppStream 2.0 portal, and your users choose an application to launch.

3. One of the following occurs:
   - If application settings persistence is enabled for your users, the application settings Virtual Hard Disk (VHD) file that stores your users' customizations and Windows settings is downloaded and mounted. Windows user login is required in this case.
     
     For information about application settings persistence, see Enable Application Settings Persistence for Your AppStream 2.0 Users (p. 146).
   - If application settings persistence is not enabled, the Windows user is already logged in.

4. Your session scripts start. If persistent storage is enabled for your users, storage connector mounting also starts. For information about persistent storage, see Enable and Administer Persistent Storage for Your AppStream 2.0 Users (p. 136).

   
   **Note**
   
   The storage connector mount doesn't need to complete for the streaming session to start. If the session scripts complete before the storage connector mount completes, the streaming session starts.
   
   For information about monitoring the mount status of storage connectors, see Use Storage Connectors with Session Scripts (p. 75).

5. Your session scripts complete or time out.

6. The users' streaming session starts.

7. The application that your users chose launches.

For information about the AppStream 2.0 dynamic application framework, see Use the AppStream 2.0 Dynamic Application Framework to Build a Dynamic App Provider (p. 165).

When you run scripts before streaming sessions begin and the AppStream 2.0 dynamic application framework is enabled, the following process occurs:
1. Your users visit the SAML 2.0 application portal for your organization, and they choose the AppStream 2.0 stack.
2. They connect to an AppStream 2.0 fleet instance that is domain-joined.
3. If application settings persistence is enabled for your users, the application settings VHD file that stores your users' customizations and Windows settings is downloaded and mounted.

4. Windows user logon occurs.

5. The application catalog displays in the AppStream 2.0 portal and your users choose an application to launch.

6. Your session scripts start. If persistent storage is enabled for your users, storage connector mounting also starts.

   **Note**
   The storage connector mount doesn't need to complete for the streaming session to start. If the session scripts complete before the storage connector mount completes, the streaming session starts.
   For information about monitoring the mount status of storage connectors, see Use Storage Connectors with Session Scripts (p. 75).

7. Your session scripts complete or time out.

8. The users' streaming session starts.

9. The application that your users chose launches.

---

**Run Scripts After Streaming Sessions End**

You can also configure your scripts to run after users' streaming sessions end. For example, you can run a script when users select **End Session** from the AppStream 2.0 toolbar, or when they reach the maximum allowed duration for the session. You can also use these session scripts to clean up your AppStream 2.0 environment before a streaming instance is terminated. For example, you can use scripts to release file locks or upload log files. When you run scripts after streaming sessions end, the following process occurs:
1. Your users’ AppStream 2.0 streaming session ends.
2. Your session termination scripts start.
Create and Specify Session Scripts

Session scripts are configured, specified, and stored within an AppStream 2.0 image. After you create your session scripts, perform the following steps on an image builder to specify the scripts.

To configure and specify session scripts

2. In the navigation pane, choose Images, Image Builder.
3. Choose an image builder that is in the Running state, and choose Connect.
4. When prompted, choose Administrator.
5. Navigate to C:\AppStream\SessionScripts, and open the config.json configuration file.
   
   For information about session script parameters, see Session Scripts Configuration File (p. 72).
6. After you finish making your changes, save and close the config.json file.
7. On the image builder desktop, open Image Assistant.
8. Optionally, specify any other applications that you want to include in the image.
9. Follow the necessary steps in Image Assistant to finish creating your image.

   If the session scripts configuration can't be validated (for example, if the .json file is not correctly formatted), you are notified when you choose Disconnect and create image.

Session Scripts Configuration File

To locate the session scripts configuration file, navigate to C:\AppStream\SessionScripts\config.json. The file is formatted as follows.

---

**Note**
The configuration file is in .json format. Verify that any text you type in this file is in valid .json format.

```json
{
  "SessionStart": {
    "executables": [
      {
        "context": "system",
        "filename": "",
        "arguments": "",
        "s3LogEnabled": true
      },
      {
        "context": "user",
        "filename": "",
```


You can use the following parameters in the session scripts configuration file.

**SessionStart/SessionTermination**

The session scripts to run in the appropriate session event based on the name of the object.

**Type:** String  
**Required:** No  
**Allowed values:** SessionStart, SessionTermination

**WaitingTime**

The maximum duration of the session scripts in seconds.

**Type:** Integer  
**Required:** No  
**Constraints:** The maximum duration is 60 seconds. If the session scripts don't complete within this duration, they will be stopped. If you require a script to continue running, launch it as a separate process.

**Executables**

The details for the session scripts to run.

**Type:** String  
**Required:** Yes  
**Constraints:** The maximum number of scripts that can run per session event is 2 (one for the user context, one for the system context).

**Context**

The context in which to run the session script.

**Type:** String
Required: Yes

Allowed values: user, system

**Filename**

The full path to the session script to run. If this parameter is not specified, the session script is not run.

**Type:** String

**Required:** No

**Constraints:** The maximum length for the file name and full path is 1,000 characters.

**Allowed values:** .bat, .exe

**Note**

You can also use Windows PowerShell files. For more information, see Using Windows PowerShell Files (p. 74).

**Arguments**

The arguments for your session script or executable file.

**Type:** String

**Required:** No

**Length constraints:** The maximum length is 1,000 characters.

**S3LogEnabled**

When the value for this parameter is set to True, an S3 bucket is created within your AWS account to store the logs created by the session script. By default, this value is set to True. For more information, see the Logging Session Script Output section later in this topic.

**Type:** Boolean

**Required:** No

**Allowed values:** True, False

---

**Using Windows PowerShell Files**

To use Windows PowerShell files, specify the full path to the PowerShell file in the **filename** parameter:

```
"filename": 
"C:\\Windows\\System32\\WindowsPowerShell\\v1.0\\powershell.exe",
```

Then specify your session script in the **arguments** parameter:

```
"arguments": "-File "C:\path\to\session\script.ps1"
```

Finally, verify that the PowerShell Execution Policy allows your PowerShell file to run.

**Logging Session Script Output**

When this option is enabled in the configuration file, AppStream 2.0 automatically captures the output from the session script that is written to the standard out. This output is uploaded to an Amazon S3 bucket in your account. You can review the log files for troubleshooting or debugging purposes.
Use Storage Connectors with Session Scripts

When AppStream 2.0 storage connectors are enabled, they begin mounting when the session start scripts run. If your script relies on the storage connectors being mounted, you can wait for the connectors to be available. AppStream 2.0 maintains the mount status of the storage connectors in the Windows registry, at the following key:

HKEY_LOCAL_MACHINE\SOFTWARE\Amazon\AppStream\Storage\<provided user name>\<Storage connector>

The registry key values are as follows:

- **Provided user name** — The user ID provided through the access mode. The access modes and value for each mode are as follows:
  - **User Pool** — The email address for the user
  - **Streaming URL** — The UserID
  - **SAML** — The NameID. If the user name includes a slash (for example, a domain user's SAMAccountName), the slash is replaced by a "-" character.
- **Storage connector** — The connector for the persistent storage option that is enabled for the user. The storage connector values are as follows:
  - **HomeFolder**
  - **GoogleDrive**
  - **OneDrive**

Each storage connector registry key contains a **MountStatus** DWORD value. The following table lists the possible values for **MountStatus**.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Storage connector not be enabled for this user</td>
</tr>
<tr>
<td>1</td>
<td>Storage connector mounting is in progress</td>
</tr>
<tr>
<td>2</td>
<td>Storage connector mounted successfully</td>
</tr>
<tr>
<td>3</td>
<td>Storage connector mounting failed</td>
</tr>
<tr>
<td>4</td>
<td>Storage connector mounting is enabled, but not mounted yet</td>
</tr>
</tbody>
</table>

Enable Amazon S3 Bucket Storage for Session Script Logs

When you enable Amazon S3 logging in your session script configuration, AppStream 2.0 captures standard output from your session script. The output is periodically uploaded to an S3 bucket within
Enable Amazon S3 Bucket Storage for Session Script Logs

Your AWS account. For every AWS Region, AppStream 2.0 creates a bucket in your account that is unique to your account and the Region.

You do not need to perform any configuration tasks to manage these S3 buckets. They are fully managed by the AppStream 2.0 service. The log files that are stored in each bucket are encrypted in transit using Amazon S3’s SSL endpoints and at rest using Amazon S3-managed encryption keys. The buckets are named in a specific format as follows:

<table>
<thead>
<tr>
<th>bucket-name</th>
<th>stack-name</th>
<th>fleet-name</th>
<th>access-mode</th>
<th>user-id-SHA-256-hash</th>
<th>session-id</th>
<th>SessionScriptsLogs/session-event</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>region-code</th>
<th>This is the AWS Region code in which the stack is created with Amazon S3 bucket storage enabled for session script logs.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>account-id-without-hyphens</th>
<th>Your AWS account identifier. The random ID ensures that there is no conflict with other buckets in that Region. The first part of the bucket name, appstream-logs, does not change across accounts or Regions.</th>
</tr>
</thead>
</table>

For example, if you specify session scripts in an image in the US West (Oregon) Region (us-west-2) on account number 123456789012, AppStream 2.0 creates an Amazon S3 bucket within your account in that Region with the name shown. Only an administrator with sufficient permissions can delete this bucket.

bucket-name/us-west-2-1234567890123-abcdefg

Disabling session scripts does not delete any log files stored in the S3 bucket. To permanently delete log files, you or another administrator with adequate permissions must do so by using the Amazon S3 console or API. AppStream 2.0 adds a bucket policy that prevents accidental deletion of the bucket.

For more information, see IAM Policies and the Amazon S3 Bucket for Application Settings Persistence in Identity and Access Management for Amazon AppStream 2.0 (p. 224).

When session scripts are enabled, a unique folder is created for each streaming session that is started.

The path for the folder where the log files are stored in the S3 bucket in your account uses the following structure:

bucket-name/stack-name/fleet-name/access-mode/user-id-SHA-256-hash/session-id/SessionScriptsLogs/session-event

<table>
<thead>
<tr>
<th>bucket-name</th>
<th>The name of the S3 bucket in which the session scripts are stored. The name format is described earlier in this section.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>stack-name</th>
<th>The name of the stack the session came from.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>fleet-name</th>
<th>The name of the fleet the session script is running on.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>access-mode</th>
<th>The identity method of the user: custom for the AppStream 2.0 API or CLI, federated for SAML, and userpool for users in the user pool.</th>
</tr>
</thead>
</table>
Enable Amazon S3 Bucket Storage for Session Script Logs

**user-id-SHA-256-hash**

The user-specific folder name. This name is created using a lowercase SHA-256 hash hexadecimal string generated from the user identifier.

**session-id**

The identifier of the user's streaming session. Each user streaming session generates a unique ID.

**session-event**

The event that generated the session script log. The event values are: `SessionStart` and `SessionTermination`.

The following example folder structure applies to a streaming session started from the test-stack and test-fleet. The session uses the API of user ID `testuser@mydomain.com`, from an AWS account ID of `123456789012`, and the settings group `test-stack` in the US West (Oregon) Region (us-west-2):

```
appstream-logs-us-west-2-1234567890123-abcdefg/test-stack/test-fleet/custom/
77e740fab628a1cd13/05yd1391-4805-3da6-f498-76f5x6746016/SessionScriptsLogs/SessionStart/
```

This example folder structure contains one log file for a user context session start script, and one log file for a system context session start script, if applicable.
Fleets and Stacks

With Amazon AppStream 2.0, you create fleet instances and stacks as part of the process of streaming applications. A fleet consists of streaming instances that run the image that you specify. A stack consists of an associated fleet, user access policies, and storage configurations.

Contents
- Fleet Type (p. 78)
- Session Context (p. 78)
- AppStream 2.0 Instance Families (p. 80)
- Create an AppStream 2.0 Fleet and Stack (p. 81)
- Customize an AppStream 2.0 Fleet to Optimize Your Users' Application Streaming Experience (p. 86)
- Update an AppStream 2.0 Fleet with a New Image (p. 96)
- Fleet Auto Scaling for Amazon AppStream 2.0 (p. 97)

Fleet Type

The fleet type determines when your instances run and how you pay for them. You can specify a fleet type when you create a fleet. You cannot change the fleet type after you create the fleet.

The following are the possible fleet types:

Always-On
Instances run all the time, even when no users are streaming applications.

On-Demand
Instances run only when users are streaming applications. Idle instances that are available for streaming are in a stopped state.

Use an Always-On fleet to provide your users with instant access to their applications. Use an On-Demand fleet to optimize your streaming charges and provide your users with access to their applications after a 1-2 minute wait. For more information, see Amazon AppStream 2.0 Pricing.

To create an On-Demand fleet, you must use a base image starting with 09-05-2017.

Session Context

You can pass parameters to your streaming application by using the create-streaming-url AppStream 2.0 API operation. If your image uses a version of the AppStream 2.0 agent that was released on or after October 30, 2018, the session context can be accessed within the image as a Windows environment variable. For information about specific environment variables, see "User and Instance Metadata for AppStream 2.0 Fleets" in Customize an AppStream 2.0 Fleet to Optimize Your Users' Application Streaming Experience (p. 86).
Using Session Context to Pass Parameters to a Streaming Application

Perform the following steps to pass parameters to your streaming application. The example uses session context to launch a specific website using Firefox.

To use session context to launch a website

1. In the left navigation pane, choose Images, Image Builder.
2. Choose the image builder to use, verify that it is in the Running state, and choose Connect.
3. Log in to the image builder by choosing Administrator on the Local User tab.
4. Create a child folder of C:\. For this example, use C:\Scripts.
5. Create a Windows batch file in the new folder. For this example, create C:\Scripts\session-context-test.bat and add a script that launches Firefox with the URL from the session context.

Do one of the following, based on the version of the AppStream 2.0 agent that your image uses (when the AppStream 2.0 agent was released):

- On or after October 30, 2018 — Use the following script, then proceed to step 6.
  ```cmd
  CD "C:\Program Files (x86)\Mozilla Firefox"
  Start firefox.exe %APPSTREAM_SESSION_CONTEXT%
  ```
- On or after September 5, 2017 and before October 30, 2018 — Follow the instructions in Using Session Context with AppStream 2.0 Agent Versions Released On or After September 5, 2017 and Before October 30, 2018 (p. 79), then proceed to step 6.
- Before September 5, 2017 — Follow the instructions in Using Session Context with AppStream 2.0 Agent Versions Released Before September 5, 2017 (p. 80), then proceed to step 6.

6. In Image Assistant, add session-context-test.bat and change the name to Firefox.

You do not need to add Firefox. This step requires that you add only the batch file.

7. Create an image, fleet, and stack. For this example, use a fleet name of session-context-test-fleet and a stack name of session-context-test-stack.

8. After the fleet is running, you can call create-streaming-url with the session-context parameter, as shown in this example.

   ```shell
   aws appstream create-streaming-url --stack-name session-context-test-stack \
   --fleet-name session-context-test-fleet \
   --user-id username --validity 10000 \
   --application-id firefox --session-context "www.amazon.com"
   ```


Using Session Context with AppStream 2.0 Agent Versions Released On or After September 5, 2017 and Before October 30, 2018

If your image uses a version of the AppStream 2.0 agent version that was released on or after September 5, 2017 and before October 30, 2018, the APPSTREAM_SESSION_CONTEXT variable is accessible only through .NET. To access the environment variable, download the SessionContextRetriever.exe
file from https://dsfpe42xwhi2e.cloudfront.net/SessionContextRetriever.exe and copy this .exe file to C:\Scripts. Then, create the following script as part of completing step 5 in Using Session Context to Pass Parameters to a Streaming Application (p. 79).

```bash
for /f "tokens=*" %f in ('C:\Scripts\SessionContextRetriever.exe')
do (set var=%f)
CD "C:\Program Files (x86)\Mozilla Firefox"
start firefox.exe %var%
```

When you're done, proceed to step 6.

**Using Session Context with AppStream 2.0 Agent Versions Released Before September 5, 2017**

If your image uses a version of the AppStream 2.0 agent that was released before September 5, 2017, use the following script for step 5 in Using Session Context to Pass Parameters to a Streaming Application (p. 79), then proceed to step 6.

```bash
CD "C:\Program Files (x86)\Mozilla Firefox"
start firefox.exe %1
```

Parameters are passed to the streaming application.

**AppStream 2.0 Instance Families**

Amazon AppStream 2.0 users stream applications from stacks that you create. Each stack is associated with a fleet. When you create a fleet, the instance type that you specify determines the hardware of the host computers used for your fleet. Each instance type offers different compute, memory, and GPU capabilities. Instance types are grouped into *instance families* based on these capabilities. For hardware specifications and pricing information, see AppStream 2.0 Pricing.

When you create a fleet or image builder, you must select an image that is compatible with the instance family on which you intend to run your fleet.

- When launching a new image builder, you are presented with a list of the images in your image registry. Select the appropriate base image.
- When launching a fleet, ensure that the private image you select was created from the appropriate base image.

The following table summarizes the available instance families and provides the base image naming format for each. Select an instance type from an instance family based on the requirements of the applications that you plan to stream on your fleet, and match the base image according to the following table.

<table>
<thead>
<tr>
<th>Instance Family</th>
<th>Description</th>
<th>Base Image Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>General Purpose</td>
<td>Basic computing resources for running web browsers and most business applications.</td>
<td>AppStream-WinServer-OperatingSystemVersion-MM-DD-YYYY</td>
</tr>
<tr>
<td>Instance Family</td>
<td>Description</td>
<td>Base Image Name</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Memory Optimized</td>
<td>Optimized for memory-intensive applications that process large amounts of data.</td>
<td>AppStream-WinServer-OperatingSystemVersion-MM-DD-YYYY</td>
</tr>
<tr>
<td>Note</td>
<td>If you plan to use AppStream 2.0 z1d-based instances, you must provision them from images that were created from AppStream 2.0 base images published on or after June 12, 2018.</td>
<td></td>
</tr>
<tr>
<td>Graphics Desktop</td>
<td>Uses NVIDIA GRID K520 GPU to support applications that benefit from or require graphics acceleration. This instance family supports DirectX, OpenGL, OpenCL, and CUDA.</td>
<td>Graphics-Desktop-Image-Builder-MM-DD-YYYY</td>
</tr>
<tr>
<td></td>
<td>This instance family is deprecated and therefore no longer available.</td>
<td></td>
</tr>
<tr>
<td>Graphics Pro</td>
<td>Uses NVIDIA Tesla M60 GPUs and provide a high-performance, workstation-like experience for graphics applications that use DirectX, OpenGL, OpenCL, or CUDA.</td>
<td>AppStream-Graphics-Pro-OperatingSystemVersion-MM-DD-YYYY</td>
</tr>
<tr>
<td>Graphics Design</td>
<td>Uses AMD FirePro S7150x2 Server GPUs and AMD Multiuser GPU technology to support graphics applications that use DirectX, OpenGL, or OpenCL.</td>
<td>AppStream-Graphics-Design-OperatingSystemVersion-MM-DD-YYYY</td>
</tr>
</tbody>
</table>

For more information, see the following:

- AppStream 2.0 Base Image Version History (p. 29)
- Amazon AppStream 2.0 Service Quotas (p. 271)
- AppStream 2.0 Pricing

---

Create an AppStream 2.0 Fleet and Stack

To stream your applications, Amazon AppStream 2.0 requires an environment that includes a fleet that is associated with a stack, and at least one application image. This tutorial describes the steps to set up a fleet and stack, and how to give users access to the stack. If you haven’t already done so, we recommend that you try the procedures in Get Started with Amazon AppStream 2.0: Set Up With Sample Applications (p. 5) first.
If you want to create an image to use, see Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47).

If you plan to join a fleet to an Active Directory domain, configure your Active Directory domain before completing the following steps. For more information, see Using Active Directory with AppStream 2.0 (p. 106).

**Tasks**
- Create a Fleet (p. 82)
- Create a Stack (p. 84)
- Provide Access to Users (p. 86)
- Clean Up Resources (p. 86)

## Create a Fleet

Set up and create a fleet from which user applications are launched and streamed.

**To set up and create a fleet**

2. Choose Get Started if you are new to the console, or Fleets from the left navigation pane. Choose Create Fleet.
3. For Step 1: Provide Fleet Details, type a unique name identifier for the fleet. Optionally, you can also specify the following:
   - **Display Name** — Type a name to display for the fleet (maximum of 100 characters).
   - **Description** — Type a description for the fleet (maximum of 256 characters).
   - **Tags** — Choose Add Tag, and type the key and value for the tag. To add more tags, repeat this step. For more information, see Tagging Your Amazon AppStream 2.0 Resources (p. 201).
4. Choose Next.
5. For Step 2: Choose an Image, choose an image that meets your needs and then choose Next.
6. For Step 3: Configure Fleet, do the following:
   - For **Choose instance type**, choose the instance type that meets the performance requirements of your applications.
   - For **Fleet type**, choose the fleet type that suits your use case. The fleet type determines its immediate availability and how you pay for it.
   - For **Maximum session duration in minutes** — Choose the maximum amount of time that a streaming session can remain active. If users are still connected to a streaming instance five minutes before this limit is reached, they are prompted to save any open documents before being disconnected. After this time elapses, the instance is terminated and replaced by a new instance. The maximum session duration that you can set in the AppStream 2.0 console is 5760 minutes (96 hours).
   - For **Disconnect timeout in minutes**, choose the amount of time that a streaming session remains active after users disconnect. If users try to reconnect to the streaming session after a disconnection or network interruption within this time interval, they are connected to their previous session. Otherwise, they are connected to a new session with a new streaming instance. If you associate a stack with a fleet for which a redirect URL is specified, after users' streaming sessions end, the users are redirected to that URL.

   If a user ends the session by choosing **End Session** on the streaming session toolbar, the disconnect timeout does not apply. Instead, the user is prompted to save any open documents,
and then immediately disconnected from the streaming instance. The instance the user was using is then terminated.

- For **idle disconnect timeout in minutes**, choose the amount of time that users can be idle (inactive) before they are disconnected from their streaming session and the **disconnect timeout in minutes** time interval begins. Users are notified before they are disconnected due to inactivity. If they try to reconnect to the streaming session before the time interval specified in **disconnect timeout in minutes** has elapsed, they are connected to their previous session. Otherwise, they are connected to a new session with a new streaming instance. Setting this value to 0 disables it. When this value is disabled, users are not disconnected due to inactivity.

  **Note**
  Users are considered idle when they stop providing keyboard or mouse input during their streaming session. File uploads and downloads, audio in, audio out, and pixels changing do not qualify as user activity. If users continue to be idle after the time interval in **idle disconnect timeout in minutes** elapses, they are disconnected.

- For **Minimum capacity**, choose a minimum number of instances for your fleet based on the minimum number of expected concurrent users.

- For **Maximum capacity**, choose a maximum number of instances for your fleet based on the maximum number of expected concurrent users.

- **IAM role (Advanced)**: When you apply an IAM role from your account to an AppStream 2.0 fleet instance, you can make AWS API requests from the fleet instance without manually managing AWS credentials. To apply an IAM role, do either of the following:
  - To use an existing IAM role in your AWS account, choose the role that you want to use from the IAM role list. The role must be accessible from the fleet instance. For more information, see Configuring an Existing IAM Role to Use With AppStream 2.0 Streaming Instances (p. 232).
  - To create a new IAM role, choose **Create new IAM role** and follow the steps in How to Create an IAM Role to Use With AppStream 2.0 Streaming Instances (p. 233).

- For **Scaling details**, specify the scaling policies that AppStream 2.0 uses to increase and decrease the capacity of your fleet. Note that the size of your fleet is limited by the minimum and maximum capacity that you specified. For more information, see Fleet Auto Scaling for Amazon AppStream 2.0 (p. 97).

7. For **Step 4: Configure Network**, do the following:

- To add internet access for fleet instances in a VPC with a public subnet, choose **Default Internet Access**. If you are providing internet access by using a NAT gateway, leave **Default Internet Access** unselected. For more information, see Internet Access (p. 10).

- For **VPC and Subnet 1**, choose a VPC and at least one subnet that has access to the network resources that your application needs. For increased fault tolerance, we recommend that you choose two subnets in different Availability Zones. For more information, see Configure a VPC with Private Subnets and a NAT Gateway (p. 12).

If you don't have your own VPC and subnet, you can use the default VPC (p. 20) or create your own. To create your own, choose the **Create a new VPC** and **Create new subnet** links to create them. Choosing these links opens the Amazon VPC console. After you create your VPC and subnets, return to the AppStream 2.0 console and choose the refresh icon to the left of the **Create a new VPC** and **Create new subnet** links to display them in the list. For more information, see Configure a VPC for AppStream 2.0 (p. 11).

- For **Security group(s)**, choose up to five security groups to associate with this fleet. If you don’t have your own security group and you don’t want to use the default security group, choose the **Create new security group** link to create one. After you create your subnets in the Amazon VPC console, return to the AppStream 2.0 console and choose the refresh icon to the left of the **Create new security group** link to display them in the list. For more information, see Security Groups in Amazon AppStream 2.0 (p. 253).

- For **Active Directory Domain (Optional)**, choose the Active Directory and organizational unit (OU) for your streaming instance computer objects. Ensure that the network access settings you
selected enable DNS resolvability and communication with your directory. For more information, see Using Active Directory with AppStream 2.0 (p. 106).

8. Choose Create.

While your fleet is being created and fleet instances are provisioned, the status of your fleets displays as Starting in the Fleets list. Choose the Refresh icon periodically to update the fleet status until the status is Running. You cannot associate the fleet with a stack and use it for streaming sessions until the status of the fleet is Running.

Create a Stack

Set up and create a stack to control access to your fleet.

To set up and create a stack

1. In the left navigation pane, choose Stacks, and then choose Create Stack.
2. For Step 1: Stack Details, Under Stack details, type a unique name identifier for the stack. Optionally, you can specify the following:
   - Display name — Type a name to display for the stack (maximum of 100 characters).
   - Description — Type a description for the stack (maximum of 256 characters).
   - Redirect URL — Specify a URL to which users are redirected after their streaming sessions end.
   - Feedback URL — Specify a URL to which users are redirected after they click the Send Feedback link to submit feedback about their application streaming experience. If you do not specify a URL, this link is not displayed.
   - Fleet — Select an existing fleet or create a new one to associate with your stack.
   - Tags — Choose Add Tag, and type the key and value for the tag. To add more tags, repeat this step. For more information, see Tagging Your Amazon AppStream 2.0 Resources (p. 201).
3. For Step 1: Stack Details, under Network Access Endpoints (Optional), you can create a private link, which is an interface VPC endpoint (interface endpoint), in your virtual private cloud (VPC). To start creating the interface endpoint, select Create PrivateLink. Selecting this link opens the VPC console. To finish creating the endpoint, follow steps 3 through 6 in To create an interface endpoint, in Creating and Streaming from Interface VPC Endpoints (p. 250).

After you create the interface endpoint, you can use it to keep streaming traffic within your VPC.

4. Choose Next.
5. For Step 2: Enable Storage, you can provide persistent storage for your users by choosing one or more of the following:
   - Home Folders — Users can save their files to their home folder and access existing files in their home folder during application streaming sessions. For information about requirements for enabling home folders, see Enable Home Folders for Your AppStream 2.0 Users (p. 137).
   - Google Drive for G Suite — Users can link their Google Drive for G Suite account to AppStream 2.0. During application streaming sessions, they can sign in to their Google Drive account, save files to Google Drive, and access their existing files in Google Drive. You can enable Google Drive for accounts in G Suite domains only, not for personal Gmail accounts.

   Note
   After you select Enable Google Drive, type the name of at least one organizational domain that is associated with your G Suite account. Access to Google Drive during application streaming sessions is limited to user accounts that are in the domains that you specify. You can specify up to 10 domains. For more information about requirements for enabling Google Drive, see Enable Google Drive for Your AppStream 2.0 Users (p. 141).
• **OneDrive for Business** — Users can link their OneDrive for Business account to AppStream 2.0. During application streaming sessions, they can sign in to their OneDrive account, save files to OneDrive, and access their existing files in OneDrive. You can enable OneDrive for accounts in OneDrive domains only, not for personal accounts.

  **Note**
  After you select Enable OneDrive, type the name of at least one organizational domain that is associated with your OneDrive account. Access to Google Drive during application streaming sessions is limited to user accounts that are in the domains that you specify. You can specify up to 10 domains. For more information about requirements for enabling OneDrive, see Enable OneDrive for Your AppStream 2.0 Users (p. 143).

6. Choose Next.

7. For **Step 3: User Settings**, select the ways in which your users can transfer data between their streaming session and their local device. Then, choose whether to enable application settings persistence. When you're done, choose Review.

**Clipboard, file transfer, and print to local device permissions options:**

• **Clipboard** — By default, users can copy and paste data between their local device and streaming applications. You can limit Clipboard options so that users can paste data to their remote streaming session only or copy data to their local device only. You can also disable Clipboard options entirely. Users can still copy and paste between applications in their streaming session.

• **File transfer** — By default, users can upload and download files between their local device and streaming session. You can limit file transfer options so that users can upload files to their streaming session only or download files to their local device only. You can also disable file transfer entirely.

  **Important**
  If your users require AppStream 2.0 file system redirection to access local drives and folders during their streaming sessions, you must enable both file upload and download. To use file system redirection, your users must have AppStream 2.0 client version 1.0.480 or later installed. For more information, see Enable File System Redirection for Your AppStream 2.0 Users (p. 187).

• **Print to local device** — By default, users can print to their local device from within a streaming application. When they choose Print in the application, they can download a .pdf file that they can print to a local printer. You can disable this option to prevent users from printing to a local device.

  **Note**
  These settings affect only whether users can use AppStream 2.0 data transfer features. If your image provides access to a browser, network printer, or other remote resource, your users might be able to transfer data to or from their streaming session in other ways.

**Application settings persistence options:**

• **Enable Application Settings Persistence** — Users' application customizations and Windows settings are automatically saved after each streaming session and applied during the next session. These settings are saved to an Amazon Simple Storage Service (Amazon S3) bucket in your account, within the AWS Region in which application settings persistence is enabled.

• **Settings Group** — The settings group determines which saved application settings are used for a streaming session from this stack. If the same settings group is applied to another stack, both stacks use the same application settings. By default, the settings group value is the name of the stack.
Note
For information about requirements for enabling and administering application settings persistence, see Enable Application Settings Persistence for Your AppStream 2.0 Users (p. 146).

8. For Step 4: Review, confirm the details for the stack. To change the configuration for any section, choose Edit and make the needed changes. After you finish reviewing the configuration details, choose Create.

After the service sets up resources, the Stacks page appears. The status of your new stack appears as Active when it is ready to use.

Provide Access to Users

After you create a stack with an associated fleet, you can provide access to users through the AppStream 2.0 user pool, SAML 2.0 [single sign-on (SSO)], or the AppStream 2.0 API. For more information, see User Pool Administration (p. 240) and Single Sign-on Access (SAML 2.0) (p. 243).

Note
Users in the AppStream 2.0 user pool can't be assigned to stacks with fleets that are joined to an Active Directory domain.

After you provide your users with access to AppStream 2.0, they can start AppStream 2.0 streaming sessions by using a web browser or by using the AppStream 2.0 client application for a supported device. If you provide access to users through the AppStream 2.0 user pool, they must use a web browser for streaming sessions. If you use SAML 2.0 or the AppStream 2.0 API, you can make the AppStream 2.0 client available to them. The AppStream 2.0 client is a native application that is designed for users who require additional functionality during their AppStream 2.0 streaming sessions. For more information, see Provide Access Through the AppStream 2.0 Client for Windows (p. 174).

Clean Up Resources

You can stop your running fleet and delete your active stack to free up resources and to avoid unintended charges to your account. We recommend stopping any unused, running fleets.

Note that you cannot delete a stack with an associated fleet.

To clean up your resources

1. In the navigation pane, choose Stacks.
2. Select the stack and choose Actions, Disassociate Fleet.
3. From Stack Details, open the Associated Fleet link to select the fleet.
4. Choose Actions, Stop. It takes about 5 minutes to stop a fleet.
5. When the status of the fleet is Stopped, choose Actions, Delete.
6. In the navigation pane, choose Stacks.
7. Select the stack and choose Actions, Delete.

Customize an AppStream 2.0 Fleet to Optimize Your Users' Application Streaming Experience

By customizing AppStream 2.0 fleet instances, you can define specific aspects of your AppStream 2.0 environment to optimize your users' application streaming experience. For example, you can persist
Persist Environment Variables

Environment variables enable you to dynamically pass settings across applications. For example, many engineering applications rely on environment variables to specify the IP address or host name of a license server to locate and check out a license from that server.

Follow the steps in these procedures to make environment variables available across your fleet instances.

Contents
- Change System Environment Variables (p. 87)
- Change User Environment Variables (p. 88)
- Create an Environment Variable That is Limited in Scope (p. 89)

Note
If you are using Active Directory and Group Policy with AppStream 2.0, keep in mind that streaming instances must be joined to an Active Directory domain to use Group Policy for environment variables. For information about how to configure the Group Policy Environment Variable preference item, see Configure an Environment Variable Item in the Microsoft documentation.

Change System Environment Variables

Follow these steps to change system environment variables across your fleet instances.

To change system environment variables on an image builder

This procedure applies only to system environment variables, not user environment variables. To change user environment variables that persist across your fleet instances, follow the steps in the next procedure.

2. In the left navigation pane, choose **Images, Image Builder**.
3. Choose the image builder on which to change system environment variables, verify that it is in the **Running** state, and choose **Connect**.
4. Log in to the image builder by doing either of the following:
   - If your image builder is not joined to an Active Directory domain, on the **Local User** tab, choose **Administrator**.
   - If your image builder is joined to an Active Directory domain, choose the **Directory User** tab, specify the credentials for a domain user account that has local administrator permissions on the image builder, then choose **Log in**.
5. Choose the Windows **Start** button, open the context (right-click) menu for **Computer**, and then choose **Properties**.
6. In the navigation pane, choose **Advanced system settings**.
7. In **System variables**, change the environment variables that you want to persist across your fleet instances, and then choose **OK**.
8. On the image builder desktop, open Image Assistant.
9. Follow the necessary steps in Image Assistant to finish creating your image. For more information, see Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47).

   The changes to the system environment variables persist across your fleet instances and are available to streaming sessions launched from those instances.

   **Note**
   Setting AWS CLI credentials as system environment variables might prevent AppStream 2.0 from creating the image.

### Change User Environment Variables

Follow these steps to change user environment variables across your fleet instances.

**To change user environment variables**

2. In the left navigation pane, choose **Images, Image Builder**.
3. Choose the image builder on which to change user environment variables, verify that it is in the **Running** state, and choose **Connect**.
4. On the **Local User** tab, choose **Template User**.
5. Choose the Windows **Start** button, **Control Panel, User Accounts**.
6. Choose **User Accounts** again. In the left navigation pane, choose **Change my environment variables**.
7. Under **User environment variables** for **DefaultProfileUser**, set or create the user environment variables as needed, then choose **OK**.
8. This disconnects your current session and opens the login menu. Log in to the image builder by doing either of the following:
   - If your image builder is not joined to an Active Directory domain, on the **Local User** tab, choose **Administrator**.
   - If your image builder is joined to an Active Directory domain, choose the **Directory User** tab, and log in as a domain user who has local administrator permissions on the image builder.
Create an Environment Variable That is Limited in Scope

Follow these steps to create an environment variable that is limited in scope to the processes that are spawned off the script. This approach is useful when you need to use the same environment variable name with different values for different applications. For example, if you have two different applications that use the environment variable "LIC_SERVER", but each application has a different value for "LIC_SERVER".

To create an environment variable that is limited in scope

2. In the left navigation pane, choose Images, Image Builder.
3. Choose the image builder on which to create an environment variable that is limited in scope, verify that it is in the Running state, and choose Connect.
4. Log in to the image builder by doing either of the following:
   - If your image builder is not joined to an Active Directory domain, on the Local User tab, choose Administrator.
   - If your image builder is joined to an Active Directory domain, choose the Directory User tab, specify the credentials for a domain user account that has local administrator permissions on the image builder, then choose Log in.
5. Create a child folder of C:\ drive for the script (for example, C:\Scripts).
6. Open Notepad to create the new script, and enter the following lines:

   ```
   set variable=value
   start " " "C:\path\to\application.exe"
   ```

   Where:

   - `variable` is the variable name to be used
   - `value` is the value for the given variable name

   **Note**
   If the application path includes spaces, the entire string must be encapsulated within quotation marks. For example:
   ```
   start " " "C:\Program Files\application.exe"
   ```

7. Choose File, Save. Name the file and save it with the .bat extension to C:\Scripts. For example, name the file LaunchApp.bat.
8. If needed, repeat steps 4 and 5 to create a script for each additional application that requires its own environment variable and values.
9. On the image builder desktop, start Image Assistant.
10. Choose Add App, navigate to C:\Scripts, and select one of the scripts that you created in step 5. Choose Open.
11. In the App Launch Settings dialog box, keep or change the settings as needed. When you're done, choose Save.
12. If you created multiple scripts, repeat steps 8 and 9 for each script.
13. Follow the necessary steps in Image Assistant to finish creating your image. For more information, see Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47).
Set Default File Associations for Your Users

The associations for application file extensions are set on a per-user basis and so are not automatically applied to all users who launch AppStream 2.0 streaming sessions. For example, if you set Adobe Reader as the default application for .pdf files on your image builder, this change is not applied to your users.

**Note**
The following steps must be performed on an image builder that is joined to an Active Directory domain. In addition, your fleet must be joined to an Active Directory domain. Otherwise, the default file associations that you set are not applied.

To set default file associations for your users

2. Choose the image builder on which to set default file associations, verify that it is in the Running state, and choose Connect.
3. Choose the Directory User tab, specify the credentials for a domain user account that has local administrator permissions on the image builder, then choose Log in.
4. Set default file associations as needed.
5. Open the Windows command prompt as an administrator.
6. At the command prompt, type the following command to export the image builder file associations as an XML file, and then press ENTER:
   
   `dism.exe /online /export-DefaultAppAssociations:c:\default_associations.xml`

   If you receive an error message stating that you cannot service a running 64-bit operating system with a 32-bit version of DISM, close the command prompt window. Open File Explorer, browse to C:\Windows\System32, right-click cmd.exe, choose Run as Administrator, and run the command again.
7. You can use either Local Group Policy Editor or the GPMC to set a default associations configuration file:
   
   - Local Group Policy Editor:
     
     On your image builder, open the command prompt as an administrator, type `gpedit.msc`, and then press ENTER.
     
     In the console tree, under Computer Configuration, expand Administrative Templates, Windows Components, and then choose File Explorer.
   
   - GPMC:
     
     In your directory or on a domain controller, open the command prompt as an administrator, type `gpmc.msc`, and then press ENTER.
     
     In the left console tree, select the OU in which you want to create a new GPO, or use an existing GPO, and then do either of the following:
     
     - Create a new GPO by opening the context (right-click) menu and choosing Create a GPO in this domain, Link it here. For Name, provide a descriptive name for this GPO.
     - Select an existing GPO.
     
     Open the context menu for the GPO, and choose Edit.
Under **User Configuration**, expand **Policies, Administrative Templates, Windows Components**, and then choose **File Explorer**.

8. Double-click **Set a default associations configuration file**.

9. In the **Set a default associations configuration file properties** dialog box, choose **Enabled**, and do one of the following:
   - If you are using Local Group Policy Editor, enter this path: `c:\default_associations.xml`.
   - If you are using the GPMC, enter a network path. For example, `\networkshare\default_associations.xml`.

10. Choose **Apply, OK**.
11. Close Local Group Policy Editor or the GPMC.
12. On the image builder desktop, open Image Assistant.
13. Follow the necessary steps in Image Assistant to finish creating your image. For more information, see **Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console** (p. 47).

The file associations that you configured are applied to the fleet instances and user streaming sessions that are launched from those instances.

**Disable Internet Explorer Enhanced Security Configuration**

Internet Explorer Enhanced Security Configuration (ESC) places servers and Internet Explorer in a configuration that limits exposure to the internet. However, this configuration can impact the AppStream 2.0 end user experience. Users who are connected to AppStream 2.0 streaming sessions may find that websites do not display or perform as expected when:

- Internet Explorer ESC is enabled on fleet instances from which users’ streaming sessions are launched
- Users run Internet Explorer during their streaming sessions
- Applications use Internet Explorer to load data

**To disable Internet Explorer Enhanced Security Configuration**

2. In the left navigation pane, choose **Images, Image Builder**.
3. Choose the image builder on which to disable Internet Explorer ESC, verify that it is in the **Running** state, and choose **Connect**.
4. Log in to the image builder by doing either of the following:
   - If your image builder is not joined to an Active Directory domain, on the **Local User** tab, choose **Administrator**.
   - If your image builder is joined to an Active Directory domain, choose the **Directory User** tab, specify the credentials for a domain user account that has local administrator permissions on the image builder, then choose **Log in**.
5. On the image builder, disable Internet Explorer ESC by doing the following:
   a. Open Server Manager. Choose the Windows **Start** button, and then choose **Server Manager**.
   b. In the left navigation pane, choose **Local Server**.
   c. In the right properties pane, choose the **On** link next to IE Enhanced Security Configuration.
   d. In the **Internet Explorer Enhanced Configuration** dialog box, choose the **Off** option under **Administrators** and **Users**, then choose **OK**.
6. In the upper right area of the image builder desktop, choose Admin Commands, Switch User.

7. This disconnects your current session and opens the login menu. Log in to the image builder by doing either of the following:

   • If your image builder is not joined to an Active Directory domain, on the Local User tab, choose Template User.
   • If your image builder is joined to an Active Directory domain, choose the Directory User tab, and log in as a domain user who does not have local administrator permissions on the image builder.

8. Open Internet Explorer and reset your settings by doing the following:

   a. In the upper right area of the Internet Explorer browser window, choose the Tools icon, then choose Internet options.
   b. Choose the Advanced tab, then choose Reset.
   c. When prompted to confirm your choice, choose Reset again.
   d. When the Reset Internet Explorer Settings message displays, choose Close.

9. Choose Admin Commands, Switch User, and then do either of the following:

   • If your image builder is not joined to an Active Directory domain, on the Local User tab, choose Administrator.
   • If your image builder is joined to an Active Directory domain, choose the Directory User tab, and log in with the same domain user account that you used in step 4.

10. On the image builder desktop, open Image Assistant.

11. In Step 2. Configure Apps, choose Save settings.

12. Follow the necessary steps in Image Assistant to finish creating your image. For more information, see Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47).

## Change the Default Internet Explorer Home Page for Users' Streaming Sessions

You can use Group Policy to change the default Internet Explorer home page for users' streaming sessions. Alternatively, if you do not have Group Policy in your environment or prefer not to use Group Policy, you can use the AppStream 2.0 Template User account instead.

### Contents

- Use Group Policy to Change the Default Internet Explorer Home Page (p. 92)
- Use the AppStream 2.0 Template User Account to Change the Default Internet Explorer Home Page (p. 94)

### Use Group Policy to Change the Default Internet Explorer Home Page

In Active Directory environments, you use the Group Policy Management (GPMC) MMC-snap-in to set a default home page that users can't change. If Active Directory is not in your environment, you can use
Local Group Policy Editor to perform this task. To set a home page that users can change, you must use the GPMC.

To use the GPMC, do the following first:

- Obtain access to a computer or an EC2 instance that is joined to your domain.
- Install the GPMC. For more information, see Installing or Removing Remote Server Administration Tools for Windows 7 in the Microsoft documentation.
- Log in as a domain user with permissions to create GPOs. Link GPOs to the appropriate organizational units (OUs).

**To change the default Internet Explorer home page by using a Group Policy administrative template**

You can use a Group Policy administrative template to set a default home page that users can't change. For more information about administrative templates, see Edit Administrative Template Policy Settings in the Microsoft documentation.

2. If you are not using Active Directory in your environment, open Local Group Policy Editor. If you are using Active Directory, open the GPMC. Locate the Scripts (Logon\Logoff) policy setting:
   - Local Group Policy Editor:
     On your image builder, open the command prompt as an administrator, type `gpedit.msc`, and then press ENTER.
     Under User Configuration, expand Administrative Templates, Windows Components, and then choose Internet Explorer.
   - GPMC:
     In your directory or on a domain controller, open the command prompt as an administrator, type `gpmc.msc`, and then press ENTER.
     In the left console tree, select the OU in which you want to create a new GPO, or use an existing GPO, and then do either of the following:
     - Create a new GPO by opening the context (right-click) menu and choosing Create a GPO in this domain, Link it here. For Name, provide a descriptive name for this GPO.
     - Select an existing GPO.
     Open the context menu for the GPO, and choose Edit.
     Under User Configuration, expand Policies, Administrative Templates, Windows Components, and then choose Internet Explorer.
3. Double-click Disable changing home page settings, choose Enabled, and in Home Page, enter a URL.
4. Choose Apply, OK.
5. Close Local Group Policy Editor or the GPMC.

**To change the default Internet Explorer home page by using Group Policy preferences**

You can use Group Policy preferences to set a default home page that users can change. For more information about working with Group Policy preferences, see Configure a Registry Item and Group Policy Preferences Getting Started Guide in the Microsoft documentation.
1. In your directory or on a domain controller, open the command prompt as an administrator, type `gpmc.msc`, and then press ENTER.

2. In the left console tree, select the OU in which you want to create a new GPO, or use an existing GPO, and then do either of the following:
   - Create a new GPO by opening the context (right-click) menu and choosing \**Create a GPO in this domain, Link it here.** For \*Name\*, provide a descriptive name for this GPO.
   - Select an existing GPO.

3. Open the context menu for the GPO, and choose **Edit**.

4. Under **User Configuration**, expand **Preferences**, and then choose **Windows Settings**.

5. Open the context (right-click) menu for **Registry** and choose **New, Registry Item**.

6. In the **New Registry Properties** dialog box, specify the following registry settings for Group Policy to configure:
   - For **Action**, choose **Update**.
   - For **Hive**, choose **HKEY_CURRENT_USER**.
   - For **Key Path**, browse to and select **HKEY_CURRENT_USER\SOFTWARE\Microsoft\Internet Explorer \Main**.
   - For **Value Name**, enter **Start Page**.
   - For **Value Data**, enter your home page URL.

7. On the **Common** tab, choose **Apply Once, Do not Re-Apply**.

   **Note**
   To enable your users to choose the **Use Default** button in their Internet Explorer browser settings and reset their default home page to your company home page, you can also set a value for **Default_Page_URL** without choosing **Apply Once** and **Do not Re-Apply**.

8. Choose **OK** and close the GPMC.

### Use the AppStream 2.0 Template User Account to Change the Default Internet Explorer Home Page

Follow these steps to use the Template User account to change the default Internet Explorer home page.

**To change the default Internet Explorer Home page by using the Template User account**


2. In the left navigation pane, choose **Images, Image Builder**.

3. Choose the image builder on which to change the default Internet Explorer home page, verify that it is in the **Running** state, and choose **Connect**.

4. Log in to the image builder by doing either of the following:
   - If your image builder is not joined to an Active Directory domain, on the **Local User** tab, choose **Template User**.
   - If your image builder is joined to an Active Directory domain, choose the **Directory User** tab, specify the credentials for a domain user account that does not have local administrator permissions on the image builder, then choose **Log in**.

5. Open Internet Explorer and complete the necessary steps to change the default home page.

6. In the upper right area of the image builder desktop, choose **Admin Commands, Switch User**.
7. This disconnects your current session and opens the login menu. Log in to the image builder by doing either of the following:
   - If your image builder is not joined to an Active Directory domain, on the Local User tab, choose Administrator.
   - If your image builder is joined to an Active Directory domain, choose the Directory User tab, and log in as a domain user who has local administrator permissions on the image builder.

8. On the image builder desktop, open Image Assistant.

9. Follow the necessary steps in Image Assistant to finish creating your image. For more information, see Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47).

User and Instance Metadata for AppStream 2.0 Fleets

AppStream 2.0 fleet instances have user and instance metadata available through Windows environment variables. You can use the following environment variables in your applications and scripts to modify your environment based on the fleet instance details.

<table>
<thead>
<tr>
<th>Environment Variable</th>
<th>Context</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AppStream_Stack_Name</td>
<td>User</td>
<td>The name of the stack that the streaming session started from.</td>
</tr>
<tr>
<td>AppStream_User_Access_Mode</td>
<td>User</td>
<td>The access mode that the user is using to stream. The values are custom, userpool, or saml.</td>
</tr>
<tr>
<td>AppStream_Session_Reservation_DateTime</td>
<td>User</td>
<td>The date and time when the user's streaming session started.</td>
</tr>
<tr>
<td>AppStream_UserName</td>
<td>User</td>
<td>The user name for the user.</td>
</tr>
<tr>
<td>AppStream_Session_ID</td>
<td>User</td>
<td>The session identifier for the user's streaming session.</td>
</tr>
<tr>
<td>APPSTREAM_SESSION_CONTEXT</td>
<td>Machine</td>
<td>The session context that was provided when the streaming URL was created.</td>
</tr>
<tr>
<td>Note</td>
<td></td>
<td>This environment variable is only available after the first application launch.</td>
</tr>
<tr>
<td>AppStream_Image_Arn</td>
<td>Machine</td>
<td>The ARN of the image that was used to create the streaming instance.</td>
</tr>
<tr>
<td>AppStream_Instance_Type</td>
<td>Machine</td>
<td>The instance type of the streaming instance. For example, stream.standard.medium.</td>
</tr>
<tr>
<td>AppStream_Resource_Type</td>
<td>Machine</td>
<td>The type of AppStream 2.0 resource. The value is either fleet or imagebuilder.</td>
</tr>
<tr>
<td>AppStream_Resource_Name</td>
<td>Machine</td>
<td>The name of the fleet.</td>
</tr>
</tbody>
</table>
Update an AppStream 2.0 Fleet with a New Image

When you create a new Amazon AppStream image, you must update your fleets to make the applications and data on the new image available to users. If your update is minor (for example, patching applications or the operating system), you can update your running fleet. When new streaming instances are created, they are created from the updated image. Changing the image on a running fleet does not disrupt users who have active streaming sessions. Unused streaming instances are replaced periodically, while streaming instances that users are connected to are terminated after the streaming sessions are finished.

You can update a fleet with a new image that runs the same operating system when the fleet is in a Running or Stopped state. However, you can update a fleet with a new image that runs a different operating system only when the fleet is in a Stopped state.

Note
The application catalog that AppStream 2.0 displays to users is based on the current image that is associated with the fleet. If the updated image contains applications that are not specified in the older image, the applications may not launch if the user streams from an instance that is based on the older image.

To update an AppStream 2.0 fleet with a new image

To apply operating system updates or make new applications available to users, create a new image that has these changes. Then, update the fleet with the new image.

2. In the left navigation pane, choose Images, Image Builder.
3. Choose the image builder to use, verify that it is in the Running state, and choose Connect.
4. Log in to the image builder by doing either of the following:
   • If your image builder is not joined to an Active Directory domain, on the Local User tab, choose Administrator.
   • If your image builder is joined to an Active Directory domain, choose the Directory User tab, type the credentials for a domain user account that has local administrator permissions on the image builder, then choose Log in.
5. Do either or both of the following as required:
   • Install updates to the operating system.
   • Install applications.

   If an application requires the Windows operating system to restart, let it do so. Before the operating system restarts, you are disconnected from your image builder. After the restart is complete, connect to the image builder again, then finish installing the application.
6. On the image builder desktop, open Image Assistant.
7. Follow the necessary steps in Image Assistant to finish creating your image. For more information, see Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47).

   After the image status changes to Available, you can update the fleet with your new image.
8. In the left navigation pane, choose Fleets.
9. Select the fleet that you want to update with the new image.
10. On the Fleet Details tab, choose Edit.
11. In the Edit Fleet dialog box, the list of available images displays in the Name list. Select the new image from the list.
12. Choose Update.
Fleet Auto Scaling for Amazon AppStream 2.0

Fleet Auto Scaling lets you change the size of your AppStream 2.0 fleet automatically to match the supply of available instances to user demand. Because one user requires one fleet instance, the size of your fleet determines the number of users who can stream concurrently. You can define scaling policies that adjust the size of your fleet automatically based on a variety of utilization metrics, and optimize the number of available instances to match user demand. You can also choose to turn off automatic scaling and make the fleet run at a fixed size.

Note
As you develop your plan for AppStream 2.0 fleet scaling, make sure that your network configuration meets your requirements. For larger deployments that require hundreds or thousands of fleet instances and involve unpredictable user demand, Fleet Auto Scaling may not be the best solution. For information about how to configure a network environment that supports effective fleet scaling and considerations for larger deployments, see VPC Setup Recommendations (p. 11).

Before you can use Fleet Auto Scaling, Application Auto Scaling needs permissions to access Amazon CloudWatch alarms and AppStream 2.0 fleets. For more information, see Network Access to Your Streaming Instance (p. 224) and Using IAM Policies to Manage Administrator Access to Application Auto Scaling (p. 229).

Note
When you use scaling, you work with the Application Auto Scaling API. For Fleet Auto Scaling to work correctly for AppStream 2.0, Application Auto Scaling requires permission to describe and update your AppStream 2.0 fleets and describe your Amazon CloudWatch alarms, and permissions to modify your fleet capacity on your behalf. For more information, see Roles Required for AppStream 2.0 and Application Auto Scaling (p. 226) and Using IAM Policies to Manage Administrator Access to Application Auto Scaling (p. 229).

The following topics provide information to help you understand and use AppStream 2.0 Fleet Auto Scaling.

Contents
- Scaling Concepts (p. 97)
- Managing Fleet Scaling Using the Console (p. 99)
- Managing Fleet Scaling Using the AWS CLI (p. 100)
- Additional Resources (p. 105)

Scaling Concepts

AppStream 2.0 scaling is provided by Application Auto Scaling. For more information, see the Application Auto Scaling API Reference.

For step-by-step guidance for working with AppStream 2.0 Fleet Auto Scaling, see Scaling Your Desktop Application Streams with Amazon AppStream 2.0 in the AWS Compute Blog.

To use Fleet Auto Scaling effectively, you must understand the following terms and concepts.

Minimum Capacity

The minimum number of fleet instances. The number of fleet instances can't be below this value, and scaling policies will not scale your fleet below this value. For example, if you set the minimum capacity for a fleet to 2, your fleet will never have less than 2 instances.
Maximum Capacity

The maximum number of fleet instances. The number of fleet instances can't be above this value, and scaling policies will not scale your fleet above this value. For example, if you set the maximum capacity for a fleet to 10, your fleet will never have more than 10 instances.

Desired Capacity

The total number of fleet instances that are either running or pending. This value represents the total number of concurrent streaming sessions that your fleet can support in a steady state. To set the value for Desired Capacity, edit Fleet Details. We do not recommend changing the Desired Capacity value manually when you use Scaling Policies.

If the value of Desired Capacity is set below the value of Minimum Capacity and a scale-out activity is triggered, Application Auto Scaling scales the Desired Capacity value up to the value of Minimum Capacity and then continues to scale out as required, based on the scaling policy. However, in this case, a scale-in activity does not adjust Desired Capacity, because it is already below the Minimum Capacity value.

If the value of Desired Capacity is set above the value of Maximum Capacity and a scale-in activity is triggered, Application Auto Scaling scales the Desired Capacity value down to the value of Maximum Capacity and then continues to scale in as required, based on the scaling policy. However, in this case, a scale-out activity does not adjust Desired Capacity, because it is already above the Maximum Capacity value.

Scaling Policy Action

The action that scaling policies perform on your fleet when the Scaling Policy Condition is met. You can choose an action based on % capacity or number of instance(s). For example, if Desired Capacity is 4 and Scaling Policy Action is set to "Add 25% capacity", Desired Capacity is increased by 25% to 5 when Scaling Policy Condition is met.

Scaling Policy Condition

The condition that triggers the action set in Scaling Policy Action. This condition includes a scaling policy metric, a comparison operator, and a threshold. For example, to scale a fleet if the utilization of the fleet is greater than 50%, your scaling policy condition should be "If Capacity Utilization > 50%".

Scaling Policy Metric

Your scaling policy is based on this metric. The following metrics are available for scaling policies:

Capacity Utilization

The percentage of instances in a fleet that are being used. You can use this metric to scale your fleet based on usage of the fleet. For example, Scaling Policy Condition: "If Capacity Utilization < 25%" perform Scaling Policy Action: "Remove 25 % capacity".

Available Capacity

The number of instances in your fleet that are available for user sessions. You can use this metric to maintain a buffer in your capacity available for users to start streaming sessions. For example, Scaling Policy Condition: "If Available Capacity < 5" perform Scaling Policy Action: "Add 5 instance(s)".

Insufficient Capacity Error

The number of session requests rejected due to lack of capacity. You can use this metric to provision new instances for users who can't start streaming sessions due to lack of capacity. For example, Scaling Policy Condition: "If Insufficient Capacity Error > 0" perform Scaling Policy Action: "Add 1 instance(s)".
Managing Fleet Scaling Using the Console

You can set up and manage fleet scaling by using the AppStream 2.0 console in either of the following two ways: During fleet creation, or any time, by using the Fleets tab. Two default scaling policies are associated with newly created fleets after launch. You can edit these policies on the Scaling Policies tab in the AppStream 2.0 console. For more information, see Create a Fleet (p. 82).

For user environments that vary in number, define scaling policies to control how scaling responds to demand. If you expect a fixed number of users or have other reasons for disabling scaling, you can set your fleet with a fixed number of instances.

To set a fleet scaling policy using the console

2. In the navigation pane, choose Fleets.
3. Select the fleet and then choose Scaling Policies.
4. Edit existing policies by choosing the edit icon next to each value. Set the desired values in the edit field and choose Update. The policy changes go into effect within a few minutes.
5. Add (create) new policies using the Add Policy link. Set the desired values in the edit field and choose Create. The new policy goes into effect within a few minutes.

You can use the Fleet Usage tab to monitor the effects of your scaling policy changes. The following is an example usage graph of scaling activity when five users connect to the fleet and then disconnect. This example is from a fleet using the following scaling policy values:

- Minimum Capacity = 1
- Maximum Capacity = 5
- Scale Out = Add 2 instances if Capacity Utilization > 75%
- Scale In = Remove 1 instance if Capacity Utilization < 25%

To set a fixed capacity fleet using the console

2. In the navigation pane, choose Fleets.
3. Select the fleet.
4. For Scaling Policies, remove all policies associated with the fleet.
5. For Fleet Details, edit the fleet to set Desired Capacity.

The fixed fleet has constant capacity based on the value that you specified as Desired Capacity. Note that a fixed fleet has the desired number of instances available at all times and the fleet must be stopped to stop billing costs for that fleet.

Managing Fleet Scaling Using the AWS CLI

You can set up and manage fleet scaling by using the AWS Command Line Interface (AWS CLI). For more advanced features such as setting up multiple scaling policies or setting scale-in and scale-out cooldown times, use the AWS CLI. Before running scaling policy commands, you must register your fleet as a scalable target. To do so, use the following register-scalable-target command:

```
aws application-autoscaling register-scalable-target
   --service-namespace appstream
   --resource-id fleet/fleetname
   --scalable-dimension appstream:fleet:DesiredCapacity
   --min-capacity 1 --max-capacity 5
```

Examples

- Example 1: Applying a Scaling Policy Based on Capacity Utilization (p. 100)
- Example 2: Applying a Scaling Policy Based on Insufficient Capacity Errors (p. 101)
- Example 3: Applying a Scaling Policy Based on Low Capacity Utilization (p. 102)
- Example 4: Change the Fleet Capacity Based on a Schedule (p. 103)
- Example 5: Applying a Target Tracking Scaling Policy (p. 104)

Example 1: Applying a Scaling Policy Based on Capacity Utilization

This AWS CLI example sets up a scaling policy that scales out a fleet by 25% if Utilization >= 75%.

The following put-scaling-policy command defines a utilization-based scaling policy:

```
aws application-autoscaling put-scaling-policy --cli-input-json file://scale-out-utilization.json
```

The contents of the file `scale-out-utilization.json` are as follows:

```json
{
   "PolicyName": "serviceName",
   "ServiceNamespace": "appstream",
   "ResourceId": "fleet/fleetname",
   "ScalableDimension": "appstream:fleet:DesiredCapacity",
   "PolicyType": "StepScaling",
   "StepScalingPolicyConfiguration": {
      "AdjustmentType": "PercentChangeInCapacity",
      "StepAdjustments": [
         {
            "MetricIntervalUpperBound": 0,
            "ScalingAdjustment": 25
         }
      ]
   }
}
```
If the command is successful, the output is similar to the following, although some details are unique to your account and Region. In this example, the policy identifier is e3425d21-16f0-d701-89fb-12f98dac64af.

```
```

Now, set up a CloudWatch alarm for this policy. Use the names, Region, account number, and policy identifier that apply to you. You can use the policy ARN returned by the previous command for the --alarm-actions parameter.

```
aws cloudwatch put-metric-alarm
--alarm-name alarmname
--alarm-description "Alarm when Capacity Utilization exceeds 75 percent"
--metric-name CapacityUtilization
--namespace AWS/AppStream
--statistic Average
--period 300
--comparison-operator GreaterThanThreshold
--dimensions "Name=Fleet,Value=fleetname"
--evaluation-periods 1 --unit Percent
```

**Example 2: Applying a Scaling Policy Based on Insufficient Capacity Errors**

This AWS CLI example sets up a scaling policy that scales out the fleet by 1 if the fleet returns an InsufficientCapacityError error.

The following command defines an insufficient capacity-based scaling policy:

```
aws application-autoscaling put-scaling-policy --cli-input-json file://scale-out-capacity.json
```

The contents of the file `scale-out-capacity.json` are as follows:

```
{
  "PolicyName": "policyname",
  "ServiceNamespace": "appstream",
  "ResourceId": "fleet/fleetname",
  "ScalableDimension": "appstream:fleet:DesiredCapacity",
  "PolicyType": "StepScaling",
  "StepScalingPolicyConfiguration": {
    "AdjustmentType": "ChangeInCapacity",
    "StepAdjustments": [
      {
        "MetricIntervalLowerBound": 0,
        "ScalingAdjustment": 1
      }
    ]
  }
}
```
"Cooldown": 120
}
}

If the command is successful, the output is similar to the following, although some details are unique to your account and Region. In this example, the policy identifier is f4495f21-0650-470c-88e6-0f393adb64fc.


Now, set up a CloudWatch alarm for this policy. Use the names, Region, account number, and policy identifier that apply to you. You can use the policy ARN returned by the previous command for the --alarm-actions parameter.

```bash
aws cloudwatch put-metric-alarm
--alarm-name alarmname
--alarm-description "Alarm when out of capacity is > 0"
--metric-name InsufficientCapacityError
--namespace AWS/AppStream
--statistic Maximum
--period 300
--threshold 0
--comparison-operator GreaterThanThreshold
--dimensions "Name=Fleet,Value=fleetname"
--evaluation-periods 1 --unit Count
```

Example 3: Applying a Scaling Policy Based on Low Capacity Utilization

This AWS CLI example sets up a scaling policy that scales in the fleet to reduce actual capacity when CapacityUtilization is low.

The following command defines an excess capacity-based scaling policy:

```bash
aws application-autoscaling put-scaling-policy --cli-input-json file://scale-in-capacity.json
```

The contents of the file scale-in-capacity.json are as follows:

```json
{
    "PolicyName": "policyname",
    "ServiceNamespace": "appstream",
    "ResourceId": "fleet/fleetname",
    "ScalableDimension": "appstream:fleet:DesiredCapacity",
    "PolicyType": "StepScaling",
    "StepScalingPolicyConfiguration": {
        "AdjustmentType": "PercentChangeInCapacity",
        "StepAdjustments": [
            {
                "MetricIntervalUpperBound": 0,
                "ScalingAdjustment": 25
            }
        ],
        "Cooldown": 360
    }
}
```
If the command is successful, the output is similar to the following, although some details are unique to your account and Region. In this example, the policy identifier is 12ab3c4d-56789-0ef1-2345-6ghi7jk8lm90.

```
```

Now, set up a CloudWatch alarm for this policy. Use the names, Region, account number, and policy identifier that apply to you. You can use the policy ARN returned by the previous command for the `--alarm-actions` parameter.

```
aws cloudwatch put-metric-alarm
--alarm-name alarmname
--alarm-description "Alarm when Capacity Utilization is less than or equal to 25 percent" 
--namespace AWS/AppStream
--statistic Average
--period 120
--threshold 25
--comparison-operator LessThanOrEqualToThreshold
--dimensions "Name=Fleet,Value=fleetname" 
--evaluation-periods 10 --unit Percent
```

Example 4: Change the Fleet Capacity Based on a Schedule

Changing your fleet capacity based on a schedule lets you scale your fleet capacity in response to predictable changes in demand. For example, at the start of a work day, you might expect a certain number of users to request streaming connections at one time. To change your fleet capacity based on a schedule, you can use the Application Auto Scaling PutScheduledAction API action or the put-scheduled-action AWS CLI command.

Before changing your fleet capacity, you can list your current fleet capacity by using the AppStream 2.0 describe-fleets AWS CLI command.

```
aws appstream describe-fleets --name fleetname
```

The current fleet capacity will appear similar to the following output (shown in JSON format):

```
{
    "ComputeCapacityStatus": {
        "Available": 1,
        "Desired": 1,
        "Running": 1,
        "InUse": 0
    },
```

Then, use the put-scheduled-action command to create a scheduled action to change your fleet capacity. For example, the following command changes the minimum capacity to 3 and the maximum capacity to 5 every day at 9:00 AM UTC.
Note
For cron expressions, specify when to perform the action in UTC. For more information, see Cron Expressions.

```bash
aws application-autoscaling put-scheduled-action --service-namespace appstream --resource-id fleet/fleetname --schedule="cron(0 9 * * ? *)" --scalable-target-action MinCapacity=3,MaxCapacity=5 --scheduled-action-name ExampleScheduledAction --scalable-dimension appstream:fleet:DesiredCapacity
```

To confirm that the scheduled action to change your fleet capacity was successfully created, run the `describe-scheduled-actions` command.

```bash
aws application-autoscaling describe-scheduled-actions --service-namespace appstream --resource-id fleet/fleetname
```

If the scheduled action was successfully created, the output appears similar to the following.

```json
{
  "ScheduledActions": [
    {
      "ScalableDimension": "appstream:fleet:DesiredCapacity",
      "Schedule": "cron(0 9 * * ? *)",
      "ResourceId": "fleet/ExampleFleet",
      "CreationTime": 1518651232.886,
      "ScheduledActionARN": "<arn>",
      "ScalableTargetAction": {
        "MinCapacity": 3,
        "MaxCapacity": 5
      },
      "ScheduledActionName": "ExampleScheduledAction",
      "ServiceNamespace": "appstream"
    }
  ]
}
```

For more information, see Scheduled Scaling in the Application Auto Scaling User Guide.

**Example 5: Applying a Target Tracking Scaling Policy**

With target tracking scaling, you can specify a capacity utilization level for your fleet.

When you create a target tracking scaling policy, Application Auto Scaling automatically creates and manages CloudWatch alarms that trigger the scaling policy. The scaling policy adds or removes capacity as required to keep capacity utilization at, or close to, the specified target value. To ensure application availability, your fleet scales out proportionally to the metric as fast as it can but scales in more gradually.

The following `put-scaling-policy` command defines a target tracking scaling policy that attempts to maintain 75% capacity utilization for an AppStream 2.0 fleet.

```bash
aws application-autoscaling put-scaling-policy --cli-input-json file://config.json
```

The contents of the file `config.json` are as follows:

```json
{
  "PolicyName":"target-tracking-scaling-policy",
```
If the command is successful, the output is similar to the following, although some details are unique to your account and Region. In this example, the policy identifier is 6d8972f3-efc8-437c-92d1-6270f29a66e7.

```
{
  "Alarms": [
    {
      "AlarmName": "TargetTracking-fleet/fleetname-AlarmHigh-d4f0770c-b46e-434a-a60f-3b36d653feca"
    },
    {
      "AlarmName": "TargetTracking-fleet/fleetname-AlarmLow-1b437334-d19b-4a63-a812-6c67aaf2910d"
    }
  ]
}
```

For more information, see Target Tracking Scaling Policies in the Application Auto Scaling User Guide.

**Additional Resources**

For step-by-step guidance for working with AppStream 2.0 Fleet Auto Scaling, see Scaling Your Desktop Application Streams with Amazon AppStream 2.0 in the AWS Compute Blog.

To learn more about using the Application Auto Scaling AWS CLI commands or API actions, see the following resources:

- application-autoscaling section of the AWS CLI Command Reference
- Application Auto Scaling API Reference
- Application Auto Scaling User Guide
Using Active Directory with AppStream 2.0

You can join your Amazon AppStream 2.0 fleets and image builders to domains in Microsoft Active Directory and use your existing Active Directory domains, either cloud-based or on-premises, to launch domain-joined streaming instances. You can also use AWS Directory Service for Microsoft Active Directory, also known as AWS Managed Microsoft AD, to create an Active Directory domain and use that to support your AppStream 2.0 resources. For more information about using AWS Managed Microsoft AD, see Microsoft Active Directory in the AWS Directory Service Administration Guide.

By joining AppStream 2.0 to your Active Directory domain, you can:

• Allow your users and applications to access Active Directory resources such as printers and file shares from streaming sessions.
• Use Group Policy settings that are available in the Group Policy Management Console (GPMC) to define the end user experience.
• Stream applications that require users to be authenticated using their Active Directory login credentials.
• Apply your enterprise compliance and security policies to your AppStream 2.0 streaming instances.

Contents

• Overview of Active Directory Domains (p. 106)
• Before You Begin Using Active Directory with AppStream 2.0 (p. 108)
• Tutorial: Setting Up Active Directory (p. 109)
• AppStream 2.0 Active Directory Administration (p. 111)
• More Info (p. 118)

Overview of Active Directory Domains

Using Active Directory domains with AppStream 2.0 requires an understanding of how they work together and the configuration tasks that you’ll need to complete. You’ll need to complete the following tasks:

1. Configure Group Policy settings as needed to define the end user experience and security requirements for applications.
2. Create the domain-joined application stack in AppStream 2.0.
3. Create the AppStream 2.0 application in the SAML 2.0 identity provider and assign it to end users either directly or through Active Directory groups.

For your users to be authenticated to a domain, several steps must occur when these users initiate an AppStream 2.0 streaming session. The following diagram illustrates the end-to-end user authentication flow from the initial browser request through SAML and Active Directory authentication.
User Authentication Flow

1. The user browses to https://applications.exampleco.com. The sign-on page requests authentication for the user.
2. The federation service requests authentication from the organization's identity store.
3. The identity store authenticates the user and returns the authentication response to the federation service.
4. On successful authentication, the federation service posts the SAML assertion to the user's browser.
5. The user's browser posts the SAML assertion to the AWS Sign-In SAML endpoint (https://signin.aws.amazon.com/saml). AWS Sign-In receives the SAML request, processes the request, authenticates the user, and forwards the authentication token to the AppStream 2.0 service.
6. Using the authentication token from AWS, AppStream 2.0 authorizes the user and presents applications to the browser.
7. The user chooses an application and is prompted to enter login information for the domain.
8. The domain controller is contacted for user authentication.
9. After being authenticated with the domain, the user's session starts with domain connectivity.

From the user’s perspective, the process happens transparently. The user starts at your organization's internal portal and lands at an AppStream 2.0 application portal, without having to enter AWS credentials. Only Active Directory domain login credentials are required.

Before a user can initiate this process, you must configure Active Directory with the required entitlements and Group Policy settings and create a domain-joined application stack.
Before You Begin Using Active Directory with AppStream 2.0

Before you use Microsoft Active Directory domains with AppStream 2.0, be aware of the following requirements and considerations.

Requirements and Considerations

- You must have a Microsoft Active Directory domain to which to join your streaming instances. If you don't have an Active Directory domain or you want to use your on-premises Active Directory environment, see Active Directory Domain Services on the AWS Cloud: Quick Start Reference Deployment.

- You must have a domain service account with permissions to create and manage computer objects in the domain that you intend to use with AppStream 2.0. For information, see How to Create a Domain Account in Active Directory in the Microsoft documentation.

When you associate this Active Directory domain with AppStream 2.0, provide the service account name and password. AppStream 2.0 uses this account to create and manage computer objects in the directory. For more information, see Granting Permissions to Create and Manage Active Directory Computer Objects (p. 111).

- When you register your Active Directory domain with AppStream 2.0, you must provide an organizational unit (OU) distinguished name. Create an OU for this purpose. The default Computers container is not an OU and cannot be used by AppStream 2.0. For more information, see Finding the Organizational Unit Distinguished Name (p. 112).

- The directories that you plan to use with AppStream 2.0 must be accessible through their fully qualified domain names (FQDNs) through the virtual private cloud (VPC) in which your streaming instances are launched. For more information, see Active Directory and Active Directory Domain Services Port Requirements in the Microsoft documentation.

- SAML 2.0-based user federation is required for application streaming from domain-joined fleets. You cannot launch sessions to domain-joined instances by using CreateStreamingURL or the AppStream 2.0 user pool.

- You must use an image that supports joining image builders and fleets to an Active Directory domain. All public images published on or after July 24, 2017 support joining an Active Directory domain. For more information, see AppStream 2.0 Base Image Version History (p. 29) and Tutorial: Setting Up Active Directory (p. 109).

- Verify your configuration for the following Group Policy settings. If required, update the settings as described in this section so that they don't block AppStream 2.0 from authenticating and logging in your domain users. Otherwise, when your users try to log in to AppStream 2.0, the login may not succeed. Instead, a message displays, notifying users that "An unknown error occurred."

  - **Computer Configuration > Administrative Templates > Windows Components > Windows Logon Options > Disable or Enable software Secure Attention Sequence** — Set this to Enabled for Services.

  - **Computer Configuration > Administrative Templates > System > Logon > Exclude credential providers** — Ensure that the following CLSID is not listed: e7c1bab5-4b49-4e64-a966-8d99686f8c7c

  - **Computer Configuration > Policies > Windows Settings > Security Settings > Security Options > Interactive Logon > Interactive Logon: Message text for users attempting to log on** — Set this to Not defined.

  - **Computer Configuration > Policies > Windows Settings > Security Settings > Local Policies > Security Options > Interactive Logon > Interactive Logon: Message title for users attempting to log on** — Set this to Not defined.
Tutorial: Setting Up Active Directory

To use Active Directory with AppStream 2.0, you must first register your directory configuration by creating a Directory Config object in AppStream 2.0. This object includes the information required to join streaming instances to an Active Directory domain. You create a Directory Config object by using the AppStream 2.0 management console, AWS SDK, or AWS CLI. You can then use your directory configuration to launch domain-joined fleets and image builders.

Tasks
- Step 1: Create a Directory Config Object (p. 109)
- Step 2: Create an Image by Using a Domain-Joined Image Builder (p. 109)
- Step 3: Create a Domain-Joined Fleet (p. 110)
- Step 4: Configure SAML 2.0 (p. 110)

Step 1: Create a Directory Config Object

The Directory Config object that you create in AppStream 2.0 will be used in later steps.

If you are using the AWS SDK, you can use the CreateDirectoryConfig operation. If you are using the AWS CLI, you can use the create-directory-config command.

To create a Directory Config object by using the AppStream 2.0 console

2. In the navigation pane, choose Directory Configs, Create Directory Config.
3. For Directory Name, provide the fully qualified domain name (FQDN) of the Active Directory domain (for example, corp.example.com). Each region can have only one Directory Config value with a specific directory name.
4. For Service Account Name, enter the name of an account that can create computer objects and that has permissions to join the domain. For more information, see Granting Permissions to Create and Manage Active Directory Computer Objects (p. 111). The account name must be in the format DOMAIN\username.
5. For Password and Confirm Password, type the directory password for the specified account.
6. For Organizational Unit (OU), type the distinguished name of at least one OU for streaming instance computer objects. The default Computers container is not an OU and cannot be used by AppStream 2.0. For more information, see Finding the Organizational Unit Distinguished Name (p. 112).
7. To add more than one OU, select the plus sign (+) next to Organizational Unit (OU). To remove OUs, choose the x icon.
8. Choose Next.
9. Review the configuration information and choose Create.

Step 2: Create an Image by Using a Domain-Joined Image Builder

Next, using the AppStream 2.0 image builder, create a new image with Active Directory domain-join capabilities. Note that the fleet and image can be members of different domains. You join the image builder to a domain to enable domain join and to install applications. Fleet domain join is discussed in the next section.
To create an image for launching domain-joined fleets

1. Follow the procedures in Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47).
2. For the base image selection step, use an AWS base image released on or after July 24, 2017. For a current list of released AWS images, see AppStream 2.0 Base Image Version History (p. 29).
3. For Step 3: Configure Network, select a VPC and subnets with network connectivity to your Active Directory environment. Select the security groups that are set up to allow access to your directory through your VPC subnets.
4. Also in Step 3: Configure Network, expand the Active Directory Domain (Optional) section, and select values for the Directory Name and Directory OU to which the image builder should be joined.
5. Review the image builder configuration and choose Create.
6. Wait for the new image builder to reach a Running state, and choose Connect.
7. Log in to the image builder in Administrator mode or as a directory user with local administrator permissions. For more information, see Granting Local Administrator Rights on Image Builders (p. 113).
8. Complete the steps in Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47) to install applications and create a new image.

Step 3: Create a Domain-Joined Fleet

Using the private image created in the previous step, create an Active Directory domain-joined fleet for streaming applications. The domain can be different than the one that you used for the image builder to create the image.

To create a domain-joined fleet

1. Follow the procedures in Create a Fleet (p. 82).
2. For the image selection step, use the image that was created in the previous step, Step 2: Create an Image by Using a Domain-Joined Image Builder (p. 109).
3. For Step 4: Configure Network, select a VPC and subnets with network connectivity to your Active Directory environment. Select the security groups that are set up to allow communication to your domain.
4. Also in Step 4: Configure Network, expand the Active Directory Domain (Optional) section and select the values for the Directory Name and Directory OU to which the fleet should be joined.
5. Review the fleet configuration and choose Create.
6. Complete the remaining steps in Create an AppStream 2.0 Fleet and Stack (p. 81) so that your fleet is associated with a stack and running.

Step 4: Configure SAML 2.0

Your users must use your SAML 2.0-based identity federation environment to launch streaming sessions from your domain-joined fleet.

To configure SAML 2.0 for single sign-on access

1. Follow the procedures in Setting Up SAML (p. 244).
2. AppStream 2.0 requires that the SAML_Subject Name ID value for the user who is logging in be provided in either of the following formats:
• *domain\username* using the sAMAccountName
• *username@domain.com* using the userPrincipalName

If you are using the sAMAccountName format, you can specify the *domain* by using either the NetBIOS name or the fully qualified domain name (FQDN).

3. Provide access to your Active Directory users or groups to enable access to the AppStream 2.0 stack from your identity provider application portal.

4. Complete the remaining steps in Setting Up SAML (p. 244).

**To log in a user with SAML 2.0**

1. Log in to your SAML 2.0 provider’s application catalog and open the AppStream 2.0 SAML application that you created in the previous procedure.

2. When the AppStream 2.0 application catalog is displayed, select an application to launch.

3. When a loading icon is displayed, you are prompted to provide a password. The domain user name provided by your SAML 2.0 identity provider appears above the password field. Enter your password, and choose **log in**.

The streaming instance performs the Windows login procedure, and the selected application opens.

**AppStream 2.0 Active Directory Administration**

Setting up and using Active Directory with AppStream 2.0 involves the following administrative tasks.

**Tasks**

• Granting Permissions to Create and Manage Active Directory Computer Objects (p. 111)
• Finding the Organizational Unit Distinguished Name (p. 112)
• Granting Local Administrator Rights on Image Builders (p. 113)
• Updating the Service Account Used for Joining the Domain (p. 114)
• Locking the Streaming Session When the User is Idle (p. 115)
• Editing the Directory Configuration (p. 116)
• Deleting a Directory Configuration (p. 116)
• Configuring AppStream 2.0 to Use Domain Trusts (p. 116)
• Managing AppStream 2.0 Computer Objects in Active Directory (p. 117)

**Granting Permissions to Create and Manage Active Directory Computer Objects**

To allow AppStream 2.0 to perform Active Directory computer object operations, you need an account with sufficient permissions. As a best practice, use an account that has only the minimum privileges necessary. The minimum Active Directory organizational unit (OU) permissions are as follows:

• Create Computer Object
• Change Password
• Reset Password
• Write Description
Before setting up permissions, you'll need to do the following first:

- Obtain access to a computer or an EC2 instance that is joined to your domain.
- Install the Active Directory User and Computers MMC snap-in. For more information, see Installing or Removing Remote Server Administration Tools for Windows 7 in the Microsoft documentation.
- Log in as a domain user with appropriate permissions to modify the OU security settings.
- Create or identify the user account, service account, or group for which to delegate permissions.

To set up minimum permissions

1. Open Active Directory Users and Computers in your domain or on your domain controller.
2. In the left navigation pane, select the first OU on which to provide domain join privileges, open the context (right-click) menu, and then choose Delegate Control.
4. For Select Users, Computers, or Groups, select the pre-created user account, service account, or group, and then choose OK.
5. On the Tasks to Delegate page, choose Create a custom task to delegate, and then choose Next.
6. Choose Only the following objects in the folder, Computer objects.
7. Choose Create selected objects in this folder, Next.
8. For Permissions, choose Read, Write, Change Password, Reset Password, Next.
10. Repeat steps 2-9 for any additional OUs that require these permissions.

If you delegated permissions to a group, create a user or service account with a strong password and add that account to the group. This account will then have sufficient privileges to connect your streaming instances to the directory. Use this account when creating your AppStream 2.0 directory configuration.

Finding the Organizational Unit Distinguished Name

When you register your Active Directory domain with AppStream 2.0, you must provide an organizational unit (OU) distinguished name. Create an OU for this purpose. The default Computers container is not an OU and cannot be used by AppStream 2.0. The following procedure shows how to obtain this name.

Note
The distinguished name must start with ou= or it cannot be used for computer objects.

Before you complete this procedure, you'll need to do the following first:

- Obtain access to a computer or an EC2 instance that is joined to your domain.
- Install the Active Directory User and Computers MMC snap-in. For more information, see Installing or Removing Remote Server Administration Tools for Windows 7 in the Microsoft documentation.
- Log in as a domain user with appropriate permissions to read the OU security properties.

To find the distinguished name of an OU

1. Open Active Directory Users and Computers in your domain or on your domain controller.
2. Under View, ensure that Advanced Features is enabled.
3. In the left navigation pane, select the first OU to use for AppStream 2.0 streaming instance computer objects, open the context (right-click) menu, and then choose Properties.
4. Choose **Attribute Editor**.
5. Under **Attributes**, for **distinguishedName**, choose **View**.
6. For **Value**, select the distinguished name, open the context menu, and then choose **Copy**.

Granting Local Administrator Rights on Image Builders

By default, Active Directory domain users do not have local administrator rights on image builder instances. You can grant these rights by using Group Policy preferences in your directory, or manually, by using the local administrator account on an image builder. Granting local administrator rights to a domain user allows that user to install applications on and create images in an AppStream 2.0 image builder.

Contents

- Using Group Policy preferences (p. 113)
- Using the local Administrators group on the image builder (p. 114)

Using Group Policy preferences

You can use Group Policy preferences to grant local administrator rights to Active Directory users or groups and to all computer objects in the specified OU. The Active Directory users or groups to which you want to grant local administrator permissions must already exist. To use Group Policy preferences, you'll need to do the following first:

- Obtain access to a computer or an EC2 instance that is joined to your domain.
- Install the Group Policy Management Console (GPMC) MMC snap-in. For more information, see Installing or Removing Remote Server Administration Tools for Windows 7 in the Microsoft documentation.
- Log in as a domain user with permissions to create Group Policy objects (GPOs). Link GPOs to the appropriate OUs.

To use Group Policy preferences to grant local administrator permissions

1. In your directory or on a domain controller, open the command prompt as an administrator, type `gpmc.msc`, and then press ENTER.
2. In the left console tree, select the OU where you will create a new GPO or use an existing GPO, and then do either of the following:
   - Create a new GPO by opening the context (right-click) menu and choosing **Create a GPO in this domain, Link it here**. For **Name**, provide a descriptive name for this GPO.
   - Select an existing GPO.
3. Open the context menu for the GPO, and choose **Edit**.
4. In the console tree, choose **Computer Configuration, Preferences, Windows Settings, Control Panel Settings**, and **Local Users and Groups**.
5. Select **Local Users and Groups** selected, open the context menu , and choose **New, Local Group**.
6. For **Action**, choose **Update**.
7. For **Group name**, choose **Administrators (built-in)**.
8. Under **Members**, choose **Add...** and specify the Active Directory user accounts or groups to which to assign local administrator rights on the streaming instance. For **Action**, choose **Add to this group**, and choose **OK**.
9. To apply this GPO to other OUs, select the additional OU, open the context menu and choose **Link an Existing GPO**.
10. Using the new or existing GPO name that you specified in step 2, scroll to find the GPO, and then choose **OK**.
11. Repeat steps 9 and 10 for additional OUs that should have this preference.
12. Choose **OK** to close the **New Local Group Properties** dialog box.
13. Choose **OK** again to close the GPMC.

To apply the new preference to the GPO, you must stop and restart any running image builders or fleets. The Active Directory users and groups that you specified in step 8 are automatically granted local administrator rights on the image builders and fleets in the OU to which the GPO is linked.

### Using the local Administrators group on the image builder

To grant Active Directory users or groups local administrator rights on your image builder, you can manually add these users or groups to the local Administrators group on the image builder. Image builders that are created from images with these rights maintain the same rights.

The Active Directory users or groups to which to grant local administrator rights must already exist.

#### To add Active Directory users or groups to the local Administrators group on the image builder

2. Connect to the image builder in Administrator mode. The image builder must be running and domain-joined. For more information, see **Tutorial: Setting Up Active Directory** (p. 109).
3. Choose **Start**, **Administrative Tools**, and then double-click **Computer Management**.
4. In the left navigation pane, choose **Local Users and Groups** and open the **Groups** folder.
5. Open the **Administrators** group and choose **Add**...
6. Select all Active Directory users or groups to which to assign local administrator rights and choose **OK**. Choose **OK** again to close the **Administrator Properties** dialog box.
8. To log in as an Active Directory user and test whether that user has local administrator rights on the image builder, choose **Admin Commands**, **Switch user**, and then enter the credentials of the relevant user.

### Updating the Service Account Used for Joining the Domain

To update the service account that AppStream 2.0 uses for joining the domain, we recommend using two separate service accounts for joining image builders and fleets to your Active Directory domain. Using two separate service accounts ensures that there is no disruption in service when a service account needs to be updated (for example, when a password expires).

#### To update a service account

1. Create an Active Directory group and delegate the correct permissions to the group.
2. Add your service accounts to the new Active Directory group.
3. When needed, edit your AppStream 2.0 Directory Config object by entering the user name and password for the new service account.
After you've set up the Active Directory group with the new service account, any new streaming instance operations will use the new service account, while in-process streaming instance operations continue to use the old account without interruption.

The service account overlap time while the in-process streaming instance operations complete is very short, no more than a day. The overlap time is needed because you shouldn't delete or change the password for the old service account during the overlap period, or existing operations can fail.

**Locking the Streaming Session When the User is Idle**

AppStream 2.0 relies on a setting that you configure in the GPMC to lock the streaming session after your user is idle for specified amount of time. To use the GPMC, you'll need to do the following first:

- Obtain access to a computer or an EC2 instance that is joined to your domain.
- Install the GPMC. For more information, see Installing or Removing Remote Server Administration Tools for Windows 7 in the Microsoft documentation.
- Log in as a domain user with permissions to create GPOs. Link GPOs to the appropriate OUs.

**To automatically lock the streaming instance when your user is idle**

1. In your directory or on a domain controller, open the command prompt as an administrator, type `gpmc.msc`, and then press ENTER.
2. In the left console tree, select the OU where you will create a new GPO or use an existing GPO, and then do either of the following:
   - Create a new GPO by opening the context (right-click) menu and choosing **Create a GPO in this domain, Link it here**. For **Name**, provide a descriptive name for this GPO.
   - Select an existing GPO.
3. Open the context menu for the GPO, and choose **Edit**.
4. Under **User Configuration**, expand **Policies**, **Administrative Templates**, **Control Panel**, and then choose **Personalization**.
5. Double-click **Enable screen saver**.
6. In the **Enable screen saver** policy setting, choose **Enabled**.
7. Choose **Apply**, and then choose **OK**.
8. Double-click **Force specific screen saver**.
9. In the **Force specific screen saver** policy setting, choose **Enabled**.
10. Under **Screen saver executable name**, enter `scrnsave.scr`. When this setting is enabled, the system displays a black screen saver on the user's desktop.
11. Choose **Apply**, and then choose **OK**.
12. Double-click **Password protect the screen saver**.
13. In the **Password protect the screen saver** policy setting, choose **Enabled**.
14. Choose **Apply**, and then choose **OK**.
15. Double-click **Screen saver timeout**.
16. In the **Screen saver timeout** policy setting, choose **Enabled**.
17. For **Seconds**, specify the length of time that users must be idle before the screen saver is applied. To set the idle time to 10 minutes, specify 600 seconds.
18. Choose **Apply**, and then choose **OK**.
19. In the console tree, under **User Configuration**, expand **Policies**, **Administrative Templates**, **System**, and then choose **Ctrl+Alt+Del Options**.
20. Double-click **Remove Lock Computer**.
21. In the **Remove Lock Computer** policy setting, choose **Disabled**.
22. Choose **Apply**, and then choose **OK**.

## Editing the Directory Configuration

After an AppStream 2.0 directory configuration has been created, you can edit it to add, remove, or modify organizational units, update the service account username, or update the service account password.

### To update a directory configuration

2. In the left navigation pane, choose **Directory Configs** and select the directory configuration to edit.
3. Choose **Actions**, **Edit**.
4. Update the fields to be changed. To add additional OUs, select the plus sign (+) next to the topmost OU field. To remove an OU field, select the x next to the field.

**Note**

At least one OU is required. OUs that are currently in use cannot be removed.

5. To save changes, choose **Update Directory Config**.
6. The information in the **Details** tab should now update to reflect the changes.

Changes to the service account user name and password do not impact in-process streaming instance operations. New streaming instance operations use the updated credentials. For more information, see [Updating the Service Account Used for Joining the Domain (p. 114)](#).

## Deleting a Directory Configuration

You can delete an AppStream 2.0 directory configuration that is no longer needed. Directory configurations that are associated with any image builders or fleets cannot be deleted.

### To delete a directory configuration

2. In the left navigation pane, choose **Directory Configs** and select the directory configuration to delete.
3. Choose **Actions**, **Delete**.
4. Verify the name in the pop-up message, and choose **Delete**.
5. Choose **Update Directory Config**.

## Configuring AppStream 2.0 to Use Domain Trusts

AppStream 2.0 supports Active Directory domain environments where network resources such as file servers, applications, and computer objects reside in one domain, and the user objects reside in another. The domain service account used for computer object operations does not need to be in the same domain as the AppStream 2.0 computer objects.

When creating the directory configuration, specify a service account that has the appropriate permissions to manage computer objects in the Active Directory domain where the file servers, applications, computer objects and other network resources reside.
Your end user Active Directory accounts must have the "Allowed to Authenticate" permissions for the following:

- AppStream 2.0 computer objects
- Domain controllers for the domain

For more information, see Granting Permissions to Create and Manage Active Directory Computer Objects (p. 111).

Managing AppStream 2.0 Computer Objects in Active Directory

AppStream 2.0 does not delete computer objects from Active Directory. These computer objects can be easily identified in your directory. Each computer object in the directory is created with the Description attribute, which specifies a fleet or an image builder instance and the name.

Computer Object Description Examples

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fleet</td>
<td>ExampleFleet</td>
<td>AppStream 2.0 - fleet:ExampleFleet</td>
</tr>
<tr>
<td>Image builder</td>
<td>ExampleImageBuilder</td>
<td>AppStream 2.0 - image-builder:ExampleImageBuilder</td>
</tr>
</tbody>
</table>

You can identify and delete inactive computer objects created by AppStream 2.0 by using the following `dsquery computer` and `dsrm` commands. For more information, see Dsquery computer and Dsrm in the Microsoft documentation.

The `dsquery` command identifies inactive computer objects over a certain period of time and uses the following format. The `dsquery` command should also be run with the parameter `-desc "AppStream 2.0*"` to display only AppStream 2.0 objects.

```
dsquery computer "OU-distinguished-name" -desc "AppStream 2.0*" -inactive number-of-weeks-since-last-login
```

- `OU-distinguished-name` is the distinguished name of the organizational unit. For more information, see Finding the Organizational Unit Distinguished Name (p. 112). If you don't provide the `OU-distinguished-name` parameter, the command searches the entire directory.
- `number-of-weeks-since-last-log-in` is the desired value based on how you want to define inactivity.

For example, the following command displays all computer objects in the OU=ExampleOU, DC=EXAMPLECO, DC=COM organizational unit that have not been logged into within the past two weeks.

```
dsquery computer OU=ExampleOU,DC=EXAMPLECO,DC=COM -desc "AppStream 2.0*" -inactive 2
```

If any matches are found, the result is one or more object names. The `dsrm` command deletes the specified object and uses the following format:

```
dsrm objectname
```
Where *objectname* is the full object name from the output of the *dsquery* command. For example, if the *dsquery* command above results in a computer object named "ExampleComputer", the *dsrm* command to delete it would be as follows:

```plaintext
dsrm "CN=ExampleComputer,OU=ExampleOU,DC=EXAMPLECO,DC=COM"
```

You can chain these commands together by using the pipe (|) operator. For example, to delete all AppStream 2.0 computer objects, prompting for confirmation for each, use the following format. Add the `-noprompt` parameter to *dsrm* to disable confirmation.

```plaintext
dsquery computer OU-distinguished-name -desc "AppStream 2.0*" -inactive number-of-weeks-since-last-log-in | dsrm -noprompt
```

### More Info

For more information related to this topic, see the following resources:

- Troubleshooting Notification Codes (p. 268)—Resolutions to notification code errors.
- Troubleshooting Active Directory Domain Join (p. 266)—Help with common difficulties.
Add Your Custom Branding to Amazon AppStream 2.0

To create a familiar experience for your users when they stream applications, you can customize the appearance of AppStream 2.0 with your own branding images, text, and website links, and you can choose from one of several color palettes. When you customize AppStream 2.0, your branding is displayed to users during application streaming sessions rather than the default AppStream 2.0 branding.

Custom Branding Options

You can customize the appearance of the streaming application catalog page by using the following branding options.

**Note**
Custom branding is not available for the user pool sign-in portal or for the email notifications that AppStream 2.0 sends to user pool users.

<table>
<thead>
<tr>
<th>Branding element</th>
<th>Description</th>
<th>Requirements and recommendations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organization logo</td>
<td>Enables you to display an image that is familiar to your users. The image appears in the header of the streaming application catalog page, which is displayed to users after they sign in to AppStream 2.0.</td>
<td>File type: .png, jpg, jpeg, or .gif</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Maximum dimensions: 1000 px x 500 px</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Maximum file size: 300 KB</td>
</tr>
<tr>
<td>Organization website links</td>
<td>Enables you to display links to helpful resources for your users, such as your organization's IT support and product marketing sites. The links are displayed in the footer of the streaming application catalog page.</td>
<td>Maximum number of links: 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Format (URL): <a href="https://example.com">https://example.com</a> or <a href="http://example.com">http://example.com</a></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Maximum length (display name): 100 letters, spaces, and numbers</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Special characters allowed (display name): @ . / # &amp; + $</td>
</tr>
<tr>
<td>Color theme</td>
<td>Applied to website links, text, and buttons. These colors are also applied as accents in the background for the streaming application catalog page.</td>
<td>Predefined themes from which to choose: 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For information about each color theme, see Color Theme Palettes (p. 121) later in this topic.</td>
</tr>
<tr>
<td>Page title</td>
<td>Displayed at the top of the browser tab</td>
<td>Maximum length: 200 letters, spaces, and numbers.</td>
</tr>
<tr>
<td>Branding element</td>
<td>Description</td>
<td>Requirements and recommendations</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td><strong>Branding</strong></td>
<td>during users' application streaming sessions.</td>
<td>Special characters allowed: @ . / # &amp; + $</td>
</tr>
</tbody>
</table>
| Favicon          | Enables your users to recognize their application streaming site in a browser full of tabs or bookmarks. The favicon icon is displayed at the top of the browser tab for the application streaming site during users' streaming sessions. | File type: .png, .jpg, .jpeg, .gif, or .ico   
Maximum dimensions: 128 px x 128 px   
Maximum file size: 50 KB |
| Redirect URL     | Enables you to specify a URL to which users are redirected when they end a streaming session. | Format: https://example.com or http://example.com   
This URL is configured in the Details page for a stack when you create or edit a stack, not in the Branding page. |
| Feedback URL     | Enables you to specify a URL for a Send Feedback link, so that your users can submit feedback. If you do not specify a URL, the Send Feedback link is not displayed. | Format: https://example.com or http://example.com   
This URL is configured in the Details page for a stack when you create or edit a stack, not in the Branding page. |

Adding Your Custom Branding to AppStream 2.0

To customize AppStream 2.0 with your organizational branding, use the AppStream 2.0 console to select the stack to customize, and then add your branding.

**To add your custom branding to AppStream 2.0**

2. In the left pane, choose **Stacks**.
3. In the stack list, select the stack to customize with your branding.
4. Choose **Branding, Custom**.
5. For **Application catalog page**, customize how the streaming application catalog page appears to users after they sign in to AppStream 2.0.
   a. For **Organization logo**, do either of the following:
      - If you are uploading a logo for the first time, choose **Upload**, and then select the image to display in the header of the streaming application catalog page.
      - If you have already uploaded a logo and need to change it, choose **Change Logo**, and then select the image to display.
   b. For **Organization website links**, specify up to three website links to display in the page footer. For each link, choose the **Add Link** button, and then enter a display name and URL. To add more links, repeat these steps for each link to add. To remove a link, choose the **Remove** button under the link URL.
For **Color theme**, choose the colors to use for your website links, body text, and buttons, and as an accent for the page background. For information about each color theme, see Color Theme Palettes (p. 121) later in this topic.

6. For **Browser tab**, customize the page title and icon to display to users at the top of their browser tab during streaming sessions.
   a. For **Page title**, enter the title to display at the top of the browser tab.
   b. For **Favicon**, do either of the following:
      - If you are uploading a favicon for the first time, choose **Upload**, and then select the image to display at the top of the browser tab.
      - If you have already uploaded a favicon and want to change it, choose **Change Logo**, and then select the image to display.

7. Do either of the following:
   - To apply your branding changes, choose **Save**. When users connect to new streaming sessions that are launched for the stack, your branding changes are displayed.
     
     **Note**
     AppStream 2.0 retains the custom branding changes that you save. If you save your custom branding changes, but then choose to restore the AppStream 2.0 default branding, your custom branding changes are saved for later use. If you restore the AppStream 2.0 default branding and decide later to reapply your custom branding, choose **Custom, Save**. In this case, the most recently saved custom branding is displayed to your users.
   - To discard your branding changes, choose **Cancel**. When prompted to confirm your choice, choose **Confirm**. If you cancel your changes, the most recently saved branding is displayed to your users.

### Specifying a Custom Redirect URL and Feedback URL

You can specify a URL to which your users are redirected when they end their streaming session, as well as a URL where your users can submit feedback. By default, AppStream 2.0 displays a **Send Feedback** link that enables users to submit feedback to AWS about the quality of their application streaming session. To enable your users to submit feedback to a site that you specify, you can provide a custom feedback URL. You can specify the redirect URL and feedback URL when you create a new stack or edit the details for an existing stack. For more information, see Create a Stack (p. 84).

### Previewing Your Custom Branding Changes

You can preview how your branding changes will appear to your users by applying your branding changes to a test stack before you apply them to a production stack, and then creating a streaming URL for the test stack. After you validate your branding changes, you can then deploy them to your production stack. For information, see Step 2: Provide Access to Users (p. 8) in Getting Started with Amazon AppStream 2.0.

### Color Theme Palettes

When you choose a color theme, the colors for that theme are applied to the website links, text, and buttons in your streaming application catalog page. A color is also applied as an accent in the
background for your streaming application catalog page. For each color in a color theme palette, the hex value is also noted.

**Color Themes**
- Red (p. 122)
- Light Blue (p. 122)
- Blue (p. 123)
- Pink (p. 124)

**Red**
The following colors are applied when you select the red color theme.

- **White (#faf9f7)** – Used as a background accent.
- **Dark grey (#404040)** – Used for the body text and in the progress spinner.

When you choose the red color theme, the website links, body text, and background accent appear in your streaming application catalog page as follows.

**Light Blue**
The following colors are applied when you select the light blue color theme:

- **Light blue (#1d83c2)** – Used for buttons and website links.
- **White (#f6f6f6)** – Used as a background accent.
- **Dark grey (#333333)** – Used for the body text and in the progress spinner.
When you choose the light blue color theme, the website links, body text, and background accent appear in your streaming application catalog page as follows.

Blue

The following colors are applied when you select the blue color theme:

- **Blue (#0070ba)** – Used for website links.
- **White (#ffffff)** – Used as a background accent.
- **Light green (#8ac53e)** – Used for buttons.
- **Grey (#666666)** – Used for the body text and in the progress spinner.

When you choose the blue color theme, the website links, body text, and background accent appear in your streaming application catalog page as follows.
Pink

The following colors are applied when you select the pink color theme:

- **Pink (#ec0069)** – Used for website links.
- **White (#ffffff)** – Used as a background accent.
- **Blue (#3159a2)** – Used for buttons.
- **Dark grey (#333333)** – Used for the body text and in the progress spinner.

When you choose the pink color theme, the website links, body text, and background accent appear in your streaming application catalog page as follows.
Embed AppStream 2.0 Streaming Sessions

You can create a dynamic, interactive, and customized experience for your users by embedding an AppStream 2.0 streaming session within your website. Embedded AppStream 2.0 streaming sessions let your users interact with 3D models, maps, and datasets directly from your website. For example, users can view training instructions or educational materials alongside their AppStream 2.0 streaming session.

Contents
• Prerequisites (p. 125)
• Recommendations and Usage Considerations (p. 126)
• Step 1: Specify a Host Domain for Embedded AppStream 2.0 Streaming Sessions (p. 126)
• Step 2: Create a Streaming URL for User Authentication (p. 126)
• Step 3: Download the Embedded AppStream 2.0 Files (p. 128)
• Step 4: Configure Your Website for Integration with AppStream 2.0 (p. 128)
• Constants, Functions, and Events for Embedded AppStream 2.0 Streaming Sessions (p. 130)

Prerequisites

To embed an AppStream 2.0 streaming session in a website, you must have the following:

• A configured AppStream 2.0 environment that includes an AppStream 2.0 image, fleet, and stack. For information about how to create these resources, see the following topics in the AppStream 2.0 Administration Guide:
  • Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47)
  or Create Your AppStream 2.0 Image Programmatically by Using the Image Assistant CLI Operations (p. 57)
  • Create a Fleet (p. 82)
  • Create a Stack (p. 84)

• A streaming URL for user authentication. SAML 2.0 and AppStream 2.0 user pools are currently not supported as authentication methods for embedded AppStream 2.0 streaming sessions.

• Optionally, you can use custom domains for embedded AppStream 2.0 streaming sessions. You can use custom domains so that your own company URL displays for users rather than an AppStream 2.0 URL. Custom domains are required if your users have web browsers that block third-party cookies.

  Note
  You can configure custom domains by using Amazon CloudFront. For information, see Using Custom Domains with AppStream 2.0.

When you use a custom domain, you must:

• Create a streaming URL that uses the same domain.

• Add appstream-custom-url-domain to the header of the webpage that will host the embedded AppStream 2.0 streaming sessions. For the header value, use the domain that your reverse proxy displays to users. For more information, see Configuration Requirements for Using Custom Domains (p. 127).
Recommendations and Usage Considerations

Consider the following recommendations and usage notes for embedded AppStream 2.0 streaming sessions.

- To maintain maximum control over the embedded AppStream 2.0 streaming experience for your users, we recommend that you configure short-lived streaming URLs that last approximately 5 seconds. Any user can inspect the contents of a webpage and view its source. This includes the document object model (DOM) and the src (source) URL of the iframe. If the URL is still valid when a user copies it, that user can paste the URL in a separate browser tab and stream the session with the standard AppStream 2.0 portal user interface, without the embed options.
- Concurrent sessions are not supported when custom domains are used for embedded AppStream 2.0 streaming sessions. Concurrent sessions occur when users start two embedded AppStream 2.0 streaming sessions either on the same webpage or across two different browser tabs.

Step 1: Specify a Host Domain for Embedded AppStream 2.0 Streaming Sessions

To embed an AppStream 2.0 streaming session in a webpage, first update your stack to specify the domain to host the embedded streaming session. This a security measure to ensure that only authorized website domains can embed AppStream 2.0 streaming sessions. AppStream 2.0 adds the domain or domains that you specify to the `Content-Security-Policy` (CSP) header. For more information, see Content Security Policy (CSP) in the Mozilla MDN Web Docs documentation.

To update your stack to specify the domain to host the embedded streaming session, use any of the following methods:

- The AppStream 2.0 console
- The `EmbedHostDomains` API action
- The `embed-host-domains` AWS command line interface (AWS CLI) command

To specify a host domain by using the AppStream 2.0 console, perform the following steps.

2. In the left navigation pane, choose `Stacks`, and select the stack that you want.
3. Choose `Edit`.
4. Expand `Embed AppStream 2.0 (Optional)`.
5. In `Host Domains`, specify a valid domain. For example: `training.example.com`.
   
   **Note**
   Embedded streaming sessions are only supported over HTTPS [TCP port 443].
6. Choose `Update`.

Step 2: Create a Streaming URL for User Authentication

You must create a streaming URL to authenticate users for embedded AppStream 2.0 streaming sessions. SAML 2.0 and user pools are currently not supported for embedded streaming sessions. To create a streaming URL, use one of the following methods:
• AppStream 2.0 console
• The CreateStreamingURL API action
• The create-streaming-url AWS CLI command

Configuration Requirements for Using Custom Domains

Whether you use custom domains to apply your company branding or to ensure that embedded AppStream 2.0 streaming sessions work with browsers that block third-party cookies, the configuration requirements are the same.

Whether you use custom domains to apply your company branding or to ensure that embedded AppStream 2.0 streaming sessions work with browsers that block third-party cookies, the configuration requirements are the same.

For web browsers that block third-party cookies, custom domains are required. AppStream 2.0 uses browser cookies to authenticate streaming sessions and lets users reconnect to an active session without being prompted to provide their user name and password every time. By default, AppStream 2.0 streaming URLs include appstream.com as the domain. When you embed a streaming session within your website, appstream.com is treated as a third-party domain. As a result, streaming sessions may be blocked when modern browsers are used that block third-party cookies by default.

To avoid embedded AppStream 2.0 streaming sessions from being blocked in this scenario, follow these steps:

1. Specify a custom domain to host your embedded AppStream 2.0 streaming sessions.
   - When you configure your custom domain, make sure that the domain is a subdomain of the webpage in which you plan to embed AppStream 2.0. For example, if you update your stack to specify training.example.com as the host domain, you can create a subdomain called content.training.example.com for your embedded streaming sessions.

2. Create a streaming URL for embedded AppStream 2.0 streaming sessions that uses the same custom subdomain. To create the streaming URL, use the CreateStreamingURL API action or the create-streaming-url AWS CLI command. You cannot use the AppStream 2.0 console to create a streaming URL in this scenario.
   - To create a streaming URL for embedded AppStream 2.0 streaming sessions, in the URL, replace appstream2.region.aws.amazon.com with your own domain.
   - By default, AppStream 2.0 streaming URLs are formatted as follows:

   ```text
   https://appstream2.region.aws.amazon.com/authenticate?parameters=authenticationcode
   ```
   - If your subdomain is content.training.example.com, your new streaming URL follows this format:

   ```text
   https://content.training.example.com/authenticate?parameters=authenticationcode
   ```

   **Note**
   When you create a custom domain, you can use the domain for embedded AppStream 2.0 streaming sessions only in the AWS Region for which it was configured. If you plan to support custom domains in multiple Regions, create a custom domain for each applicable Region. Also, embedded streaming sessions are only supported over HTTPS [TCP port 443].

3. Add appstream-custom-url-domain to the header of the webpage that will host the embedded streaming sessions. For the header value, use the domain that your reverse proxy displays to users. For example:

   ```text
   Header name: appstream-custom-url-domain
   ```
Step 3: Download the Embedded AppStream 2.0 Files

To host embedded AppStream 2.0 streaming sessions, you must download and configure the provided AppStream 2.0 API JavaScript file.

1. On the Embedding AppStream 2.0 in Your Website webpage, choose the link in step 1 to download the AppStream 2.0 Embed Kit .zip file, appstream_embed_<version>.zip.
2. Navigate to the location where you downloaded the .zip file, and extract the contents of the file.
3. The extracted contents of the file comprise one folder, appstream-embed. In addition to the COPYRIGHT.txt and THIRD_PARTY_NOTICES.txt file, this folder contains the following two files:
   - appstream-embed.js — Provides the embedded AppStream 2.0 API. This JavaScript file includes the functions and API actions for configuring and controlling your embedded AppStream 2.0 streaming session.
   - embed-sample.html — Describes how to use the embedded AppStream 2.0 API to initialize a streaming session, call functions, and listen for events. This sample file expands on the information in this topic, to provide an example use case for developers.

Step 4. Configure Your Website for Integration with AppStream 2.0

The following sections provide information about how to configure your webpage to host embedded AppStream 2.0 streaming sessions.

Contents
- Import the appstream-embed JavaScript File (p. 128)
- Initialize and Configure the AppStream.Embed Interface Object (p. 129)
- Examples for Hiding Items in the AppStream 2.0 User Interface (p. 129)

Import the appstream-embed JavaScript File

1. On the webpage where you plan to embed the AppStream 2.0 streaming session, import the appstream-embed.js file into the webpage by adding the following code:

   ```html
   <script type="text/javascript" src="/appstream_embed.js"></script>
   ```

2. Next, create an empty container div. The ID of the div that you set is passed into the AppStream 2.0 embed constructor. It's then used to inject an iframe for the streaming session. To create the div, add the following code:

   ```html
   <div id="appstream-container"></div>
   ```
Initialize and Configure the AppStream.Embed Interface Object

To initialize the AppStream.Embed interface object in JavaScript, you must add code that creates an AppStream.Embed object with options for the streaming URL and user interface configuration. These options, and the div ID that you created, are stored in an object called `appstreamOptions`.

The following example code shows how to initialize the AppStream.Embed interface object.

```javascript
var appstreamOptions = {
};
appstreamEmbed = new AppStream.Embed("appstream-container", appstreamOptions);
```

In the code, replace `sessionURL` and `userInterfaceConfig` with your own values.

**Note**
The value specified for `userInterfaceConfig` hides the entire AppStream 2.0 toolbar. This value, which is included as an example, is optional.

**sessionUrl**
The streaming URL that you created by using the AppStream 2.0 console, the `CreateStreamingURL` API action, or the `create-streaming-url` AWS CLI command. This parameter is case-sensitive.

  **Type:** String  
  **Required:** Yes

**userInterfaceConfig**
The configuration that generates the initial state of the user interface elements. The configuration is a key-value pair.

The key, `AppStream.Embed.Options.HIDDEN_ELEMENTS`, specifies the user interface objects that are initially hidden when the embedded AppStream 2.0 streaming session is initialized. Later, you can return both hidden and visible objects by using the `getInterfaceState` parameter.

The value is an array of constants (toolbar buttons). For a list of constants that you can use, see [Working with HIDDEN_ELEMENTS](p. 131).

  **Type:** Map (key:value)  
  **Required:** No

Examples for Hiding Items in the AppStream 2.0 User Interface

The examples in this section show how to hide items in the AppStream 2.0 user interface from users during their embedded AppStream 2.0 streaming sessions.

**Examples**
- Example 1: Hide the entire AppStream 2.0 toolbar (p. 130)
- Example 2: Hide a specific button on the AppStream 2.0 toolbar (p. 130)
- Example 3: Hide multiple buttons on the AppStream 2.0 toolbar (p. 130)
Example 1: Hide the entire AppStream 2.0 toolbar

To prevent users from accessing any button on the AppStream 2.0 toolbar during embedded streaming sessions, use the `AppStream.Embed.Elements.TOOLBAR` constant. This constant lets you hide all AppStream 2.0 toolbar buttons.

```javascript
var appstreamOptions = {
    sessionURL: 'https://appstream2.region.aws.amazon.com/authenticate?
parameters=authenticationcode...',
    userInterfaceConfig:{[AppStream.Embed.Options.HIDDEN_ELEMENTS]:
};
```

Example 2: Hide a specific button on the AppStream 2.0 toolbar

You can display the AppStream 2.0 toolbar, while preventing users from accessing a specific toolbar button during embedded streaming sessions. To do so, specify the constant for the button that you want to hide. The following code uses the `AppStream.Embed.Elements.FILES_BUTTON` constant to hide the My Files button. This prevents users from accessing persistent storage options during embedded streaming sessions.

```javascript
var appstreamOptions = {
    sessionURL: 'https://appstream2.region.aws.amazon.com/authenticate?
parameters=authenticationcode...',
    userInterfaceConfig:{[AppStream.Embed.Options.HIDDEN_ELEMENTS]:
    [AppStream.Embed.Elements.FILES_BUTTON]}
};
```

Example 3: Hide multiple buttons on the AppStream 2.0 toolbar

You can display the AppStream 2.0 toolbar, while preventing users from accessing more than one toolbar button during embedded streaming sessions. To do so, specify the constants for the buttons that you want to hide. The following code uses the `AppStream.Embed.Elements.END_SESSION_BUTTON` and `AppStream.Embed.Elements.FULLSCREEN_BUTTON` constants to hide the End Session and Fullscreen buttons.

```
Note
Separate each constant with a comma, with no preceding or following space.
```

```javascript
var appstreamOptions = {
    sessionURL: 'https://appstream2.region.aws.amazon.com/authenticate?
parameters=authenticationcode... (https://appstream2.region.aws.amazon.com/#/)',
    userInterfaceConfig:{[AppStream.Embed.Options.HIDDEN_ELEMENTS]:
};
```

Constants, Functions, and Events for Embedded AppStream 2.0 Streaming Sessions

The following topics provide reference information for constants, functions, and events that you can use to configure embedded AppStream 2.0 streaming sessions.

Contents

- Working with HIDDEN_ELEMENTS (p. 131)
The following AppStream 2.0 user interface elements can be passed into the `HIDDEN_ELEMENTS` configuration option when an embedded AppStream 2.0 streaming session is initialized.

**Working with `HIDDEN_ELEMENTS`**

The following AppStream 2.0 user interface elements can be passed as constants into the `HIDDEN_ELEMENTS` configuration option when an embedded AppStream 2.0 streaming session is initialized.

<table>
<thead>
<tr>
<th>AppStream.Embed.Elements.TOOLBAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>AppStream.Embed.Elements.FULLSCREEN_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.END_SESSION_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.TOOLBAR</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.CATALOG_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.WINDOW_SWITCHER_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.FILES_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.CATALOG_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.WINDOW_SWITCHER_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.FILES_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.CATALOG_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.WINDOW_SWITCHER_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.FILES_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.CATALOG_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.WINDOW_SWITCHER_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.FILES_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.CATALOG_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.WINDOW_SWITCHER_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.FILES_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.CATALOG_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.WINDOW_SWITCHER_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.FILES_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.CATALOG_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.WINDOW_SWITCHER_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.FILES_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.CATALOG_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.WINDOW_SWITCHER_BUTTON</td>
</tr>
<tr>
<td>AppStream.Embed.Elements.FILES_BUTTON</td>
</tr>
</tbody>
</table>

The following three elements can be passed as strings into `HIDDEN_ELEMENTS`, rather than as constants.

<table>
<thead>
<tr>
<th>String</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>'adminCommandsButton'</td>
<td>When you are connected to an AppStream 2.0 image builder, the <strong>Admin Commands</strong> button displays on the top right corner of the session toolbar. Passing this string into <code>HIDDEN_ELEMENTS</code> hides the <strong>Admin Commands</strong> button.</td>
</tr>
<tr>
<td>'softKeyboardButton'</td>
<td>During AppStream 2.0 streaming sessions on touch-enabled devices, users can tap the keyboard icon on the AppStream 2.0 toolbar to display the on-screen keyboard. Passing this string into <code>HIDDEN_ELEMENTS</code> hides the keyboard icon.</td>
</tr>
<tr>
<td>'keyboardShortcutsButton'</td>
<td>During AppStream 2.0 streaming sessions on touch-enabled devices, users can tap the Fn icon to display keyboard shortcuts. Passing this string into <code>HIDDEN_ELEMENTS</code> hides the Fn icon.</td>
</tr>
</tbody>
</table>

**Functions for the `AppStream.Embed` Object**

The following table lists the functions that can be performed on the `AppStream.Embed` object.
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AppStream.Embed(containerId:string, options:object)</td>
<td>The AppStream.Embed object constructor. This constructor initializes and communicates with the AppStream.Embed object, and it uses a div container ID. The ID is used to inject the iframe. It also injects an object that includes the configuration options for appstreamOptions (sessionURL and HIDDEN_ELEMENTS).</td>
</tr>
<tr>
<td>endSession()</td>
<td>This function ends the streaming session, but does not destroy the iframe. If you specify a redirect URL, the iframe attempts to load the URL. Depending on the CORS headers of the page, the URL may not load.</td>
</tr>
<tr>
<td>launchApp(appId:string)</td>
<td>This function programmatically launches an application with the application ID that was specified during image creation.</td>
</tr>
<tr>
<td>launchAppSwitcher()</td>
<td>This function sends the AppSwitcher command to the AppStream 2.0 portal. This triggers the AppSwitcher command on the instance.</td>
</tr>
<tr>
<td>getSessionState()</td>
<td>This function returns an object for sessionStatus. For more information, see Events for Embedded AppStream 2.0 Streaming Sessions (p. 132).</td>
</tr>
<tr>
<td>getUserInterfaceState()</td>
<td>This function returns an object for UserInterfaceState. The object contains the key-value pairs for the following:</td>
</tr>
<tr>
<td></td>
<td>sessionStatus: State enumeration</td>
</tr>
<tr>
<td></td>
<td>sessionTerminationReason: String</td>
</tr>
<tr>
<td></td>
<td>sessionDisconnectionReason: String</td>
</tr>
<tr>
<td></td>
<td>For more information, see Events for Embedded AppStream 2.0 Streaming Sessions (p. 132).</td>
</tr>
<tr>
<td>addEventListener(name, callback)</td>
<td>This function adds a callback function to call when the specified event is triggered. For a list of the events that can be triggered, see Events for Embedded AppStream 2.0 Streaming Sessions (p. 132).</td>
</tr>
<tr>
<td>removeEventListener(name, callback)</td>
<td>This function removes the callback for the specified events.</td>
</tr>
<tr>
<td>destroy()</td>
<td>This function deletes the iframe and cleans up resources. This function does not affect streaming sessions that are in progress.</td>
</tr>
</tbody>
</table>

**Events for Embedded AppStream 2.0 Streaming Sessions**

The following table lists the events that can be triggered during embedded AppStream 2.0 streaming sessions.
### Event Data Description

<table>
<thead>
<tr>
<th>Event</th>
<th>Data</th>
<th>Description</th>
</tr>
</thead>
</table>
| AppStream.Embed.Events.SESSION_STATE_CHANGE | sessionTerminationReason: String  
    sessionDisconnectionReason: String | This event is triggered when any session state change occurs. The event includes a map of the states that changed. To retrieve the full session state, use the `getSessionState()` function. Following are the session states:  
    - AppStream.Embed.SessionStatus.Unknown — The session has not started and is not reserved  
    - AppStream.Embed.SessionStatus.Reserved — The session is reserved but has not started  
    - AppStream.Embed.SessionStatus.Started — The user connected to the session and started streaming  
    - AppStream.Embed.SessionStatusDisconnected — The user disconnected from the session  
    - AppStream.Embed.SessionStatus.Ended — The session was marked as ended or expired |
| AppStream.Embed.Events.SESSION_INTERFACE_STATE_CHANGE | isFullscreen: Boolean  
    isSoftKeyboardVisible: Boolean | This event is triggered when any session state change occurs. The |
Examples for Adding Event Listeners and Ending an Embedded AppStream 2.0 Streaming Session

The examples in this section show how to do the following:

- Add event listeners for embedded AppStream 2.0 streaming sessions.
- Programmatically end an embedded AppStream 2.0 streaming session.

**Example 1: Add event listeners for embedded AppStream 2.0 streaming sessions**

To add event listeners for session state changes, session interface state changes, and session errors during embedded streaming sessions, use the following code:

```javascript
appstreamEmbed.addEventListener(AppStream.Embed.Events.SESSION_STATE_CHANGE, updateSessionStateCallback);
appstreamEmbed.addEventListener(AppStream.Embed.Events.SESSION_INTERFACE_STATE_CHANGE, updateUserInterfaceStateCallback);
appstreamEmbed.addEventListener(AppStream.Embed.Events.SESSION_ERROR, errorCallback);
```


The `updateSessionStateCallback`, `updateUserInterfaceStateCallback`, and `errorCallback` functions are ones that you implement. These functions are passed into the `addEventListener` function and called when an event is triggered.

**Example 2: Programmatically end an embedded AppStream 2.0 streaming session**

To end an embedded AppStream 2.0 streaming sessions, use the following function:
appstreamEmbed.endSession();
Enable and Administer Persistent Storage for Your AppStream 2.0 Users

Amazon AppStream 2.0 supports the following persistent storage options for users in your organization:

- Home folders
- Google Drive for G Suite
- OneDrive for Business

You can enable one or more options for your organization. As an AppStream 2.0 administrator, you must understand how to perform the following tasks to enable and administer persistent storage for your users.

Contents

- Enable and Administer Home Folders for Your AppStream 2.0 Users (p. 136)
- Enable and Administer Google Drive for Your AppStream 2.0 Users (p. 140)
- Enable and Administer OneDrive for Business for Your AppStream 2.0 Users (p. 143)

Enable and Administer Home Folders for Your AppStream 2.0 Users

AppStream 2.0 supports the following persistent storage options for users in your organization:

- Home folders
- Google Drive for G Suite
- OneDrive for Business

You can enable one or more options for your organization. When you enable home folders for an AppStream 2.0 stack, users of the stack can access a persistent storage folder during their application streaming sessions. No further configuration is required for your users to access their home folder. Data stored by users in their home folder is automatically backed up to an Amazon Simple Storage Service bucket in your AWS account and is made available to those users in subsequent sessions.

Files and folders are encrypted in transit using Amazon S3’s SSL endpoints. Files and folders are encrypted at rest using Amazon S3-managed encryption keys.

Note
Home folders are stored on fleet instances in the following default locations:

- Non-domain-joined instances: C:\Users\PhotonUser\My Files\Home Folder
- Domain-joined instances: C:\Users\%username%\My Files\Home Folder

As an administrator, use the applicable path if you configure your applications to save to the home folder. In some cases, your users may not be able to find their home folder because some
applications do not recognize the redirect that displays the home folder as a top-level folder in File Explorer. If this is the case, your users can access their home folder by browsing to the same directory in File Explorer.

Contents
- Enable Home Folders for Your AppStream 2.0 Users (p. 137)
- Administer Your Home Folders (p. 137)

Enable Home Folders for Your AppStream 2.0 Users

Before enabling home folders, you must do the following:

- Check that you have the correct AWS Identity and Access Management (IAM) permissions for Amazon S3 actions. For more information, see Using IAM Policies to Manage Administrator Access to the Amazon S3 Bucket for Home Folders and Application Settings Persistence (p. 230).
- Use an image that was created from an AWS base image released on or after May 18, 2017. For a current list of released AWS images, see AppStream 2.0 Base Image Version History (p. 29).
- Enable network connectivity to Amazon S3 from your virtual private cloud (VPC) by configuring internet access or a VPC endpoint for Amazon S3. For more information, see Networking and Access for Amazon AppStream 2.0 (p. 10) and Using Amazon S3 VPC Endpoints for Home Folders and Application Settings Persistence (p. 21).

You can enable or disable home folders while creating a stack (see Create a Stack (p. 84)), or after the stack is created by using the AWS Management Console for AppStream 2.0, AWS SDK, or AWS CLI. For each AWS Region, home folders are backed up by an Amazon S3 bucket.

The first time you enable home folders for an AppStream 2.0 stack in an AWS Region, the service creates an Amazon S3 bucket in your account in that same Region. The same bucket is used to store the content of home folders for all users and all stacks in that Region. For more information, see Amazon S3 Bucket Storage (p. 138).

Note
For guidance that you can provide your users to help them get started with using home folders during AppStream 2.0 streaming sessions, see Use Home Folders (p. 278).

To enable home folders while creating a stack

- Follow the steps in Create a Stack (p. 84), and make sure that Enable Home Folders is selected.

To enable home folders for an existing stack

2. In the left navigation pane, choose Stacks, and select the stack for which to enable home folders.
3. Below the stacks list, choose Storage and select Enable Home Folders.
4. In the Enable Home Folders dialog box, choose Enable.

Administer Your Home Folders

Contents
- Disable Home Folders (p. 138)
- Amazon S3 Bucket Storage (p. 138)
- Home Folder Formats (p. 139)
Disable Home Folders

You can disable home folders for a stack without losing user content already stored in home folders. Disabling home folders for a stack has the following effects:

- Users who are connected to active streaming sessions for the stack receive an error message. They are informed that they can no longer store content in their home folder.
- Home folders do not appear for any new sessions that use the stack with home folders disabled.
- Disabling home folders for one stack does not disable it for other stacks.
- Even if home folders are disabled for all stacks, AppStream 2.0 does not delete the user content.

To restore access to home folders for the stack, enable home folders again by following the steps described earlier in this topic.

To disable home folders while creating a stack

- Follow the steps in Create a Stack (p. 84) and make sure that the Enable Home Folders option is cleared.

To disable home folders for an existing stack

2. In the left navigation pane, choose Stacks, and select the stack.
3. Below the stacks list, choose Storage and clear Enable Home Folders.
4. In the Disable Home Folders dialog box, type CONFIRM (case-sensitive) to confirm your choice, then choose Disable.

Amazon S3 Bucket Storage

AppStream 2.0 manages user content stored in home folders by using Amazon S3 buckets created in your account. For every AWS Region, AppStream 2.0 creates a bucket in your account. All user content generated from streaming sessions of stacks in that Region is stored in that bucket. The buckets are fully managed by the service without any input or configuration from an administrator. The buckets are named in a specific format as follows:

```
apppstream2-36fb080bb8-region-code-account-id-without-hyphens
```

Where `region-code` is the AWS Region code in which the stack is created and `account-id-without-hyphens` is your AWS account ID. The first part of the bucket name, `appstream2-36fb080bb8-`, does not change across accounts or Regions.

For example, if you enable home folders for stacks in the US West (Oregon) Region (us-west-2) on account number 123456789012, the service creates an Amazon S3 bucket in that Region with the name shown. Only an administrator with sufficient permissions can delete this bucket.

```
apppstream2-36fb080bb8-us-west-2-123456789012
```

As mentioned earlier, disabling home folders for stacks does not delete any user content stored in the Amazon S3 bucket. To permanently delete user content, an administrator with adequate access must do so from the Amazon S3 console. AppStream 2.0 adds a bucket policy that prevents accidental deletion of
the bucket. For more information, see Using IAM Policies to Manage Administrator Access to the Amazon S3 Bucket for Home Folders and Application Settings Persistence (p. 230).

**Additional Resources**

For more information about managing Amazon S3 buckets and best practices, see the following topics in the Amazon Simple Storage Service Developer Guide:

- You can provide offline access to user data for your users with Amazon S3 policies. For more information, see Allow Users to Access a Personal "Home Directory" in Amazon S3.
- You can enable file versioning for content stored in Amazon S3 buckets used by AppStream 2.0. For more information, see Using Versioning.

**Home Folder Formats**

When home folders are enabled, each user is provided with one unique folder in which to store their content. The folder is created and maintained as a unique Amazon S3 object within the bucket for that Region. The hierarchy of a user folder depends on how the user launches a streaming session, as described in the following sections.

**AWS SDKs and AWS CLI**

For sessions launched using CreateStreamingURL or create-streaming-url the user folder structure is as follows:

```
bucket-name/user/custom/user-id-SHA-256-hash/
```

Where `bucket-name` is in the format shown in Amazon S3 Bucket Storage (p. 138) and `user-id-SHA-256-hash` is the user-specific folder name created using a lowercase SHA-256 hash hexadecimal string generated from the `UserId` value passed to the CreateStreamingURL API operation or create-streaming-url command. For more information, see CreateStreamingURL in the Amazon AppStream 2.0 API Reference and create-streaming-url in the AWS CLI Command Reference.

The following example folder structure applies to session access using the API or AWS CLI with a `UserId` testuser@mydomain.com, account id 123456789012 in the US West (Oregon) Region (us-west-2):

```
appstream2-36fb080bb8-us-west-2-123456789012/user/custom/
a0bcb1da1f480d9b5b3e90f91243143eac04cfccfbdc777e740fab628a1cd13/
```

You can identify the folder for a user by generating the lowercase SHA-256 hash value of the `UserId` using websites or open source coding libraries available online.

**SAML**

For sessions created using SAML federation, the user folder structure is as follows:

```
bucket-name/user/federated/user-id-SHA-256-hash/
```

In this case, `user-id-SHA-256-hash` is the folder name created using a lowercase SHA-256 hash hexadecimal string generated from the NameID SAML attribute value passed in the SAML federation request. To differentiate users who have the same name but belong to two different domains, send the SAML request with NameID in the format domainname\username. For more information, see Single Sign-on Access (SAML 2.0) (p. 243).

The following example folder structure applies to session access using SAML federation with NameID SAMPLEDOMAIN\testuser, account ID 123456789012 in the US West (Oregon) Region:
When part or all of the NameID string is capitalized (as the domain name `SAMPLEDOMAIN` is in the example), AppStream 2.0 generates the hash value based on the capitalization used in the string. Using this example, the hash value for `SAMPLEDOMAIN\testuser` is `8DD9A642F511609454D344D53CB861A71190E44FED2B8AF9FDE0C507012A9901`. In the folder for that user, this value is displayed in lowercase, as follows: `8dd9a642f511609454d344d53cb861a71190e44fed2b8af9fde0c507012a9901`.

You can identify the folder for a user by generating the SHA-256 hash value of the NameID using websites or open source coding libraries available online.

**Using the AWS Command Line Interface or AWS SDKs**

You can enable and disable home folders for a stack by using the AWS CLI or AWS SDKs.

Use the following `create-stack` command to enable home folders while creating a new stack:

```
aws appstream create-stack --name ExampleStack --storage-connectors type=HOMEFOLDERS
```

Use the following `update-stack` command to enable home folders for an existing stack:

```
aws appstream update-stack --name ExistingStack --storage-connectors type=HOMEFOLDERS
```

Use the following command to disable home folders for an existing stack. This command does not delete any user data.

```
aws appstream update-stack --name ExistingStack --delete-storage-connectors
```

---

**Enable and Administer Google Drive for Your AppStream 2.0 Users**

Amazon AppStream 2.0 supports the following persistent storage options for users in your organization:

- Google Drive for G Suite
- OneDrive for Business
- Home folders

You can enable one or more options for your organization. When you enable Google Drive for G Suite for an AppStream 2.0 stack, users of the stack can link their Google Drive for G Suite account to AppStream 2.0. Then they can sign into their Google Drive for G Suite account and access their Google Drive folder during application streaming sessions. Any changes that they make to files or folders in Google Drive during those sessions are automatically backed up and synchronized, so that they are available to users outside of their streaming sessions.

**Important**

You can enable Google Drive for G Suite for accounts in your G Suite domains only, but not for personal Gmail accounts.

**Contents**

- Enable Google Drive for Your AppStream 2.0 Users (p. 141)
Enable Google Drive for Your AppStream 2.0 Users

Before enabling Google Drive, you must do the following:

- Have an active G Suite account with a valid organizational domain and user accounts in the domain to use with AppStream 2.0.
- Configure an AppStream 2.0 stack with an associated fleet.

The fleet must use an image that uses a version of the AppStream 2.0 agent released on or after May 31, 2018. For more information, see AppStream 2.0 Agent Version History (p. 39). The fleet must also have access to the internet.
- Add Amazon AppStream 2.0 as a trusted app in one or more domains associated with your G Suite account. You can enable Google Drive for up to 10 domains.

Follow these steps to add Amazon AppStream 2.0 as a trusted app in your G Suite domains.

To add Amazon AppStream 2.0 as a trusted app in your G Suite domains

2. Choose Dashboard.
3. Choose the main menu in the upper left of the window (to the left of the Google Admin title), then choose Security, Settings.
4. Choose API Permissions.
5. At the bottom of the API Access list, choose the Trusted Apps link.
6. Choose the Whitelist an App [plus sign (+) icon] in the bottom right of the window.
7. In the Add APP to Trusted List dialog box, do the following. When you're done, choose Add:
   - For Select App Type, choose Web Application.
   - For OAuth2 Client ID, type the Amazon AppStream 2.0 OAuth client ID for your AWS Region. For a list of client IDs, see the table that follows this procedure.
8. Confirm that Amazon AppStream 2.0 appears in the list of trusted apps.

Amazon AppStream 2.0 OAuth2 client IDs

<table>
<thead>
<tr>
<th>Region</th>
<th>Amazon AppStream 2.0 OAuth client ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>US East (N. Virginia)</td>
<td>266080779488-15n5q5nkiclp6m524qibnwmhbsg0hk92.apps.googleusercontent.com</td>
</tr>
<tr>
<td>US West (Oregon)</td>
<td>1026466167591-i4jmemrggsjomp9tnkkcs5tmiggfiuujb.apps.googleusercontent.com</td>
</tr>
<tr>
<td>Asia Pacific (Seoul)</td>
<td>562383781419-aml2dnvt050tmdltsvr36i81js40dj.apps.googleusercontent.com</td>
</tr>
<tr>
<td>Asia Pacific (Singapore)</td>
<td>856871139998-4elia2nldb5j6gtv4c1rdtel1fhlgec8vs.apps.googleusercontent.com</td>
</tr>
<tr>
<td>Asia Pacific (Sydney)</td>
<td>151535156524-b889372osskprm4dtclpm53mo3m9omp.apps.googleusercontent.com</td>
</tr>
<tr>
<td>Asia Pacific (Tokyo)</td>
<td>922579247628-qpl19kplgh3hu5du2lpbjs4qbg6jpm2.apps.googleusercontent.com</td>
</tr>
</tbody>
</table>
Region | Amazon AppStream 2.0 OAuth client ID
---|---
Europe (Frankfurt) | 643727794574-1se5360a77184je9j3ap12obov1lib76q.apps.googleusercontent.com
Europe (Ireland) | 599492309098-098muc7ojfjfo9vua5rm5u9q2k3mlok3j.apps.googleusercontent.com
AWS GovCloud (US-West) | 96065833880-ltfkb2vfd7c65nt7s24r7t8le5bc9bl.apps.googleusercontent.com

Note
For more information about using AppStream 2.0 in AWS GovCloud (US-West), see Amazon AppStream 2.0 in the AWS GovCloud (US) User Guide.

Follow these steps to enable Google Drive for your AppStream 2.0 users.

**To enable Google Drive while creating a stack**
- Follow the steps in Create a Stack (p. 84), make sure that Enable Google Drive is selected, and that you have specified at least one organizational domain associated with your G Suite account.

**To enable Google Drive for an existing stack**
2. In the left navigation pane, choose Stacks, and select the stack for which to enable Google Drive.
3. Below the stacks list, choose Storage and select Enable Google Drive for G Suite.
4. In the Enable Google Drive for G Suite dialog box, in G Suite domain name, type the name of at least one organizational domain that is associated with your G Suite account. To specify another domain, choose Add another domain, and type the name of the domain.
5. After you add domain names, choose Enable.

Note
For guidance that you can provide your users to help them get started with using Google Drive during AppStream 2.0 streaming sessions, see Use Google Drive (p. 279).

**Disable Google Drive for Your AppStream 2.0 Users**

You can disable Google Drive for a stack without losing user content that is already stored on Google Drive. Disabling Google Drive for a stack has the following effects:
- Users who are connected to active streaming sessions for the stack receive an error message. They are informed that they do not have permissions to access their Google Drive.
- Any new sessions that use the stack with Google Drive disabled do not display Google Drive.
- Only the specific stack for which Google Drive is disabled is affected.
- Even if Google Drive is disabled for all stacks, AppStream 2.0 does not delete the user content stored in their Google Drive.

Follow these steps to disable Google Drive for an existing stack.

**To disable Google Drive for an existing stack**
2. In the left navigation pane, choose Stacks, and select the stack for which to disable Google Drive.
3. Below the stacks list, choose Storage, and clear the Enable Google Drive for G Suite option.
4. In the Disable Google Drive for G Suite dialog box, type CONFIRM (case-sensitive) to confirm your choice, then choose Disable.

When users of the stack start their next AppStream 2.0 streaming session, they can no longer access their Google Drive folder from within that session and future sessions.

Enable and Administer OneDrive for Business for Your AppStream 2.0 Users

AppStream 2.0 supports the following persistent storage options for users in your organization:

- OneDrive for Business
- Google Drive for G Suite
- Home folders

You can enable one or more options for your organization. When you enable OneDrive for Business for an AppStream 2.0 stack, users of the stack can link their OneDrive for Business account to AppStream 2.0. Then they can sign into their OneDrive for Business account and access their OneDrive folder during application streaming sessions. Any changes that they make to files or folders in OneDrive during those sessions are automatically backed up and synchronized, so that they are available to users outside of their streaming sessions.

**Important**
You can enable OneDrive for Business for accounts in your OneDrive domains only, but not for personal accounts. AppStream 2.0 requires that you configure your Microsoft Azure Active Directory environment to allow end-user consent to applications. For more information, see Configure how end-users consent to applications in the Azure Active Directory Application management documentation.

The admin consent workflow lets administrators grant access to applications that require administrator approval. If the admin consent workflow is configured in your Azure Active Directory environment, contact AWS Support. For information about how to contact AWS Support, see AWS Support Center.

**Contents**

- Enable OneDrive for Your AppStream 2.0 Users (p. 143)
- Disable OneDrive for Your AppStream 2.0 Users (p. 144)

Enable OneDrive for Your AppStream 2.0 Users

Before enabling OneDrive, you must do the following:

- Have an active Microsoft Office 365 or OneDrive for Business account with a valid organizational domain and user accounts in the domain to use with AppStream 2.0.
- Configure an AppStream 2.0 stack with an associated fleet.

The fleet must use an image that uses a version of the AppStream 2.0 agent released on or after July 26, 2018. For more information, see AppStream 2.0 Agent Version History (p. 39). The fleet must also have access to the internet.

Follow these steps to enable OneDrive for your AppStream 2.0 users.
To enable OneDrive while creating a stack

- Follow the steps in Create a Stack (p. 84), make sure that Enable OneDrive is selected, and that you have specified at least one organizational domain that is associated with your OneDrive for Business account.

To enable OneDrive for an existing stack

2. In the left navigation pane, choose Stacks, and select the stack for which to enable OneDrive.
3. Below the stacks list, choose Storage, and select Enable OneDrive for Business.
4. In the Enable OneDrive for Business dialog box, in OneDrive domain name, type the name of at least one organizational domain that is associated with your OneDrive account. To specify another domain, choose Add another domain, and type the name of the domain.
5. After you add OneDrive domain names, choose Enable.

Before your users can use OneDrive with AppStream 2.0, you must provide them with permissions to link their OneDrive account with third-party web applications. To do so, follow the steps in the next section.

Important
You must configure your Microsoft Azure Active Directory environment to allow end-user consent to applications. For more information, see Configure how end-users consent to applications in the Azure Active Directory Application management documentation.

Provide Your Users with Permissions to Link OneDrive with AppStream 2.0

You must enable Integrated Apps in your Office 365 or OneDrive for Business admin console before users can link their OneDrive for Business account to AppStream 2.0.

1. Sign in to Office 365 or the OneDrive for Business admin console.
2. In the left navigation pane of the console, choose Settings, Services & add-ins.
3. From the list of services and add-ins, choose Integrated Apps.
4. On the Integrated apps page, turn on the option to allow users in your organization to let third party web apps access their Office 365 information.

Note
For guidance that you can provide your users to help them get started with using OneDrive during AppStream 2.0 streaming sessions, see Use OneDrive for Business (p. 281).

Disable OneDrive for Your AppStream 2.0 Users

You can disable OneDrive for a stack without losing user content that is already stored on OneDrive. Disabling OneDrive for a stack has the following effects:

- Users who are connected to active streaming sessions for the stack receive an error message. They are informed that they do not have permissions to access their OneDrive.
- Any new sessions that use the stack with OneDrive disabled do not display OneDrive.
- Only the specific stack for which OneDrive is disabled is affected.
- Even if OneDrive is disabled for all stacks, AppStream 2.0 does not delete the user content stored in their OneDrive.

Follow these steps to disable OneDrive for an existing stack.
To disable OneDrive for an existing stack

2. In the left navigation pane, choose Stacks, and select the stack for which to disable OneDrive.
3. Below the stacks list, choose Storage, and clear Enable OneDrive for Business option.
4. In the Disable OneDrive for Business dialog box, type CONFIRM (case-sensitive) to confirm your choice, then choose Disable.

When users of the stack start their next AppStream 2.0 streaming session, they can no longer access their OneDrive folder from within that session and future sessions.
Enable Application Settings Persistence for Your AppStream 2.0 Users

AppStream 2.0 supports persistent application settings. This means that your users' application customizations and Windows settings are automatically saved after each streaming session and applied during the next session. Examples of persistent application settings that your users can configure include, but are not limited to, browser favorites, settings, webpage sessions, application connection profiles, plugins, and UI customizations. These settings are saved to an Amazon Simple Storage Service (Amazon S3) bucket in your account, within the AWS Region in which application settings persistence is enabled. They are available in each AppStream 2.0 streaming session.

Note
Standard Amazon S3 charges may apply to data that is stored in your S3 bucket. For more information, see Amazon S3 Pricing.

Contents
- How Application Settings Persistence Works (p. 146)
- Enabling Application Settings Persistence (p. 147)
- Administer the VHDs for Your Users' Application Settings (p. 149)

How Application Settings Persistence Works

Persistent application settings are saved to a Virtual Hard Disk (VHD) file. This file is created the first time a user streams an application from a stack on which application settings persistence is enabled. If the fleet associated with the stack is based on an image that contains default application and Windows settings, the default settings are used for the user's first streaming session. For more information about default settings, see Step 4: Create Default Application and Windows Settings in Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47).

When the streaming session ends, the VHD is unmounted and uploaded to an Amazon S3 bucket within your account. The bucket is created when you enable persistent application settings for the first time for a stack in an AWS Region. The bucket is unique to your AWS account and the Region. The VHD is encrypted in transit using Amazon S3 SSL endpoints, and at rest using Amazon S3-managed encryption keys.

The VHD is mounted to the streaming instance in both C:\Users\%username% and D:\%username%. If your instance is not joined to an Active Directory domain, the Windows user name is PhotonUser. If your instance is joined to an Active Directory domain, the Windows user name is that of the logged in user.

Application settings persistence does not work across different operating system versions. For example, if you enable application settings persistence on a stack and the stack is associated with a fleet that uses a Windows Server 2012 R2 image, if you update the fleet to use an image that runs a different operating system (such as Windows Server 2016), settings from previous streaming sessions are not saved for users of the stack. Instead, after you update the fleet to use the new image, when users launch a streaming session from a fleet instance, a new Windows user profile is created. However, if you apply an update to the same operating system on the image, users' customizations and settings from previous streaming sessions are saved. When updates to the same operating system are applied to an image, the same Windows user profile is used when users launch a streaming session from the fleet instance.
Important
AppStream 2.0 supports applications that rely on the Microsoft Data Protection API only when the streaming instance is joined to a Microsoft Active Directory domain. In cases where a streaming instance is not joined to an Active Directory domain, the Windows user, PhotonUser, is different on each fleet instance. Due to the way in which the DPAPI security model works, users' passwords don't persist for applications that use DPAPI in this scenario. In cases where streaming instances are joined to an Active Directory domain and the user is a domain user, the Windows user name is that of the logged in user, and users' passwords persist for applications that use DPAPI.

AppStream 2.0 automatically saves all files and folders in this path, except for the following folders:

- Contacts
- Desktop
- Documents
- Downloads
- Links
- Pictures
- Saved Games
- Searches
- Videos

Files and folders created outside of these folders are saved within the VHD and synced to Amazon S3. The default VHD maximum size is 1GB. The size of the saved VHD is the total size of the files and folders that it contains. AppStream 2.0 automatically saves the HKEY_CURRENT_USER registry hive for the user.

Note
The entire VHD must be downloaded to the streaming instance before a streaming session can begin. For this reason, a VHD that contains a large amount of data can delay the start of the streaming session. For more information, see Best Practices for Enabling Application Settings Persistence (p. 148).

When you enable application settings persistence, you must specify a settings group. The settings group determines which saved application settings are used for a streaming session from this stack. AppStream 2.0 creates a new VHD file for the settings group that is stored separately within the S3 bucket in your AWS account. If the settings group is shared between stacks, the same application settings are used in each stack. If a stack requires its own application settings, specify a unique settings group for the stack.

Enabling Application Settings Persistence

Contents
- Prerequisites for Enabling Application Settings Persistence (p. 147)
- Best Practices for Enabling Application Settings Persistence (p. 148)
- How to Enable Application Settings Persistence (p. 148)

Prerequisites for Enabling Application Settings Persistence

To enable application settings persistence, you must first do the following:
Best Practices for Application Settings Persistence

• Check that you have the correct AWS Identity and Access Management (IAM) permissions for Amazon S3 actions. For more information, see the IAM Policies and the Amazon S3 Bucket for Home Folders section in Identity and Access Management for Amazon AppStream 2.0 (p. 224).
• Use an image that was created from a base image published by AWS on or after December 7, 2017. For a current list of released AWS base images, see AppStream 2.0 Base Image Version History (p. 29).
• Associate the stack on which you plan to enable this feature with a fleet based on an image that uses a version of the AppStream 2.0 agent released on or after August 29, 2018. For more information, see AppStream 2.0 Agent Version History (p. 39).
• Enable network connectivity to Amazon S3 from your virtual private cloud (VPC) by configuring internet access or a VPC endpoint for Amazon S3. For more information, see the Home Folders and VPC Endpoints section in Networking and Access for Amazon AppStream 2.0 (p. 10).

Best Practices for Enabling Application Settings Persistence

To enable application settings persistence without providing internet access to your instances, use a VPC endpoint. This endpoint must be in the VPC to which your AppStream 2.0 instances are connected. You must attach a custom policy to enable AppStream 2.0 access to the endpoint. For information about how to create the custom policy, see the Home Folders and VPC Endpoints section in Networking and Access for Amazon AppStream 2.0 (p. 10). For more information about private Amazon S3 endpoints, see VPC Endpoints and Endpoints for Amazon S3 in the Amazon VPC User Guide.

How to Enable Application Settings Persistence

You can enable or disable application settings persistence while creating a stack or after the stack is created by using the AppStream 2.0 console, AppStream 2.0 API, an AWS SDK, or the AWS Command Line Interface (CLI). For each AWS Region, persistent application settings are stored in an S3 bucket in your account.

The first time you enable application settings persistence for a stack in an AWS Region, AppStream 2.0 creates an S3 bucket in your AWS account in the same Region. The same bucket stores the application settings VHD file for all users and all stacks in that AWS Region. For more information, see Amazon S3 Bucket Storage in Administer the VHDs for Your Users' Application Settings (p. 149).

To enable application settings persistence while creating a stack

• Follow the steps in Create a Stack (p. 84), and make sure that Enable Application Settings Persistence is selected.

To enable application settings persistence for an existing stack

2. In the left navigation pane, choose Stacks, and select the stack for which to enable application settings persistence.
3. Below the stacks list, choose User Settings, Application Settings Persistence, Edit.
4. In the Application Settings Persistence dialog box, choose Enable Application Settings Persistence.
5. Confirm the current settings group or type the name of a new settings group. When you're done, choose Update.

New streaming sessions now have application settings persistence enabled.
Administer the VHDs for Your Users' Application Settings

Contents

- Amazon S3 Bucket Storage (p. 149)
- Reset a User's Application Settings (p. 150)
- Enable Amazon S3 Object Versioning and Revert a User's Application Settings (p. 151)
- Increase the Size of the Application Settings VHD (p. 151)

Amazon S3 Bucket Storage

When you enable application settings persistence, your users' application customizations and Windows settings are automatically saved to a Virtual Hard Disk (VHD) file that is stored in an Amazon S3 bucket created in your AWS account. For every AWS Region, AppStream 2.0 creates a bucket in your account that is unique to your account and the Region. All application settings configured by your users are stored in the bucket for that Region.

You do not need to perform any configuration tasks to manage these S3 buckets; they are fully managed by the AppStream 2.0 service. The VHD file that is stored in each bucket is encrypted in transit using Amazon S3's SSL endpoints and at rest using Amazon S3-managed encryption keys. The buckets are named in a specific format as follows:

```
appstream-app-settings-region-code-account-id-without-hyphens-random-identifier
```

- **region-code**
  This is the AWS Region code in which the stack is created with application settings persistence.

- **account-id-without-hyphens**
  Your AWS account ID. The random identifier ensures there is no conflict with other buckets in that Region. The first part of the bucket name, `appstream-app-settings`, does not change across accounts or Regions.

For example, if you enable application settings persistence for stacks in the US West (Oregon) Region (us-west-2) on account number 123456789012, AppStream 2.0 creates an Amazon S3 bucket within your account in that Region with the name shown. Only an administrator with sufficient permissions can delete this bucket.

```
appstream-app-settings-us-west-2-1234567890123-abcdefg
```

Disabling application settings persistence does not delete any VHDs stored in the S3 bucket. To permanently delete settings VHDs, you or another administrator with adequate permissions must do so by using the Amazon S3 console or API. AppStream 2.0 adds a bucket policy that prevents accidental deletion of the bucket. For more information, see IAM Policies and the Amazon S3 Bucket for Application Settings Persistence in Identity and Access Management for Amazon AppStream 2.0 (p. 224).

When application settings persistence is enabled, a unique folder is created for each settings group to store the settings VHD. The hierarchy of the folder in the S3 bucket depends on how the user launches a streaming session, as described in the following section.

The path for the folder where the settings VHD is stored in the S3 bucket in your account uses the following structure:
The name of the S3 bucket in which users' application settings are stored. The name format is described earlier in this section.

The settings group value. This value is applied to one or more stacks that share the same application settings.

The identity method of the user: custom for the AppStream 2.0 API or CLI, federated for SAML, and userpool for user pool users.

The user-specific folder name. This name is created using a lowercase SHA-256 hash hexadecimal string generated from the user ID.

The following example folder structure applies to a streaming session that is accessed using the API or CLI with a user ID of testuser@mydomain.com, an AWS account ID of 123456789012, and the settings group test-stack in the US West (Oregon) Region (us-west-2):

```
appstream-app-settings-us-west-2-1234567890123-abcdefg/Windows/v2/test-stack/custom/
a0bcb1dal1f480d9b5b3e90f91243143eac04cfccfbdc777e740fab628a1cd13
```

You can identify the folder for a user by generating the lowercase SHA-256 hash value of the user ID using websites or open source coding libraries available online.

## Reset a User's Application Settings

To reset a user's application settings, you must find and delete the VHD and associated metadata file from the S3 bucket in your AWS account. Make sure that you do not do this during a user's active streaming session. After you delete the user's VHD and the metadata file, the next time the user launches a session from a streaming instance that has application settings persistence enabled, AppStream 2.0 creates a new settings VHD for that user.

### To reset a user's application settings

1. Open the Amazon S3 console at [https://console.aws.amazon.com/s3/](https://console.aws.amazon.com/s3/).
2. In the **Bucket name** list, choose the S3 bucket that contains the application settings VHD that you want to reset.
3. Locate the folder that contains the VHD. For more information about how to navigate the S3 bucket folder structure, see [Amazon S3 Bucket Storage](#) earlier in this topic.
4. In the **Name** list, select the check box next to the VHD and the REG, choose **More**, and then choose **Delete**.
5. In the **Delete objects** dialog box, verify that the VHD and the REG are listed, and then choose **Delete**.

The next time the user streams from a fleet on which application settings persistence is enabled with the applicable settings group, a new application settings VHD is created. This VHD is saved to the S3 bucket at the end of the session.
Enable Amazon S3 Object Versioning and Revert a User's Application Settings

You can use Amazon S3 object versioning and lifecycle policies to manage your users' application settings when your users change them. With Amazon S3 object versioning, you can preserve, retrieve, and restore every version of the settings VHD. This enables you to recover from both unintended user actions and application failures. When versioning is enabled, after each streaming session, a new version of the application settings VHD is synced to Amazon S3. The new version does not overwrite the previous version, so if an issue with your users' settings occurs, you can revert to a previous version of the VHD.

**Note**
Each version of the application settings VHD is saved to Amazon S3 as a separate object and is charged accordingly.

Object versioning is not enabled by default in your S3 bucket, so you must explicitly enable it.

**To enable object versioning for your application settings VHD**

1. Open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. In the **Bucket name** list, choose the S3 bucket that contains the application settings VHD on which to enable object versioning.
3. Choose **Properties**.
4. Choose **Versioning, Enable versioning**, and then choose **Save**.

To expire older versions of your application settings VHDs, you can use Amazon S3 lifecycle policies. For information, see How Do I Create a Lifecycle Policy for an S3 Bucket? in the Amazon Simple Storage Service Console User Guide.

**To revert a user's application settings VHD**

You can revert to a previous version of a user's application settings VHD by deleting newer versions of the VHD from the applicable S3 bucket. Do not do this when the user has an active streaming session.

1. Open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. In the **Bucket name** list, choose the S3 bucket that contains the user's application settings VHD version to revert to.
3. Locate and select the folder that contains the VHD. For information about how to navigate the S3 bucket folder structure, see Amazon S3 Bucket Storage earlier in this topic.
   
   When you select the folder, the settings VHD and associated metadata file display.
4. To display a list of the VHD and metadata file versions, choose **Show**.
5. Locate the version of the VHD to revert to.
6. In the **Name** list, select the check boxes next to the newer versions of the VHD and associated metadata files, choose **More**, and then choose **Delete**.
7. Verify that the application settings VHD that you want to revert to and the associated metadata file are the newest versions of these files.

The next time the user streams from a fleet on which application settings persistence is enabled with the applicable settings group, the reverted version of the user's settings displays.

**Increase the Size of the Application Settings VHD**

The default VHD maximum size is 1 GB. If a user requires additional space for application settings, you can download the applicable application settings VHD to a Windows computer to expand it. Then,
replace the current VHD in the S3 bucket with the larger one. Do not do this when the user has an active streaming session.

**To increase the size of the application settings VHD**

**Note**
The full VHD must be downloaded before a user can stream applications. Increasing the size of an application settings VHD can increase the time it takes for users to start application streaming sessions.

1. Open the Amazon S3 console at [https://console.aws.amazon.com/s3/](https://console.aws.amazon.com/s3/).
2. In the **Bucket name** list, choose the S3 bucket that contains the application settings VHD to expand.
3. Locate and select the folder that contains the VHD. For information about how to navigate the S3 bucket folder structure, see *Amazon S3 Bucket Storage* earlier in this topic.

When you select the folder, the settings VHD and associated metadata file display.
4. Download the Profile.vhdx file to a directory on your Windows computer. Do not close your browser after the download completes, because you'll use the browser again later to upload the expanded VHD.
5. To use Diskpart to increase the size of the VHD to 2 GB, open the command prompt as an administrator, and type the following commands.

   ```
   diskpart
   select vdisk file="C:\path\to\application\settings\profile.vhdx"
   expand vdisk maximum=2000
   ```

6. Then, type the following Diskpart commands to find and attach the VHD, and display the list of volumes:

   ```
   select vdisk file="C:\path\to\application\settings\profile.vhdx"
   attach vdisk
   list volume
   ```

   In the output, make note of the volume number with the label "AppStreamUS". In the next step, you select this volume so that you can enlarge it.

7. Type the following command:

   ```
   select volume ###
   ```

   where ### is the number in the list volume output.
8. Type the following command:

   ```
   extend
   ```
9. Type the following commands to confirm that the size of the partition on the VHD increased as expected (2 GB in this example):

   ```
   diskpart
   select vdisk file="C:\path\to\application\settings\profile.vhdx"
   list volume
   ```
10. Type the following command to detach the VHD so that it can be uploaded:

    ```
    detach vdisk
    ```
11. Return to your browser with the Amazon S3 console, choose **Upload**, **Add files**, and then select the enlarged VHD.
12. Choose **Upload**.

After the VHD is uploaded, the next time the user streams from a fleet on which application settings persistence is enabled with the applicable settings group, the larger application settings VHD is available.
Enable Regional Settings for Your AppStream 2.0 Users

AppStream 2.0 lets you or your users configure certain Windows settings that are specific to your users’ location or language.

Contents
- Configure Default Regional Settings for Your AppStream 2.0 Users (p. 154)
- Enable Your AppStream 2.0 Users to Configure Their Regional Settings (p. 160)

Configure Default Regional Settings for Your AppStream 2.0 Users

In AppStream 2.0, users can configure their streaming sessions to use settings that are specific to their location or language. For more information, see Enable Your AppStream 2.0 Users to Configure Their Regional Settings (p. 160). You can also configure your fleets to use default settings that are specific to your users’ location or language. In particular, you can apply the following Windows settings to your fleets:

- **Time Zone** — Determines the system time used by Windows and any applications that rely on the operating system time. AppStream 2.0 makes available the same options for this setting as Windows Server 2012 R2, Windows Server 2016, and Windows Server 2019.
- **Display Language** — Determines the display language used by the Windows operating system and certain Windows applications.
- **System Locale** — Determines the code pages (ANSI, MS-DOS, and Macintosh) and bitmap font files that Windows uses for non-Unicode applications in different languages.
- **User Locale** (also known as culture) — Determines the conventions used by Windows and any applications that query the Windows culture when formatting dates, numbers, or currencies or when sorting strings.
- **Input Method** — Determines the keystroke combinations that can be used to enter characters in another language.

Currently, AppStream 2.0 supports English and Japanese only for these language settings.

Contents
- Specify a Default Time Zone (p. 155)
- Specify a Default Display Language (p. 155)
- Specify a Default System Locale (p. 157)
- Specify a Default User Locale (p. 157)
- Specify a Default Input Method (p. 158)
- Special Considerations for Application Settings Persistence (p. 159)
- Special Considerations for Japanese Language Settings (p. 160)
Specify a Default Time Zone

To specify a default time zone to be used in your users' streaming sessions, perform the following steps.

2. In the left navigation pane, choose Images, Image Builder.
3. Choose the image builder that you want to use, choose Connect, and log in as Administrator.
4. On the image builder desktop, choose the Windows Start button, and choose Control Panel.
5. Choose Clock, Language, and Region, then Date and Time, then Change time zone.
6. In the Time zone list, choose a time zone, and choose OK.

   **Note**
   Currently, AppStream 2.0 supports only UTC and (UTC+9:00) Osaka, Sapporo, Tokyo.
7. To apply any change to the time zone setting, restart your image builder. To do so, choose the Windows Start button, and choose Windows PowerShell. In PowerShell, use the restart-computer cmdlet.
8. While Windows restarts, the AppStream 2.0 login prompt displays. Wait for 10 minutes before you log in to the image builder again. Otherwise, you may receive an error. After 10 minutes, you can log in as Administrator.
9. If required, configure additional default regional or language settings. Otherwise, on the image builder desktop, open Image Assistant and install and configure applications for streaming.
10. After you finish configuring your image builder, follow the necessary steps in Image Assistant to finish creating your image. For information about how to create an image, see Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47).
11. Do one of the following:
   - Create a new fleet and choose your new image for the fleet. For more information, see Create an AppStream 2.0 Fleet and Stack (p. 81).
   - Update an existing fleet to use the new image.
12. Associate your fleet with the stack that is assigned to the users for whom you are configuring the default settings.

   The default time zone setting that you configured is applied to the fleet instances and user streaming sessions that are launched from those instances.

   **Note**
   Your users can change their time zone from the default setting that you configured. They can configure their regional settings during an application streaming session, as described in Enable Your AppStream 2.0 Users to Configure Their Regional Settings (p. 160). Also, if a user previously selected a time zone when streaming from any fleet instance in the same AWS Region, the user-specified time zone setting automatically overrides any default time zone setting you specify through your image builder.

Specify a Default Display Language

There are two ways to specify the default display language for your users' streaming sessions. Use the AppStream 2.0 default application and Windows settings feature, or configure your image builder while logged in as Administrator.

   **Note**
   Changing the display language in Windows also automatically changes the user locale and input method to match the language and region of the display language. If you want all three settings to match, you do not need to separately change the user locale or input method.
To specify a default display language by using the AppStream 2.0 default application and Windows settings feature, perform the following steps.

2. In the left navigation pane, choose Images, Image Builder.
3. Choose the image builder that you want to use, choose Connect, and log in as Template User.
4. On the image builder desktop, choose the Windows Start button, and choose Control Panel.
5. Choose Clock, Language, and Region, then Language, Add a language.
6. Choose a language, and choose Add.

Note
Currently, AppStream 2.0 supports only English (United States) and Japanese.

7. The language that you selected appears in the list of languages you added to Windows. Choose the language that you just added. Then choose Move up until the language appears at the top of the language list.
8. Choose Advanced Settings. Under Override for Windows display language, choose your language from the list.
9. If you want to use the input method associated with the language that you added, under Override for default input method, choose the input method for the language.
10. Choose Save. When prompted to log off, choose Log off now.
11. When prompted, log in again to the image builder as Template User. Confirm that Windows is using the display language that you selected.
12. In the upper right area of the image builder desktop, choose Admin Commands, Switch User.

13. When prompted, log in as Administrator.
14. If required, configure additional default regional or language settings. Otherwise, on the image builder desktop, open Image Assistant and install and configure applications for streaming.
15. In Step 2 of the Image Assistant process, choose Save settings.
16. Follow the necessary steps in Image Assistant to finish creating your image. For information about how to create an image, see Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47).
17. Do one of the following:
   - Create a new fleet and choose your new image for the fleet. For information, see Create an AppStream 2.0 Fleet and Stack (p. 81).
   - Update an existing fleet to use the new image.
18. Associate your fleet with the stack that is assigned to the users for whom you are configuring the default settings.

The default display language and associated user locale and input method settings that you configured are applied to the fleet instances and user streaming sessions that are launched from those instances.

Alternatively, you can configure a default display language while logged in to the image builder as Administrator. If you chose different display languages while you were logged in under the Template User and Administrator accounts and you chose Save settings in Step 2 of the Image Assistant process, the Template User settings take precedence.
Specify a Default System Locale

To specify a default system locale for your users’ streaming sessions, perform the following steps.

2. In the left navigation pane, choose Images, Image Builder.
3. Choose the image builder that you want to use, choose Connect, and log in as Administrator.
4. On the image builder desktop, choose the Windows Start button, and choose Control Panel.
5. Choose Clock, Language, and Region, then Region.
6. In the Region dialog box, choose the Formats tab.
7. Choose Change system locale.
8. In the Region Settings dialog box, in the Current system locale list, choose a language and region.
   
   **Note**
   Currently, AppStream 2.0 supports only English (United States) and Japanese (Japan).
9. Choose OK to close the Region Settings dialog box, and choose OK again to close the Region dialog box.
10. When prompted to restart your computer, allow Windows to restart.
11. While Windows restarts, the AppStream 2.0 login prompt displays. Wait for 10 minutes before you log in to the image builder again. Otherwise, you may receive an error. After 10 minutes, you can log in as Administrator.
12. If required, configure additional default regional or language settings. Otherwise, on the image builder desktop, open Image Assistant and install and configure applications for streaming. After you finish configuring your image builder, follow the necessary steps in Image Assistant to finish creating your image. For information about how to create an image, see Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47).
13. Do one of the following:
   - Create a new fleet and choose your new image for the fleet. For more information, see Create an AppStream 2.0 Fleet and Stack (p. 81).
   - Update an existing fleet to use the new image.
14. Associate your fleet with the stack that is assigned to the users for whom you are configuring the default settings.

The default system locale setting that you configured is applied to the fleet instances and user streaming sessions that are launched from those instances.

Specify a Default User Locale

To specify a default user locale for your users’ streaming sessions, perform the following steps.

**Note**
Your users can change their user locale and input method from the default settings that you configured. They can change to any one of 11 different supported locales and nine different supported input methods. To do so, they can configure their regional settings during application streaming sessions, as described in Enable Your AppStream 2.0 Users to Configure Their Regional Settings (p. 160). Also, if a user previously selected a user locale or input method when streaming from any fleet instance in the same Region, those user-specified settings automatically override any default user locale and input method that you specify through your image builder.
Specify a Default Input Method

Note
If you plan to configure the display language and you want the input method and display language to match, you do not need to change the input method. Changing the display language in Windows also automatically changes the user locale and input method to match the language and region of the display language. If you want all three settings to match, you do not need to separately change the user locale or input method.
2. In the left navigation pane, choose Images, Image Builder.
3. Choose the image builder that you want to use, choose Connect, and log in as Administrator.
4. On the image builder desktop, choose the Windows Start button, and choose Control Panel.
5. Choose Clock, Language, and Region, then Language, Add a language.
6. Choose a language, and choose Add.

   **Note**
   Currently, AppStream 2.0 supports only **English (United States)** and **Japanese**.

7. The language that you chose appears in the list of languages you added to Windows.
8. Choose Advanced Settings. Under *Override for default input method*, choose the input method for the language you added.
9. Choose Save.
10. Log off and log in again. To do so, choose the Windows Start button on the image builder desktop. Choose ImageBuilderAdmin, Sign out. When prompted, log in as Administrator.
11. If required, configure additional default regional or language settings. Otherwise, on the image builder desktop, open Image Assistant and install and configure applications for streaming.
12. In Step 2 of the Image Assistant process, choose Save settings.
13. Follow the necessary steps in Image Assistant to finish creating your image. For information about how to create an image, see Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47).
14. Do one of the following:
   - Create a new fleet and choose your new image for the fleet. For information, see Create an AppStream 2.0 Fleet and Stack (p. 81).
   - Update an existing fleet to use the new image.
15. Associate your fleet with the stack that is assigned to the users for whom you are configuring the default settings.

   The default input method that you configured is applied to the fleet instances and user streaming sessions that are launched from those instances.

   **Note**
   Your users can change their input method from the default setting that you configured to any one of nine different supported input methods. They can configure this setting by configuring their regional settings during application streaming sessions, as described in Enable Your AppStream 2.0 Users to Configure Their Regional Settings (p. 160). Also, if a user previously selected an input method when streaming from any fleet instance in the same Region, that user-specified setting automatically overrides any default input method that you specify through your image builder.

### Special Considerations for Application Settings Persistence

When you create a stack in the AppStream 2.0 console, in **Step 3: User Settings**, if you use the same settings group under **Application settings persistence** as another stack that uses different regional settings, only one set of regional settings is used for both stacks. For each user, the default regional settings for the stack that the user logs into first automatically override the default regional settings of any other stacks in the same application settings group. To avoid this problem, do not use the same application settings group for two different stacks that have different regional settings.
Special Considerations for Japanese Language Settings

This section describes key points to keep in mind when configuring Japanese language settings for your AppStream 2.0 users.

AWS CLI

Changing the Windows system locale to Japanese requires that your image builder have AWS Command Line Interface (AWS CLI) version 1.16.30 or later installed. To update the version of AWS CLI on your image builder, follow the steps in Installing the AWS Command Line Interface.

Japanese Keyboards

If your image builder input method is set to Japanese when you create an image, AppStream 2.0 automatically configures your image to use a Japanese keyboard. Any fleets that use the image are also automatically configured to use Japanese keyboards. However, if you want to use a Japanese keyboard within your image builder session, update the following registry settings for the HKEY_LOCAL_MACHINE \SYSTEM\CurrentControlSet\Services\i8042prt\Parameters registry key:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>LayerDriver JPN</td>
<td>REG_SZ</td>
<td>kbd106.dll</td>
</tr>
<tr>
<td>OverrideKeyboardIdentifier</td>
<td>REG_SZ</td>
<td>PCAT_106KEY</td>
</tr>
<tr>
<td>OverrideKeyboardSubtype</td>
<td>DWORD</td>
<td>2</td>
</tr>
<tr>
<td>OverrideKeyboardType</td>
<td>DWORD</td>
<td>7</td>
</tr>
</tbody>
</table>

After changing these settings, restart your image builder. To do so, choose the Windows Start button, and choose Windows PowerShell. In PowerShell, use the restart-computer cmdlet.

Enable Your AppStream 2.0 Users to Configure Their Regional Settings

Users can configure their Amazon AppStream 2.0 streaming sessions to use settings that are specific to their location or language. In particular, users can configure the following settings:

- **Time zone** — Determines the system time used by Windows and any applications that rely on the operating system time. AppStream 2.0 makes available the same options for this setting as the Windows Server version used in your fleet.
- **Locale** (also known as culture) — Determines the conventions used by Windows and any applications that query the Windows culture when formatting dates, numbers, or currencies or when sorting strings. For a list of locales that AppStream 2.0 supports, see Supported Locales (p. 161).
- **Input method** — Determines the keystroke combinations that can be used to input characters in another language.

If users change regional settings during their streaming sessions, the changes are applied to any future streaming sessions in the same AWS Region.
Note
For guidance that you can provide your users to help them get started with configuring their regional settings, see Configure Regional Settings (p. 284).

Contents
- Supported Locales (p. 161)
- Enable Regional Settings for Your AppStream 2.0 Users (p. 161)

Supported Locales

AppStream 2.0 supports the following locales:

<table>
<thead>
<tr>
<th>Locale</th>
<th>Language culture name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chinese (Simplified, China)</td>
<td>zh-CN</td>
</tr>
<tr>
<td>Chinese (Simplified, Singapore)</td>
<td>zh-SG</td>
</tr>
<tr>
<td>Chinese (Traditional, Taiwan)</td>
<td>zh-TW</td>
</tr>
<tr>
<td>Dutch (The Netherlands)</td>
<td>nl-NL</td>
</tr>
<tr>
<td>English (Australia)</td>
<td>en-AU</td>
</tr>
<tr>
<td>English (Canada)</td>
<td>en-CA</td>
</tr>
<tr>
<td>English (United Kingdom)</td>
<td>en-GB</td>
</tr>
<tr>
<td>English (United States)</td>
<td>en-US</td>
</tr>
<tr>
<td>French (France)</td>
<td>fr-FR</td>
</tr>
<tr>
<td>German (Germany)</td>
<td>de-DE</td>
</tr>
<tr>
<td>Italian (Italy)</td>
<td>it-IT</td>
</tr>
<tr>
<td>Japanese (Japan)</td>
<td>ja-JP</td>
</tr>
<tr>
<td>Korean (Korea)</td>
<td>ko-KR</td>
</tr>
<tr>
<td>Portuguese (Brazil)</td>
<td>pt-BR</td>
</tr>
<tr>
<td>Spanish (Spain, International Sort)</td>
<td>es-ES</td>
</tr>
<tr>
<td>Thai (Thailand)</td>
<td>th-TH</td>
</tr>
</tbody>
</table>

Enable Regional Settings for Your AppStream 2.0 Users

To enable users to configure regional settings for a given stack during their AppStream 2.0 streaming sessions, your stack must be associated with a fleet based on an image that uses a version of the AppStream 2.0 agent released on or after June 6, 2018. For more information, see AppStream 2.0 Agent Version History (p. 39). Additionally, your image must have Windows PowerShell 5.1 or later installed. Images created from AppStream 2.0 base images published on or after June 12, 2018 meet both criteria. Images created from AppStream 2.0 base images published before June 12, 2018 do not have Windows PowerShell 5.1 by default.
To update an existing image to include Windows PowerShell 5.1

1. Launch a new image builder using your existing image as the base image by doing the following:
   a. In the left navigation pane in the AppStream 2.0 console, choose Images.
   b. Choose the Image Builder tab, Launch Image Builder, and then select your existing image.
   c. If you are prompted to update the AppStream 2.0 agent when you launch the image builder, select the check box, and then choose Start.
2. Once your new image builder is running, connect to it and log in with a user account that has local administrator permissions.
3. From the image builder desktop, open Windows PowerShell. Choose the Windows Start button, and then choose Windows PowerShell.
4. At the PowerShell command prompt, type the command $PSVersionTable to determine the version of Windows PowerShell that is installed on your image builder. If your image builder does not include Windows PowerShell 5.1 or later, use the following steps to install it.
5. Open a web browser and follow the steps in Install and Configure WMF 5.1 in the Microsoft documentation, making sure that you download the Windows Management Framework (WMF) 5.1 package for Windows Server 2012 R2. WMF 5.1 includes Windows PowerShell 5.1.
6. At the end of the WMF 5.1 installation process, the installer prompts you to restart your computer. Choose Restart Now to restart the image builder.
7. Wait about 10 minutes before logging in to your image builder, even though AppStream 2.0 prompts you to do so immediately. Otherwise, you might encounter an error.
8. After logging in to your image builder again, open Windows PowerShell and type the command $PSVersionTable to confirm that Windows PowerShell 5.1 is installed on your image builder.
9. Use the image builder to create a new image. This new image now includes the latest versions of the AppStream 2.0 agent and Windows PowerShell.
10. Update your fleet to use the new image by doing the following:
    a. In the left navigation pane in the AppStream 2.0 console, choose Fleets, and then choose the fleet associated with the stack for which you want to enable regional settings.
    b. On the Fleet Details tab, choose Edit.
    c. In Image name, choose the new image to use for the fleet.

For more information about using image builders to create images, see Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47).
Manage App Entitlement with the Dynamic Application Framework

Amazon AppStream 2.0 supports dynamically building the application catalog that displays for your users when they stream from an AppStream 2.0 stack. You can use the API operations provided by AppStream 2.0 to develop a dynamic app provider that modifies, in real time, the applications that users can access on the streaming instance. Alternatively, you can implement a third-party dynamic app provider that uses these API operations.

**Note**
This feature requires an AppStream 2.0 fleet that is joined to a Microsoft Active Directory domain. For more information, see Using Active Directory with AppStream 2.0 (p. 106).

**Contents**
- Example API Operations Work Flow for the Dynamic Application Framework (p. 163)
- Use the AppStream 2.0 Dynamic Application Framework to Build a Dynamic App Provider (p. 165)
- Enable and Test Dynamic App Providers (p. 170)
- Third-Party Dynamic App Providers (p. 171)

Example API Operations Work Flow for the Dynamic Application Framework

The following diagram is an example of the API operations flow between AppStream 2.0 and a third-party application provider.
1. Connect to AppStream 2.0, log in to Windows

2. Detect Windows login event

3. Fetch application entitlements

4. AddApp

5. Data store for application entitlements

6. Select application to launch

7. Launch application
1. The user connects to AppStream 2.0. A fleet streaming instance is assigned to the user and Windows login occurs.
2. Your service or agent detects the Windows logon event and determines the user who is logging in to Windows.
3. The service or agent fetches the application entitlements for the user. In the example diagram, the application entitlements are stored in a database. This information can be stored and retrieved in different ways. For example, application entitlements may be fetched from server software, or group names in Active Directory may be parsed to locate the application identifiers (IDs).
4. Your dynamic app provider calls the AppStream 2.0 agent AddApplications API operation with the application metadata for the applications that the user should have.
5. The AppStream 2.0 agent dynamically updates the application catalog with the modified application list.
6. The user selects an application to launch.
7. The application is launched by using the application metadata specified by your service or agent.

From the user’s perspective, the process happens transparently. The user connects to AppStream 2.0 and logs in to the fleet instance. After login, the list of applications specified in the image and provided by your dynamic app provider displays for the user.

Use the AppStream 2.0 Dynamic Application Framework to Build a Dynamic App Provider

The AppStream 2.0 dynamic application framework provides API operations within an AppStream 2.0 streaming instance that you can use to build a dynamic app provider. Dynamic app providers can use the API operations provided to modify the catalog of applications that your users can access in real time. The applications managed by the dynamic app providers can be within the image, or they can be off-instance, such as from a Windows file share or an application virtualization technology.

Note
This feature requires an AppStream 2.0 fleet that is joined to a Microsoft Active Directory domain. For more information, see Using Active Directory with AppStream 2.0 (p. 106).

Contents
- About the Dynamic Application Framework (p. 165)
- Dynamic Application Framework Thrift Definitions and Named Pipe Name (p. 165)
- API Actions for Managing App Entitlement for AppStream 2.0 (p. 167)

About the Dynamic Application Framework

The dynamic application framework uses the Apache Thrift software framework for inter-process messaging. It is exposed through Named Pipes in Windows. Using the Thrift framework allows you to build your dynamic app provider in your software development language of choice. The dynamic application framework consists of three API operations: AddApplications, RemoveApplications, and ClearApplications.

Dynamic Application Framework Thrift Definitions and Named Pipe Name

Thrift enables you to use simple definition files provided by AppStream 2.0 to compile RPC clients. The RPC clients let you communicate with the AppStream 2.0 agent software running on a streaming...
instance. For information about how to compile the RPC client for your language, see the Apache Thrift documentation. After you compile the Thrift libraries for the language of your choice, build a Thrift client by using the Named Pipe transport. Use D56C0258-2173-48D5-B0E6-1EC85AC67893 as the pipe name.

### AppStreamServer.thrift

```csharp
namespace csharp AppStream.ApplicationCatalogService.Model

include "AppStreamServerMessages.thrift"

const string ServiceEndpoint = "D56C0258-2173-48D5-B0E6-1EC85AC67893";

service ApplicationCatalogService
{
}
```

### AppStreamServerMessages.thrift

```csharp
namespace csharp AppStream.ApplicationCatalogService.Model

struct AddApplicationsRequest
{
    1: required string userSid;
    2: required list<Application> applications;
}

struct AddApplicationsResponse
{
}

struct RemoveApplicationsRequest
{
    1: required string userSid;
    2: required list<string> applicationIds;
}

struct RemoveApplicationsResponse
{
}

struct ClearApplicationsRequest
{
    1: required string userSid;
}

struct ClearApplicationsResponse
{
}
```
API Actions for Managing App Entitlement for AppStream 2.0

You can use the following API operations to manage application entitlement for AppStream 2.0.

AddApplicationsRequest operation

Adds applications to the application catalog for AppStream 2.0 users. The application catalog displayed by AppStream 2.0 includes the applications that you add by using this API operation and the applications that you add in the image. After you add applications by using one or both of these methods, your users can launch the applications.

Request syntax

```
string userSid;
list<Application> applications;
```

Request parameters

userSid

The SID of the user who the request applies to.

Type: String

Required: Yes

Length constraints: Minimum length of 1, maximum length of 208 characters.

applications

The list of applications that the request applies to.
Type: String
Required: Yes

**Application object**

Describes the application metadata required to display and launch the application. The application identifier must be unique and not in conflict with other applications specified through the API operation or the image.

**id**

The identifier of the application being specified. This value, which corresponds to the application_name value in an AppStream 2.0 applications report, is provided when a user launches the application. When you enable usage reports (p. 207), for each day that users launch at least one application during their streaming sessions, AppStream 2.0 exports an applications report to your Amazon S3 bucket. For more information about applications reports, see Applications Report Fields (p. 213).

Type: String
Required: Yes

Length constraints: Minimum length of 1, maximum length of 512 characters.

**displayName**

The display name of the application being specified. This name is displayed to the user in the application catalog.

Type: String
Required: Yes

Length constraints: Minimum length of 1, maximum length of 512 characters.

**launchPath**

The Windows file system path to the executable of the application to be launched.

Type: String
Required: Yes

Length constraints: Minimum length of 1, maximum length of 32,767 characters.

**iconData**

The base-64 encoded image to display in the application catalog. The image must be in one of the following formats: .png, .jpeg, or .jpg.

Type: String
Required: Yes

Length constraints: Minimum length of 1, maximum length of 1,000,000 characters.

**launchParams**

The parameters used to launch the application.

Type: String
RemoveApplicationsRequest operation

Removes applications that were added by using the AddApplicationsRequest operation. The applications are removed from the application catalog for the user. After applications are removed, they can't be launched. If an application is still running, AppStream 2.0 does not close it. Applications that are specified directly in the AppStream 2.0 image can't be removed.

Request syntax

```java
string userSid;
list<Application> applications;
```

Request parameters

**userSid**

The SID of the user the request applies to.

**Type:** String

**Required:** Yes

**Length constraints:** Minimum length of 1, maximum length of 208 characters.

**applications**

The list of applications that the request applies to.

**Type:** String

**Required:** Yes

ClearApplicationsRequest operation

Removes all applications that were added to the application catalog by using the AddApplicationsRequest operation. After applications are removed, they can't be launched. If the applications are running when the ClearApplicationsRequest operation is used, AppStream 2.0 does not close them. Applications that are specified directly in the AppStream 2.0 image can't be removed.

Request syntax

```java
string userSid;
```
Enable and Test Dynamic App Providers

Dynamic app providers must first be enabled within an AppStream 2.0 image. After you enable these providers, they can manage applications for users on the streaming instance.

Enable Dynamic App Providers

To enable this capability, you must add your dynamic app provider details to a configuration file on the image builder. The image builder must be joined to a Microsoft Active Directory domain. Perform the following steps on an image builder, then you can test your dynamic apps to verify that they function as expected. Finally, finish creating your image.

**Note**
Third-party dynamic app providers may modify the configuration file during install. For installation instructions, see the documentation for the applicable provider.

**To enable dynamic app providers**

1. In the left navigation pane, choose **Images, Image Builder**.
2. Choose an image builder that is joined to a Microsoft Active Directory domain. Verify that the image builder is in the **Running** state, and choose **Connect**.
3. When prompted, choose **Administrator**.
4. Navigate to `C:\ProgramData\Amazon\AppStream\AppCatalogHelper\DynamicAppCatalog\`, and open the **Agents.json** configuration file.
5. In the **Agents.json** file, add the following entries:

   ```
   "DisplayName": "<Uninstall hive display name value>",
   "Path": "<C:\path\to\client\application>"
   ```

   **DisplayName** must match the **DisplayName** registry value for the `HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Uninstall` key created for your application.

6. Install your dynamic app provider.
7. On the image builder desktop, open Image Assistant.
8. Optionally, install any other applications that you want to include in the image.
9. In Image Assistant, on the **1. Add Apps** page, select the **Enable dynamic app providers** check box.
10. On the same page, if you installed other applications as described in step 8, choose **+Add App**, and specify the applications to add.

   **Note**

   When you use a dynamic app provider, you don't need to specify any applications in the image. If you specify applications in the image, they can't be removed by dynamic app providers.
11. Proceed to the steps in the next section to test your dynamic app provider.

Test Dynamic App Providers (Optional)

After you enable your dynamic app provider on an image builder, you can test the provider to verify that it functions as expected. To do so, perform the following steps before you finish creating the image.

To test dynamic app providers

1. Do one of the following:

   • If you are already logged on as an Administrator to the image builder on which you enabled dynamic app providers, you must switch to an account that does not have local administrator permissions on the image builder. To do so, in the upper right corner of the image builder session toolbar, choose Admin Commands, Switch User.

   • If you are not already logged on to the image builder, in the left navigation pane, choose Images, Image Builder. Choose the image builder on which you enabled your dynamic app providers. Verify that the image builder is in the Running state, and choose Connect.

2. When prompted, choose Directory User, and log on with a domain user account that does not have local administrator permissions on the image builder.

3. On the image builder desktop, open Image Assistant, if it is not already open.

4. On the Test Apps page, if you specified any applications in the image that are not from the dynamic app provider, they display first in the list. It may take a few moments for applications from dynamic app providers to appear in the list.

5. Choose an application from the list and open it to verify that it functions as expected.

6. After you finish testing, in the lower right corner of the Test Apps page, choose Switch user.

7. Choose Administrator, and log back into the image builder.

8. Follow the necessary steps in Image Assistant to finish creating your image. For information about how to create an image, see Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47).

   AppStream 2.0 automatically optimizes the agents that are specified in the Agents.json configuration file.

Third-Party Dynamic App Providers

The following third-party providers use the AppStream 2.0 dynamic application framework to manage application entitlements and delivery in real time.

<table>
<thead>
<tr>
<th>Dynamic app provider</th>
<th>More information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Liquidware Labs FlexApp</td>
<td>FlexApp — FlexApp layering delivers applications to any Windows desktop environment, independent of the Windows operating system version.</td>
</tr>
</tbody>
</table>
### Dynamic app provider

<table>
<thead>
<tr>
<th>Dynamic app provider</th>
<th>More information</th>
</tr>
</thead>
<tbody>
<tr>
<td>SynchroNet CLICK™</td>
<td>CLICK™ — CLICK™ is a User Automation Platform (UAP) that leverages the power of serverless technology to automate user, workspace, and application life cycles.</td>
</tr>
</tbody>
</table>

For more information about using the AppStream 2.0 dynamic application framework to develop your own dynamic app provider, see Use the AppStream 2.0 Dynamic Application Framework to Build a Dynamic App Provider (p. 165).
Provide Your Users with Access to AppStream 2.0

Users can access AppStream 2.0 streaming sessions by using either a web browser or the AppStream 2.0 client on a supported device. The following topics provide guidance to help you provide your users with access to AppStream 2.0 for application streaming.

For guidance that you can provide your users to help them get started with application streaming, see Guidance for AppStream 2.0 Users (p. 272).

Contents
• Provide Access Through a Web Browser (p. 173)
• Provide Access Through the AppStream 2.0 Client for Windows (p. 174)

Provide Access Through a Web Browser

Users can access AppStream 2.0 through an HTML5-capable web browser on a desktop computer such as a Windows, Mac, Chromebook, or Linux computer. HTML5-capable web browsers that can be used include the following:

• Google Chrome
• Mozilla Firefox
• Safari
• Microsoft Edge
• Microsoft Internet Explorer version 11 or later

No browser extensions or plugins are required to use AppStream 2.0 in a web browser.

Users can also access AppStream 2.0 fleet streaming sessions on the following browsers and devices:

• Chrome or Safari on an iPad (iOS 11 or later)
• Android (Android 8 or later)
• Microsoft Surface Pro (Windows 10) tablet

AppStream 2.0 is not supported on devices that have screen resolutions smaller than 1024x768 pixels.

Dual-Monitor Support

AppStream 2.0 provides dual-monitor support for streaming sessions that are started on the following web browsers:

• Google Chrome
• Mozilla Firefox
• Safari

For browser-based streaming sessions on dual monitors, a maximum display resolution of 2560x1440 pixels is supported per monitor. If your users require more than two monitors, or a display resolution that is greater than 2560x1440 pixels per monitor, the AppStream 2.0 client is available.

**Note**
Dual monitors are not supported on mobile devices or for embedded AppStream 2.0 streaming sessions.

In addition to user connections for streaming sessions, AppStream 2.0 also supports the use of dual monitors for administrative connections to image builders.

### Touchscreen Device Support

AppStream 2.0 supports gestures on touch-enabled iPads, Android tablets, and Windows devices. All touch events are passed through to the streaming session and handled according to Windows conventions. Examples of supported touch gestures include long-tap to right-click, swipe to scroll, pinch to zoom, and two-finger rotation for supporting applications.

**Note**
To enable support for gestures on touch-enabled devices, your AppStream 2.0 image must use a version of the AppStream 2.0 agent released on or after March 7, 2019. For more information, see [AppStream 2.0 Agent Version History](p. 39).

For guidance that you can provide your users to help them get started with using touch-enabled devices during their AppStream 2.0 streaming sessions, see [Touchscreen Devices](p. 273).

### Provide Access Through the AppStream 2.0 Client for Windows

Your users can start AppStream 2.0 streaming sessions by using the AppStream 2.0 client application for a supported device or by using a web browser.

The AppStream 2.0 client is a native application that is designed for users who require additional functionality during their AppStream 2.0 streaming sessions. This client is suitable for users who:

- Require support for more than two monitors or 4K resolution.
- Use their USB devices with applications streamed through AppStream 2.0.
- Use keyboard shortcuts during their streaming sessions.
- Require seamless access to local drives and folders during their streaming sessions.
- Prefer to interact with remote streaming applications in much the same way as they interact with locally installed applications.

The following topics provide information to help you provide user access through the AppStream 2.0 client. For information about how to provide user access AppStream 2.0 through a web browser, see [Provide Access Through a Web Browser](p. 173).

**Contents**

- System Requirements and Feature Support (p. 175)
- Install and Configure the AppStream 2.0 Client (p. 176)
- AppStream 2.0 Client Version History (p. 197)
System Requirements and Feature Support

This topic provides information to help you understand the requirements for the AppStream 2.0 client and supported features.

System Requirements

The AppStream 2.0 client requires the following:

- Operating system — Windows 7, Windows 8, or Windows 10 (32-bit or 64-bit)
- RAM — 2 GB minimum
- Hard drive space — 200 MB minimum
- Local administrator rights — Used if you want to install the AppStream 2.0 USB driver for USB driver support
- An AppStream 2.0 image that uses the latest AppStream 2.0 agent or agent versions published on or after November 14, 2018. For information about AppStream 2.0 agent versions, see AppStream 2.0 Agent Version History (p. 39).

Feature Support

The AppStream 2.0 client supports the following features.

Features

- Multiple Monitors (up to 2K Resolution) (p. 175)
- Multiple Monitors (up to 4K Resolution) (p. 175)
- USB Devices Qualified by AppStream 2.0 (p. 175)
- Native Application Mode (p. 175)
- Keyboard Shortcuts (p. 176)
- Relative Mouse Offset (p. 176)

Multiple Monitors (up to 2K Resolution)

The following AppStream 2.0 instance types support up to 4 monitors and a maximum display resolution of 2560x1440 pixels per monitor: General Purpose, Memory Optimized, Compute Optimized, Graphics Design, and Graphics Pro.

Multiple Monitors (up to 4K Resolution)

The following AppStream 2.0 instance types support up to 2 monitors and a maximum display resolution of 4096x2160 pixels per monitor: Graphics Design and Graphics Pro.

USB Devices Qualified by AppStream 2.0

All categories of the Connexion 3D Mouse are already qualified by AppStream 2.0. By default, all USB devices are disabled. To enable your users to use other USB devices with AppStream 2.0, you must qualify the devices. For more information, see Qualify USB Devices for Use with Streaming Applications (p. 183).

Native Application Mode

Native application mode provides a familiar experience for your users during their AppStream 2.0 streaming sessions. When your users connect to AppStream 2.0 in this mode, they can work with their...
remote streaming applications in much the same way that they work with applications that are installed on their local PC. Each streaming application in native application mode opens in its own window, and application icons appear on the taskbar on your users' local PC.

If you want your users to connect to AppStream 2.0 in classic mode only, you can configure the NativeAppModeDisabled registry value to disable native application mode. For more information, see Choose Whether to Disable Native Application Mode (p. 180).

For more information about native application mode and classic mode, and for guidance that you can provide to your users, see AppStream 2.0 Client Connection Modes (p. 276).

Requirements

To enable this feature for your users, you must use an image that uses a version of the AppStream 2.0 agent (p. 39) released on or after February 19, 2020. In addition, version 1.1.129 or later of the AppStream 2.0 client must be installed on your users' PCs. For more information about client versions, see AppStream 2.0 Client Version History (p. 197).

If AppStream 2.0 client version 1.1.129 or later is installed on your users' PC, but you are not using an image that uses an agent version released on or after February 19, 2020, the client falls back to classic mode—even if native application mode is selected.

Known Issues

When users try to dock or undock tabs in one browser window into separate windows during a streaming session in native application mode, their remote streaming browser doesn't work the same way as a local browser. To perform this task during a streaming session in native application mode, users must press the Alt key until their browser tabs are docked into separate browser windows.

Keyboard Shortcuts

Most operating system keyboard shortcuts are supported. Supported keyboard shortcuts include Alt + Tab, Clipboard shortcuts (Ctrl + X, Ctrl + C, Ctrl + V), Esc, and Alt + F4

Relative Mouse Offset

This feature can be used with applications such as Minecraft.

Install and Configure the AppStream 2.0 Client

You can have your users install the AppStream 2.0 client themselves, or you can install the AppStream 2.0 client for them by running PowerShell scripts remotely.

You must qualify the USB devices that you want to enable your users to use with their streaming session. If their USB device is not qualified, it won't be detected by AppStream 2.0 and can't be shared with the session.

The following topics describe how to install and configure the AppStream 2.0 client.

Contents

- Have Your Users Install the AppStream 2.0 Client Themselves (p. 177)
- Install the AppStream 2.0 Client And Customize the Client Experience for Your Users (p. 177)
- Update the AppStream 2.0 Enterprise Deployment Tool, Client, and USB Driver Manually (p. 182)
- Qualify USB Devices for Use with Streaming Applications (p. 183)
- Use the AppStream 2.0 Client to Start a Streaming Session (p. 185)
- Share a USB Device with an AppStream 2.0 Streaming Session (p. 186)
• Redirect a Streaming Session from the Web Browser to the AppStream 2.0 Client (p. 186)
• Enable File System Redirection for Your AppStream 2.0 Users (p. 187)

Have Your Users Install the AppStream 2.0 Client Themselves

For step-by-step guidance that you can provide your users to help them install the AppStream 2.0 client, see Setup (p. 275).

Important
If your organization has deployed antivirus software that prevents users from running .exe files, you must add an exception to allow your users to run the AppStream 2.0 client installation .exe program. Otherwise, when users try to install the client, either nothing happens, or they receive an error after they start the installation program.

After users install the client, if you plan to let your users use USB devices during their AppStream 2.0 streaming sessions, the following requirements must be met:

• You must qualify the USB devices that can be used with AppStream 2.0. For more information, see Qualify USB Devices for Use with Streaming Applications (p. 183).
• After their devices are qualified, your users must share the devices with AppStream 2.0 every time they start a new streaming session. For guidance that you can provide your users to them complete this task, see USB Devices (p. 277).

Install the AppStream 2.0 Client And Customize the Client Experience for Your Users

The following sections describe how to install the AppStream 2.0 client and customize the client experience for your users. If you plan to download and install the client for your users, first download the Enterprise Deployment Tool. You can then run PowerShell scripts to install the AppStream 2.0 client and configure client settings remotely.

Contents
• Download the Enterprise Deployment Tool (p. 177)
• Install the AppStream 2.0 Client and USB Driver (p. 178)
• Accessing AppStream 2.0 with the AppStream 2.0 Client (p. 178)
• Set the StartURL Registry Value for AppStream 2.0 Client Users (p. 178)
• Set the TrustedDomains Registry Value to Enable Other Domains for the AppStream 2.0 Client (p. 179)
• Choose Whether to Disable Automatic Client Updates (p. 180)
• Choose Whether to Disable Native Application Mode (p. 180)
• Configure Additional AppStream 2.0 Client Settings for Your Users (p. 180)
• Using Group Policy to Customize AppStream 2.0 Client Experience (p. 182)

Download the Enterprise Deployment Tool

The Enterprise Deployment Tool includes the AppStream 2.0 client installation files and a Group Policy administrative template.

1. To download the Enterprise Deployment Tool, on the bottom right of the AppStream 2.0 supported clients page, select the Enterprise Deployment Tool link. This link opens a .zip file that contains the required files for the latest version of the tool.
2. To extract the required files, navigate to the location where you downloaded the tool, right-click the AmazonAppStreamClient_EnterpriseSetup_<version> folder, and choose Extract All. The folder contains two installation programs and a Group Policy administrative template:

- AppStream 2.0 client installer (AmazonAppStreamClientSetup_<version>.msi) — Installs the AppStream 2.0 client.
- AppStream 2.0 USB driver installer (AmazonAppStreamUsbDriverSetup_<version>.exe) — Installs the AppStream 2.0 USB driver that is required to use USB devices with applications streamed through AppStream 2.0.
- AppStream 2.0 client Group Policy administrative template (as2_client_config.adm) — Lets you configure the AppStream 2.0 client through Group Policy.

Install the AppStream 2.0 Client and USB Driver

After you download the AppStream 2.0 client installation files, run the following PowerShell script on users' computers to install the AppStream 2.0 client and USB driver silently.

**Note**
To run this script, you must be logged into the applicable computer with **Administrator** permissions. You can also run the script remotely under the **System** account on startup.

```
Start-Process msiexec.exe -Wait -ArgumentList '/i AmazonAppStreamClientSetup_<version>.msi /quiet'
Start-Process AmazonAppStreamUsbDriverSetup_<version>.exe -Wait -ArgumentList '/quiet'
```

After you install the Enterprise Deployment tool on users' computers, the AppStream 2.0 client is also installed automatically for current users. For new users, the client is installed the first time that they log in to their computer. However, if users uninstall the AppStream 2.0 client, the client isn’t installed again until you update the AppStream 2.0 Enterprise Deployment Tool.

Accessing AppStream 2.0 with the AppStream 2.0 Client

By default, when users launch the AppStream 2.0 client, they can connect only to URLs that include the AppStream 2.0 domain. You can let client users access domains other than the AppStream 2.0 domain by doing either of the following:

- Set the **StartURL** registry value to specify a custom URL that users can access, such as the URL for your organization's login portal.
- Set the **TrustedDomains** registry value to specify trusted domains that users can access.

Set the StartURL Registry Value for AppStream 2.0 Client Users

You can use the **StartUrl** registry value to set a custom URL that is populated in the AppStream 2.0 client when a user launches the client. You can create this HKLM registry key while installing the client so that your users don’t need to specify a URL when they launch the client.

After the AppStream 2.0 client is installed, you can run the following PowerShell script to create this registry key, or you can use the administrative template that is included in the AppStream 2.0 client Enterprise Deployment Tool to configure the client through Group Policy.

Replace the **StartUrl** value with a URL for your identity provider (IdP). The URL must use a certificate that is trusted by the device. This means that the certificate that is used by the **StartUrl** webpage must contain a Subject Alternative Name (SAN) that includes the URL’s domain name. For example, if your **StartUrl** is set to https://appstream.example.com, the SSL certificate must have a SAN that includes appstream.example.com.
Note
To run this script, you must be logged into the applicable computer with Administrator permissions. You can also run the script remotely under the System account on startup.

```
$registryPath="HKLM:\Software\Amazon\AppStream Client"
New-Item -Path "HKLM:\Software\Amazon" -Name "AppStream Client" -Force
New-ItemProperty -Path $registryPath -Name "StartUrl" -Value "https://www.example.com" -PropertyType String -Force | Out-Null
```

Set the TrustedDomains Registry Value to Enable Other Domains for the AppStream 2.0 Client

You can configure the AppStream 2.0 client to connect to URLs in trusted domains that you specify. For example, you may want to let users connect to any URL in your organizational domain or to any URL in one or more of your IdP domains. When you specify the URL, use the following format: *.example-idp.com.

You can specify a list of trusted domains in a comma-separated format. Add this list as a registry value to the AppStream 2.0 TrustedDomains HKLM registry key. We recommend that you create this registry key and specify the list of trusted domains when you install the AppStream 2.0 client or, if you are using Microsoft Active Directory, through Group Policy. That way, your users can connect to a URL in any trusted domain immediately after the client is installed.

After the AppStream 2.0 client is installed, you can run the following PowerShell script to create this registry key, or you can use the administrative template that is included in the AppStream 2.0 client Enterprise Deployment Tool to configure the client through Group Policy.

Replace the TrustedDomains value with a comma-separated list for one or more of your organizational or IdP domains. The certificate used by the trusted domain webpage must contain a SAN that includes the URL's domain. For example, if your trusted domain includes *.example.com, and users specify https://appstream.example.com, the SSL certificate must have a SAN that includes appstream.example.com.

```
$registryPath="HKLM:\Software\Amazon\AppStream Client"
New-Item -Path "HKLM:\Software\Amazon" -Name "AppStream Client" -Force
New-ItemProperty -Path $registryPath -Name "TrustedDomains" -Value ",*.example1.com, *.example2.com, aws.amazon.com" -PropertyType String -Force | Out-Null
```

The following are requirements and considerations for formatting trusted domain names.

- The following characters are supported: a-z, 0-9, -, *
- DNS treats the * character either as a wildcard or as an asterisk character (ASCII 42), depending on where it appears in the domain name. Following are restrictions when using * as a wildcard in the name of a DNS record:
  - The * must replace the leftmost label in a domain name. For example, *.example.com or *.prod.example.com. If you include * in any other position, such as prod.*.example.com, DNS treats it as an asterisk character (ASCII 42), not as a wildcard.
  - The * must replace the entire label. For example, you can't specify *prod.example.com or prod*.example.com.
  - The * applies to the subdomain level that includes the *, and to all the subdomains of that subdomain. For example, if an entry is named *.example.com, the AppStream 2.0 client allows zenith.example.com, acme.zenith.example.com, and pinnacle.acme.zenith.example.com.
Choose Whether to Disable Automatic Client Updates

By default, when a new version of the AppStream 2.0 client is available, the client updates automatically to the latest version. However, you can disable automatic updates by setting the value for the `AutoUpdateDisabled` registry key to `true`. You can create this registry key when you install the AppStream 2.0 client. That way, the client is not updated automatically whenever a new version is available.

After the AppStream 2.0 client is installed, you can run the following PowerShell script to create this registry key. Alternatively, you can use the administrative template that is included in the AppStream 2.0 client Enterprise Deployment Tool to configure the client through Group Policy.

**Note**

To run this script, you must be logged into the applicable computer with **Administrator** permissions. You can also run the script remotely under the **System** account on startup.

```
$registryPath="HKLM:\Software\Amazon\AppStream Client"
New-Item -Path "$registryPath" -Name "AppStream Client" -Force
New-ItemProperty -Path $registryPath -Name "AutoUpdateDisabled" -Value "True" -PropertyType String -Force | Out-Null
```

Choose Whether to Disable Native Application Mode

By default, the AppStream 2.0 client can run in either classic mode or native application mode. However, you can configure the `NativeAppModeDisabled` registry value to disable native application mode by setting its value to `true`. You can create this HKLM registry key when you install the AppStream 2.0 client. If the value is set to `true`, the client runs in classic mode only. For information about native application mode, see Native Application Mode (p. 175).

After the AppStream 2.0 client is installed, you can run the following PowerShell script to create this registry key. Alternatively, you can use the administrative template that is included in the AppStream 2.0 client Enterprise Deployment Tool to configure the client through Group Policy.

**Note**

To run this script, you must be logged into the applicable computer with **Administrator** permissions. You can also run the script remotely under the **System** account on startup.

```
$registryPath="HKLM:\Software\Amazon\AppStream Client"
New-Item -Path "$registryPath" -Name "AppStream Client" -Force
New-ItemProperty -Path $registryPath -Name "NativeAppModeDisabled" -Value "True" -PropertyType String -Force | Out-Null
```

Configure Additional AppStream 2.0 Client Settings for Your Users

The AppStream 2.0 client uses registry keys to configure the following additional client settings:

- AppStream 2.0 client End-User License Agreement (EULA) acceptance
- AppStream 2.0 client EULA version accepted
- Diagnostic logging for the AppStream 2.0 client
- Automatic updates for the USB driver that is used to pass USB drivers to AppStream 2.0
- Enabling hardware rendering in the AppStream 2.0 client
- Setting custom folder paths for file system redirection in the AppStream 2.0 client

The following table summarizes the registry values for additional client settings that you can use to customize the AppStream 2.0 client experience for your users.
Note
These values are case-sensitive.

<table>
<thead>
<tr>
<th>Value</th>
<th>Registry path</th>
<th>Type</th>
<th>Description</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>EULAAccepted</td>
<td>HKCU\Software \Amazon \Appstream \Client</td>
<td>String</td>
<td>Set this value to true to accept the AppStream 2.0 client EULA on behalf of your users.</td>
<td>true / false</td>
</tr>
<tr>
<td>AcceptedEULAVersion</td>
<td>HKCU\Software \Amazon \Appstream \Client</td>
<td>String</td>
<td>The version of EULA that is accepted. If the current version of the AppStream 2.0 client EULA is different than the version of the EULA that is accepted, users are prompted to accept the current version of the EULA.</td>
<td>1.0</td>
</tr>
<tr>
<td>DiagnosticInfoCollectionAllowed</td>
<td>HKCU\Software \Amazon \Appstream \Client</td>
<td>String</td>
<td>Set this value to true to enable AppStream 2.0 to automatically send diagnostic logs from the AppStream 2.0 client to AppStream 2.0.</td>
<td>true / false</td>
</tr>
<tr>
<td>USBDriverOptIn</td>
<td>HKCU\Software \Amazon \Appstream \Client</td>
<td>String</td>
<td>Set this value to true to enable AppStream 2.0 to automatically update the USB driver that is used to pass USB drivers to AppStream 2.0.</td>
<td>true / false</td>
</tr>
<tr>
<td>HardwareRenderingEnabled</td>
<td>HKCU\Software \Amazon \Appstream \Client</td>
<td>String</td>
<td>Set this value to true to enable hardware rendering in the AppStream 2.0 client.</td>
<td>true / false</td>
</tr>
<tr>
<td>FileRedirectionCustomDefaultFolders</td>
<td>HKCU\Software \Amazon \Appstream \Client</td>
<td>String</td>
<td>Set this value to include at least one folder path Valid folder path</td>
<td></td>
</tr>
</tbody>
</table>
### Value Registry path Type Description Data

| \Appstream Client | \Appstream Client | for file system redirection. Separate multiple folder paths by using '|'. By default, the following folder paths are specified: | \Appstream Client for file system redirection. Separate multiple folder paths by using '|'. By default, the following folder paths are specified: | \Appstream Client for file system redirection. Separate multiple folder paths by using '|'. By default, the following folder paths are specified: |

After the AppStream 2.0 client is installed, you can run the following PowerShell script to create these registry keys. If you don’t want to create all of the registry keys, modify the script as needed to create only the registry keys that you want. Alternatively, you can use the administrative template that is provided in the AppStream 2.0 client Enterprise Deployment Tool to configure the client through Group Policy.  

**Note**  
You must set the following entries for each user.

```powershell
$registryPath = "HKCU:\Software\Amazon\AppStream Client"  
New-Item -Path "$registryPath" -Name "AppStream Client" -Force  
New-ItemProperty -Path $registryPath -Name "EULAAccepted" -Value "true" -PropertyType String -Force | Out-Null  
New-ItemProperty -Path $registryPath -Name "AcceptedEULAVersion" -Value "1.0" -PropertyType String -Force | Out-Null  
New-ItemProperty -Path $registryPath -Name "DiagnosticInfoCollectionAllowed" -Value "true" -PropertyType String -Force | Out-Null  
New-ItemProperty -Path $registryPath -Name "USBDriverOptIn" -Value "true" -PropertyType String -Force | Out-Null  
New-ItemProperty -Path $registryPath -Name "HardwareRenderingEnabled" -Value "true" -PropertyType String -Force | Out-Null  
New-ItemProperty -Path $registryPath -Name "FileRedirectionCustomDefaultFolders" -Value "%USERPROFILE%\Desktop|%USERPROFILE%\Documents|%USERPROFILE%\Downloads" -PropertyType String -Force | Out-Null  
```

### Using Group Policy to Customize AppStream 2.0 Client Experience

You can use the administrative template that is provided in the AppStream 2.0 client Enterprise Deployment Tool to configure the client through Group Policy. To learn how to load administrative templates into the Group Policy Management Console, see Recommendations for managing Group Policy administrative template (.adm) files in the Microsoft Support documentation.

### Update the AppStream 2.0 Enterprise Deployment Tool, Client, and USB Driver Manually

By default, the AppStream 2.0 client and USB driver are updated automatically when a new client version is released. However, if you used the Enterprise Deployment Tool to install the AppStream 2.0 client for your users and you disabled automatic updates, you must update the AppStream 2.0 Enterprise Deployment Tool, client, and USB driver manually. To do so, perform the following steps to run the required PowerShell commands on users’ computers.
Install and Configure the AppStream 2.0 Client

Note
To run these commands, you must either be logged into the applicable computer as Administrator, or you can run the script remotely under the SYSTEM account on startup.

1. Uninstall the AppStream 2.0 Enterprise Deployment Tool silently:
   ```powershell
   Start-Process msiexec.exe -Wait -ArgumentList '/x AmazonAppStreamClientSetup_<existing_version>.msi /quiet'
   ```

2. Uninstall the AppStream 2.0 USB driver silently:
   ```powershell
   Start-Process -Wait AmazonAppStreamUsbDriverSetup_<existing_version>.exe -ArgumentList '/uninstall /quiet /norestart'
   ```

3. Uninstall the AppStream 2.0 client silently:
   ```powershell
   Start-Process "$env:LocalAppData\AppStreamClient\Update.exe" -ArgumentList '--uninstall'
   ```
   Note
   This process also removes the registry keys that are used to configure the AppStream 2.0 client. After you reinstall the AppStream 2.0 client, you must recreate these keys.

4. Clean the application installation directory:
   ```powershell
   Remove-Item -Path $env:LocalAppData\AppStreamClient -Recurse -Confirm:$false –Force
   ```

5. Restart the computer:
   ```powershell
   Restart-computer
   ```

6. Install the latest version of the AppStream 2.0 Enterprise Deployment Tool silently:
   ```powershell
   Start-Process msiexec.exe -Wait -ArgumentList '/i AmazonAppStreamClientSetup_<new_version>.msi /quiet'
   ```

7. Install the latest version of the AppStream 2.0 USB driver silently:
   ```powershell
   Start-Process AmazonAppStreamUsbDriverSetup_<new_version>.exe -Wait -ArgumentList '/quiet'
   ```

Qualify USB Devices for Use with Streaming Applications

To qualify your users' USB devices so that the devices can be used with streaming applications, perform these steps.

1. If you haven't already done so, install the AppStream 2.0 client. For information, see Install and Configure the AppStream 2.0 Client (p. 176).
2. Connect the USB device that you want to qualify to your computer.
3. Navigate to `C:\Users\<logged-in-user>\AppData\Local\AppStreamClient`, and double-click `dcvusblist.exe`.
4. In the DCV - USB devices dialog box, the list of USB devices that are connected to your local computer displays. The Filter column displays the filter string for every USB device. Right-click the list entry for a USB device that you want to enable, and choose Copy filter string.
5. On your desktop, choose the Windows Start button, and search for Notepad. Double-click Notepad to open a new file, copy the filter string to the file, and save it. Later, you'll use the filter string to qualify the USB device.

6. Launch a new image builder. For more information, see Launch an Image Builder to Install and Configure Streaming Applications (p. 26).

7. After your image builder is in the Running state, perform the following steps to create a streaming URL and connect to the image builder by using the AppStream 2.0 client.
   a. With your image builder selected in the list, choose Actions, Create streaming URL.
   b. In the Create streaming URL dialog box, choose Copy link, and copy and paste the web address into a separate file for later use. You'll use this URL to reconnect to the image builder in step 12.
   c. Choose Launch in Client.
   d. If the Launch Application dialog box displays and prompts you to choose the application to use when opening the link, choose Amazon AppStream, Open link. To prevent this dialog box from displaying the next time you perform this step to connect to an image builder, select the Remember my choice for amazonappstream links check box.
   e. If the AppStream 2.0 client displays links to the AWS Customer Agreement, AWS Service Terms, and the AWS Privacy Notice, and third-party notices, review this information, and then choose Finish.
   f. If the client sign in page displays, the web address field is prepopulated with the streaming URL. Choose Connect.
   g. If prompted, log in to the image builder as Administrator.

8. After you are connected to the image builder, if your USB device requires you to install drivers before you use it, download and install the drivers on the image builder. For example, if you use the Connexion 3D mouse, you must download and install the required Connexion drivers on the image builder.

9. On your image builder desktop, choose the Windows Start button, and search for Notepad. Right-click Notepad, and choose Run as Administrator.

10. Choose File, Open, and open the following file: C:\ProgramData\Amazon\Photon\DCV\usb_device_whitelist.txt. You can also allow an entire category of devices or all devices from a specific manufacturer by using wildcard expressions in the usb_device_whitelist.txt file.

11. Copy the filter string from your local computer to the image builder. The filter string for a specific USB device is a comma-separated string of the following fields: Name, Base Class, SubClass, Protocol, ID Vendor, ID Product, Support Autoshare, and Skip Reset. For detailed information about these strings, see the next section, Working with USB Device Filter Strings (p. 184).

12. Disconnect from your image builder, restart it, and reconnect to it by using the AppStream 2.0 client. To do so, open the AppStream 2.0 and paste the streaming URL that you created in step 7 into the client sign in web address field, and choose Connect.

13. On the image builder, test your USB device to confirm that it works as expected.

14. Before your users can use the USB device in an AppStream 2.0 session, they must share the device with their session first. After you finish testing, for guidance that you can provide your users to help them perform this task, see USB Devices (p. 277).

15. If the USB device works with the image builder as expected, create an image. For more information, see Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47).

16. After you finish creating the image, update your AppStream 2.0 fleet to use the new image.

**Working with USB Device Filter Strings**

This section describes the filter strings that are available for qualifying USB devices for AppStream 2.0 streaming sessions. It also provides guidance for working with these strings. The following filter strings are available:
Install and Configure the AppStream 2.0 Client

- **Name** — By default, the value for this filter string is the name of the device. You can optionally specify your own value.

- **Base Class,SubClass,Protocol** — The USB class code for the device. For more information, see Defined Class Codes.

- **ID Vendor (VID)** — A unique identifier that is assigned by the USB organization to the manufacturer of the USB device.

- **ID Product (PID)** — A unique identifier that assigned by the manufacturer to the USB device.

- **Support Autoshare** — Lets the AppStream 2.0 client automatically share the device when a streaming session starts. Set this value to 1 to allow automatic device sharing or to 0 to not allow automatic device sharing.

- **Skip Reset** — By default, when a USB device is shared by AppStream 2.0 with a streaming session, the device is reset to ensure that it functions correctly. However, some USB devices don't function correctly during the streaming session if they are reset. To prevent this problem from occurring, set the value for this filter string to 1 to instruct the AppStream 2.0 client not to reset the device while it is shared with a streaming session. To ensure that the device is reset while it is shared with a streaming session, set this value to 0. When you set a value for **Skip Reset**, make sure that you set the value for **Support Autoshare** to 0 or 1.

The filter string that is copied from the local computer is specific to a USB device. In some cases, you may want to allow an entire class of devices instead of allowing every possible USB device. For example, you may want to allow your users to use any kind of WACOM design tablets or use any USB mass storage device. In such scenarios, you can provide wildcard characters for specific filter string fields. If you don't know the VID and PID for your USB devices, you can search for this information in the USB ID database.

The following examples show how to configure filter strings for USB device sharing during streaming sessions:

- Allow all mass storage devices automatically on starting a streaming session — "Mass storage, 8, *, *, *, *,1,0"
- Allow all Wacom devices automatically on starting a streaming session — "Wacom Tablets, 3, *, *, 1386, *,1,0"
- Allow all devices that provide an audio interface - "Audio, 1, *, *, *, *,1,0"
- Allow device X but don't reset it while the device is shared. Also don't share the device automatically on starting a streaming session — "X, Y, *, *, 1386, *,0,1"

**Use the AppStream 2.0 Client to Start a Streaming Session**

After you install the AppStream 2.0 client on your users' local computers, they can use the client to connect to a streaming session. You can configure one or more of the following connection methods, as required for your organization:

- **The AppStream 2.0 API**
- **SAML 2.0 (p. 244)**
- **The AppStream 2.0 user pool (console) (p. 239)**

For each of these connection methods, the user experience is as follows:

**AppStream 2.0 API**

In the AppStream 2.0 client sign-in page, users enter the streaming URL that you created as the web address, and then choose **Connect**.
SAML 2.0

If you use SAML 2.0 to federate your users to an AppStream 2.0 stack, you must create a registry value to configure the AppStream 2.0 client with a pre-populated URL whenever the client is launched. The URL must use a certificate that is trusted by the device. The certificate must contain a Subject Alternative Name (SAN) that includes the URL's domain name. For more information, see Set the StartURL Registry Value for AppStream 2.0 Client Users (p. 178).

User pool

In the AppStream 2.0 client sign-in page, users enter the URL that was provided in your welcome email, and then choose Connect.

**Note**

For step-by-step guidance that you can provide your users to help them with connecting to AppStream 2.0 and starting a streaming session, see the following topics:

- Connect by Using a Web Browser (p. 273)
- Connect by Using the AppStream 2.0 Client for Windows (p. 276)

Share a USB Device with an AppStream 2.0 Streaming Session

Before users share their USB devices with an AppStream 2.0 session, the USB devices must be qualified. Otherwise, when users start a streaming session, their USB device is not detected by AppStream 2.0 and cannot be shared with the session. For more information, see Qualify USB Devices for Use with Streaming Applications (p. 183).

Redirect a Streaming Session from the Web Browser to the AppStream 2.0 Client

You can configure AppStream 2.0 to redirect a streaming session from a web browser to the AppStream 2.0 client. That way, when your users sign in to AppStream 2.0 and start a streaming session in their web browser, their session is redirected to the AppStream 2.0 client. To do so, perform these steps.

1. Use the AppStream 2.0 CreateStreamingURL API action to generate a streaming URL.
2. Add the following prefix for the custom AppStream 2.0 client handler to the streaming URL:

   ```
   amazonappstream:
   ```

   Together, the prefix and streaming URL are formatted as follows:

   ```
   amazonappstream:base64encoded(streamingURL)
   ```

3. When users are redirected to the streaming URL, their browser detects that the link must be opened by the AppStream 2.0 client.
4. Users are prompted to choose whether they want to start the streaming session by using the AppStream 2.0 client.
5. After the prompt, either of the following occurs:

   - If the AppStream 2.0 client is installed by the user, they can choose to continue the streaming session by using the AppStream 2.0 client.
   - If the AppStream 2.0 client is not installed, the browser behavior varies as follows:
     - Chrome — No message displays.
     - Firefox — A message states that the user will need a new app to open Amazon AppStream.
     - Microsoft Edge — No message displays.
     - Internet Explorer — A message notifies the user that the AppStream 2.0 client is not installed.
In this case, users can select the Download AppStream Client link to download the client. After they download the client, they can install it, and refresh their browser to start the streaming session by using the client.

Enable File System Redirection for Your AppStream 2.0 Users

AppStream 2.0 file system redirection lets users who have the AppStream 2.0 client installed access files on their local computer from within their streaming session. When you enable file system redirection, you can specify the list of local drives and folders that your users can choose to access. When users sign in to AppStream 2.0 and start a streaming session, they can choose the drive or folder that they want to access from the list. Then they can share the drive or folder with AppStream 2.0. The drive or folder remains available for them to access during their streaming sessions. Users can stop sharing their local drives or folders at any time.

Prerequisites for File System Redirection

To enable AppStream 2.0 file redirection, you must meet the following prerequisites:

- You must use an image that uses a version of the AppStream 2.0 agent released on or after August 8, 2019. For more information, see AppStream 2.0 Agent Version History (p. 39).
- Your users must have AppStream 2.0 client version 1.0.480 or later installed. For more information, see AppStream 2.0 Client Version History (p. 197).
- File upload and download must be enabled on the stack that your users access for streaming sessions. For instructions, see the next section.

How to Enable File System Redirection

Perform the following steps to enable both file upload and download on the stack that your users access for streaming sessions.

2. In the left navigation pane, choose Stacks.
3. Choose the stack for which you want to enable file system redirection.
4. Choose the User Settings tab, and then expand the Clipboard, file transfer, and local print permissions section.
5. Verify that Upload and download is selected for File transfer. If not, choose Edit, and then choose Upload and download.
6. Choose Update.

Make default drives and folders available for your users to share

By default, when you enable file direction for users of a stack, the following drives and folders are made available for those users to share within their streaming session:

- Drives:
  - All local hard disks (physical drives, such as the C Drive and D Drive)
  - All virtual drives (network and virtual drives such as mapped drive letters, Google Drive, and OneDrive)
  - All local USB drives
- Folders:
  - %USERPROFILE%\Desktop
These drive and folder paths prepopulate the **Share your local drives and folders** dialog box. This dialog box displays when users sign in to AppStream 2.0, start a streaming session, and choose **Settings, Local Resources**, and **Local Drives and Folders**.

You can change or define your own default drive and folder paths by modifying the registry. You can also use the administrative template file that is provided in the AppStream 2.0 client Enterprise Deployment Tool. This template lets you configure the client by using Group Policy. For more information, see *Configure the AppStream 2.0 Client for Your Users* in *Install and Configure the AppStream 2.0 Client* (p. 176).

When users access their shared local drives and folders during a streaming session, the corresponding paths appear with backslashes replaced by underscores. They are also suffixed with the name of the local computer and a drive letter. For example, for a user with the user name janedoe and a computer name of ExampleCorp-123456, the default Desktop, Documents, and Downloads folder paths appear as follows:

- C_Users_janedoe_Desktop (\ExampleCorp-123456) (F:)
- C_Users_janedoe_Documents (\ExampleCorp-123456) (G:)
- C_Users_janedoe_Downloads (\ExampleCorp-123456) (H:)

**Provide Your AppStream 2.0 Users with Guidance for Working with File System Redirection**

To help your users understand how to work with file redirection during their streaming sessions, you can provide them with the following information.

**Guidance for Users**

AppStream 2.0 file redirection lets you access files on your local computer from within your AppStream 2.0 streaming session. To use file redirection, open the AppStream 2.0 client, connect to a streaming session, and choose the drives and folders that you want to share. After you share a local drive or folder, you can access all files in the shared drive or folder from within your streaming session. You can stop sharing local drives and folders at any time.

**Important**

To use AppStream 2.0 file redirection, you must have the AppStream 2.0 client installed on your local computer. File redirection is not available when you connect to AppStream 2.0 by using a web browser.

**To share local drives and folders**

1. Open the AppStream 2.0 client and connect to a streaming session.
2. In your AppStream 2.0 session, in the top left area, choose the **Settings** icon, and then choose **Local Resources, Local Drives and Folders**.
The Share your local drives and folders dialog box displays the drives and folders that your administrator has made available for you to share. You can share all or specific drives and folders, or just one. You can also add your own drives and folders. To share drives and folders, do one of the following:

- To share all local drives and folders displayed in the Share your local drives and folders dialog box, in the upper right of the dialog box, choose Share All. To apply your changes to future streaming sessions, in the lower right of the dialog box, choose Save my configuration to save your changes.
• To share a specific local drive or folder, select the drive or folder that you want to access, and choose **Share**, **Save my configuration**. To share another local drive or folder, repeat these steps as needed.
If the local drive or folder that you want to share doesn't display, you can add it. For example, your administrator may make your entire local C Drive available for you to share. However, you may only need to access a specific folder on that drive. In this case, you can add the folder that you need and share the folder only. To choose a specific folder, do the following:

- In the lower left of the **Share your local drives and folders** dialog box, choose the **Add Folder** icon.
• Browse to the folder that you want to share, and choose **OK**.

• The folder that you selected is now available to share. Select the folder, and choose **Share**, **Save my configuration**. To add another local drive or folder, repeat these steps as needed.
After you share a local drive or folder, perform the following steps to access files in the shared drive or folder from within your streaming session.

### To access files in a shared local drive or folder

1. Open the AppStream 2.0 client and connect to a streaming session.
2. In your AppStream 2.0 session, open the application that you want to use.
3. From your application interface, choose **File Open**, and browse to the file that you want to access. The following screenshot shows how shared local drives and folders appear in the Notepad++ browse dialog box for Jane Doe, when she browses for a file.
In the browse dialog box, the corresponding paths for her shared drives and folders are shown in the red box. The paths appear with backslashes replaced by underscores. At the end of each path is the name of Jane's computer, ExampleCorp-123456, and a drive letter.

4. When you're done working with the file, use the File Save or File Save As command to save it to the location that you want.

If you want to stop sharing a local drive or folder, perform the following steps.

**To stop sharing local drives and folders**

1. Open the AppStream 2.0 client and connect to a streaming session.
2. In your AppStream 2.0 session, in the top left area, choose the Settings icon, and then choose Local Resources, Local Drives and Folders.

   The Share your local drives and folders dialog box displays the drives and folders that your administrator has made available for you to share, and any that you added, if applicable. To stop sharing one or more local drives and folders, do either of the following:

   - To stop sharing all shared local drives and folders, choose Unshare All, Save my configuration.
To stop sharing a specific shared local drive or folder, select the drive or folder, and choose **Unshare**, **Save my configuration**. To stop sharing another local drive or folder, repeat these steps as needed.
You can delete local drives and folders that you add to the **Share your local drives and folders** dialog box. However, you can't delete local drives or folders that your administrator has made available for you to share. Also, if you have already shared a local drive or folder, you must stop sharing it before you can delete it.

**To delete local drives and folders**

1. Open the AppStream 2.0 client and connect to a streaming session.
2. In your AppStream 2.0 session, in the top left area, choose the **Settings** icon, and then choose **Local Resources, Local Drives and Folders**.
   
   The **Share your local drives and folders** dialog box displays the drives and folders that your administrator has made available for you to share. If you added any drives or folders, they display as well.
3. Select the local drive or folder that you want to delete, and then choose **Delete, Save my configuration**.
AppStream 2.0 Client Version History

The AppStream 2.0 client is a native application that is designed for users who require additional functionality during their AppStream 2.0 streaming sessions. The following table describes the latest updates that are available in released versions of the AppStream 2.0 client.

For more information about the client, see Provide Access Through the AppStream 2.0 Client for Windows (p. 174).

<table>
<thead>
<tr>
<th>Client version</th>
<th>Release date</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1.137</td>
<td>03-08-2020</td>
<td>• Reverts the updates in version 1.1.136</td>
</tr>
</tbody>
</table>
| 1.1.136        | 03-05-2020   | • Adds support for defining trusted subdomains for user connections in a DNS TXT record  
                     • Users can choose to upload diagnostic logs |
<table>
<thead>
<tr>
<th>Client version</th>
<th>Release date</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>to AWS automatically, or manually, on an as-needed basis</td>
</tr>
</tbody>
</table>
| 1.1.129       | 02-28-2020   | • Adds support for native application mode  
• Improves the user interface for the DCV Printer experience  
• Resolves an issue with using Surface Pro Pen with streaming applications  
• Resolves an issue with downloading files with file names that have international characters |
| 1.0.525       | 12-12-2019   | • Resolves a DPI issue that causes the mouse cursor to point to the wrong location when a user clicks on an application during a streaming session |
| 1.0.511       | 10-16-2019   | • Adds support for up to 4 monitors with a maximum display resolution of 2560x1440 pixels per monitor  
• Adds support for up to 2 monitors with a maximum display resolution of 4096x2160 pixels per monitor on the Graphics Design and Graphics Pro instance types  
• Adds support for seamless user connections to streaming sessions that were started using custom uniform resource identifier (URI) redirects  
• Adds support for adding trusted domains for start URLs |
<table>
<thead>
<tr>
<th>Client version</th>
<th>Release date</th>
<th>Changes</th>
</tr>
</thead>
</table>
| 1.0.499        | 09-26-2019   | • Resolves an issue with client-side hardware rendering  
• Resolves an issue with the client not working correctly when Bluetooth headsets are connected to the local computer |
| 1.0.480        | 08-20-2019   | • Adds support for AppStream 2.0 file system redirection |
| 1.0.467        | 07-29-2019   | • Includes fixes and enhancements to ensure compatibility with updates made to AppStream 2.0 portal endpoints |
| 1.0.407        | 05-16-2019   | • Adds support for configuring the amount of time that users can be idle (inactive) before they are disconnected from their streaming session. For more information, see “Create a Fleet” in Create an AppStream 2.0 Fleet and Stack (p. 81).  
• Resolves an issue with the “session alert” window appearing when a SAML 2.0 session has expired  
• Includes bug fixes for printing a document to a print server |
| 1.0.375        | 03-07-2019   | • Adds touch screen support on Windows PCs  
• Adds support for automatically connecting USB devices when a new streaming session starts  
• Adds support for running session scripts  
• Adds support for delivering virtualized applications using the AppStream 2.0 dynamic application framework APIs |
<table>
<thead>
<tr>
<th>Client version</th>
<th>Release date</th>
<th>Changes</th>
</tr>
</thead>
</table>
| 1.0.320        | 01-19-2019   | • Adds multi monitor support for Graphics Design instances  
• Adds support for client display scaling factors greater than 100 percent  
• Adds support for AppStream 2.0 regional settings  
• Adds support for the AppStream 2.0 user pool  
• Adds support for honoring client-side proxy settings |
| 1.0.247        | 11-20-2018   | Initial release |
Tagging Your Amazon AppStream 2.0 Resources

AWS enables you to assign metadata to your AWS resources in the form of tags. You can use these tags to help manage your AppStream 2.0 image builders, images, fleets, and stacks, and also organize data, including billing data.

You can:

- Logically group resources in different ways (for example, by purpose, owner, or environment).
  
  This is useful when you have many resources of the same type.
- Quickly identify a specific resource based on the tags that you've assigned to it
- Identify and control AWS costs

For example, you can identify and group AppStream 2.0 fleets that are in different environments (such as Development or Production) or that are assigned to different business units (such as HR or Marketing). You can then track the associated AWS costs for these fleets on a detailed level. To do this, sign up to get your AWS account bill with tag key values included. For more information about setting up a cost allocation report with tags, see Monthly Cost Allocation Report in the AWS Billing and Cost Management User Guide.

Contents

- Tagging Basics (p. 201)
- Tag Restrictions (p. 202)
- Adding Tags during Resource Creation in the AppStream 2.0 Console (p. 202)
- Adding, Editing, and Deleting Tags for Existing Resources in the AppStream 2.0 Console (p. 202)
- Working with Tags by Using the AppStream 2.0 API, an AWS SDK, or AWS CLI (p. 203)

Tagging Basics

Tags consist of a key-value pair, similar to other AWS services tags. To tag a resource, you specify a key and a value for each tag. A key can be a general category, such as "project", "owner", or "environment", with specific associated values, and you can share the same key and value across multiple resources. You can tag an AppStream 2.0 resource immediately after you create it or later on. If you delete a resource, the tags are removed from that resource on deletion. However, other AppStream 2.0 and AWS resources that have the same tag key are not impacted.

You can edit tag keys and values, and you can remove tags from a resource at any time. You can set the value of a tag to an empty string, but you can't set the name of a tag to null. If you add a tag that has the same key as an existing tag on that resource, the new value overwrites the old value. If you delete a resource, any tags for the resource are also deleted.

Note

If you plan to set up a monthly cost allocation report to track AWS costs for AppStream 2.0 resources, keep in mind that tags added to existing AppStream 2.0 resources appear in your
Tag Restrictions

- The maximum number of tags per AppStream 2.0 resource is 50.
- The maximum key length is 128 Unicode characters in UTF-8.
- The maximum value length is 256 Unicode characters in UTF-8.
- Tag keys and values are case-sensitive.
- Do not use the "aws:" prefix in your tag names or values because it is a system tag that is reserved for AWS use. You cannot edit or delete tag names or values with this prefix. Tags with this prefix do not count against your tags per resource limit.
- Generally allowed characters are: letters, numbers, and spaces representable in UTF-8, and the following special characters: + - = . _ : / @.
- Although you can share the same key and value across multiple resources, you cannot have duplicate keys on the same resource.
- You can add tags for resources during resource creation. You can also add, edit, and delete tags for resources that are already created.

Adding Tags during Resource Creation in the AppStream 2.0 Console

When you create a resource in the AppStream 2.0 console, you can add one or more tags to manage the resource. For more information, see the following topics:

- Image builders — Launch an Image Builder to Install and Configure Streaming Applications (p. 26), step 4
- Images — Step 6: Finish Creating Your Image (p. 50), step 1
- Fleets — Create a Fleet (p. 82), step 3
- Stacks — Create a Stack (p. 84), step 2

Adding, Editing, and Deleting Tags for Existing Resources in the AppStream 2.0 Console

You can add, edit, and delete tags for existing resources by using the AppStream 2.0 console.

To add, edit, or delete tags for an existing AppStream 2.0 resource

2. From the navigation bar, select the Region that contains the resource for which you want to add, edit, or delete tags.
3. In the navigation pane, select the resource type. The resource type can be an image builder, image, fleet, or stack.
4. Select the resource from the resource list.
5. Choose **Tags, Add/Edit Tags**, and then do one or more of the following:
   - To add a tag, choose **Add Tag**, and type the key and value for each tag.
   - To edit a tag, modify the key and value for the tag as needed.
   - To delete a tag, choose the **Delete** icon (X) for the tag.
6. Choose **Save**.

### Working with Tags by Using the AppStream 2.0 API, an AWS SDK, or AWS CLI

If you're using the AppStream 2.0 API, an AWS SDK, or the AWS Command Line Interface (AWS CLI), you can use the following AppStream 2.0 operations with the `tags` parameter to add tags when you create new resources.

**Note**
You can use spaces in tag keys and values. To indicate a space when you use the AWS CLI, use "\s" (without the quotation marks).

<table>
<thead>
<tr>
<th>Task</th>
<th>AWS CLI</th>
<th>API Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add one or more tags for a new fleet</td>
<td>create-fleet</td>
<td>CreateFleet</td>
</tr>
<tr>
<td>Add one or more tags for a new image builder</td>
<td>create-imagebuilder</td>
<td>CreateImageBuilder</td>
</tr>
<tr>
<td>Add one or more tags for a new stack</td>
<td>create-stack</td>
<td>CreateStack</td>
</tr>
</tbody>
</table>

You can use the following AppStream 2.0 operations to add, edit, remove, or list tags for existing resources:

<table>
<thead>
<tr>
<th>Task</th>
<th>AWS CLI</th>
<th>API Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add or overwrite one or more tags for a resource</td>
<td>tag-resource</td>
<td>TagResource</td>
</tr>
<tr>
<td>Remove one or more tags for a resource</td>
<td>untag-resource</td>
<td>UntagResource</td>
</tr>
<tr>
<td>List one or more tags for a resource</td>
<td>list-tags-for-resource</td>
<td>ListTagsForResource</td>
</tr>
</tbody>
</table>

When you use the AppStream 2.0 API, an AWS SDK, or AWS CLI actions to add, edit, remove, or list tags for an existing AppStream 2.0 resource, specify the resource by using its Amazon Resource Name (ARN). An ARN uniquely identifies an AWS resource and uses the following general syntax.

```
arn:aws:appstream:region:account:resourceType/resourceName
```
Working with Tags by Using the AppStream 2.0 API, an AWS SDK, or AWS CLI

**region**

The AWS Region in which the resource was created (for example, us-east-1).

**account**

The AWS account ID, with no hyphens (for example, 123456789012).

**resourceType**

The type of resource. You can tag the following AppStream 2.0 resource types: image-builder, image, fleet, and stack.

**resourceName**

The name of the resource.

For example, you can obtain the ARN for an AppStream 2.0 fleet by using the AWS CLI `describe-fleets` command. Copy the following command.

```
aws appstream describe-fleets
```

For an environment that contains a single fleet named `TestFleet`, the ARN for this resource would appear in the JSON output similar to the following.

```
"Arn": "arn:aws:appstream:us-east-1:123456789012:fleet/TestFleet"
```

After you obtain the ARN for this resource, you can add two tags by using the `tag-resource` command:

```
aws appstream tag-resource --resource arn:aws:appstream:us-east-1:123456789012:fleet/TestFleet --tags Environment=Test,Department=IT
```

The first tag, `Environment=Test`, indicates that the fleet is in a test environment. The second tag, `Department=IT`, indicates that the fleet is in the IT department.

You can use the following command to list the two tags that you added to the fleet.

```
aws appstream list-tags-for-resource --resource arn:aws:appstream:us-east-1:123456789012:fleet/TestFleet
```

For this example, the JSON output appears as follows:

```
{
    "Tags": {
        "Environment": "Test",
        "Department": "IT"
    }
}
```
Monitoring and Reporting

Monitoring and reporting are an important part of maintaining the reliability, availability, and performance of your Amazon AppStream 2.0 streaming instances and providing your users with a responsive streaming experience.

Contents
- Monitoring Amazon AppStream 2.0 Resources (p. 205)
- AppStream 2.0 Usage Reports (p. 207)
- Logging AppStream 2.0 API Calls with AWS CloudTrail (p. 218)

Monitoring Amazon AppStream 2.0 Resources

AppStream 2.0 publishes metrics to Amazon CloudWatch to enable detailed tracking and deep dive analysis. These statistics are recorded for an extended period so you can access historical information and gain a better perspective on how your fleets are performing. For more information, see the Amazon CloudWatch User Guide.

Viewing Fleet Usage Using the Console

You can monitor Amazon AppStream 2.0 using the AppStream 2.0 console or the CloudWatch console.

To view fleet usage in the AppStream 2.0 console
2. In the left pane, choose Fleets.
3. Select a fleet and choose its Fleet Usage tab.
4. By default, the graph displays the following metrics: ActualCapacity, InUseCapacity, and CapacityUtilization. You can select additional metrics to graph or change the period.

To view fleet usage in the CloudWatch console
2. In the left pane, choose Metrics.
3. Choose the AppStream namespace and then choose Fleet Metrics.
4. Select the metrics to graph.

AppStream 2.0 Metrics and Dimensions

Amazon AppStream 2.0 sends the following metrics and dimension information to Amazon CloudWatch.

Amazon AppStream 2.0 Metrics

AppStream 2.0 sends metrics to CloudWatch one time every minute. The AWS/AppStream namespace includes the following metrics.
<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActualCapacity</td>
<td>The total number of instances that are available for streaming or are currently streaming.</td>
</tr>
<tr>
<td></td>
<td>ActualCapacity = AvailableCapacity + InUseCapacity</td>
</tr>
<tr>
<td></td>
<td>Units: Count</td>
</tr>
<tr>
<td></td>
<td>Valid statistics: Average, Minimum, Maximum</td>
</tr>
<tr>
<td>AvailableCapacity</td>
<td>The number of idle instances currently available for user sessions.</td>
</tr>
<tr>
<td></td>
<td>AvailableCapacity = ActualCapacity - InUseCapacity</td>
</tr>
<tr>
<td></td>
<td>Units: Count</td>
</tr>
<tr>
<td></td>
<td>Valid statistics: Average, Minimum, Maximum</td>
</tr>
<tr>
<td>CapacityUtilization</td>
<td>The percentage of instances in a fleet that are being used, using the following formula.</td>
</tr>
<tr>
<td></td>
<td>CapacityUtilization = (InUseCapacity/ActualCapacity) * 100</td>
</tr>
<tr>
<td></td>
<td>Monitoring this metric helps with decisions about increasing or decreasing the value of a fleet's desired capacity.</td>
</tr>
<tr>
<td></td>
<td>Units: Percent</td>
</tr>
<tr>
<td></td>
<td>Valid statistics: Average, Minimum, Maximum</td>
</tr>
<tr>
<td>DesiredCapacity</td>
<td>The total number of instances that are either running or pending. This represents the total number of concurrent streaming sessions your fleet can support in a steady state.</td>
</tr>
<tr>
<td></td>
<td>DesiredCapacity = ActualCapacity + PendingCapacity</td>
</tr>
<tr>
<td></td>
<td>Units: Count</td>
</tr>
<tr>
<td></td>
<td>Valid statistics: Average, Minimum, Maximum</td>
</tr>
<tr>
<td>InUseCapacity</td>
<td>The number of instances currently being used for streaming sessions. One InUseCapacity count represents one streaming session.</td>
</tr>
<tr>
<td></td>
<td>Units: Count</td>
</tr>
<tr>
<td></td>
<td>Valid statistics: Average, Minimum, Maximum</td>
</tr>
<tr>
<td>PendingCapacity</td>
<td>The number of instances being provisioned by AppStream 2.0. Represents the additional number of streaming sessions the fleet can support after provisioning is complete. When provisioning starts, it usually takes 10-20 minutes for an instance to become available for streaming.</td>
</tr>
</tbody>
</table>
### Metric & Description

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RunningCapacity</td>
<td>The total number of instances currently running. Represents the number of concurrent streaming sessions that can be supported by the fleet in its current state. This metric is provided for Always-On fleets only, and has the same value as the ActualCapacity metric.</td>
</tr>
<tr>
<td></td>
<td>Units: Count</td>
</tr>
<tr>
<td>InsufficientCapacity</td>
<td>The number of session requests rejected due to lack of capacity. You can set alarms to use this metric to be notified of users waiting for streaming sessions.</td>
</tr>
<tr>
<td></td>
<td>Units: Count</td>
</tr>
</tbody>
</table>

### Dimensions for Amazon AppStream 2.0 Metrics

To filter the metrics provided by Amazon AppStream 2.0, use the following dimension.

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fleet</td>
<td>The name of the fleet.</td>
</tr>
</tbody>
</table>

### AppStream 2.0 Usage Reports

You can subscribe to Amazon AppStream 2.0 usage reports to receive detailed reports about how your users are using the service. Each day that users launch one or more streaming sessions in your AWS account, two .csv files are exported to your Amazon Simple Storage Service (Amazon S3) account.

**Note**

To enable AppStream 2.0 usage reports, you must use an image that uses a version of the AppStream 2.0 agent released on or after May 7, 2019.

### Contents

- Enable AppStream 2.0 Usage Reports (p. 207)
- AppStream 2.0 Usage Reports Fields (p. 209)
- Create Custom Reports and Analyze AppStream 2.0 Usage Data (p. 213)

### Enable AppStream 2.0 Usage Reports

To receive usage reports, you subscribe to them by using the AppStream 2.0 console, the AWS Command Line Interface (AWS CLI), or the CreateUsageReportSubscription API operation. You must enable usage reports separately for each AWS Region for which you want to receive usage data.
**Note**
You can start or stop your subscription to usage reports at any time. There is no charge for subscribing to usage reports, but standard Amazon S3 charges may apply to reports that are stored in your S3 bucket. For more information, see Amazon S3 Pricing.

To subscribe to usage reports for AppStream 2.0 by using the AppStream 2.0 console, perform the following steps.

2. Choose the AWS Region for which you want to enable usage reports.
3. In the navigation pane, choose **Usage Reports**.
4. Choose **Enabled**, and then choose **Apply**.

If you enabled on-instance session scripts and Amazon S3 logging for your session script configuration, AppStream 2.0 created an S3 bucket to store the script output. The bucket is unique to your account and Region. When you enable usage reporting in this case, AppStream 2.0 uses the same bucket to store your usage reports. If you haven't already enabled on-instance session scripts, when you enable usage reports, AppStream 2.0 creates a new S3 bucket in the following location:

```
appstream-logs-[region-code]-[account-id-without-hyphens]-random-identifier
```

- **region-code**
  The AWS Region code for the Region in which usage reporting is enabled.

- **account-id-without-hyphens**
  Your AWS account identifier. The random ID ensures that there is no conflict with other buckets in the same Region. The first part of the bucket name, `appstream-logs`, does not change across accounts or Regions.

For example, if you enable usage reporting in the US West (Oregon) Region (us-west-2) on account number 123456789012, AppStream 2.0 creates an Amazon S3 bucket within your account in that Region similar to the name shown in the following example:

```
appstream-logs-us-west-2-1234567890123-abcdefghijkl
```

Only an administrator with sufficient permissions can delete this bucket.

**AppStream 2.0 Sessions Reports**

For each day that users launch at least one streaming session in your AWS account, AppStream 2.0 exports a sessions report to your Amazon S3 bucket. The report, named `daily-session-report-[YYYY]-[MM]-[DD].csv`, is stored in a nested folder structure in your Amazon S3 account, using the following folder path:

```
[bucket_name]/sessions/schedule=DAILY/year=[YYYY]/month=[MM]/day=[DD]/
```

This nesting structure facilitates partitioning if you choose to query your reports by using Amazon Athena. Athena is a serverless, interactive query service that you can use to analyze data stored in your S3 buckets using standard SQL. For more information, see Create Custom Reports and Analyze AppStream 2.0 Usage Data (p. 213).

Each user session is described in a single record in a sessions report. Sessions reports are generated daily according to UTC time within 24 hours of the close of the day that is the subject of the report. If a session spans more than one day, the session record appears in the sessions report corresponding to the
day in which the session ends. For information about the data included in sessions reports, see Sessions Report Fields (p. 209).

**AppStream 2.0 Applications Reports**

For each day that users launch at least one application during their streaming sessions, AppStream 2.0 exports an applications report to your Amazon S3 bucket. The report, named `daily-app-report-[YYYY]-[MM]-[DD].csv`, is stored in a nested folder structure in your Amazon S3 account, using the following folder path:

```
[bucket_name]/applications/schedule=DAILY/year=[YYYY]/month=[MM]/day=[DD]/
```

This nesting structure facilitates partitioning if you choose to query your reports by using Amazon Athena. Athena is a serverless, interactive query service that you can use to analyze data stored in your S3 buckets using standard SQL. For more information, see Create Custom Reports and Analyze AppStream 2.0 Usage Data (p. 213).

Each application launch is described in a single record in an applications report. For example, if a user launches five separate applications during a session, five separate records appear in the relevant applications report. An application is recorded as launched if any of the following events occurs:

- The application is launched directly when the session begins, because the application ID is embedded in either the streaming URL or the relay state.
- A user chooses the application from the application catalog when launching a new streaming session.
- A user chooses the application from the application catalog list during a streaming session.

The applications report doesn’t include applications that are launched in other ways. For example, if you provide users with access to Windows Explorer or PowerShell, and users use those tools to launch applications directly, or if another program or script launches an application, those application launches are not included in the applications report.

Applications reports are generated daily according to UTC time within 24 hours of the close of the day that is the subject of the report. If a session spans more than one day, applications launched during the session are reflected in the applications report corresponding to the day in which the session ends. For information about the data included in applications reports, see Applications Report Fields (p. 213).

**AppStream 2.0 Usage Reports Fields**

This topic provides information about the fields included in AppStream 2.0 usage reports.

**Contents**

- Sessions Report Fields (p. 209)
- Applications Report Fields (p. 213)

**Sessions Report Fields**

The following table describes the fields included in AppStream 2.0 sessions reports.

<table>
<thead>
<tr>
<th>Field name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user_session_id</td>
<td>The unique identifier (ID) for the session.</td>
</tr>
<tr>
<td>aws_account_id</td>
<td>The AWS account ID.</td>
</tr>
<tr>
<td>region</td>
<td>The AWS Region.</td>
</tr>
<tr>
<td>Field name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>session_start_time</td>
<td>The date and time that the session started.</td>
</tr>
<tr>
<td></td>
<td>Format: ISO 8601 (UTC)</td>
</tr>
<tr>
<td>session_end_time</td>
<td>The date and time that the session ended.</td>
</tr>
<tr>
<td></td>
<td>Format: ISO 8601 (UTC)</td>
</tr>
<tr>
<td>session_duration_in_seconds</td>
<td>The duration of the session in seconds.</td>
</tr>
<tr>
<td>user_id</td>
<td>The unique ID for the user within the authentication type.</td>
</tr>
<tr>
<td>user_arn</td>
<td>The Amazon Resource Name (ARN) for the user.</td>
</tr>
<tr>
<td>authentication_type</td>
<td>The method used to authenticate the user.</td>
</tr>
<tr>
<td></td>
<td>Possible values: CUSTOM</td>
</tr>
<tr>
<td>authentication_type_user_id</td>
<td>The concatenation of the user ID and authentication type, which uniquely identify the user for the purpose of assessing user fees. For more information, see AppStream 2.0 Pricing.</td>
</tr>
<tr>
<td>fleet_name</td>
<td>The name of the fleet associated with the session.</td>
</tr>
<tr>
<td>stack_name</td>
<td>The name of the stack associated with the session.</td>
</tr>
<tr>
<td>instance_type</td>
<td>The AppStream 2.0 instance type used for the session. For a list of instance types, see AppStream 2.0 Pricing.</td>
</tr>
<tr>
<td>eni_private_ip_address</td>
<td>The IP address of the elastic network interface used by the AppStream 2.0 instance for network communications.</td>
</tr>
<tr>
<td>connected_at_least_once</td>
<td>Indicates whether the user connected to the session at least once.</td>
</tr>
<tr>
<td></td>
<td>Possible values: true</td>
</tr>
<tr>
<td>Field name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>client_ip_addresses</td>
<td>The IP addresses associated with the user device or devices used to connect to the session. If the user connected and then disconnected from the session more than once, up to the last 10 distinct IP addresses are stored, separated by semicolons.</td>
</tr>
<tr>
<td>google_drive_enabled</td>
<td>Indicates whether Google Drive was enabled as a persistent storage option for the session. For more information, see Enable and Administer Google Drive for Your AppStream 2.0 Users (p. 140). Possible values: true</td>
</tr>
<tr>
<td>one_drive_enabled</td>
<td>Indicates whether OneDrive was enabled as a persistent storage option for the session. For more information, see Enable and Administer Google Drive for Your AppStream 2.0 Users (p. 140). Possible values: true</td>
</tr>
<tr>
<td>home_folders_storage_location</td>
<td>The Amazon S3 bucket used for files that are stored using home folders.</td>
</tr>
<tr>
<td>user_settings_clipboard_copy_from_local_device</td>
<td>Indicates whether the user was able to copy data from the local device to the streaming session using the clipboard during the session. Possible values: ENABLED</td>
</tr>
<tr>
<td>user_settings_clipboard_copy_to_local_device</td>
<td>Indicates whether the user was able to copy data from the streaming session to the local device using the clipboard during the session. Possible values: ENABLED</td>
</tr>
<tr>
<td>Field name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>user_settings_file_upload</td>
<td>Indicates whether the user was able to upload files from the local device to the streaming session during the session.</td>
</tr>
<tr>
<td></td>
<td>Possible values: ENABLED</td>
</tr>
<tr>
<td>user_settings_file_download</td>
<td>Indicates whether the user was able to download files from the streaming session to the local device during the session.</td>
</tr>
<tr>
<td></td>
<td>Possible values: ENABLED</td>
</tr>
<tr>
<td>user_settings_printing_to_local_device</td>
<td>Indicates whether the user was able to print files from the streaming session to the local device during the session.</td>
</tr>
<tr>
<td></td>
<td>Possible values: ENABLED</td>
</tr>
<tr>
<td>application_settings_enabled</td>
<td>Indicates whether application settings persistence was enabled for the session.</td>
</tr>
<tr>
<td></td>
<td>Possible values: true</td>
</tr>
<tr>
<td>domain Joined</td>
<td>Indicates whether the AppStream 2.0 streaming instance was joined to an Active Directory domain at session launch. For more information, see Using Active Directory with AppStream 2.0 (p. 106).</td>
</tr>
<tr>
<td></td>
<td>Possible values: Y</td>
</tr>
<tr>
<td>max_session_duration</td>
<td>The maximum allowed duration of the session in seconds.</td>
</tr>
<tr>
<td>session_type</td>
<td>The session type.</td>
</tr>
<tr>
<td></td>
<td>Possible values: ALWAYS_ON</td>
</tr>
<tr>
<td>schedule</td>
<td>The frequency that reports are generated.</td>
</tr>
<tr>
<td></td>
<td>Possible value: DAILY</td>
</tr>
<tr>
<td>year</td>
<td>The year of the report.</td>
</tr>
<tr>
<td>month</td>
<td>The month of the report.</td>
</tr>
<tr>
<td>day</td>
<td>The day of the report.</td>
</tr>
</tbody>
</table>
Applications Report Fields

The following table describes the fields included in AppStream 2.0 applications reports.

<table>
<thead>
<tr>
<th>Field name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user_session_id</td>
<td>The unique identifier (ID) for the session.</td>
</tr>
<tr>
<td>application_name</td>
<td>The name of the application, as specified in Image Assistant. This value is provided when a user launches an application.</td>
</tr>
<tr>
<td>schedule</td>
<td>The frequency with which reports are generated. Possible value: DAILY</td>
</tr>
<tr>
<td>year</td>
<td>The year of the report.</td>
</tr>
<tr>
<td>month</td>
<td>The month of the report.</td>
</tr>
<tr>
<td>day</td>
<td>The day of the report.</td>
</tr>
</tbody>
</table>

Create Custom Reports and Analyze AppStream 2.0 Usage Data

Amazon Athena is a serverless, interactive query service that you can use to analyze data stored in your S3 buckets using standard SQL queries. You can use Athena to aggregate your usage reports or generate other types of custom reports.

Contents
- Create an AWS Glue Crawler (p. 213)
- Create a Data Catalog by Using the AWS Glue Crawler (p. 216)
- Create and Run Athena Queries (p. 216)
- Working with Athena Queries (p. 217)

Create an AWS Glue Crawler

AWS Glue is a fully managed extract, transform, and load (ETL) service that lets you create a database from your Amazon S3 data and query that database by using Athena. This database is also referred to as an AWS Glue Data Catalog. An AWS Glue crawler can automatically detect the schema of your Amazon S3 data and create the corresponding database and tables. AppStream 2.0 provides an AWS CloudFormation template that you can use to create the necessary AWS Glue resources.

**Important**
Completing the steps in the following procedure creates an AWS Glue crawler. However, these steps don’t start the crawler. To start the crawler, you must perform the steps in the next procedure. For more information about AWS Glue crawlers, see Defining Crawlers.

To create an AWS Glue crawler
2. Choose the AWS Region for which you have subscribed to usage reports.
3. In the navigation pane, choose **Usage Reports**, and verify that usage reports logging is enabled.
4. On the **Report Details** tab, in the paragraph next to **Analytics**, choose the **CloudFormation template** link.

Choosing the link opens the AWS CloudFormation console, where you can review the parameters of the AWS CloudFormation stack specified by the template before you run it. The template, when run, creates an AWS Glue crawler and several sample Athena queries.

5. On the **Specify Details** page, next to **ScheduleExpression**, either keep the default value or specify a different cron expression value for the frequency that you want to run the crawler. Do not change any other default value. When you're done, choose **Next**.

By default, the crawler is scheduled to run on a daily basis, but you can configure the crawler to run weekly, monthly, or on another frequency. For information about cron syntax, see **Cron Expressions**.

6. On the **Options** page, keep the default values, and choose **Next**.
7. On the **Review** page, select the check box next to "I acknowledge that AWS CloudFormation might create IAM resources with custom names," and then choose **Create**.

You must have sufficient AWS Glue and AWS Identity and Access Management (IAM) permissions to create and run the AWS CloudFormation stack. If you don't have the required permissions, ask your AWS account administrator either to perform these steps in your account or to grant you the following permissions.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "athena:CreateNamedQuery",
        "athena:BatchGetNamedQuery",
        "athena:GetNamedQuery",
        "athena:StartQueryExecution",
        "athena:GetQueryResults",
        "athena:GetQueryExecution",
        "athena:GetNamedQuery",
        "cloudformation:DescribeStacks",
        "cloudformation:GetStackPolicy",
        "cloudformation:DescribeStackEvents",
        "cloudformation:CreateStack",
        "cloudformation:GetTemplate",
        "cloudformation:DescribeChangeSets",
        "cloudformation:ListStackResources",
        "iam:GetRole",
        "iam:CreateRole",
        "iam:GetRolePolicy",
        "s3:GetBucketLocation",
        "s3:ListBucketMultipartUploads",
        "s3:GetBucket",
        "s3:ListMultipartUploadParts",
        "s3:GetObject",
        "s3:AbortMultipartUpload"
      ],
      "Resource": [
        "arn:aws:cloudformation::*:stack/AppStreamUsageReports/**",
        "arn:aws:athena::*:workgroup/primary",
        "arn:aws:s3:::aws-athena-query-results-*"
      ]
    }
  ]
}
```


},

{ "Effect": "Allow",
 "Action": [
   "iam:AttachRolePolicy",
   "iam:PutRolePolicy",
   "s3:GetObject",
   "s3:ListBucket"
],
 "Resource": [
   "arn:aws:s3:::appstream-logs-*",
   "arn:aws:iam::*:role/AppStreamUsageReports-AppStreamUsageReportGlueRole*"
]
},

{ "Effect": "Allow",
 "Action": [
   "iam:PassRole"
],
 "Resource": [
   "arn:aws:iam::*:role/AppStreamUsageReports-AppStreamUsageReportGlueRole*"
],
 "Condition": {
   "StringEquals": {
     "iam:PassedToService": "glue.amazonaws.com"
   }
 }
},

{ "Effect": "Allow",
 "Action": [
   "cloudformation:GetTemplateSummary",
   "glue:GetResourcePolicy",
   "glue:GetCrawlers",
   "glue:BatchGetCrawlers",
   "glue:GetClassifiers",
   "glue:CreateClassifier",
   "glue:ListCrawlers",
   "glue:GetTags",
   "glue:GetCrawlerMetrics",
   "glue:GetClassifier",
   "tag:GetResources"
],
 "Resource": "*"
},

{ "Effect": "Allow",
 "Action": "athena:RunQuery",
 "Resource": "arn:aws:athena:*:*:workgroup/primary"
},

{ "Effect": "Allow",
 "Action": [
   "glue:GetTables",
   "glue:GetPartitions",
   "glue:GetTable"
],
 "Resource": [
   "arn:aws:glue::*:table/appstream-usage/*",
   "arn:aws:glue::*:database/appstream-usage",
   "arn:aws:glue::*:catalog"
]
},
Create a Data Catalog by Using the AWS Glue Crawler

The AWS Glue crawler, when run, creates a Data Catalog and schema that are mapped to the structure of your sessions and applications reports. Each time a new report is stored in your Amazon S3 bucket, you must run the crawler to update your AWS Glue Data Catalog with the data from the new report.

**Note**
Charges may apply to the running of your AWS Glue crawler. For more information, see AWS Glue Pricing.

2. Choose the AWS Region for which you have subscribed to usage reports.
3. Select the check box next to the crawler named `appstream-usage-sessions-crawler`, and then choose **Run crawler**. Repeat this step for the crawler named `appstream-usage-apps-crawler`.

Performing these steps runs the crawlers and schedules them to run automatically according to the schedule specified in the AWS CloudFormation stack.

4. After both crawlers finish running, in the navigation pane, choose **Databases**. A database named `appstream-usage`, which represents your usage reports, displays. This database is an AWS Glue Data Catalog that was created when `appstream-usage-sessions-crawler` and `appstream-usage-apps-crawler` were run.
5. To view the tables in the database, choose `appstream-usage`, **Tables**. Two tables, `applications` and `sessions`, which represent your applications and sessions usage reports respectively, display. Choose either table to view its schema.

You can now query these tables in Athena by using SQL.

**Create and Run Athena Queries**

To query your usage reports by using Athena, perform the following steps.
Create Custom Reports

Note
Charges may apply to Athena queries that you run. For more information, see Amazon Athena Pricing.

2. Under Database, choose appstream-usage.
3. In the query pane, enter a SQL query, and choose Run query.

Working with Athena Queries

This section provides SQL queries that you can run in Athena to analyze the usage reports data in your Amazon S3 bucket.

To create a consolidated report of all sessions in a given month, run the following query:

```
SELECT *
FROM "appstream-usage"."sessions"
WHERE year='four-digit-year'
AND month='two-digit-month'
```

You can also perform join operations between the applications and sessions tables in your query. For example, to view the distinct users who launched each application in a given month, run the following query:

```
SELECT DISTINCT apps.application_name, sessions.user_id
FROM "appstream-usage"."applications" apps
INNER JOIN "appstream-usage"."sessions" sessions ON (apps.user_session_id = sessions.user_session_id AND sessions.year='four-digit-year' AND sessions.month='two-digit-month')
WHERE apps.year='four-digit-year'
AND apps.month='two-digit-month'
ORDER BY 1, 2
```

Athena query results are stored as .csv files in an Amazon S3 bucket in your account that is named aws-athena-query-results-account-id-without-hyphens-region-code. For ease in locating query results, choose Save as and provide a name for your query before you run it. You can also choose the download icon in the Athena Results pane to download the results of the query as a .csv file.

To enhance performance and reduce costs, Athena uses partitioning to reduce the amount of data scanned in queries. For more information, see Partitioning Data. Usage reports are partitioned in your Amazon S3 buckets by year, month, and day. You can restrict your queries to certain date range partitions using the year, month, and day fields as conditions in your queries. For example, the following query ingests only the sessions reports for the week of May 19, 2019.

```
SELECT SUBSTRING(session_start_time, 1, 10) AS report_date,
       COUNT(DISTINCT user_session_id) AS num_sessions
FROM "appstream-usage"."sessions"
WHERE year='2019'
   AND month='05'
   AND day BETWEEN '19' and '25'
GROUP BY 1
ORDER BY 1
```

In contrast, the following query produces identical results, but because it isn't restricted to any partitions, it ingests all sessions reports stored in your Amazon S3 bucket.

```
SELECT SUBSTRING(session_start_time, 1, 10) AS report_date,
       COUNT(DISTINCT user_session_id) AS num_sessions
FROM "appstream-usage"."sessions"
WHERE year='2019'
   AND month='05'
GROUP BY 1
ORDER BY 1
```
If a session spans more than one day, the session and application records appear in the sessions and applications reports, respectively, corresponding to the day in which the session ended. For this reason, if you need to find records that relate to all sessions that were active during a given date range, consider expanding the partition set of your query by the maximum session length you have configured for your fleets.

For example, to view all sessions that were active for a given fleet during a calendar month, where the fleet had a maximum session duration of 100 hours, run the following query to expand your partition set by five days.

```sql
SELECT *
FROM "appstream-usage"."sessions"
WHERE fleet_name = 'fleet_name'
AND session_start_time BETWEEN '2019-05-01' AND '2019-06-01'
AND year='2019'
AND (month='05' OR (month='06' AND day<='05'))
ORDER BY session_start_time
```

The AWS CloudFormation template that created the AWS Glue crawlers also created and saved several sample queries in your Athena account that you can use to analyze your usage data. These sample queries include the following:

- Aggregated monthly session report
- Average session length per stack
- Number of sessions per day
- Total streaming hours per user
- Distinct users per app

To use any of these queries, perform the following steps.

2. Choose Saved Queries. The five queries noted before this procedure should display. The name of each query begins with "AS2." For example, "AS2_users_per_app_curr_mo."
3. To run a query, choose the query name rather than the option next to the name.
4. The text of the query appears in the query pane. Choose Run query.

To view these queries in a separate AWS CloudFormation template, see [athena-sample-queries-appstream-usage-data_template.yml](https://aws-code-samples.s3.amazonaws.com/athena-sample-queries-appstream-usage-data_template.yml) in the AWS Code Sample Catalog.

### Logging AppStream 2.0 API Calls with AWS CloudTrail

Amazon AppStream 2.0 is integrated with AWS CloudTrail. CloudTrail is a service that provides a record of actions taken by a user, role, or an AWS service in AppStream 2.0. CloudTrail captures API calls for AppStream 2.0 as events. The calls captured include calls from the AppStream 2.0 console and code calls to the AppStream 2.0 API operations. If you create a trail, you can enable continuous delivery of
CloudTrail events to an Amazon S3 bucket, including events for AppStream 2.0. If you don’t configure a trail, you can still view the most recent events in the CloudTrail console in Event history. You can use the information collected by CloudTrail to determine details such as request information. For example, CloudTrail collects the following information: What request was made to AppStream 2.0, the IP address from which the request was made, who made the request, and when it was made.

To learn more about CloudTrail, including how to configure and enable it, see the AWS CloudTrail User Guide.

AppStream 2.0 Information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When supported event activity occurs in AppStream 2.0, 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 AppStream 2.0, 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

AppStream 2.0 supports logging the following actions as events in CloudTrail log files:

- AssociateFleet
- BatchAssociateUserStack
- BatchDisassociateUserStack
- CopyImage
- CreateDirectoryConfig
- CreateFleet
- CreateImageBuilder
- CreateImageBuilderStreamingURL
- CreateStack
- CreateStreamingURL
- DeleteDirectoryConfig
- DeleteFleet
- DeleteImage
- DeleteImageBuilder
- DeleteImagePermissions
- DeleteStack
- DescribeDirectoryConfigs
- DescribeFleets
- DescribeImageBuilders
- DescribeImagePermissions
Example: AppStream 2.0 Log File Entries

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.

**Example: AppStream 2.0 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.

The following example shows a CloudTrail log entry that demonstrates the `AssociateFleet` event.

```json
{
   "eventVersion": "1.05",
   "userIdentity": {
      "type": "AssumedRole",
      "principalId": "AIDACKCEVSQ6C2EXAMPLE:janeroe",
      "arn": "arn:aws:sts::123456789012:assumed-role/Admin/janeroe",
      "accountId": "123456789012",
      "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
      "sessionContext": {
         "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2019-03-12T06:41:50Z"
         }
      }
   }
}
```
The following example shows a CloudTrail log entry that demonstrates the CreateImage event when an image is created using the AppStream 2.0 image builder.
Security in Amazon AppStream 2.0

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that is 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. For information about the compliance programs that apply to AppStream 2.0, 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 AppStream 2.0. It shows you how to configure AppStream 2.0 to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your AppStream 2.0 resources.

**Contents**

- Data Protection in Amazon AppStream 2.0 (p. 222)
- Identity and Access Management for Amazon AppStream 2.0 (p. 224)
- Logging and Monitoring in Amazon AppStream 2.0 (p. 235)
- Compliance Validation for Amazon AppStream 2.0 (p. 236)
- Resilience in Amazon AppStream 2.0 (p. 237)
- Infrastructure Security in Amazon AppStream 2.0 (p. 237)
- Security Groups in Amazon AppStream 2.0 (p. 253)
- Update Management in Amazon AppStream 2.0 (p. 254)

**Data Protection in Amazon AppStream 2.0**

Amazon AppStream 2.0 conforms to the AWS shared responsibility model, which includes regulations and guidelines for data protection. AWS is responsible for protecting the global infrastructure that runs all the AWS services. AWS maintains control over data hosted on this infrastructure, including the security configuration controls for handling customer content and personal data. AWS customers and APN partners, acting either as data controllers or data processors, are responsible for any personal data that they put in the AWS Cloud.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual user accounts with AWS Identity and Access Management (IAM), so that 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 TLS to communicate with AWS resources.
- 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 personal data that is stored in Amazon S3.

We strongly recommend that you never put sensitive identifying information, such as your customers' account numbers, into free-form fields or metadata, such as function names and tags. Any data that you enter into metadata might get picked up for inclusion in diagnostic logs. When you provide a URL to an external server, don't include credentials information in the URL to validate your request to that server.

For more information about data protection, see the AWS Shared Responsibility Model and GDPR blog post on the AWS Security Blog.

**Encryption at Rest**

AppStream 2.0 fleet instances are ephemeral in nature. After a user's streaming session is finished, the underlying instance and its associated Amazon Elastic Block Store (Amazon EBS) volume are terminated. In addition, AppStream 2.0 periodically recycles unused instances for freshness.

When you enable application settings persistence (p. 146) or home folders (p. 137) for your users, the data that is generated by your users and stored in Amazon Simple Storage Service buckets is encrypted at rest. AWS Key Management Service is a service that combines secure, highly available hardware and software to provide a key management system scaled for the cloud. Amazon S3 uses AWS Managed CMKs to encrypt your Amazon S3 object data.

**Encryption in Transit**

The following table provides information about how data is encrypted in transit. Where applicable, other data protection methods for AppStream 2.0 are also listed.

<table>
<thead>
<tr>
<th>Data</th>
<th>Network path</th>
<th>How protected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web assets</td>
<td>Between AppStream 2.0 users and AppStream 2.0</td>
<td>Encrypted using TLS 1.2</td>
</tr>
<tr>
<td>This traffic includes assets such as images and JavaScript files.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pixel and related streaming traffic</td>
<td>Between AppStream 2.0 users and AppStream 2.0</td>
<td>Encrypted using 256-bit Advanced Encryption Standard (AES-256)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transported using TLS 1.2</td>
</tr>
<tr>
<td>API traffic</td>
<td>Between AppStream 2.0 users and AppStream 2.0</td>
<td>Encrypted using TLS 1.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Requests to create a connection are signed using SigV4</td>
</tr>
<tr>
<td>Application settings and home folder data generated by users</td>
<td>Between AppStream 2.0 users and Amazon S3</td>
<td>Encrypted using Amazon S3 SSL endpoints</td>
</tr>
<tr>
<td>Applicable when application settings persistence and home folders are enabled.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Identity and Access Management for Amazon AppStream 2.0

Identity and Access Management for Amazon AppStream 2.0

Your security credentials identify you to services in AWS and grant you unlimited use of your AWS resources, such as your AppStream 2.0 resources. You can use features of AppStream 2.0 and AWS Identity and Access Management (IAM) to allow other users, services, and applications to use your AppStream 2.0 resources without sharing your security credentials.

You can use IAM to control how other users use resources in your AWS account, and you can use security groups to control access to your AppStream 2.0 streaming instances. You can allow full use or limited use of your AppStream 2.0 resources.

Contents

• Network Access to Your Streaming Instance (p. 224)
• Using AWS Managed Policies and Linked Roles to Manage Administrator Access to AppStream 2.0 Resources (p. 224)
• Using IAM Policies to Manage Administrator Access to Application Auto Scaling (p. 229)
• Using IAM Policies to Manage Administrator Access to the Amazon S3 Bucket for Home Folders and Application Settings Persistence (p. 230)
• Using an IAM Role to Grant Permissions to Applications and Scripts Running on AppStream 2.0 Streaming Instances (p. 232)

Network Access to Your Streaming Instance

A security group acts as a stateful firewall that controls what traffic is allowed to reach your streaming instances. When you launch an AppStream 2.0 streaming instance, assign it to one or more security groups. Then, add rules to each security group that control traffic for the instance. You can modify the rules for a security group at any time. The new rules are automatically applied to all instances to which the security group is assigned.

For more information, see Security Groups in Amazon AppStream 2.0 (p. 253).

Using AWS Managed Policies and Linked Roles to Manage Administrator Access to AppStream 2.0 Resources

By default, IAM users don't have the permissions required to create or modify AppStream 2.0 resources, or perform tasks by using the AppStream 2.0 API. This means that these users can't perform these
actions in the AppStream 2.0 console or by using AppStream 2.0 AWS CLI commands. To allow IAM users to create or modify resources and perform tasks, attach an IAM policy to the IAM users or groups that require those permissions.

When you attach a policy to a user, group of users, or IAM role, it allows or denies the users permission to perform the specified tasks on the specified resources.

Contents

- AWS Managed Policies Required to Access AppStream 2.0 Resources (p. 225)
- Roles Required for AppStream 2.0 and Application Auto Scaling (p. 226)
- Checking for the AmazonAppStreamServiceAccess Service Role and Policies (p. 227)
- Checking for the ApplicationAutoScalingForAmazonAppStreamAccess Service Role and Policies (p. 227)
- Checking for the AWSServiceRoleForApplicationAutoScaling_AppStreamFleet Service-Linked Role and Policies (p. 228)

AWS Managed Policies Required to Access AppStream 2.0 Resources

To provide full administrative or read-only access to AppStream 2.0, you must attach one of the following AWS managed policies to the IAM users or groups that require those permissions. An AWS managed policy is a standalone policy that is created and administered by AWS. For more information, see AWS Managed Policies in the IAM User Guide.

AmazonAppStreamFullAccess

This managed policy provides full administrative access to AppStream 2.0 resources. To manage AppStream 2.0 resources and perform API actions through the AWS Command Line Interface (AWS CLI), AWS SDK, or AWS Management Console, you must have the permissions defined in this policy.

If you sign into the AppStream 2.0 console as an IAM user, you must attach this policy to your IAM user account. If you sign in through console federation, you must attach this policy to the IAM role that was used for federation.

AmazonAppStreamReadOnlyAccess

This managed policy provides read-only access to AppStream 2.0 resources.

The AppStream 2.0 console uses two additional actions that provide functionality that is not available through the AWS CLI or AWS SDK. The AmazonAppStreamFullAccess and AmazonAppStreamReadOnlyAccess policies both provide permissions for these actions.

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
<th>Access Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetImageBuilders</td>
<td>Grants permission to retrieve a list that describes one or more specified image builders, if the image builder names are provided. Otherwise, all image builders in the account are described.</td>
<td>Read</td>
</tr>
<tr>
<td>GetParametersForThemeAssetUpload</td>
<td>Grants permission to upload theme assets for</td>
<td>Write</td>
</tr>
</tbody>
</table>
Access to AppStream 2.0 Resources

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
<th>Access Level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>custom branding. For more information, see Add Your Custom Branding to Amazon AppStream 2.0 (p. 119).</td>
<td></td>
</tr>
</tbody>
</table>

## Roles Required for AppStream 2.0 and Application Auto Scaling

In AWS, IAM roles are used to give permissions to an AWS service so it can access AWS resources. The policies that are attached to the role determine which AWS resources the service can access and what it can do with those resources. For AppStream 2.0, in addition to having the permissions defined in the AmazonAppStreamFullAccess policy, you must also have the following roles in your AWS account, with the required policies attached.

### Roles

- AmazonAppStreamServiceAccess (p. 226)
- ApplicationAutoScalingForAmazonAppStreamAccess (p. 226)
- AWSServiceRoleForApplicationAutoScaling_AppStreamFleet (p. 226)

These roles are automatically created for you, with the required IAM policies attached, when you get started with the AppStream 2.0 service in an AWS Region. To get started with AppStream 2.0, you must have either of the following permissions:

- **AdministratorAccess** permissions
- Permissions to create an IAM role and attach IAM policies to a role

**Note**

To create the AWSServiceRoleForApplicationAutoScaling_AppStreamFleet role, you must have also have the iam:CreateServiceLinkedRole permission.

### AmazonAppStreamServiceAccess

While AppStream 2.0 resources are being created, the AppStream 2.0 service makes API calls to other AWS services on your behalf by assuming this role. To create fleets, you must have this service role in your account. If this service role is not in your AWS account and the required IAM permissions and trust relationship policies are not attached, you cannot create AppStream 2.0 fleets.

### ApplicationAutoScalingForAmazonAppStreamAccess

Automatic scaling is a feature of AppStream 2.0 fleets. To configure scaling policies, you must have this service role in your AWS account. If this service role is not in your AWS account and the required IAM permissions and trust relationship policies are not attached, you cannot scale AppStream 2.0 fleets.

### AWSServiceRoleForApplicationAutoScaling_AppStreamFleet

Application Auto Scaling uses a service-linked role to perform automatic scaling on your behalf. A service-linked role is an IAM role that is linked directly to an AWS service. This role includes all the permissions that the service requires to call other AWS services on your behalf.

For more information, see Service-Linked Roles in the Application Auto Scaling User Guide.
Checking for the AmazonAppStreamServiceAccess Service Role and Policies

Complete the steps in this section to check whether the AmazonAppStreamServiceAccess service role is present and has the correct policies attached. If this role is not in your account and must be created, you or an administrator with the required permissions must perform the steps to get started with AppStream 2.0 in your AWS account.

To check whether the AmazonAppStreamServiceAccess IAM service role is present

1. Open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane, choose Roles.
3. In the search box, type amazonappstreamservice to narrow the list of roles to select, and then choose AmazonAppStreamServiceAccess. If this role is listed, select it to view the role Summary page.
4. On the Permissions tab, confirm whether the AmazonAppStreamServiceAccess permissions policy is attached.
5. Return to the role Summary page.
6. On the Trust relationships tab, choose Show policy document, and then confirm whether the AmazonAppStreamServiceAccess trust relationship policy is attached and follows the correct format. If so, the trust relationship is correctly configured. Choose Cancel and close the IAM console.

AmazonAppStreamServiceAccess trust relationship policy

The AmazonAppStreamServiceAccess trust relationship policy must include the AppStream 2.0 service as the principal. A principal is an entity in AWS that can perform actions and access resources. This policy must also include the sts:AssumeRole action. The following policy configuration defines AppStream 2.0 as a trusted entity.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "appstream.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Checking for the ApplicationAutoScalingForAmazonAppStreamAccess Service Role and Policies

Complete the steps in this section to check whether the ApplicationAutoScalingForAmazonAppStreamAccess service role is present and has the correct policies attached. If this role is not in your account and must be created, you or an administrator with the required permissions must perform the steps to get started with AppStream 2.0 in your AWS account.
To check whether the ApplicationAutoScalingForAmazonAppStreamAccess IAM service role is present

1. Open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane, choose Roles.
3. In the search box, type applicationautoscaling to narrow the list of roles to select, and then choose ApplicationAutoScalingForAmazonAppStreamAccess. If this role is listed, select it to view the role Summary page.
4. On the Permissions tab, confirm whether the ApplicationAutoScalingForAmazonAppStreamAccess permissions policy is attached.
5. Return to the role Summary page.
6. On the Trust relationships tab, choose Show policy document, and then confirm whether the ApplicationAutoScalingForAmazonAppStreamAccess trust relationship policy is attached and follows the correct format. If so, the trust relationship is correctly configured. Choose Cancel and close the IAM console.

**ApplicationAutoScalingForAmazonAppStreamAccess trust relationship policy**

The ApplicationAutoScalingForAmazonAppStreamAccess trust relationship policy must include the Application Auto Scaling service as the principal. This policy must also include the sts:AssumeRole action. The following policy configuration defines Application Auto Scaling as a trusted entity.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "application-autoscaling.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Checking for the **AWSServiceRoleForApplicationAutoScaling_AppStreamFleet Service-Linked Role and Policies**

Complete the steps in this section to check whether the AWSServiceRoleForApplicationAutoScaling_AppStreamFleet service-linked role is present and has the correct policies attached. If this role is not in your account and must be created, you or an administrator with the required permissions must perform the steps to get started with AppStream 2.0 in your AWS account.

To check whether the AWSServiceRoleForApplicationAutoScaling_AppStreamFleet IAM service-linked role is present

1. Open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane, choose Roles.
3. In the search box, type applicationautoscaling to narrow the list of roles to select, and then choose AWSServiceRoleForApplicationAutoScaling_AppStreamFleet. If this role is listed, select it to view the role Summary page.
4. On the Permissions tab, confirm whether the `AWSApplicationAutoscalingAppStreamFleetPolicy` permissions policy is attached.

5. Return to the Role summary page.

6. On the Trust relationships tab, choose Show policy document, and then confirm whether the `AWSServiceRoleForApplicationAutoScaling_AppStreamFleet` trust relationship policy is attached and follows the correct format. If so, the trust relationship is correctly configured. Choose Cancel and close the IAM console.

**AWSServiceRoleForApplicationAutoScaling_AppStreamFleet trust relationship policy**

The `AWSServiceRoleForApplicationAutoScaling_AppStreamFleet` trust relationship policy must include `appstream.application-autoscaling.amazonaws.com` as the principal. This policy must also include the `sts:AssumeRole` action. The following policy configuration defines `appstream.application-autoscaling.amazonaws.com` as a trusted entity.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "appstream.application-autoscaling.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

**Using IAM Policies to Manage Administrator Access to Application Auto Scaling**

Automatic scaling for fleets is made possible by a combination of the AppStream 2.0, Amazon CloudWatch, and Application Auto Scaling APIs. AppStream 2.0 fleets are created with AppStream 2.0, alarms are created with CloudWatch, and scaling policies are created with Application Auto Scaling.

In addition to having the permissions defined in the `AmazonAppStreamFullAccess` policy, the IAM user that accesses fleet scaling settings must have the required permissions for the services that support dynamic scaling. IAM users must have permissions to use the actions shown in the following example policy.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "appstream:*",
        "application-autoscaling:*",
        "cloudwatch:DeleteAlarms",
        "cloudwatch:DescribeAlarmsForMetric",
        "cloudwatch:DisableAlarmActions",
        "cloudwatch:DescribeAlarms",
        "cloudwatch:EnableAlarmActions",
        "cloudwatch:ListMetrics",
        "cloudwatch:PutMetricAlarm",
        "iam:PassRole",
```
You can also create your own IAM policies to set more specific permissions for calls to the Application Auto Scaling API. For more information, see Authentication and Access Control in the Application Auto Scaling User Guide.

Using IAM Policies to Manage Administrator Access to the Amazon S3 Bucket for Home Folders and Application Settings Persistence

The following examples show how you can use IAM policies to manage access to the Amazon S3 bucket for home folders and application settings persistence.

Examples
- Deleting the Amazon S3 Bucket for Home Folders and Application Settings Persistence (p. 230)
- Restricting Administrator Access to the Amazon S3 Bucket for Home Folders and Application Settings Persistence (p. 231)

Deleting the Amazon S3 Bucket for Home Folders and Application Settings Persistence

AppStream 2.0 adds an Amazon S3 bucket policy to the buckets that it creates to prevent them from being accidentally deleted. To delete an S3 bucket, you must first delete the S3 bucket policy. Following are the bucket policies that you must delete for home folders and application settings persistence.

Home folders policy

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PreventAccidentalDeletionOfBucket",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:DeleteBucket",
      "Resource": "arn:aws:s3:::appstream2-36fb080bb8-region-code-account-id-without-hyphens"
    }
  ]
}
```

Application settings persistence policy

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PreventAccidentalDeletionOfBucket",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:DeleteBucket",
      "Resource": "arn:aws:s3:::appstream2-36fb080bb8-region-code-account-id-without-hyphens"
    }
  ]
}
```
Restricting Administrator Access to the Amazon S3 Bucket for Home Folders and Application Settings Persistence

By default, administrators who can access the Amazon S3 buckets created by AppStream 2.0 can view and modify content that is part of users’ home folders and persistent application settings. To restrict administrator access to the S3 buckets that contain user files, we recommend applying the S3 bucket access policy based on the following template:

```json
{
    "Sid": "RestrictedAccess",
    "Effect": "Deny",
    "NotPrincipal": {
        "AWS": [
            "arn:aws:iam::account:role/service-role/AmazonAppStreamServiceAccess",
            "arn:aws:sts::account:assumed-role/AmazonAppStreamServiceAccess/PhotonSession",
            "arn:aws:iam::account:user/IAM-user-name"
        ],
    },
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::home-folder-or-application-settings-persistence-s3-bucket-region-account"
}
```

This policy allows S3 bucket access only to the users specified and to the AppStream 2.0 service. For every IAM user who should have access, replicate the following line:

```
"arn:aws:iam::account:user/IAM-user-name"
```

In the following example, the policy restricts access to the home folder S3 bucket for anyone other than IAM users marymajor and johnstiles. It also allows access to the AppStream 2.0 service, in AWS Region US West (Oregon) for account ID 123456789012.

```json
{
    "Sid": "RestrictedAccess",
    "Effect": "Deny",
    "NotPrincipal": {
        "AWS": [
            "arn:aws:iam::123456789012:role/service-role/AmazonAppStreamServiceAccess",
            "arn:aws:sts::123456789012:assumed-role/AmazonAppStreamServiceAccess/PhotonSession",
            "arn:aws:iam::123456789012:user/maymarymajor",
            "arn:aws:iam::123456789012:user/johnstiles"
        ],
    },
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::appstream2-36fb080bb8-us-west-2-123456789012"
}
```
Using an IAM Role to Grant Permissions to Applications and Scripts Running on AppStream 2.0 Streaming Instances

Applications and scripts that run on AppStream 2.0 streaming instances must include AWS credentials in their AWS API requests. You can create an IAM role to manage these credentials. An IAM role specifies a set of permissions that you can use to access AWS resources. This role is not uniquely associated with one person, however. Instead, it can be assumed by anyone that needs it.

You can apply an IAM role to an AppStream 2.0 streaming instance. When the streaming instance switches to (assumes) the role, the role provides temporary security credentials. Your application or scripts use these credentials to perform API actions and management tasks on the streaming instance. AppStream 2.0 manages the temporary credential switch for you.

Best Practices for Using IAM Roles With AppStream 2.0 Streaming Instances

When you use IAM roles with AppStream 2.0 streaming instances, we recommend that you follow these practices:

- Limit the permissions that you grant to AWS API actions and resources.
  
  Follow least privilege principles when you create and attach IAM policies to the IAM roles associated with AppStream 2.0 streaming instances. When you use an application or script that requires access to AWS API actions or resources, determine the specific actions and resources that are required. Then, create policies that allow the application or script to perform only those actions. For more information, see Grant Least Privilege in the IAM User Guide.

- Create an IAM role for each AppStream 2.0 resource.
  
  Creating a unique IAM role for each AppStream 2.0 resource is a practice that follows least privilege principles. Doing so also lets you modify permissions for a resource without affecting other resources.

- Limit where the credentials can be used.
  
  IAM policies let you define the conditions under which your IAM role can be used to access a resource. For example, you can include conditions to specify a range of IP addresses that requests can come from. Doing so prevents the credentials from being used outside of your environment. For more information, see Use Policy Conditions for Extra Security in the IAM User Guide.

Configuring an Existing IAM Role to Use With AppStream 2.0 Streaming Instances

This topic describes how to configure an existing IAM role so that you can use it with image builders and fleet streaming instances.
Prerequisites

The IAM role that you want to use with an AppStream 2.0 image builder or fleet streaming instance must meet the following prerequisites:

- The IAM role must be in the same AWS account as the AppStream 2.0 streaming instance.
- The IAM role cannot be a service role.
- The trust relationship policy that is attached to the IAM role must include the AppStream 2.0 service as the principal. A principal is an entity in AWS that can perform actions and access resources. The policy must also include the sts:AssumeRole action. This policy configuration defines AppStream 2.0 as a trusted entity.
- If you are applying the IAM role to an image builder, the image builder must run a version of the AppStream 2.0 agent released on or after September 3, 2019. If you are applying the IAM role to a fleet, the fleet must use an image that uses a version of the agent released on or after the same date. For more information, see AppStream 2.0 Agent Version History (p. 39).

To enable the AppStream 2.0 service principal to assume an existing IAM role

To perform the following steps, you must sign into the account as an IAM user who has the permissions required to list and update IAM roles. If you don't have the required permissions, ask your AWS account administrator either to perform these steps in your account or to grant you the required permissions.

1. Open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane, choose Roles.
3. In the list of roles in your account, choose the name of the role that you want to modify.
4. Choose the Trust relationships tab, and then choose Edit trust relationship.
5. Under Policy Document, verify that the trust relationship policy includes the sts:AssumeRole action for the appstream.amazonaws.com service principal:

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "appstream.amazonaws.com"
        ],
      "Action": "sts:AssumeRole"
      }
    }
  ]
}
```

6. When you are finished editing your trust policy, choose Update Trust Policy to save your changes.
7. The IAM role that you selected will display in the AppStream 2.0 console. This role grants permissions to applications and scripts to perform API actions and management tasks on streaming instances.

How to Create an IAM Role to Use With AppStream 2.0 Streaming Instances

This topic describes how to create a new IAM role so that you can use it with image builders and fleet streaming instances.
1. Open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane, choose **Roles**, and then choose **Create role**.
3. For **Select type of trusted entity**, choose **AWS service**.
4. From the list of AWS services, choose **AppStream 2.0**.
5. Under **Select your use case**, **AppStream 2.0 — Allows AppStream 2.0 instances to call AWS services on your behalf** is already selected. Choose **Next: Permissions**.
6. If possible, select the policy to use for the permissions policy or choose **Create policy** to open a new browser tab and create a new policy from scratch. For more information, see step 4 in the procedure **Creating IAM Policies (Console)** in the **IAM User Guide**.

After you create the policy, close that tab and return to your original tab. Select the check box next to the permissions policies that you want AppStream 2.0 to have.

7. (Optional) Set a permissions boundary. This is an advanced feature that is available for service roles, but not service-linked roles. For more information, see **Permissions Boundaries for IAM Entities** in the **IAM User Guide**.
8. Choose **Next: Tags**. You can optionally attach tags as key-value pairs. For more information, see **Tagging IAM Users and Roles** in the **IAM User Guide**.
9. Choose **Next: Review**.
10. For **Role name**, type a role name that is unique within your AWS account. Because other AWS resources might reference the role, you can't edit the name of the role after it has been created.
11. For **Role description**, keep the default role description or type a new one.
12. Review the role, and then choose **Create role**.

### How to Use the IAM Role With AppStream 2.0 Streaming Instances

After you create an IAM role, you can apply it to an image builder or fleet streaming instance when you launch the image builder or create a fleet. You can also apply an IAM role to existing fleets. For information about how to apply IAM role when you launch an image builder, see **Launch an Image Builder to Install and Configure Streaming Applications** (p. 26). For information about how to apply IAM role when you create a fleet, see **Create a Fleet** (p. 82).

When you apply an IAM role to your image builder or fleet streaming instance, AppStream 2.0 retrieves temporary credentials and creates the **appstream_machine_role** credential profile on the instance. The temporary credentials are valid for 6 hours, and new credentials retrieved every hour. The previous credentials do not expire, so you can use them for as long as they are valid. You can use the credential profile to call AWS services programmatically by using the AWS Command Line Interface (AWS CLI), AWS Tools for PowerShell, or the AWS SDK with the language of your choice.

When you make the API calls, specify **appstream_machine_role** as the credential profile. Otherwise, the operation fails due to insufficient permissions.

AppStream 2.0 assumes the specified role while the streaming instance is provisioned. Because AppStream 2.0 uses the elastic network interface that is attached to your VPC for AWS API calls, your application or script must wait for the elastic network interface to become available before making AWS API calls. If API calls are made before the elastic network interface is available, the calls fail.

The following examples show how you can use the **appstream_machine_role** credential profile to describe streaming instances (EC2 instances) and to create the Boto client. Boto is the Amazon Web Services (AWS) SDK for Python.

**Describe Streaming Instances (EC2 instances) by Using the AWS CLI**
AWS ec2 describe-instances --region us-east-1 --profile appstream_machine_role

Describe Streaming Instances (EC2 instances) by Using AWS Tools for PowerShell

You must use AWS Tools for PowerShell version 3.3.563.1 or later, with the Amazon Web Services SDK for .NET version 3.3.103.22 or later. You can download the AWS Tools for Windows installer, which includes AWS Tools for PowerShell and the Amazon Web Services SDK for .NET, from the AWS Tools for PowerShell website.

Get-EC2Instance -Region us-east-1 -ProfileName appstream_machine_role

Creating the Boto Client by Using the AWS SDK for Python

session = boto3.Session(profile_name='appstream_machine_role')

Logging and Monitoring in Amazon AppStream 2.0

Monitoring is an important part of maintaining the reliability, availability, and performance of Amazon AppStream 2.0. This topic describes the services and tools that AWS provides for monitoring your AppStream 2.0 resources and responding to potential incidents.

Amazon CloudWatch Alarms

Amazon CloudWatch alarms let you watch a single metric over a time period that you specify. If the metric exceeds a given threshold, a notification is sent to an Amazon Simple Notification Service topic or AWS Auto Scaling policy. CloudWatch alarms do not invoke actions that are in a particular state. Instead, the state must have changed and been maintained for a specified number of periods. For more information, see Monitoring Amazon AppStream 2.0 Resources (p. 205).

Note
AppStream 2.0 currently can't be configured as a target for CloudWatch Events. For a list of services that you can configure as targets for CloudWatch events, see What Is Amazon CloudWatch Events.

AWS CloudTrail

AWS CloudTrail provides a record of actions taken by a user, role, or an AWS service in AppStream 2.0. This record lets you determine the request that was made to AppStream 2.0, the IP address from which the request was made, who made the request, when it was made, and additional details. For more information, see Logging AppStream 2.0 API Calls with AWS CloudTrail (p. 218).

AWS Trusted Advisor

AWS Trusted Advisor inspects your AWS environment and then recommends ways to save money, improve system availability and performance, or help close security gaps. Trusted Advisor uses best practices collected from a wide variety of AWS customers. All AWS customers have access to five Trusted Advisor checks. If you have a Business or Enterprise support plan, you can view all Trusted Advisor checks.

When you enable application settings persistence (p. 146) or home folders (p. 137) for your users, the data that is generated by your users is stored in Amazon S3 buckets. Trusted Advisor contains the following checks related to Amazon S3:

- Logging configuration of Amazon S3 buckets.
- Security checks for Amazon S3 buckets that have open access permissions.
Compliance Validation for Amazon AppStream 2.0

Third-party auditors assess the security and compliance of Amazon AppStream 2.0 as part of multiple AWS compliance programs. AppStream 2.0 compliance, certifications, and accreditations include the following:

- Health Insurance Portability and Accountability Act (HIPAA)
- Health Information Trust Alliance Common Security Framework (HITRUST CSF)
- Federal Information Processing Standard (FIPS) 140-2
- Payment Card Industry Data Security Standard (PCI DSS)
- AWS System and Organization Controls (SOC) Reports
- International Organization for Standardization (ISO)
- Undergoing Third-Party Organization (3PAO) assessment for FedRAMP Moderate and FedRAMP High

For information about how to use AppStream 2.0 FIPS endpoints for administrative use or streaming, see the section called “FIPS Endpoints” (p. 252).

For a list of AWS services in scope of specific compliance programs, see AWS Services in Scope by Compliance Program. 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 AppStream 2.0 is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. AWS provides the following resources to help with compliance:

- Security and Compliance Quick Start Guides – These deployment guides discuss architectural considerations and provide steps for deploying security- and compliance-focused baseline environments on AWS.
- Architecting for HIPAA Security and Compliance Whitepaper – This whitepaper describes how companies can use AWS to create HIPAA-compliant applications.
- AWS Compliance Resources – This collection of workbooks and guides might apply to your industry and location.
• Evaluating Resources with Rules in the AWS Config Developer Guide – The AWS Config service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.
• AWS Security Hub – This AWS service provides a comprehensive view of your security state within AWS that helps you check your compliance with security industry standards and best practices.

Resilience in Amazon AppStream 2.0

The AWS global infrastructure is built around AWS Regions and Availability Zones. Regions provide multiple physically separated and isolated Availability Zones, which are connected through 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.

Infrastructure Security in Amazon AppStream 2.0

As a managed service, Amazon AppStream 2.0 is protected by the AWS global network security procedures that are described in the Amazon Web Services: Overview of Security Processes whitepaper.

use published AWS API calls to access AppStream 2.0 through the network. Clients must support Transport Layer Security (TLS) 1.0 or later. We recommend TLS 1.2 or later. Clients must also support cipher suites with perfect forward secrecy (PFS) such as Ephemeral Diffie-Hellman (DHE) or Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed using an access key ID and a secret access key that is associated with an IAM principal. Or you can use the AWS Security Token Service (AWS STS) to generate temporary security credentials to sign requests.

The following topics provide additional information about AppStream 2.0 infrastructure security.

Contents
• Network Isolation (p. 237)
• Isolation on Physical Hosts (p. 238)
• Controlling Network Traffic (p. 238)
• Authentication of Corporate Users (p. 239)
• AppStream 2.0 Interface VPC Endpoints (p. 249)
• Protecting Data in Transit with FIPS Endpoints (p. 252)

Network Isolation

A virtual private cloud (VPC) is a virtual network in your own logically isolated area in the AWS Cloud. Use separate VPCs to isolate infrastructure by workload or organizational entity.

A subnet is a range of IP addresses in a VPC. When you launch an instance, you launch it into a subnet in your VPC. Use subnets to isolate the tiers of your application (for example, web, application, and database) within a single VPC. Use private subnets for your instances if they should not be accessed directly from the internet.
Isolation on Physical Hosts

Different streaming instances on the same physical host are isolated from each other as though they are on separate physical hosts. The hypervisor isolates CPU and memory, and the instances are provided virtualized disks instead of access to the raw disk devices.

When you stop or terminate a streaming instance, the memory allocated to it is scrubbed (meaning, it's set to zero) by the hypervisor before it is allocated to a new instance, and every block of storage is reset. This ensures that your data is not exposed to another instance.

Controlling Network Traffic

To help control network traffic to your AppStream 2.0 streaming instances, consider these options:

- When you launch an Amazon AppStream streaming instance, you launch it into a subnet in your VPC. You can deploy streaming instances in a private subnet if they should not be accessible from the internet.
- To provide internet access to your streaming instances in a private subnet, use a NAT gateway. For more information, see Configure a VPC with Private Subnets and a NAT Gateway (p. 12).
- Security groups that belong to your VPC let you control the network traffic between AppStream 2.0 streaming instances and VPC resources such as license servers, file servers, and database servers. Security groups also isolate traffic between your streaming instances and AppStream 2.0 management services.

Use security groups to restrict access to your streaming instances. For example, you can allow traffic only from the address ranges for your corporate network. For more information, see Security Groups in Amazon AppStream 2.0 (p. 253).

- You can stream from AppStream 2.0 streaming instances in your VPC without going through the public internet. To do so, use an interface VPC endpoint (interface endpoint). For more information, see Creating and Streaming from Interface VPC Endpoints (p. 250).

You can also call AppStream 2.0 API operations from your VPC without sending traffic over the public internet by using an interface endpoint. For more information, see Access AppStream 2.0 API Operations and CLI Commands Through an Interface VPC Endpoint (p. 251).

- Use IAM roles and policies to manage administrator access to AppStream 2.0, Application Auto Scaling, and Amazon S3 buckets. For more information, see Network Access to Your Streaming Instance (p. 224), Using IAM Policies to Manage Administrator Access to Application Auto Scaling (p. 229), Restricting Administrator Access to the Amazon S3 Bucket for Home Folders and Application Settings Persistence (p. 231).

- You can use SAML 2.0 to federate authentication to AppStream 2.0. For more information, see Amazon AppStream 2.0 Service Quotas (p. 271).

Note

For smaller AppStream 2.0 deployments, you can use AppStream 2.0 user pools. By default, user pools support a maximum of 50 users. For more information about AppStream 2.0 quotas (also referred to as limits), see Amazon AppStream 2.0 Service Quotas (p. 271).
deployments that must support 100 or more AppStream 2.0 users, we recommend using SAML 2.0.

Authentication of Corporate Users

The following topics provide information about Amazon AppStream 2.0 user authentication and authorization.

Contents

- AppStream 2.0 User Pools (p. 239)
- Single Sign-on Access (SAML 2.0) (p. 243)

AppStream 2.0 User Pools

The AppStream 2.0 user pool provides a simplified way to manage access to applications for your users through a persistent portal for each AWS Region. This feature is a built-in alternative to user management through Active Directory (p. 106) and SAML 2.0 federation (p. 243).

Note

By default, AppStream 2.0 user pools support a maximum of 50 users. For deployments that must support 100 or more AppStream 2.0 users, we recommend using SAML 2.0.

Note

Stacks can't be assigned to users in the user pool if the stacks are associated with a fleet that is joined to an Active Directory domain.

The AppStream 2.0 user pool provides the following key features:

- Users can access application stacks through a persistent URL and login credentials by using their email address and a password that they choose.
- Users' email addresses are case-sensitive. During login, if they specify an email address that doesn't use the same capitalization as the email address specified when their user pool account was created, a "user does not exist" error message displays.
- You can assign multiple stacks to users. Doing so enables AppStream 2.0 to display multiple application catalogs to users when they log in.
- When you create new users, a welcome email is automatically sent to them. The email includes instructions, a login portal link, and a temporary password for connecting to the login portal.
- After you create users, they are enabled unless you specifically disable them.
- You can control which users have access to which application stacks, or disable access completely.

User Pool End User Experience

The following steps summarize the initial connection experience for users in the user pool.

1. You create new users in the Region you want by specifying their email addresses.
2. AppStream 2.0 sends them a welcome email.
3. You assign one or more stacks to the users.
4. AppStream 2.0 sends them an optional notification email. This email includes information about how to access the stacks that are newly assigned to them.
5. The users connect to the login portal by entering the information included in the welcome email, and they set a permanent password. The login portal link never expires and can be used any time.
6. They sign in to AppStream 2.0 by entering their email address and permanent password.
7. After they sign in, the users can view their application catalogs.

The login portal link provided in the welcome email should be saved for future use, as it does not change and is valid for all users in the user pool. The login portal URL and users in the user pool are managed on a per-Region basis.

**Resetting a Forgotten Password**

If users forget their password, follow these steps to connect to the login portal link (provided in the welcome email) and choose a new password.

**To choose a new password**

1. Open the AppStream 2.0 login portal by using the login link provided in the welcome email.
2. Choose **Forgot Password?**.
3. Type the email address that you used to create the user in the user pool, and choose **Next**.
   
   Your email address is case-sensitive. During login, if your email address doesn't use the same capitalization as the email address specified when your user pool account was created, a "user does not exist" error message displays.
4. Check your email for the password reset request message. If you are having difficulty finding the email, check your spam email folder. Type the verification code from the email in **Verification Code**.
   
   **Note**
   
   The verification code is valid for 24 hours. If a new password is not chosen within this time, request a new verification code.
5. Following the password rules shown, type and confirm your new password. Choose **Reset Password**.

**User Pool Administration**

To create and manage users in the user pool, sign in to the AppStream 2.0 console for the AWS Region you want and choose **User Pool** in the left navigation pane. The User Pool dashboard supports bulk operations on a list of users for some actions. You can select multiple users on which to perform the same action from the **Actions** list. Users in the user pool are created and managed on a per-Region basis.

AppStream 2.0 does not support bulk user creation or disable. However, you can use Amazon Cognito with the **CreateStreamingURL** API action to manage access efficiently for multiple users. Amazon Cognito user pools let you quickly create your own directory to sign up and sign in users. In addition, you can use Amazon Cognito user pools to store user profiles. For information about how to integrate AppStream 2.0 with your Cognito User Pool, see the **Create a SaaS Portal with Amazon AppStream 2.0** project.

**Note**

AppStream 2.0 sends email to users on your behalf when a new user is created or a user is assigned to a stack. To ensure the email is delivered, add no-reply@accounts.aws-region-code.amazonappstream.com to your allow list, where **aws-region-code** is a valid AWS Region code in which you are working. If users are having difficulty finding the emails, ask them to check their "spam" email folder.

**Tasks**

- Creating a User (p. 241)
- Assigning Stacks to Users (p. 241)
- Unassigning Stacks from Users (p. 242)
- Disabling Users (p. 242)
- Enabling Users (p. 242)
Creating a User

You must enter a valid and unique email address for each new user within a Region. However, you can reuse an email address for a new user in another Region.

When you create a new user, be aware of the following:

- You can enable or disable a user, but you cannot delete a user by using the AppStream 2.0 console. To delete a user, you must use the DeleteUser API action.
- You cannot change the email address, first name, or last name for a user that you have already created. To change this information for a user, disable the user. Then, recreate the user (as a new user) and specify the updated information as needed.
- Users' email addresses are case-sensitive. During login, if they specify an email address that doesn't use the same capitalization as the email address specified when their user pool account was created, a "user does not exist" error message displays.
- You can assign one or more stacks to the user after the user is created.

To create a new user

2. In the left navigation pane, choose User Pool, Create User.
3. For Email, type the unique email address for the user.
4. Type the user's first name and last name in the corresponding fields. These fields need not be unique.
5. Choose Create User.

After users are created, AppStream 2.0 sends them a welcome email. This email includes the login portal link, the login email address to be used, and a temporary password. By browsing to the login portal and typing their temporary password, users can set a permanent password to access their applications.

By default, the new user's status is Enabled, meaning you can assign one or more stacks to the user or perform other administrative actions.

Assigning Stacks to Users

You can assign one or more stacks to one or more users in the user pool. After they are assigned to at least one stack, users can log in to AppStream 2.0 and launch applications. If users are assigned to more than one stack, they are presented with a list of stacks as catalogs to choose from before launching applications.

Note

Stacks can't be assigned to users if the stacks are associated with a fleet that is joined to an Active Directory domain.

To assign a stack to users

2. In the left navigation pane, choose User Pool and select the users you want.
3. Choose Actions, Assign stack. For more information, see Using Active Directory with AppStream 2.0 (p. 106).
4. Review the list to confirm that the correct users are specified. For Stack, choose the stack you want to assign.
5. By default, **Send email notification to user** is enabled. Clear this option if you do not want to send the notification email to users now.

6. Choose **Assign stack**.

### Unassigning Stacks from Users

You can unassign a stack from one or more users in the user pool. After a stack is unassigned from users, they can't launch applications from the stack.

**To unassign a stack from users**

2. In the left navigation pane, choose **User Pool** and select the users you want.
3. Choose **Actions, Unassign stack**.
4. Review the list to confirm that the correct users are specified. For **Stack**, choose the stack you want to unassign. The list includes all stacks, assigned or unassigned.
5. Choose **Unassign stack**.

### Disabling Users

You can disable one or more users in the user pool, one at a time. After they are disabled, users can no longer log in to AppStream 2.0 until they are re-enabled. This action does not delete users. If users are connected when you disable them, their sessions remain active until the session cookie expires (about one hour). Stack assignments for the users are retained. If the users are re-enabled, their stack assignments become active again.

**To disable a user**

2. In the left navigation pane, choose **User Pool** and select the user you want.
3. Choose **Actions, Disable user**.
4. Confirm that the correct user is specified, and choose **Disable User**.

### Enabling Users

You can enable one or more users in the user pool, one at a time. After they are enabled, users can log in to AppStream 2.0 and launch applications from the stacks to which they are assigned. If the users were disabled, these assignments are retained.

**To enable users**

2. In the left navigation pane, choose **User Pool** and select the user you want.
3. Choose **Actions, Enable user**.
4. Confirm that the correct user is specified, and choose **Enable User**.

### Re-Sending Welcome Email

You can re-send the welcome email with connection instructions to users in the user pool. Unused passwords expire after seven days. To provide a new temporary password, you must re-send the welcome email. This option is only available until users set their permanent password. If they've already set their password and forgotten it, they can set a new one. For more information, see **Resetting a Forgotten Password** (p. 240).
To resend the welcome email for a user

2. In the left navigation pane, choose User Pool and select the user you want.
3. For User Details, choose Resend welcome email.
4. Confirm that the success message displays at the top of the User Pool dashboard.

Single Sign-on Access (SAML 2.0)

Amazon AppStream 2.0 supports identity federation to AppStream 2.0 stacks through Security Assertion Markup Language 2.0 (SAML 2.0). You can use an identity provider (IdP) that supports SAML 2.0—such as Active Directory Federation Services (AD FS) in Windows Server, Ping One Federation Server, or Okta—to provide an onboarding flow for your AppStream 2.0 users.

This feature offers your users the convenience of one-click access to their AppStream 2.0 applications using their existing identity credentials. You also have the security benefit of identity authentication by your IdP. By using your IdP, you can control which users have access to a particular AppStream 2.0 stack.

Example Authentication Workflow

The following diagram illustrates the authentication flow between AppStream 2.0 and a third-party identity provider (IdP). In this example, the administrator has set up a sign-in page to access AppStream 2.0, called applications.exampleco.com. The webpage uses a SAML 2.0–compliant federation service to trigger a sign-on request. The administrator has also set up a user to allow access to AppStream 2.0.

1. The user browses to https://applications.exampleco.com. The sign-on page requests authentication for the user.
2. The federation service requests authentication from the organization's identity store.
3. The identity store authenticates the user and returns the authentication response to the federation service.
4. On successful authentication, the federation service posts the SAML assertion to the user's browser.
5. The user's browser posts the SAML assertion to the AWS Sign-In SAML endpoint (https://signin.aws.amazon.com/saml). AWS Sign-In receives the SAML request, processes the request, authenticates the user, and forwards the authentication token to AppStream 2.0.
6. Using the authentication token from AWS, AppStream 2.0 authorizes the user and presents applications to the browser.
From the user's perspective, this process happens transparently. The user starts at your organization's internal portal and is automatically redirected to an AppStream 2.0 application portal without being required to enter AWS credentials.

Setting Up SAML

To enable users to sign in to AppStream 2.0 by using their existing credentials, and start streaming applications, you can set up identity federation using SAML 2.0. To do this, use an IAM role and a relay state URL to configure your SAML 2.0-compliant identity provider (IdP) and enable AWS to permit your federated users to access an AppStream 2.0 stack. The IAM role grants users the permissions to access the stack. The relay state is the stack portal to which users are forwarded after successful authentication by AWS.

Contents

- Prerequisites (p. 244)
- Step 1: Create a SAML Identity Provider in AWS IAM (p. 244)
- Step 2: Create a SAML 2.0 Federation IAM Role (p. 244)
- Step 3: Embed an Inline Policy for the IAM Role (p. 245)
- Step 4: Configure Your SAML-Based IdP (p. 246)
- Step 5: Create Assertions for the SAML Authentication Response (p. 246)
- Step 6: Configure the Relay State of Your Federation (p. 247)

Prerequisites

Complete the following prerequisites before configuring your SAML 2.0 connection.

1. Configure your SAML-based IdP to establish a trust relationship with AWS.
   - Inside your organization's network, configure your identity store to work with a SAML-based IdP. For configuration resources, see AppStream 2.0 Integration with SAML 2.0 (p. 248).
   - Use your SAML-based IdP to generate and download a federation metadata document that describes your organization as an IdP. This signed XML document is used to establish the relying party trust. Save this file to a location that you can access from the IAM console later.
2. Use the AppStream 2.0 management console to create an AppStream 2.0 stack. You need the stack name to create the IAM policy and to configure your IdP integration with AppStream 2.0, as described later in this topic.
   - You can create an AppStream 2.0 stack by using the AppStream 2.0 management console, AWS CLI, or AppStream 2.0 API. For more information, see Create an AppStream 2.0 Fleet and Stack (p. 81).

Step 1: Create a SAML Identity Provider in AWS IAM

First, create a SAML IdP in AWS IAM. This IdP defines your organization's IdP-to-AWS trust relationship using the metadata document generated by the IdP software in your organization. For more information, see Creating and Managing a SAML Identity Provider (AWS Management Console) in the IAM User Guide.

Step 2: Create a SAML 2.0 Federation IAM Role

Next, create a SAML 2.0 federation IAM role. This step establishes a trust relationship between IAM and your organization's IdP, which identifies your IdP as a trusted entity for federation.

To create an IAM role for the SAML IdP

1. Open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane, choose Roles, Create role.
3. For Role type, choose SAML 2.0 federation.
4. For SAML Provider, select the SAML IdP that you created.
   
   Important
   Do not choose either of the two SAML 2.0 access methods (Allow programmatic access only or Allow programmatic and AWS Management Console access).
5. For Attribute, choose SAML:sub_type.
6. For Value, type persistent. This step restricts role access to only SAML user streaming requests that include a SAML subject type assertion with a value of persistent. If the SAML:sub_type is persistent, your IdP sends the same unique value for the NameID element in all SAML requests from a particular user. For more information about the SAML:sub_type assertion, see the Uniquely Identifying Users in SAML-Based Federation section in Using SAML-Based Federation for API Access to AWS.
7. Review your SAML 2.0 trust information, confirming the correct trusted entity and condition, and then choose Next: Permissions.
8. On the Attach permissions policies page, choose Next: Tags.
9. (Optional) Type a key and value for each tag that you want to add. For more information, see Tagging IAM Users and Roles.
10. When you're done, choose Next: Review. You create and embed an inline policy for this role later.
11. For Role name, type a name that helps you identify the purpose of this role. Because various entities might reference the role, you cannot edit the name of the role after it has been created.
12. (Optional) For Role description, type a description for the new role.
13. Review the role details and choose Create role.

Step 3: Embed an Inline Policy for the IAM Role

Next, embed an inline IAM policy for the role that you created. When you embed an inline policy, the permissions in the policy cannot be inadvertently attached to the wrong principal entity. The inline policy provides federated users with access to the AppStream 2.0 stack that you created.

1. In the details for the IAM role that you created, choose the Permissions tab, and then choose Add inline policy. The Create policy wizard starts.
2. In Create policy, choose the JSON tab.
3. Copy and paste the following JSON policy into the JSON window and modify the resource by entering your AWS Region Code, account ID, and stack name. In the following policy, "Action": "appstream:Stream" is the action that provides your AppStream 2.0 users with permissions to connect to streaming sessions on the stack that you created.

```json
```
Choose a value for `REGION-CODE` that corresponds to the AWS Region where your AppStream 2.0 stack exists. Replace `STACK-NAME` with the name of the stack. Note that this value is case-sensitive, so the case in the stack name that you specify in this policy must match the case in the AppStream 2.0 stack name as it appears in the Stacks dashboard of the AppStream 2.0 management console.


**Step 4: Configure Your SAML-Based IdP**

Next, depending on your SAML-based IdP, you may need to manually update your IdP to trust AWS as a service provider by uploading the `saml-metadata.xml` file at https://signin.aws.amazon.com/static/saml-metadata.xml to your IdP. This step updates your IdP's metadata. For some IdPs, the update may already be configured. If this is the case, proceed to the next step.

If this update is not already configured in your IdP, review the documentation provided by your IdP for information about how to update the metadata. Some providers give you the option to type the URL, and the IdP obtains and installs the file for you. Others require you to download the file from the URL and then provide it as a local file.

**Step 5: Create Assertions for the SAML Authentication Response**

Next, depending on your SAML-based IdP, you may need to configure the information that the IdP passes as SAML attributes to AWS as part of the authentication response. For some IdPs, this information may already be configured. If this is the case, proceed to the next step.

If this information is not already configured in your IdP, provide the following:

- **SAML Subject NameID** – The unique identifier for the user who is signing in.
  
  *Note*
  
  For stacks with domain-joined fleets, the NameID value for the user must be provided in the format of "domain\username" using the sAMAccountName or "username@domain.com" using userPrincipalName. If you are using the sAMAccountName format, you can specify the domain by using either the NetBIOS name or the fully qualified domain name (FQDN). For more information, see Using Active Directory with AppStream 2.0 (p. 106).

- **SAML Subject Type** (with a value set to `persistent`) – Setting the value to `persistent` ensures that your IdP sends the same unique value for the NameID element in all SAML requests from a particular user. Make sure that your IAM policy includes a condition to only allow SAML requests with a SAML sub_type set to `persistent`, as described in the section called “Step 2: Create a SAML 2.0 Federation IAM Role” (p. 244).

- **Attribute element with the Name attribute set to `https://aws.amazon.com/SAML/Attributes/Role`** – This element contains one or more `AttributeValue` elements that list the IAM role and SAML IdP to which the user is mapped by your IdP. The role and IdP are specified as a comma-delimited pair of ARNs.

- **Attribute element with the Name attribute set to `https://aws.amazon.com/SAML/Attributes/RoleSessionName`** – This element contains one `AttributeValue` element that provides an identifier for the AWS temporary credentials that are issued for SSO. The value in the `AttributeValue` element must be between 2 and 64 characters long, can contain only alphanumeric characters, underscores, and the following characters: + (plus sign), = (equals sign), , (comma), . (period), @ (at symbol), and - (hyphen). It cannot contain spaces. The value is typically a user ID (bobsmit) or an email address (bobsmit@example.com). It should not be a value that includes a space, such as a user's display name (Bob Smith).

- **Attribute element with the `SessionDuration` attribute set to `https://aws.amazon.com/SAML/Attributes/SessionDuration (optional)`** – This element contains one `AttributeValue` element that specifies the maximum amount of time that a federated streaming session for a user can remain active.
before reauthentication is required. The default value is 60 minutes. For more information, see the *An optional Attribute element with the SessionDuration attribute set to* https://aws.amazon.com/SAML/Attributes/SessionDuration section in Configuring SAML Assertions for the Authentication Response.

For more information about how to configure these elements, see Configuring SAML Assertions for the Authentication Response in the IAM User Guide. For information about specific configuration requirements for your IdP, see the documentation for your IdP.

**Step 6: Configure the Relay State of Your Federation**

Finally, use your IdP to configure the relay state of your federation to point to the AppStream 2.0 stack relay state URL. After successful authentication by AWS, the user is directed to the AppStream 2.0 stack portal, defined as the relay state in the SAML authentication response.

The format of the relay state URL is as follows:

```
https://relay-state-region-endpoint?stack=stackname&accountId=aws-account-id-without-hyphens
```

Construct your relay state URL from your AWS account ID, stack name, and the relay state endpoint associated with the Region in which your stack is located.

Optionally, you can specify the name of the application that you want to launch automatically. To find the application name, select the image in the AppStream 2.0 console, choose the Applications tab, and note the name that displays in the Application Name column. Alternatively, if you haven't yet created the image, connect to the image builder where you installed the application, and open Image Assistant. The names of applications display in the Add Apps tab.

```
https://relay-state-region-endpoint?stack=stackname&accountId=aws-account-id-without-hyphens&app=application-name-to-launch
```

The following table lists the relay state endpoints for the Regions where AppStream 2.0 is available. If you want your users to stream using a FIPS-compliant connection, you must use a FIPS-compliant endpoint. For more information about FIPS endpoints, see the section called “FIPS Endpoints” (p. 252).

<table>
<thead>
<tr>
<th>Region</th>
<th>Relay state endpoint</th>
</tr>
</thead>
<tbody>
<tr>
<td>US East (N. Virginia)</td>
<td><a href="https://appstream2.us-east-1.aws.amazon.com/saml">https://appstream2.us-east-1.aws.amazon.com/saml</a></td>
</tr>
<tr>
<td></td>
<td><em>(FIPS)</em> <a href="https://appstream2-fips.us-east-1.aws.amazon.com/saml">https://appstream2-fips.us-east-1.aws.amazon.com/saml</a></td>
</tr>
<tr>
<td>US West (Oregon)</td>
<td><a href="https://appstream2.us-west-2.aws.amazon.com/saml">https://appstream2.us-west-2.aws.amazon.com/saml</a></td>
</tr>
<tr>
<td></td>
<td><em>(FIPS)</em> <a href="https://appstream2-fips.us-west-2.aws.amazon.com/saml">https://appstream2-fips.us-west-2.aws.amazon.com/saml</a></td>
</tr>
<tr>
<td>Asia Pacific (Seoul)</td>
<td><a href="https://appstream2.ap-northeast-2.aws.amazon.com/saml">https://appstream2.ap-northeast-2.aws.amazon.com/saml</a></td>
</tr>
<tr>
<td>Asia Pacific (Singapore)</td>
<td><a href="https://appstream2.ap-southeast-1.aws.amazon.com/saml">https://appstream2.ap-southeast-1.aws.amazon.com/saml</a></td>
</tr>
<tr>
<td>Asia Pacific (Sydney)</td>
<td><a href="https://appstream2.ap-southeast-2.aws.amazon.com/saml">https://appstream2.ap-southeast-2.aws.amazon.com/saml</a></td>
</tr>
<tr>
<td>Region</td>
<td>Relay state endpoint</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------</td>
</tr>
<tr>
<td>Asia Pacific (Tokyo)</td>
<td><a href="https://appstream2.ap-northeast-1.amazonaws.com/saml">https://appstream2.ap-northeast-1.amazonaws.com/saml</a></td>
</tr>
<tr>
<td>Europe (Frankfurt)</td>
<td><a href="https://appstream2.eu-central-1.amazonaws.com/saml">https://appstream2.eu-central-1.amazonaws.com/saml</a></td>
</tr>
<tr>
<td>Europe (Ireland)</td>
<td><a href="https://appstream2.eu-west-1.amazonaws.com/saml">https://appstream2.eu-west-1.amazonaws.com/saml</a></td>
</tr>
</tbody>
</table>

**Note**: For more information about using AppStream 2.0 in AWS GovCloud (US-West), see Amazon AppStream 2.0 in the AWS GovCloud (US) User Guide.

### AppStream 2.0 Integration with SAML 2.0

The following links help you configure third-party SAML 2.0 identity provider solutions to work with AppStream 2.0.

<table>
<thead>
<tr>
<th>IdP solution</th>
<th>More information</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS Single Sign-On (SSO)</td>
<td>Enable federation with AWS Single Sign-On and Amazon AppStream 2.0 — Describes how to use AWS SSO to federate user access to your AppStream 2.0 applications with their existing enterprise credentials.</td>
</tr>
<tr>
<td>Active Directory Federation Services (AD FS) for Windows Server</td>
<td>Enabling Identity Federation with AD FS 3.0 and Amazon AppStream 2.0 — Describes how to provide users with SSO access to AppStream 2.0 by using their existing enterprise credentials. You can configure federated identities for AppStream 2.0 by using AD FS 3.0.</td>
</tr>
<tr>
<td>Azure Active Directory (Azure AD)</td>
<td>Enabling Federation with Azure AD Single Sign-On and Amazon AppStream 2.0 — Describes how to configure federated user access for Amazon AppStream 2.0 by using Azure AD SSO for enterprise applications.</td>
</tr>
<tr>
<td>GG4L School Passport™</td>
<td>Enabling Identity Federation with GG4L's School Passport™ and Amazon AppStream 2.0 — Describes how to configure GG4L's School Passport™ to federate login to AppStream 2.0.</td>
</tr>
<tr>
<td>Google</td>
<td>Setting up G Suite SAML 2.0 federation with Amazon AppStream 2.0 — Describes how to use the G Suite Admin console to set up SAML federation to AppStream 2.0 for users in G Suite domains.</td>
</tr>
<tr>
<td>Okta</td>
<td>How to Configure SAML 2.0 for Amazon AppStream 2.0 — Describes how to use Okta to set up SAML federation to AppStream 2.0. For stacks that are joined to a domain,</td>
</tr>
</tbody>
</table>
### IdP solution | More information
--- | ---
| Ping Identity | the "Application username format" must be set to "AD user principal name". Configuring an SSO connection to Amazon AppStream 2.0 — Describes how to set up single sign-on (SSO) to AppStream 2.0. | Shibboleth | Single Sign-On: Integrating AWS, OpenLDAP, and Shibboleth — Describes how to set up the initial federation between the Shibboleth IdP and the AWS Management Console. You must complete the following additional steps to enable federation to AppStream 2.0. Step 4 of the AWS Security whitepaper describes how to create IAM roles that define the permissions that federated users have to the AWS Management Console. After you create these roles and embed the inline policy as described in the whitepaper, modify this policy so that it provides federated users with permissions to access only an AppStream 2.0 stack. To do this, replace the existing policy with the policy noted in Step 3: Embed an Inline Policy for the IAM Role, in Setting Up SAML (p. 244). When you add the stack relay state URL as described in Step 6: Configure the Relay State of Your Federation, in Setting Up SAML (p. 244), add the relay state parameter to the federation URL as a target request attribute. For information about configuring relay state parameters, see the SAML 2.0 section in the Shibboleth documentation. | VMware WorkSpace ONE | Federating Access to Amazon AppStream 2.0 from VMware Workspace ONE — Describes how to use the VMware Workspace ONE platform to federate user access to your AppStream 2.0 applications. |

For solutions to common problems you may encounter, see Troubleshooting (p. 256).

For more information about additional supported SAML providers, see Integrating Third-Party SAML Solution Providers with AWS in the IAM User Guide.

## AppStream 2.0 Interface VPC Endpoints

A virtual private cloud (VPC) is a virtual network in your own logically isolated area in the AWS Cloud. If you use Amazon Virtual Private Cloud to host your AWS resources, you can establish a private connection between your VPC and AppStream 2.0. You can use this connection to enable AppStream 2.0 to communicate with your resources on your VPC without going through the public internet.

Interface endpoints are powered by AWS PrivateLink, a technology that lets you keep streaming traffic within a VPC that you specify by using private IP addresses. When you use the VPC with an AWS Direct Connect or AWS Virtual Private Network (VPN) tunnel, you can keep the streaming traffic within your network.

The following topics provide information about AppStream 2.0 interface endpoints.
Creating and Streaming from Interface VPC Endpoints

You can use an interface VPC endpoint in your AWS account to restrict all network traffic between your Amazon VPC and AppStream 2.0 to the Amazon network. After you create this endpoint, you configure your AppStream 2.0 stack or image builder to use it.

Prerequisites

Before you set up interface VPC endpoints for AppStream 2.0, be aware of the following prerequisites:

- Internet connectivity is required to authenticate users and deliver the web assets that AppStream 2.0 requires to function. The streaming interface endpoint maintains the streaming traffic within your VPC. Streaming traffic includes pixel, USB, user input, audio, clipboard, file upload and download, and printer traffic. To allow this traffic, you must allow the domains listed in Allowed Domains (p. 24).
- The network to which your users' devices are connected must be able to route traffic to the interface endpoint.
- The security groups that are associated with the interface endpoint must allow inbound access to port 443 (TCP) and ports 1400-1499 (TCP) from the IP address range from which your users connect.
- The network access control list for the subnets must allow outbound traffic from ephemeral network ports 1024-65535 (TCP) to the IP address range from which your users connect.
- You must have an IAM permissions policy in your AWS account that provides permissions to perform the `ec2:DescribeVpcEndpoints` API action. By default, this permission is defined in the IAM policy that is attached to the AmazonAppStreamServiceAccess role. If you have the required permissions, this service role is automatically created by AppStream 2.0, with the required IAM policies attached, when you get started with the AppStream 2.0 service in an AWS Region. For more information, see Identity and Access Management for Amazon AppStream 2.0 (p. 224).

To create an interface endpoint

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, choose **Endpoints**, Create Endpoint.
3. Choose **Create Endpoint**.
4. For **Service category**, ensure that **AWS services** is selected.
5. For **Service Name**, choose `com.amazonaws.<AWS Region>.appstream.streaming`.
6. Specify the following information. When you're done, choose **Create endpoint**.
   - For **VPC**, choose a VPC in which to create the interface endpoint. You can choose a different VPC than the VPC with AppStream 2.0 resources.
   - For **Subnets**, choose the subnets (Availability Zones) in which to create the endpoint network interfaces. We recommend that you choose subnets in at least two Availability Zones.
   - Ensure that the **Enable Private DNS Name** check box is selected.
   - For **Security group**, choose the security groups to associate with the endpoint network interfaces.

   **Note**
   The security groups must provide inbound access to the ports from the IP address range from which your users connect.

While your interface endpoint is being created, the status of the endpoint in the console appears as **Pending**. After your endpoint is created, the status changes to **Available**.
To update a stack to use the interface endpoint that you created for streaming sessions, perform the following steps.

**To update a stack to use a new interface endpoint**

   
   Ensure that you open the console in the same AWS Region as the interface endpoint that you want to use.
2. In the navigation pane, choose **Stacks**, and then choose the stack that you want.
3. Choose the **Network Access Endpoints** tab, and then choose **Edit**.
4. In the **Edit Network Access Endpoint** dialog box, for **Streaming Endpoint**, choose the endpoint through which you want to stream traffic.
5. Choose **Update**.

Traffic for new streaming sessions will be routed through this endpoint. However, traffic for current streaming sessions continues to be routed through the previously specified endpoint.

**Note**

Users cannot stream using the internet endpoint when an interface endpoint is specified.

**Access AppStream 2.0 API Operations and CLI Commands Through an Interface VPC Endpoint**

If you use Amazon Virtual Private Cloud to host your AWS resources, you can connect directly to AppStream 2.0 API operations or command line interface (CLI) commands through an interface VPC endpoint (interface endpoint) in your virtual private cloud (VPC) instead of connecting over the internet.

Interface endpoints are powered by AWS PrivateLink, a technology that lets you keep streaming traffic within a VPC that you specify by using private IP addresses. When you use an interface endpoint, communication between your VPC and AppStream 2.0 is conducted entirely and securely within the AWS network.

**Note**

This topic describes how to access the AppStream 2.0 API operations and CLI commands through an interface endpoint. For information about how to create and stream from AppStream 2.0 interface endpoints, see Creating and Streaming from Interface VPC Endpoints (p. 250).

**Prerequisites**

To use interface endpoints, you must meet the following prerequisites:

- The security groups that are associated with the interface endpoint must allow inbound access to port 443 (TCP) from the IP address range from which your users connect.
- The network access control list for the subnets must allow outbound traffic from ephemeral network ports 1024-65535 (TCP) to the IP address range from which your users connect.

**Create an Interface Endpoint to Access AppStream 2.0 API Operations and CLI Commands**

Perform the following steps to create an interface endpoint.

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, choose **Endpoints, Create Endpoint**.
3. Choose Create Endpoint.
4. For Service category, ensure that AWS services is selected.
5. For Service Name, choose com.amazonaws.<AWS Region>.appstream.api.
6. Specify the following information. When you’re done, choose Create endpoint.
   - For VPC, select a VPC in which to create the interface endpoint.
   - For Subnets, select the subnets (Availability Zones) in which to create the endpoint network interfaces. We recommend that you choose subnets in at least two Availability Zones.
   - Optionally, you can select the Enable Private DNS Name check box.
     Note
     If you select this option, ensure that you configure VPC and DNS as needed to support private DNS. For more information, see Private DNS in the Amazon VPC User Guide.
   - For Security group, select the security groups to associate with the endpoint network interfaces.
     Note
     The security groups must provide inbound access to the ports from the IP address range from which your users connect.

While your interface endpoint is being created, the status of the endpoint in the console appears as Pending. After your endpoint is created, the status changes to Available.

Use an Interface Endpoint to Access AppStream 2.0 API Operations and CLI Commands

After the status of the interface VPC endpoint that you create changes to Available, you can use the endpoint to access AppStream 2.0 API operations and CLI commands. To do so, specify the endpoint-url parameter with the DNS name of the interface endpoint when you use these operations and commands. The DNS name is publicly resolvable, but it only successfully routes traffic in your VPC.

The following example shows how to specify the DNS name of the interface endpoint when you use the describe-fleets CLI command:

```bash
aws appstream describe-fleets --endpoint-url <vpc-endpoint-id>.api.appstream.<aws-region>.vpce.amazonaws.com
```

The following example shows how to specify the DNS name of the interface endpoint when you instantiate the AppStream 2.0 Boto3 Python client:

```python
appstream2client = boto3.client('appstream',region_name='<aws-region>',endpoint_url='<vpc-endpoint-id>.api.appstream.<aws-region>.vpce.amazonaws.com'
```

Subsequent commands using the appstream2client object automatically use the interface endpoint that you specified.

If you enabled the private DNS host names on the interface endpoint, you don’t need to specify the endpoint URL. The AppStream 2.0 API DNS host name that the API and CLI use by default resolves within your VPC. For more information about private DNS host names, see Private DNS in the Amazon VPC User Guide.

Protecting Data in Transit with FIPS Endpoints

By default, when you communicate with the AppStream 2.0 service, whether as an administrator using the AppStream 2.0 console, the AWS Command Line Interface (AWS CLI), or an AWS SDK, or as a user streaming from an image builder or a fleet instance, all data in transit is encrypted using SSL.
Alternatively, AppStream 2.0 offers FIPS-compliant endpoints (FIPS endpoints) in all United States AWS Regions where AppStream 2.0 is available. When you use a FIPS endpoint, all data in transit is encrypted using cryptographic standards that comply with Federal Information Processing Standard (FIPS) 140-2. For information about FIPS endpoints, including a list of AppStream 2.0 endpoints, see Federal Information Processing Standard (FIPS) 140-2.

### FIPS Endpoints for Administrative Use

To specify a FIPS endpoint when you run an AWS CLI command for AppStream 2.0, use the `endpoint-url` parameter. The following example uses the AppStream 2.0 FIPS endpoint in the US West (Oregon) Region to retrieve a list of all stacks in the Region:

```
aws appstream describe-stacks --endpoint-url https://appstream2-fips.us-west-2.amazonaws.com
```

To specify a FIPS endpoint for AppStream 2.0 API operations, use the procedure in your AWS SDK for specifying a custom endpoint.

### FIPS Endpoints for User Streaming Sessions

If you use SAML 2.0 or a streaming URL to authenticate users, you can configure FIPS-compliant connections for your users’ streaming sessions.

To use a FIPS-compliant connection for users who authenticate using SAML 2.0, specify an AppStream 2.0 FIPS endpoint when you configure the relay state of your federation. For more information about constructing a relay state URL for identity federation using SAML 2.0, see Setting Up SAML (p. 244).

To configure a FIPS-compliant connection for users who authenticate through a streaming URL, specify an AppStream 2.0 FIPS endpoint when you call the `CreateStreamingURL` or `CreateImageBuilderStreamingURL` operation from the AWS CLI or an AWS SDK. A user who connects to a streaming instance using the resulting URL is connected through a FIPS-compliant connection. The following example uses the AppStream 2.0 FIPS endpoint in the US East (Virginia) Region to generate a FIPS-compliant streaming URL:

```
aws appstream create-streaming-url --stack-name stack-name --fleet-name fleet-name --user-id user-id --endpoint-url https://appstream2-fips.us-east-1.amazonaws.com
```

### Exceptions

FIPS-compliant connections are not supported in the following scenarios:

- Administration of AppStream 2.0 through the AppStream 2.0 console
- Streaming sessions for users who authenticate using the AppStream 2.0 user pool feature
- Streaming using an interface VPC endpoint
- Generating FIPS-compliant streaming URLs through the AppStream 2.0 console
- Connections to your Google Drive or OneDrive storage accounts where your storage provider does not provide a FIPS endpoint

### Security Groups in Amazon AppStream 2.0

You can provide additional access control to your VPC from streaming instances in a fleet or an image builder in Amazon AppStream 2.0 by associating them with VPC security groups. Security groups that
belong to your VPC allow you to control the network traffic between AppStream 2.0 streaming instances and VPC resources such as license servers, file servers, and database servers. For more information, see Security Groups for your VPC in the Amazon VPC User Guide.

The rules that you define for your VPC security group are applied when the security group is associated with a fleet or image builder. The security group rules determine what network traffic is allowed from your streaming instances. For more information, see Security Group Rules in the Amazon VPC User Guide.

You can associate up to five security groups while launching a new image builder or while creating a new fleet. You can also associate security groups to an existing fleet or change the security groups of a fleet. For more information, see Working with Security Groups in the Amazon VPC User Guide.

If you don't select a security group, your image builder or fleet is associated with the default security group for your VPC. For more information, see Default Security Group for Your VPC in the Amazon VPC User Guide.

Use these additional considerations when using security groups with AppStream 2.0.

- All end user data, such as internet traffic, home folder data, or application communication with VPC resources, are affected by the security groups associated with the streaming instance.
- Streaming pixel data is not affected by security groups.
- If you have enabled default internet access for your fleet or image builder, the rules of the associated security groups must allow internet access.

You can create or edit rules for your security groups or create new security groups using the Amazon VPC console.

- **To associate security groups with an image builder** — Follow the instructions at Launch an Image Builder to Install and Configure Streaming Applications (p. 26).
- **To associate security groups with a fleet**
  - **While creating the fleet** — Follow the instructions at Create a Fleet (p. 82).
  - **For an existing fleet** — Edit the fleet settings using the AWS Management Console.

You can also associate security groups to your fleets using the AWS CLI and SDKs.

- **AWS CLI** — Use the create-fleet and update-fleet commands.
- **AWS SDKs** — Use the CreateFleet and UpdateFleet API operations.

For more information, see the AWS Command Line Interface User Guide and Tools for Amazon Web Services.

### Update Management in Amazon AppStream 2.0

AppStream 2.0 provides an automated way to update your image builder with newer AppStream 2.0 software. When your images are configured to always use the latest AppStream 2.0 agent version, your streaming instances are automatically updated with the latest features, performance improvements, and security updates that are available from AWS. For information about how to manage AppStream 2.0 agent versions, see Manage AppStream 2.0 Agent Versions (p. 36).

You are responsible for installing and maintaining the updates for the Windows operating system, your applications, and their dependencies. For more information, see Keep Your AppStream 2.0 Image Up-to-Date (p. 55).
To manage updates for applications on your streaming instances, you can use any automatic update services provided. You can also follow the recommendations for installing updates provided by the application vendor.
Troubleshooting

If you encounter difficulties when working with Amazon AppStream 2.0, consult the following troubleshooting resources.

Contents

- General Troubleshooting (p. 256)
- Troubleshooting Image Builders (p. 260)
- Troubleshooting Fleets (p. 264)
- Troubleshooting Active Directory Domain Join (p. 266)
- Troubleshooting Notification Codes (p. 268)

General Troubleshooting

The following are possible general issues you might have while using Amazon AppStream 2.0.

Issues

- SAML federation is not working. The user is not authorized to view AppStream 2.0 applications. (p. 256)
- After federating from an ADFS portal, my streaming session doesn't start. I am getting the error "Sorry connection went down". (p. 257)
- I get an invalid redirect URI error. (p. 257)
- My image builders and fleets never reach the running state. My DNS servers are in a Simple AD directory. (p. 257)
- My stack's home folders aren't working correctly. (p. 257)
- My users can't access their home folder directory from one of our applications. (p. 257)
- I've enabled application settings persistence for my users, but their persistent application settings aren't being saved or loaded. (p. 258)
- I've enabled app settings persistence for my users, but for certain streaming apps, my users' passwords aren't persisting across sessions. (p. 258)
- Google Chrome data is filling the VHD file that contains my users' persistent application settings. This is preventing their settings from persisting. How can I manage the Chrome profile? (p. 259)
- My users can't copy and paste between their local device and their streaming session. (p. 259)
- Some keyboard shortcuts aren't working for users during their streaming sessions. (p. 260)

SAML federation is not working. The user is not authorized to view AppStream 2.0 applications.

This might happen because the inline policy that is embedded for the SAML 2.0 federation IAM role does not include permissions to the stack ARN. The IAM role is assumed by the federated user who is accessing an AppStream 2.0 stack. Edit the role permissions to include the stack ARN. For more information, see
After federating from an ADFS portal, my streaming session doesn't start. I am getting the error "Sorry connection went down".

Set the claim rule's **Incoming Claim Type** for the **NameID** SAML attribute to **UPN** and try the connection again.

**I get an invalid redirect URI error.**

This error occurs due to a malformed or invalid AppStream 2.0 stack relay state URL. Make sure that the relay state configured in your federation setup is the same as the stack relay state that is displayed in the stack details through the AppStream 2.0 console. If they are the same and the problem still persists, contact AWS Support. For more information, see Single Sign-on Access (SAML 2.0) (p. 243).

**My image builders and fleets never reach the running state. My DNS servers are in a Simple AD directory.**

AppStream 2.0 relies on the DNS servers within your VPC to return a non-existent domain (NXDOMAIN) response for local domain names that don't exist. This enables the AppStream 2.0-managed network interface to communicate with the management servers.

When you create a directory with Simple AD, AWS Directory Service creates two domain controllers that also function as DNS servers on your behalf. Because the domain controllers don't provide the NXDOMAIN response, they can't be used with AppStream 2.0.

**My stack's home folders aren't working correctly.**

Problems with home folder backup to an S3 bucket can occur in the following scenarios:

- There is no internet connectivity from the streaming instance, or there is no access to the private Amazon S3 VPC endpoint, if applicable.
- Network bandwidth consumption is too high. For example, multiple large files are being downloaded or streamed by the user while the service is trying to back up a home folder that contains large files to Amazon S3.
- An administrator deleted the bucket created by the service.
- An administrator incorrectly edited the Amazon S3 permissions for the **AmazonAppStreamServiceAccess** service role.

For more information, see the Amazon Simple Storage Service Console User Guide and Amazon Simple Storage Service Developer Guide.

**My users can't access their home folder directory from one of our applications.**

Some applications do not recognize the redirect that displays the home folder as a top-level folder in File Explorer. If this is the case, your users can access their home folder from within an application during a...
streaming session by choosing **File Open** from the application interface and browsing to either of the following directories:

- Non-domain-joined: `C:\Users\PhotonUser\My Files\Home Folder`
- Domain-joined: `C:\Users\%username%\My Files\Home Folder`

I've enabled application settings persistence for my users, but their persistent application settings aren't being saved or loaded.

AppStream 2.0 automatically saves application settings that are created in certain locations on the Windows instance. The settings are saved only if your application saves them to one of these locations. For a list of supported locations, see *How Application Settings Persistence Works* (p. 146). If your application is configured to save to `C:\Users\%username%` and your users' settings for the application aren't persisting between sessions, the mount point might not be created. This prevents the settings from being saved to the VHD file that contains your users' persistent application settings.

To resolve this issue, follow these steps:

1. On the fleet instance, open File Explorer and browse to the user profile directory at `C:\Users\%username%`.
2. Confirm whether this directory contains a symlink, and then do either of the following:
   - If there is a symlink, confirm that it points to `D:\%username%`.
   - If there isn't a symlink, try to delete the `C:\Users\%username%` directory.

   If you can't delete this directory, identify the file in the directory that is preventing it from being deleted and the application that created the file. Then contact the application vendor for information about how to change the file permissions or move the file.

   If you can delete this directory, contact AWS Support for further guidance to resolve this issue. For more information, see [AWS Support Center](https://aws.amazon.com/support/).

I've enabled app settings persistence for my users, but for certain streaming apps, my users’ passwords aren't persisting across sessions.

This issue occurs when:

- Users are streaming applications such as Microsoft Outlook, which use the [Microsoft Data Protection API](https://docs.microsoft.com/en-us/windows/win32/seceditor/dpapi).
- App settings persistence is enabled for streaming instances that are not joined to Active Directory domains.

In cases where a streaming instance is not joined to an Active Directory domain, the Windows user, PhotonUser, is different on each fleet instance. Due to the way in which the DPAPI security model works, users’ passwords don’t persist for applications that use DPAPI in this scenario. In cases where streaming instances are joined to an Active Directory domain and the user is a domain user, the Windows user name is that of the logged in user, and users’ passwords persist for applications that use DPAPI.
Google Chrome data is filling the VHD file that contains my users' persistent application settings. This is preventing their settings from persisting. How can I manage the Chrome profile?

By default, Google Chrome stores both user data and the local disk cache in the Windows user profile. To prevent the local disk cache data from filling the VHD file that contains users' persistent application settings, configure Chrome to save only the user data. To do so, on the fleet instance, open the command line as an administrator and start Chrome with the following parameters to change the location of the disk cache:

```
chrome.exe --disk-cache-dir C:\path-to-unsaved-location\n```

Running Chrome with these parameters prevents the disk cache from being persisted between AppStream 2.0 sessions.

My users can't copy and paste between their local device and their streaming session.

AppStream 2.0 takes advantage of the W3C specification for enabling asynchronous clipboard operations in web applications. This enables users to copy and paste content between their local device and their streaming session in the same ways that they copy and paste between applications on their local device, including using keyboard shortcuts.

The only browser that currently supports the W3C asynchronous clipboard specification is Google Chrome version 66 or later, which supports copying and pasting only for text. For all other browsers, users can use the clipboard feature in the AppStream 2.0 web portal, which provides a dialog box for copying or pasting text.

If your users run into issues using the clipboard during their streaming sessions, you can provide them with the following information:

- I'm using Chrome version 66 or later, and keyboard shortcuts aren't working.

  Chrome displays a prompt for you to choose whether to allow AppStream 2.0 to access content copied to the clipboard. Choose **Allow** to enable pasting to your remote session. If you're copying text from your remote session to your local device, both the Chrome application and the tab containing your streaming session must stay in focus on your local device long enough for the text to be copied from your streaming session. Small amounts of text should be copied almost immediately, but for large amounts of text, you may need to wait 1 to 2 seconds before switching away from Chrome or from the tab containing your streaming session. The time required to copy the text may vary based on network conditions.

- Copying and pasting doesn't work when I try to copy and paste a large amount of text.

  AppStream 2.0 has a 2 MB limit for the amount of text that you can copy and paste between your local device and your streaming session. If you try to copy more than 2 MB, no text is copied. This limit doesn't apply if you try to copy and paste text between applications on your local device or between applications in your streaming session. If you need to copy or paste text greater than 2 MB between your local device and your streaming session, you can divide it into smaller chunks or upload it as a file instead.

- I'm using the AppStream 2.0 web portal clipboard feature to paste text to my streaming session and it's not working.
In some cases, after you paste text into the clipboard dialog box and the dialog box closes, nothing happens when you try to use keyboard shortcuts to paste the text in your streaming session. This issue occurs because when the clipboard dialog box appears, it takes the focus away from your streaming application. After the dialog box closes, the focus may not automatically return to your streaming application. Clicking your streaming application should return the focus to it and enable you to use keyboard shortcuts to paste your text into your streaming session.

Some keyboard shortcuts aren't working for users during their streaming sessions.

The following keyboard shortcuts work on users' local computers, but are not passed to AppStream 2.0 streaming sessions:

Windows:
- Win+L
- Ctrl+Alt+Del

Mac:
- Ctrl+F3
- All shortcuts that use Alt or Option key combinations

This issue is due to the following limitations on users' local computers:
- The keyboard shortcuts are filtered by the operating system that is running on users' local computers and not propagated to the browsers on which users are accessing AppStream 2.0. This behavior applies to the Windows Win+L and Ctrl+Alt+Del keyboard shortcuts and Mac Ctrl+F3 keyboard shortcut.
- When used with web applications, some keyboard shortcuts are filtered by the browser and don’t generate an event for the web applications. As a result, the web applications can’t respond to the keyboard shortcuts typed by users.
- The keyboard shortcuts are translated by the browser before a keyboard event is generated and so are not translated correctly. For example, Alt key combinations and Option key combinations on Mac computers are translated as if they are Alt Graph key combinations on Windows. When this occurs, the results are not as the users intend when they use these key combinations.

Troubleshooting Image Builders

The following are possible issues you might have while using Amazon AppStream 2.0 image builders.

Issues
- I cannot connect to the internet from my image builder. (p. 261)
- When I tried installing my application, I see an error that the operating system version is not supported. (p. 261)
- I want to use a Windows PowerShell script to open my applications. (p. 261)
- I want to make ClickOnce applications available to users. (p. 261)
- When I connect to my image builder, I see a login screen asking me to enter Ctrl+Alt+Delete to log in. However, my local machine intercepts the key strokes. (p. 262)
- When I switched between admin and test modes, I saw a request for a password. I don't know how to get a password. (p. 262)
I cannot connect to the internet from my image builder.

Image builders cannot communicate to the internet by default. To resolve this issue, launch your image builder in a VPC subnet that has internet access. You can enable internet access from your VPC subnet using a NAT gateway. Alternatively, you can configure an internet gateway in your VPC, and attach an Elastic IP address to your image builder. For more information, see Networking and Access for Amazon AppStream 2.0 (p. 10).

When I tried installing my application, I see an error that the operating system version is not supported.

Only applications that can be installed on Windows Server 2012 R2, Windows Server 2016, and Windows Server 2019 can be added to an AppStream 2.0 image. Check if your application is supported on one of these three operating systems, as applicable for your image builder.

I want to use a Windows PowerShell script to open my applications.

You can use Windows PowerShell scripts to open your applications in the fleet instance. You may want to do this to configure the application or environment before the application opens. To launch a Windows PowerShell script for your application, specify the PowerShell .exe file in Image Assistant. Navigate to C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe, and specify the following launch parameters:

-file "C:\Path\To\PowerShell\Script.ps1"

Note
To allow the specified script to open the application, you must override the PowerShell script execution policy. To do so, add -ExecutionPolicy Bypass to the launch parameter.

I want to make ClickOnce applications available to users.

To make a ClickOnce application available to your AppStream 2.0 users, you must install the application on your image builder first as an Administrator, and then as a Template User. Because ClickOnce
When I connect to my image builder, I see a login screen asking me to enter Ctrl+Alt+Delete to log in. However, my local machine intercepts the key strokes. If your client may intercept certain key combinations locally instead of sending them to the image builder session. To reliably send the Ctrl+Alt+Delete key combination to the image builder, choose Admin Commands, Send Ctrl+Alt+Delete. The Admin Commands menu is available on the top right corner of the image builder session toolbar.

When I switched between admin and test modes, I saw a request for a password. I don't know how to get a password.

AppStream 2.0 usually logs you into the user mode that you choose automatically. On some occasions, the switch may not happen automatically. If a password is requested, choose Admin Commands, Log me in. This sends a one-time password, securely, to your image builder and pastes it into the Password field.

I get an error when I add my installed application.

Check if your application type is supported. You can add applications of the types .exe, .lnk, and .bat.

Check if your application is installed under the C:\Users folder hierarchy. Any application installed under C:\Users is not supported. Select a different installation folder under C:\ when installing the application.
I accidentally quit a background service on the image builder and got disconnected. I am now unable to connect to that image builder.

Try stopping the image builder, restarting it and connecting to it again. If the problem persists, you must launch (create) a new image builder. Do not stop any background services running on the image builder instance. Doing so may interrupt your image builder session or interfere with the image creation.

The application fails to launch in test mode.

Check if your application requires elevated user privileges or any special permissions that are usually available only to an administrator. The Image Builder Test mode has the same limited permissions on the image builder instance as your end users have on an AppStream 2.0 test fleet. If your applications require elevated permissions, they do not launch in the Image Builder Test mode.

The application could not connect to a network resource in my VPC.

Check if the image builder was launched in the correct VPC subnet. You may also need to verify that the route tables in your VPC are configured to allow a connection.

I customized my image builder desktop, but my changes are not available when connecting to a session after launching a fleet from the image I created.

Changes that are saved as part of a local user session, such as time settings, are not persisted when creating an image. To persist any local user session changes, add them to the local group policy on the image builder instance.

My application is missing a command line parameter when launching.

You can provide a command line parameter when using image builder to add an application to an image. If the launch parameters for the application do not change for each user, you can enter them while adding an application to the image in the image builder instance.

If the launch parameters are different for every launch, you can pass them programmatically when using the CreateStreamingURL API. Set the sessionContext and applicationID parameters in the API fields. The sessionContext is included as a command line option when launching the application.

If the launch parameters must be computed on the fly, you can launch your application using a script. You can parse the sessionContext parameter within your script before launching your application with a computed parameter.
I am unable to use my image with a fleet after installing an antivirus application.

You can install any tools, including antivirus programs, on your AppStream 2.0 stack by using the image builder before creating an image. However, these programs should not block any network ports or stop any processes that are used by the AppStream 2.0 service. We recommend testing your application in Image Builder Test mode before creating an image and attempting to use it with a fleet.

My image creation failed.

Verify that you did not make any changes to AppStream 2.0 services before starting the image creation. Try creating your image again; if it fails, contact AWS Support. For more information, see AWS Support Center.

The Image Assistant create-image operation failed with an error message that access to the PrewarmManifest.txt is denied

The application optimization manifest was created with elevated privileges. To create the image, do either of the following, and then try again:

- Run the Image Assistant command line interface (CLI) executable file (Image-Assistant.exe) with administrator privileges.
- Delete the application optimization manifest file.

Troubleshooting Fleets

The following are possible issues that might occur when users connect to streaming sessions launched from fleet instances.

Issues

- My applications won't work correctly unless I use the Internet Explorer defaults. How do I restore the Internet Explorer default settings? (p. 264)
- I need to persist environment variables across my fleet instances. (p. 265)
- I want to change the default Internet Explorer home page for my users. (p. 265)
- When my users end a streaming session and then start a new one, they see a message that says no streaming resources are available. (p. 266)

My applications won't work correctly unless I use the Internet Explorer defaults. How do I restore the Internet Explorer default settings?

If your AppStream 2.0 environment includes applications that render elements, you might need to restore the Internet Explorer default settings to enable full enable access to the internet.

To automatically restore the Internet Explorer default settings

2. In the left navigation pane, choose **Images, Image Builder**.
3. Choose the image builder on which to restore the Internet Explorer default settings, verify that it is in the **Running** state, and choose **Connect**.
4. Log in to the image builder by doing either of the following:
   - If your image builder is not joined to an Active Directory domain, on the **Local User** tab, choose **Template User**.
   - If your image builder is joined to an Active Directory domain, choose the **Directory User** tab, enter the credentials for a domain user that does not have local administrator permissions on the image builder, then choose **Log in**.
5. Open Internet Explorer and reset your settings by doing the following:
   a. In the upper right area of the Internet Explorer browser window, choose the **Tools** icon, then choose **Internet options**.
   b. Choose the **Advanced** tab, then choose **Reset**.
   c. When prompted to confirm your choice, choose **Reset** again.
   d. When the **Reset Internet Explorer Settings** message displays, choose **Close**.
6. In the upper right area of the image builder desktop, choose **Admin Commands, Switch User**.

7. This disconnects your current session and opens the login menu. Do either of the following:
   - If your image builder is not joined to an Active Directory domain, on the **Local User** tab, choose **Administrator**.
   - If your image builder is joined to an Active Directory domain, choose the **Directory User** tab, and log in as a domain user who has local administrator permissions on the image builder.
8. On the image builder desktop, open Image Assistant.
9. Follow the necessary steps in Image Assistant to finish creating your image. For more information, see Tutorial: Create a Custom AppStream 2.0 Image by Using the AppStream 2.0 Console (p. 47).

**I need to persist environment variables across my fleet instances.**

Environment variables enable you to dynamically pass settings across applications. You can make user environment variables and system environment variables available across your fleet instances. You can also create environment variables with limited scope, which is useful when you need to use the same environment variable with different values across different applications. For more information, see Persist Environment Variables (p. 87).

**I want to change the default Internet Explorer home page for my users.**

You can use Group Policy to set the default home page in Internet Explorer for your users. You can also enable users to change the default page that you set. For more information, see Change the Default Internet Explorer Home Page for Users’ Streaming Sessions (p. 92).
When my users end a streaming session and then start a new one, they see a message that says no streaming resources are available.

When a user ends a session, AppStream 2.0 terminates the underlying instance and creates a new instance if needed to meet the desired capacity of the fleet. If a user tries to start a new session before AppStream 2.0 creates the new instance and all other instances are in use, the user will receive an error stating that no streaming resources are available. If your users start and stop sessions frequently, consider increasing your fleet capacity. For more information, see Fleet Auto Scaling for Amazon AppStream 2.0 (p. 97). Or, consider increasing the maximum session duration for your fleet and instructing your users to close their browser during periods of inactivity rather than ending their session.

Troubleshooting Active Directory Domain Join

The following are possible issues you might have while setting up and using Active Directory with Amazon AppStream 2.0. For help troubleshooting notification codes, see Troubleshooting Notification Codes (p. 268).

Issues

- My image builders and fleet instances are stuck in the PENDING state. (p. 266)
- My users aren't able to log in with the SAML application. (p. 266)
- My fleet instances work for one user but don't cycle correctly. (p. 267)
- My user Group Policy objects aren't applying successfully. (p. 267)
- My AppStream 2.0 streaming instances aren't joining the Active Directory domain. (p. 267)
- User login is taking a long time to complete on a domain-joined streaming session. (p. 268)
- My users can't access a domain resource in a domain-joined streaming session but they can access the resource from a domain-joined image builder. (p. 268)

My image builders and fleet instances are stuck in the PENDING state.

Image builders and fleet instances can take up to 25 minutes to move into a ready state and become available. If your instances are taking longer than 25 minutes to become available, in Active Directory, verify whether new computer objects were created in the correct organizational units (OUs). If there are new objects, the streaming instances will be available soon. If the objects aren't there, check the directory configuration details in your AppStream 2.0 Directory Config: Directory name (the fully qualified domain name of the directory, service account username and password, and the OU distinguished name.

Image builder and fleet errors are displayed in the AppStream 2.0 console on the Notifications tab for the fleet or image builder. Fleet errors are also available using the AppStream 2.0 API via the DescribeFleets operation, or the CLI command describe-fleets.

My users aren't able to log in with the SAML application.

AppStream 2.0 relies on the SAML Subject "NameID" attribute from your identity provider to populate the username field to log in your user. The username can either be formatted as "domain\username", or...
"user@domain.com". If you are using "domain\username" format, domain can either be the NetBIOS name or the fully qualified domain name. If using "user@domain.com" format, the UserPrincipalName attribute can be used. If you've verified your SAML_Signature attribute is configured correctly and the problem persists, contact AWS Support. For more information, see AWS Support Center.

My fleet instances work for one user but don't cycle correctly.

Fleet instances are cycled after a user completes a session, ensuring that each user has a new instance. When the cycled fleet instance is brought online, it joins the domain using the computer name of the previous instance. To ensure that this operation happens successfully, the service account requires Change Password and Reset Password permissions on the organizational unit (OU) to which the computer object is joining. Check the service account permissions and try again. If the problem persists, contact AWS Support. For more information, see AWS Support Center.

My user Group Policy objects aren't applying successfully.

By default, computer objects apply computer-level policies based on the OU in which the computer object resides, while applying user-level policies based on the OU in which the user resides. If your user-level policies aren't being applied, you can do one of the following:

- Move the user-level policies to the OU in which the user Active Directory object resides
- Enable computer-level "loopback processing," which applies the user-level policies in the computer object OU.

For more information, see Loopback processing of Group Policy at Microsoft Support.

My AppStream 2.0 streaming instances aren't joining the Active Directory domain.

The Active Directory domain to use with AppStream 2.0 must be accessible through its fully qualified domain name (FQDN) via the VPC in which your streaming instances are launched.

To test that your domain is accessible

1. Launch an Amazon EC2 instance in the same VPC, subnet, and security groups that you use with AppStream 2.0.
2. Manually join the EC2 instance to your Active Directory domain using the FQDN (for example, yourdomain.exampleco.com) with the service account that you intend to use with AppStream 2.0. Use the following command in a Windows PowerShell console:

   ```
   netdom join computer /domain:FQDN /OU:path /ud:user /pd:password
   ```

   If this manual join fails, proceed to the next step.
3. If you cannot manually join to your domain, open a command prompt and verify that you can resolve the FQDN using the nslookup command. For example:

   ```
   nslookup yourdomain.exampleco.com
   ```
User login is taking a long time to complete on a domain-joined streaming session.

AppStream 2.0 performs a Windows login action after users provide their domain password. After successful authentication, AppStream 2.0 launches the application. The login and launch times are impacted by many variables, such as network contention for the domain controllers or the time it takes to apply Group Policy settings to the streaming instance. If domain authentication takes too long to complete, try performing the following actions.

- Minimize the network latency from your AppStream 2.0 Region to your domain controllers by choosing the correct domain controllers. For example, if your fleet is in us-east-1, use domain controllers with high bandwidth and low latency to us-east-1 through Active Directory Sites and Services zone mappings. For more information, see Active Directory Sites and Services in the Microsoft documentation.
- Ensure that your Group Policy settings and user login scripts don't take prohibitively long to apply or run.

If your domain users' login to AppStream 2.0 fails with the message "An unknown error occurred," you may need to update the Group Policy settings described in Before You Begin Using Active Directory with AppStream 2.0 (p. 108). Otherwise, these settings may prevent AppStream 2.0 from authenticating and logging in your domain users.

My users can't access a domain resource in a domain-joined streaming session but they can access the resource from a domain-joined image builder.

Confirm that your fleet is created in the same VPC, subnets, and security groups as your image builder, and that your user has the appropriate permissions to access and use the domain resource.

Troubleshooting Notification Codes

The following are notification codes and resolution steps for notifications you may see while setting up and using Amazon AppStream 2.0. These notifications can be found in the Notifications tab in the
AppStream 2.0 console, after selecting an image builder or fleet. Fleet notifications can also be obtained using the AppStream 2.0 API operation DescribeFleets, or using the describe-fleets CLI command.

**Active Directory Internal Service**

Following is a notification code and resolution steps for an internal service error that you might encounter while setting up and using Active Directory with Amazon AppStream 2.0.

**INTERNAL_SERVICE_ERROR**

*Message:* The user name or password is incorrect.

*Resolution:* This error may occur when the computer object that was created in the Microsoft Active Directory domain for the resource was deleted or disabled. You can resolve this error by enabling the computer object in the Active Directory domain, then starting the resource again. You may also need to reset the computer object account in the Active Directory domain. If you continue to encounter this error, contact AWS Support. For more information, see AWS Support Center.

**Active Directory Domain Join**

The following are notification codes and resolution steps for issues with domain join that you might encounter while setting up and using Active Directory with Amazon AppStream 2.0.

**DOMAIN_JOIN_ERROR_ACCESS_DENIED**

*Message:* Access is denied.

*Resolution:* The service account specified in the directory configuration does not have permissions to create the computer object, or reuse an existing one. Validate the permissions and start the image builder or fleet. For more information, see Granting Permissions to Create and Manage Active Directory Computer Objects (p. 111).

**DOMAIN_JOIN_ERROR_LOGON_FAILURE**

*Message:* The username or password is incorrect.

*Resolution:* The service account specified in the directory configuration has an invalid username or password. Update the configuration and re-create the image builder or fleet that had the error.

**DOMAIN_JOIN_NERR_PASSWORD_EXPIRED**

*Message:* The password of this user has expired.

*Resolution:* The password for the service account specified in the AppStream 2.0 directory configuration has expired. Change the password for the service account in your Active Directory domain, then update the configuration, and re-create the image builder or fleet that had the error.

**DOMAIN_JOIN_ERROR_DS_MACHINE_ACCOUNT_QUOTA_EXCEEDED**

*Message:* Your computer could not be joined to the domain. You have exceeded the maximum number of computer accounts you are allowed to create in this domain. Contact your system administrator to have this limit reset or increased.

*Resolution:* The service account specified on the directory configuration does not have permissions to create the computer object, or reuse an existing one. Validate the permissions and start the image builder or fleet. For more information, see Granting Permissions to Create and Manage Active Directory Computer Objects (p. 111).

**DOMAIN_JOIN_ERROR_INVALID_PARAMETER**

*Message:* A parameter is incorrect. This error is returned if the LpName parameter is NULL or the NameType parameter is specified as NetSetupUnknown or an unknown nametype.
Resolution: This error can occur when the distinguished name for the OU is incorrect. Validate the OU chosen. If you continue to encounter this error, contact AWS Support. For more information, see AWS Support Center.

DOMAIN_JOIN_ERROR_MORE_DATA

Message: More data is available.

Resolution: This error can occur when the distinguished name for the OU is incorrect. Validate the OU chosen. If you continue to encounter this error, contact AWS Support. For more information, see AWS Support Center.

DOMAIN_JOIN_ERROR_NO_SUCH_DOMAIN

Message: The specified domain either does not exist or could not be contacted.

Resolution: The streaming instance was unable to contact your Active Directory domain. To ensure network connectivity, confirm your VPC, subnet, and security group settings. For more information, see My AppStream 2.0 streaming instances aren't joining the Active Directory domain. (p. 267)

DOMAIN_JOIN_NERR_WORKSTATION_NOT_STARTED

Message: The Workstation service has not been started.

Resolution: An error occurred starting the Workstation service. Ensure that the service is enabled in your image. If you continue to encounter this error, contact AWS Support. For more information, see AWS Support Center.

DOMAIN_JOIN_ERROR_NOT_SUPPORTED

Message: The request is not supported. This error is returned if a remote computer was specified in the lpServer parameter and this call is not supported on the remote computer.

Resolution: Contact AWS Support for assistance. For more information, see AWS Support Center.

DOMAIN_JOIN_ERROR_FILE_NOT_FOUND

Message: The system cannot find the file specified.

Resolution: This error occurs when an invalid organizational unit (OU) distinguished name is provided. The distinguished name must start with ou=. Validate the OU distinguished name and try again. For more information, see Finding the Organizational Unit Distinguished Name (p. 112).

DOMAIN_JOIN_INTERNALSERVICE_ERROR

Message: The account already exists.

Resolution: This error can occur in either of the following scenarios:

- The service account specified in the directory configuration does not have permissions to create the computer object or reuse an existing one. If this is the case, validate the permissions and start the image builder or fleet. For more information, see Granting Permissions to Create and Manage Active Directory Computer Objects (p. 111).

- After AppStream 2.0 creates the computer object, the object is moved from the OU in which it was created. In this case, the first image builder or fleet is created successfully, but any new image builder or fleet that uses the computer object fails. When Active Directory searches for the computer object in the specified OU and detects that an object with the same name exists elsewhere in the domain, the domain join does not succeed.
Amazon AppStream 2.0 Service Quotas

By default, AWS sets quotas (also referred to as limits) for the resources that you can create and the number of users who can use the service. To request a quota increase, use the AppStream 2.0 Limits form.

The following table lists the quotas for each AppStream 2.0 resource. Where no default quota is listed for a specific instance family (p. 80) or instance type, the quota is 0.

**Default Quotas Per AWS Region Per Account**

<table>
<thead>
<tr>
<th>Resource</th>
<th>Default Quota</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stacks</td>
<td>5</td>
</tr>
<tr>
<td>Fleets</td>
<td>5</td>
</tr>
<tr>
<td>Fleet instances*</td>
<td></td>
</tr>
<tr>
<td>• Stream.standard.medium: 5</td>
<td></td>
</tr>
<tr>
<td>• Stream.standard.large: 5</td>
<td></td>
</tr>
<tr>
<td>• Stream.graphics-design.large: 2</td>
<td></td>
</tr>
<tr>
<td>Image builder instances</td>
<td></td>
</tr>
<tr>
<td>• Stream.standard.medium: 5</td>
<td></td>
</tr>
<tr>
<td>• Stream.standard.large: 5</td>
<td></td>
</tr>
<tr>
<td>• Stream.graphics-design.large: 1</td>
<td></td>
</tr>
<tr>
<td>Images</td>
<td>5</td>
</tr>
<tr>
<td>Number of AWS accounts an image can be shared with</td>
<td>100</td>
</tr>
<tr>
<td>Concurrent image copies</td>
<td>2 per destination Region</td>
</tr>
<tr>
<td>Image copies (per month)</td>
<td>20</td>
</tr>
<tr>
<td>Users in the user pool</td>
<td>50</td>
</tr>
</tbody>
</table>

*For fleets that have **Default Internet Access** enabled, the quota is 100 fleet instances. If your deployment must support more than 100 concurrent users, use the **NAT gateway configuration** (p. 12) instead. For more information about enabling internet access for a fleet, see **Internet Access** (p. 10).
Guidance for AppStream 2.0 Users

If you are an AppStream 2.0 administrator, you can provide your users with the guidance in this section to help them get started with using AppStream 2.0.

If you are a user who now has access to AppStream 2.0, the topics in this section will help you use AppStream 2.0 for application streaming. With AppStream 2.0, your administrator makes your applications available for you to access remotely, so that you don't have to install the applications on your own device. To access your applications, connect to AppStream 2.0 and start an application streaming session.

Contents
- AppStream 2.0 Access Methods and Clients (p. 272)
- File Storage Options (p. 278)
- Configure Regional Settings (p. 284)

AppStream 2.0 Access Methods and Clients

You can connect to AppStream 2.0 by using a web browser or the AppStream 2.0 client for Windows.

Contents
- Web Browser Access (p. 272)
- AppStream 2.0 Client Application for Windows (p. 275)

Web Browser Access

The following information will help you get started with using a web browser to connect to AppStream 2.0 and stream applications.

Contents
- Requirements (p. 272)
- Setup (p. 273)
- Connect to AppStream 2.0 (p. 273)
- Monitors and Display Resolution (p. 273)
- Touchscreen Devices (p. 273)

Requirements

You can connect to AppStream 2.0 from any location by using an HTML5-capable web browser. Supported browsers include the following:

- Google Chrome
- Mozilla Firefox
• Safari
• Microsoft Edge
• Microsoft Internet Explorer version 11 or later

Setup

No browser extensions or plugins are required to use AppStream 2.0 in a web browser.

Connect to AppStream 2.0

To connect to AppStream 2.0 and start an application streaming session, perform these steps.

1. If you received a welcome email that notifies you to start accessing your apps using AppStream 2.0, open the email and click the Login page link.
2. The AppStream 2.0 sign-in page opens in your browser.
3. Type your email address and the temporary password that was provided in the email, and choose Log in.
4. When prompted, type a new password, confirm it, and then choose Set Password.
5. The AppStream 2.0 application portal opens, displaying applications that your administrator has made available to you for streaming.
6. Click an application to start it.

Monitors and Display Resolution

You can use dual monitors for application streaming sessions that are started on the following web browsers:

• Google Chrome
• Mozilla Firefox
• Safari

For browser-based streaming sessions on dual monitors, a maximum display resolution of 2560x1440 pixels is supported per monitor. If you require more than two monitors, or a display resolution that is greater than 2560x1440 pixels per monitor, you must use the AppStream 2.0 client.

Touchscreen Devices

AppStream 2.0 supports gestures on touch-enabled iPads, Android tablets, and Windows devices. Examples of supported touch gestures include long-tap to right-click, swipe to scroll, pinch to zoom, and two-finger rotation for supporting applications.

To display the on-screen keyboard on an iPad or Android tablet, tap the keyboard icon on the AppStream 2.0 toolbar. The keyboard icon turns blue, and you can use the on-screen keyboard to input text within the streaming application. Tap the keyboard icon again to hide the on-screen keyboard.

Tap the Fn icon to display a row of Windows-specific keys and keyboard shortcuts.
Following is an example of how Windows-specific keys and keyboard shortcuts are displayed when the Fn icon is tapped. Swipe to the left on the shortcut toolbar to display more keys.

To use a key combination that includes the Windows Control key, tap the Ctrl key on the shortcut toolbar and then type any key on either the on-screen keyboard or the shortcut toolbar. Tapping the Ctrl key changes the color to blue. In this case, any other key that you select is interpreted as a key combination that includes the Control key.

Tap the Ctrl key again to release it. For example, to use the keyboard shortcut Ctrl + F, tap the Ctrl key on the shortcut toolbar, and then type the f key on the on-screen keyboard. Tap the Ctrl key on the shortcut toolbar again to release the Control key. To use shortcuts that include the Alt or Shift keys, tap the Alt key and/or the Shift key on the shortcut toolbar in the same way. You can use the Shift key on the shortcut toolbar only for keyboard shortcuts. This key doesn’t affect the capitalization of keys that you type on the on-screen keyboard.

**Note**

The remote keyboard, the on-screen keyboard that is displayed when a user taps the keyboard icon on the AppStream 2.0 toolbar, is different than the local keyboard, the on-screen keyboard that a touch-enabled device automatically displays when a user taps inside an input control in a locally running application. During AppStream 2.0 streaming sessions, the remote keyboard can be used to input text into streaming applications only. Users can display or hide the remote keyboard only by tapping the keyboard icon on the AppStream 2.0 toolbar. A blue keyboard icon on the AppStream 2.0 toolbar indicates that the remote keyboard is active. The local keyboard can be used to input text into elements of the AppStream 2.0 web portal, including the My Files dialog box. This keyboard can’t be used, however, to input text into streaming applications. Also, users can’t display or hide it by using the keyboard icon on the AppStream 2.0 toolbar.

AppStream 2.0 doesn’t display the keyboard icon or the Fn icon when streaming to a Windows device because Windows already provides a way to display an on-screen keyboard that includes common Windows keys. To display the on-screen keyboard on a Windows computer, tap the keyboard icon in the Windows system tray. If the keyboard icon doesn’t appear in the Windows
system tray, switch to Windows tablet mode. Tap the keyboard icon in the Windows system tray again to hide the on-screen keyboard.

AppStream 2.0 Client Application for Windows

The following information will help you get started with using the AppStream 2.0 client for Windows to connect to AppStream 2.0 and stream applications.

Contents

• Features (p. 275)
• Requirements (p. 275)
• Setup (p. 275)
• Connect to AppStream 2.0 (p. 276)
• USB Devices (p. 277)
• Monitors and Display Resolution (p. 278)

Features

The AppStream 2.0 client for Windows is an application that you install on your Windows PC. This application provides additional capabilities that are not available when you access AppStream 2.0 by using a web browser. For example, the AppStream 2.0 client lets you do the following:

• Use more than two monitors or 4K resolution.
• Use your USB devices with applications streamed through AppStream 2.0.
• Use keyboard shortcuts during your streaming sessions.
• Access your local drives and folders during your streaming sessions.
• Interact with your remote streaming applications in much the same way as you interact with locally installed applications.

Requirements

The AppStream 2.0 client for Windows must be installed on a PC that meets the following requirements:

• Operating system — Windows 7, Windows 8, or Windows 10 (32-bit or 64-bit)
• RAM — 2 GB minimum
• Hard drive space — 200 MB minimum

In addition, to install the AppStream 2.0 USB driver for USB driver support, you must have local administrator rights on your PC.

Setup

To install the client, perform these steps.

1. On the PC where you want to install the AppStream 2.0 client, download the AppStream 2.0 client for Windows application from AppStream 2.0 supported clients.
2. Navigate to the location where you downloaded the application .exe file, and double-click the file to begin the installation.
Important
If nothing happens when you double-click the file, or an error message displays, contact your network administrator. Your organization may be using antivirus software that prevents the AppStream 2.0 client installation program from running.

3. If a message notifies you that Windows Defender SmartScreen prevented an unrecognized app from starting, choose the More info link. For App, verify that AmazonAppStreamClientSetup_<version-number>.exe displays, and then choose Run anyway.
4. The installation wizard displays links to the AWS Customer Agreement, AWS Service Terms, and the AWS Privacy Notice, and third-party notices. Review this information, and choose Next.
5. On the Client Diagnostics page, to enable the AppStream 2.0 client to automatically upload device logs to help with troubleshooting issues, keep Client logging selected, and choose Next.
6. On the Optional Components page, to enable your USB devices to be used with streaming applications, select the AppStream 2.0 Client USB Driver check box, and choose Finish.
7. If the AppStream 2.0 USB driver wizard setup wizard opens, choose Install.
8. If prompted by User Account Control to choose whether to allow the app to make changes to your device, choose Yes.
9. A message notifies you when the USB driver installation is complete. Choose Close.
10. The AppStream 2.0 sign-in page opens. For information about how to connect to AppStream 2.0 and start an application streaming session, see How to Connect to AppStream 2.0 (p. 277).

Connect to AppStream 2.0

After the AppStream 2.0 client for Windows is installed on your PC, you can use it to connect to AppStream 2.0.

AppStream 2.0 Client Connection Modes

The AppStream 2.0 client provides two connection modes: Native application mode and classic mode. The connection mode that you choose determines the options that are available to you during application streaming, and how your streaming applications function and display.

Native application mode

Native application mode lets you work with remote streaming applications in much the same way that you work with applications that are installed on your local PC.

When you connect to AppStream 2.0 in native application mode, the AppStream 2.0 Application Launcher window opens and displays the list of applications that are available for you to stream. When you open a streaming application in this mode, the AppStream 2.0 Application Launcher window remains open, and the application opens in its own window. During your streaming session, the remote streaming application functions in much the same way as a locally installed application. The application icon displays in the taskbar of your local PC, just as the icons do for your local applications. Unlike the icons for your local applications, however, the icons for your streaming applications in native application mode include the AppStream 2.0 logo.

During your AppStream 2.0 streaming session, you can switch quickly between your locally installed applications and your remote streaming applications by clicking the taskbar icon of the remote or local application you want to work with. You can also switch AppStream 2.0 connection modes. If you want to work in classic mode instead, you can switch from native application mode to classic mode.

Classic mode

When you use classic application mode, you work with remote streaming applications within the AppStream 2.0 session window. If your administrator has made more than one application available to you, you can open multiple applications during your session. All applications that you open, however, display within the same AppStream 2.0 session window.
When you connect to AppStream 2.0 in classic mode, the AppStream 2.0 Application Launcher window opens and displays the list of applications that are available for you to stream. When you open a streaming application in this mode, the Application Launcher window closes, and the application opens in the AppStream 2.0 session window.

If your administrator has not disabled native application mode, you can switch from classic mode to native application mode. For more information, see How to Switch AppStream 2.0 Connection Modes (p. 277).

How to Connect to AppStream 2.0

To connect to AppStream 2.0 and start an application streaming session, perform these steps.

1. On your local PC where the AppStream 2.0 client is installed, in the lower left of your screen, choose the Windows search icon on the taskbar, and type AppStream in the Search box.
2. In the search results, select Amazon AppStream to start the AppStream 2.0 client.
3. On the AppStream 2.0 sign-in page, do either of the following:
   - To use native application mode, keep the Start in native application mode check box selected.
   - To use classic mode, clear the Start in native application mode check box.
4. If your AppStream 2.0 administrator has provided you with a web address (URL) to use to connect to AppStream 2.0 for application streaming, enter the URL, and choose Connect.

If you want to use your USB devices with streaming applications, you must first share your device with AppStream 2.0. For more information, see USB Devices (p. 277).

How to Switch AppStream 2.0 Connection Modes

If your administrator has not disabled native application mode for your streaming sessions, you can switch between native application mode and classic mode.

Switch from native application mode to classic mode

1. In the upper left of the AppStream 2.0 Application Launcher window, choose the Settings icon, and then choose Switch to classic mode.
2. When you switch to classic mode, the Application Launcher window closes and the AppStream 2.0 session window opens. Any application that you are streaming in native application mode opens within the AppStream 2.0 session window.

Perform the following steps to switch from classic mode to native application mode.

Switch from classic mode to native application mode

1. In the upper left of the AppStream 2.0 session window, choose the Settings icon, and choose Switch to native application mode.
2. When you switch from classic mode back to native application mode, the AppStream 2.0 session window closes and the AppStream 2.0 Application Launcher window opens. Any application that you are streaming in classic mode opens in a separate window.

USB Devices

To use a USB device with streaming applications, you must share the device with AppStream 2.0 every time you start a new streaming session.
1. Use the AppStream 2.0 client to start a streaming session.
2. In the top left area, choose the Settings icon, and then choose USB Devices.
3. If your USB device is connected to your computer, the USB device name appears in the dialog box. If your USB device is not detected, contact your AppStream 2.0 administrator for assistance.
4. Switch the Share toggle key next to the name of the USB device that you want to share with the streaming session.

Your USB device is now available for use with your streaming applications.

**Important**
USB devices can't be simultaneously used between local and remote applications. So after you share a USB device with a streaming session, you can't use it with applications on your local computer. To use your USB device on your local computer, switch the Share toggle key next to the name of the USB device that you want to use locally. This disables sharing with the streaming session.

5. You can also enable your USB device to automatically connect when a new streaming session starts. To do so, select the option next to the toggle key for the USB device that you want to connect. After you enable this option, when your next streaming session starts, the USB device is automatically connected.

**Monitors and Display Resolution**

The AppStream 2.0 client supports the following:

- Multiple monitors (up to 2K resolution) — Up to 4 monitors and a maximum display resolution of 2560x1440 pixels per monitor
- Multiple monitors (4K resolution) — Up to 2 monitors and a maximum display resolution of 4096x2160 pixels per monitor

**File Storage Options**

If your AppStream 2.0 administrator has enabled it, you can use one or more of the following storage options for your files and folders during application streaming sessions.

- Home folders (p. 278)
- Google Drive (p. 279)
- OneDrive for Business (p. 281)

**Use Home Folders**

If your AppStream 2.0 administrator has enabled this file storage option, when you are signed in to an AppStream 2.0 streaming session, you can use your home folder. You can do the following with your home folder:

- Open and edit files and folders that you store in your home folder. Content that is stored in your home folder is specific to you and cannot be accessed by other users.
- Upload and download files between your local computer and your home folder. AppStream 2.0 continuously checks for the most recently modified files and folders and backs them up to your home folder.
- When you are working in an application, you can access files and folders that are stored in your home folder. Choose File Open from the application and browse to the file or folder that you want to open.
To save your changes in a file to your home folder, choose **File Save** from the application interface, and browse to the location in your home folder where you want to save the file.

- You can also access your home folder by choosing **My Files** from the web view session toolbar.

  **Note**
  If your home folder doesn't appear, view your home folder files by browsing to the following directory in File Explorer: C:\Users\PhotonUser\My Files\Home Folder.

---

### To upload and download files between your local computer and your home folder

1. In the top left of the AppStream 2.0 toolbar, choose the **My Files** icon.
2. Navigate to an existing folder, or choose **Add Folder** to create a new folder.
3. When the folder that you want is displayed, do one of the following:
   - To upload a file to the folder, select the file that you want to upload, and choose **Upload**.
   - To download a file from the folder, select the file that you want to download, choose the down arrow to the right of the file name, and choose **Download**.

---

### Use Google Drive

If your AppStream 2.0 administrator has enabled this file storage option, you can add your Google Drive account to AppStream 2.0. After you add your account and you sign in to an AppStream 2.0 streaming session, you can do the following in Google Drive:

- Open and edit files and folders that you store in Google Drive. Content that is stored in your Google Drive is specific to you. Other users cannot access your content unless you choose to share it.
- Upload and download files between your local computer and Google Drive. Any changes that you make to your files and folders in Google Drive during a streaming session are automatically backed up and synchronized. They are available to you when you sign in to your Google Drive account and access Google Drive outside of your streaming session.
- When you are working in an application, you can access your files and folders that are stored in Google Drive. Choose **File, Open** from the application interface and browse to the file or folder that you want to open. To save your changes in a file to your Google Drive, choose **File, Save** from the application and browse to the location in Google Drive where you want to save the file.
- You can also access Google Drive by choosing **My Files** from top left of the AppStream 2.0 toolbar.

---

### To add your Google Drive account to AppStream 2.0

To access your Google Drive during AppStream 2.0 streaming sessions, you must first add your Google Drive account to AppStream 2.0.

1. In the top left of the AppStream 2.0 toolbar, choose the **My Files** icon.
2. In the **My Files** dialog box, choose **Add Storage**.

3. Choose **Google Drive**.

4. Choose the domain for your Google Drive account.

5. The **Sign in with Google** dialog box displays. Type the user name and password for your Google Drive account when prompted.

6. After your Google Drive account is added to AppStream 2.0, your Google Drive folder displays in **My Files**.
7. To work with your files and folders in Google Drive, choose the Google Drive folder and browse to the file or folder you want. If you do not want to work with files in Google Drive during this streaming session, close the My Files dialog box.

To upload and download files between your local computer and your Google Drive

1. In the top left of the AppStream 2.0 toolbar, choose the My Files icon.
2. In the My Files dialog box, choose Google Drive.
3. Navigate to an existing folder, or choose Add Folder to create a new folder.
4. When the folder that you want displays, do one of the following:
   • To upload a file to the folder, select the file that you want to upload, and choose Upload.
   • To download a file from the folder, select the file that you want to download, choose the down arrow to the right of the file name, and choose Download.

Use OneDrive for Business

If your AppStream 2.0 administrator has enabled this file storage option, you can add your OneDrive account to AppStream 2.0. After you add your account and sign in to an AppStream 2.0 streaming session, you can do the following in OneDrive:

• Open and edit files and folders that you store in OneDrive. Content that is stored in OneDrive is specific to you. Other users cannot access your content unless you choose to share it.
• Upload and download files between your local computer and OneDrive. Any changes that you make to your files and folders in OneDrive during a streaming session are automatically backed up and synchronized. They are available to you when you sign in to your OneDrive account and access OneDrive outside of your streaming session.

• When you are working in an application, you can access your files and folders that are stored in OneDrive. Choose File, Open from the application interface and browse to the file or folder that you want to open. To save your changes in a file to OneDrive, choose File, Save from the application and browse to the location in OneDrive where you want to save the file.

• You can also access OneDrive by choosing My Files from the top left of the AppStream 2.0 toolbar.

To add your OneDrive account to AppStream 2.0

To access your OneDrive during AppStream 2.0 streaming sessions, you must first add your OneDrive account to AppStream 2.0.

1. In the top left of the AppStream 2.0 toolbar, choose the My Files icon.
2. In the My Files dialog box, choose Add Storage.
3. Choose OneDrive.
4. Under Login accounts, choose the domain for your OneDrive account.
5. The **Sign in** dialog box displays. Type the user name and password for your account when prompted, then sign in.

6. After your OneDrive account is added to AppStream 2.0, your OneDrive folder displays in **My Files**.

7. To work with your files and folders in OneDrive, choose the **OneDrive** folder and browse to the file or folder you want. If you do not want to work with files in OneDrive during this streaming session, close the **My Files** dialog box.

### To upload and download files between your local computer and your OneDrive

1. In the top left of the AppStream 2.0 toolbar, choose the **My Files** icon.
2. In the **My Files** dialog box, choose **OneDrive**.
3. Navigate to an existing folder, or choose **Add Folder** to create a new folder.
4. When the folder that you want displays, do one of the following:
   - To upload a file to the folder, select the file that you want to upload, and choose **Upload**.
   - To download a file from the folder, select the file that you want to download, choose the down arrow to the right of the file name, and choose **Download**.
To remove OneDrive permissions from AppStream 2.0

If you no longer want to use OneDrive during your AppStream 2.0 streaming sessions, follow these steps to remove OneDrive permissions from AppStream 2.0.

**Note**
You can restore these permissions at any time during an AppStream 2.0 streaming session.

1. Sign in to Office 365 or OneDrive for Business.
2. In the right pane, under **My accounts**, choose **My account**.
3. On the account dashboard page, in **App permissions**, choose **Change app permissions**.
4. On the **App permissions** page, under **Amazon AppStream 2.0**, choose **Revoke**.

Configure Regional Settings

You can configure regional settings so that your AppStream 2.0 streaming sessions use settings that are specific to your location or language. Changes that you make during your streaming session are applied to future streaming sessions.

To configure regional settings for your AppStream 2.0 streaming sessions

1. In the top left of the AppStream 2.0 toolbar, choose the **Settings** icon, and then choose **Regional settings**.
2. In the **Regional settings** dialog box, set the following options as needed. When you’re done, choose **Save**.
   - **Time zone** — Determines the system time used by Windows and any applications that rely on the operating system time.
   - **Locale** (also known as culture) — Determines how Windows displays numbers, currency, time, and dates. AppStream 2.0 supports the following locales: Chinese (Simplified and Traditional), Dutch, English, French, German, Italian, Japanese, Korean, Portuguese, Spanish, and Thai.
   - **Input method** — Determines the keystroke combinations that can be used to input characters in another language.
Document History for Amazon AppStream 2.0

- **API version:** 2016-12-01

The following table describes important additions to the AppStream 2.0 service (including AppStream 2.0 base image, AppStream 2.0 agent, and AppStream 2.0 client releases) and to the Amazon AppStream 2.0 Administration Guide documentation from June 4, 2018 onward. We also update the documentation frequently to address the feedback that you send us.

For notification about these updates, you can subscribe to the Amazon AppStream 2.0 RSS feed.

<table>
<thead>
<tr>
<th>update-history-change</th>
<th>update-history-description</th>
<th>update-history-date</th>
</tr>
</thead>
<tbody>
<tr>
<td>AppStream 2.0 base image update</td>
<td>Updates for Base, Graphics Design, Graphics Pro: Includes Microsoft Windows updates up to February 11, 2020; AWS CLI version 1.17.5; and Amazon SSM Agent version 2.3.842.0</td>
<td>March 18, 2020</td>
</tr>
<tr>
<td>AppStream 2.0 base image update</td>
<td>Adds support for Graphics g4dn instances (Windows Server 2012 R2); also includes Microsoft Windows updates up to February 11, 2020; AWS CLI version 1.17.5; and Amazon SSM Agent version 2.3.842.0</td>
<td>March 16, 2020</td>
</tr>
<tr>
<td>AppStream 2.0 client update: version 1.137</td>
<td>Reverts the updates in version 1.1.136</td>
<td>March 8, 2020</td>
</tr>
<tr>
<td>AppStream 2.0 client update: version 1.136</td>
<td>Adds support for defining trusted subdomains for user connections in a DNS TXT record, and provides other enhancements and fixes</td>
<td>March 5, 2020</td>
</tr>
<tr>
<td>AppStream 2.0 base image update</td>
<td>Adds support for Graphics g4dn instances (Windows Server 2016, Windows Server 2019); also includes Microsoft Windows updates up to February 11, 2020; AWS CLI version 1.17.5; and Amazon SSM Agent version 2.3.842.0</td>
<td>March 5, 2020</td>
</tr>
<tr>
<td>Documentation update: Support for native application mode</td>
<td>Created “Native application mode” and updated other content as needed.</td>
<td>February 28, 2020</td>
</tr>
<tr>
<td>AppStream 2.0 client update: version 1.129</td>
<td>Adds support for native application mode and provides other enhancements and fixes</td>
<td>February 28, 2020</td>
</tr>
<tr>
<td>Date</td>
<td>Release Description</td>
<td></td>
</tr>
<tr>
<td>--------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>February 19, 2020</td>
<td>Adds support for native application mode</td>
<td></td>
</tr>
<tr>
<td>January 13, 2020</td>
<td>Updates for Graphics Design: Adds support for Windows Server 2019, with Microsoft Windows updates up to November 12, 2019</td>
<td></td>
</tr>
<tr>
<td>January 13, 2020</td>
<td>Resolves multiple issues</td>
<td></td>
</tr>
<tr>
<td>December 23, 2019</td>
<td>Created &quot;Security in Amazon AppStream 2.0&quot; and updated other content as needed.</td>
<td></td>
</tr>
<tr>
<td>December 12, 2019</td>
<td>Resolves a DPI issue that causes the mouse cursor to point to the wrong location when a user clicks on an application during a streaming session</td>
<td></td>
</tr>
<tr>
<td>December 12, 2019</td>
<td>Includes Microsoft Windows updates up to December 12, 2019; AWS CLI version 1.16.284; and Amazon SSM Agent version 2.3.760.0</td>
<td></td>
</tr>
<tr>
<td>November 21, 2019</td>
<td>Updated &quot;AppStream 2.0 Instance Families.&quot;</td>
<td></td>
</tr>
<tr>
<td>November 13, 2019</td>
<td>AppStream 2.0 assemblies are now signed, including executables and installer packages</td>
<td></td>
</tr>
<tr>
<td>November 1, 2019</td>
<td>Created &quot;Embed AppStream 2.0 Streaming Sessions&quot; and updated other content as needed.</td>
<td></td>
</tr>
<tr>
<td>October 16, 2019</td>
<td>Adds support for up to 4 monitors and provides other enhancements</td>
<td></td>
</tr>
<tr>
<td>October 9, 2019</td>
<td>Created Amazon AppStream 2.0 in the AWS GovCloud (US) User Guide and updated the relay state endpoint table in &quot;Setting Up SAML&quot; and other content as needed.</td>
<td></td>
</tr>
<tr>
<td>October 8, 2019</td>
<td>Modifies the AppStream 2.0 storage connector to no longer bypass the system proxy server</td>
<td></td>
</tr>
<tr>
<td>Documentation update: FIPS-compliant endpoints</td>
<td>Created &quot;Protecting Data in Transit with FIPS Endpoints&quot; and updated other content as needed.</td>
<td>October 7, 2019</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>AppStream 2.0 client update: version 1.0.499</td>
<td>Resolves issues with client-side hardware rendering and with the client not working correctly when Bluetooth headsets are connected to the local computer.</td>
<td>September 26, 2019</td>
</tr>
<tr>
<td>AppStream 2.0 agent update</td>
<td>Resolves multiple issues</td>
<td>September 23, 2019</td>
</tr>
<tr>
<td>Documentation update: Support for applying IAM roles to AppStream 2.0 streaming instances</td>
<td>Created &quot;Using an IAM Role to Grant Permissions to Applications and Scripts Running on AppStream 2.0 Streaming Instances&quot; and updated other content as needed.</td>
<td>September 9, 2019</td>
</tr>
<tr>
<td>AppStream 2.0 base image update</td>
<td>Updates for Graphics Design instances: Includes Microsoft Windows updates up to August 13th, 2019; AWS CLI version 1.16.222; and AMD driver 24.20.13028.3002</td>
<td>September 5, 2019</td>
</tr>
<tr>
<td>AppStream 2.0 agent update</td>
<td>Adds support for applying IAM roles to AppStream 2.0 streaming instances</td>
<td>September 3, 2019</td>
</tr>
<tr>
<td>Documentation update: AppStream 2.0 file system redirection</td>
<td>Created &quot;Enable File System Redirection for Your AppStream 2.0 Users&quot; and updated other content as needed.</td>
<td>August 20, 2019</td>
</tr>
<tr>
<td>Documentation update: Interface VPC endpoints</td>
<td>Created &quot;Creating and Streaming From Interface VPC Endpoints.&quot; Also created &quot;Access AppStream 2.0 API Operations and CLI Commands Through an Interface VPC Endpoint&quot; and updated other content as needed.</td>
<td>August 19, 2019</td>
</tr>
<tr>
<td>AppStream 2.0 client update: version 1.0.480</td>
<td>Adds support for AppStream 2.0 file system redirection</td>
<td>August 14, 2019</td>
</tr>
<tr>
<td>Update Type</td>
<td>Details</td>
<td>Date</td>
</tr>
<tr>
<td>-------------</td>
<td>---------</td>
<td>------</td>
</tr>
<tr>
<td>AppStream 2.0 agent update</td>
<td>Adds support for AppStream 2.0 file system redirection</td>
<td>August 8, 2019</td>
</tr>
<tr>
<td>Documentation update: Programmatic AppStream 2.0 image creation</td>
<td>Created &quot;Create Your AppStream 2.0 Image Programmatically&quot; and updated other content as needed.</td>
<td>August 1, 2019</td>
</tr>
<tr>
<td>AppStream 2.0 agent update</td>
<td>Adds support for creating AppStream 2.0 images programmatically</td>
<td>July 26, 2019</td>
</tr>
<tr>
<td>AppStream 2.0 base image update</td>
<td>Adds support for Windows Server 2016 and Windows Server 2019</td>
<td>June 10, 2019</td>
</tr>
<tr>
<td>AppStream 2.0 base image update</td>
<td>Includes Microsoft Windows updates up to May 14, 2019</td>
<td>May 28, 2019</td>
</tr>
<tr>
<td>Documentation update: AppStream 2.0 Usage reports</td>
<td>Created &quot;AppStream 2.0 Usage Reports&quot; and updated other content as needed.</td>
<td>May 21, 2019</td>
</tr>
<tr>
<td>Documentation update: Support for disconnecting idle users</td>
<td>Updated &quot;Create a Fleet&quot; in &quot;Create an AppStream 2.0 Fleet and Stack.&quot;</td>
<td>May 17, 2019</td>
</tr>
<tr>
<td>AppStream 2.0 client update: version 1.0.407</td>
<td>Adds support for configuring the amount of time that users can be idle (inactive) before they are disconnected from their streaming session</td>
<td>May 16, 2019</td>
</tr>
<tr>
<td>AppStream 2.0 agent update</td>
<td>Adds support for configuring the amount of time that users can be idle (inactive) before they are disconnected from their streaming session. Also adds support for subscribing to AppStream 2.0 usage reports.</td>
<td>May 7, 2019</td>
</tr>
<tr>
<td>AppStream 2.0 base image update</td>
<td>Includes Microsoft Windows updates up to April 20, 2019; AWS CLI version 1.16.126; and NVIDIA Graphics Driver 412.16 for Graphics Pro instances</td>
<td>April 29, 2019</td>
</tr>
<tr>
<td>Documentation update: Logging AppStream 2.0 API calls with AWS CloudTrail</td>
<td>Created &quot;Logging AppStream 2.0 API Calls with AWS CloudTrail.&quot;</td>
<td>April 25, 2019</td>
</tr>
<tr>
<td>Documentation update: HIPAA compliance</td>
<td>Created &quot;AppStream 2.0 Compliance.&quot;</td>
<td>March 28, 2019</td>
</tr>
<tr>
<td>Documentation update: Gesture support</td>
<td>Created &quot;Touchscreen Devices&quot; and updated other content as needed.</td>
<td>March 13, 2019</td>
</tr>
<tr>
<td>AppStream 2.0 client update: version 1.0.375</td>
<td>Adds touch screen support on Windows PCs and support for: Automatically connecting USB devices when a new streaming session starts, running session scripts, and delivering virtualized applications using the AppStream 2.0 dynamic application framework APIs</td>
<td>March 7, 2019</td>
</tr>
<tr>
<td>AppStream 2.0 agent update</td>
<td>Adds support for gestures on touch-enabled iPads, Android tablets, and Windows devices</td>
<td>March 7, 2019</td>
</tr>
<tr>
<td>New Region: Seoul</td>
<td>Updated the relay state endpoint table in &quot;Setting Up SAML&quot; and other content as needed.</td>
<td>February 13, 2019</td>
</tr>
<tr>
<td>Documentation update: Session scripts</td>
<td>Created &quot;Use Session Scripts to Manage Your AppStream 2.0 Users' Streaming Experience&quot; in &quot;Images&quot; and updated other content as needed.</td>
<td>January 27, 2019</td>
</tr>
<tr>
<td>AppStream 2.0 base image update</td>
<td>Includes Microsoft Windows updates up to December 10, 2018; AWS CLI version 1.16.84; and NVIDIA Graphics Driver 391.58 for Graphics Pro instances</td>
<td>January 22, 2019</td>
</tr>
<tr>
<td>AppStream 2.0 agent update</td>
<td>Adds support for using on-instance session scripts. Also adds support for adding tags to the following AppStream 2.0 resource types during resource creation: image builders, images, fleets, and stacks.</td>
<td>January 22, 2019</td>
</tr>
<tr>
<td>AppStream 2.0 client update: version 1.0.320</td>
<td>Adds support for AppStream 2.0 dynamic application framework APIs, AppStream 2.0 regional settings, the AppStream 2.0 user pool, and provides other enhancements</td>
<td>January 19, 2019</td>
</tr>
<tr>
<td>Documentation update: Default regional settings</td>
<td>Created &quot;Configure Default Regional Settings for Your AppStream 2.0 Users&quot; and updated other content as needed.</td>
<td>December 13, 2018</td>
</tr>
<tr>
<td>Date</td>
<td>Version/Update Details</td>
<td></td>
</tr>
<tr>
<td>-------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>December 7, 2018</td>
<td>Documentation update: Dynamic application framework. Created &quot;Manage App Entitlement&quot; and updated other content as needed.</td>
<td></td>
</tr>
<tr>
<td>December 4, 2018</td>
<td>AppStream 2.0 agent update. Adds support for using the AppStream 2.0 dynamic application framework to build a dynamic app provider. Also adds support for using a Japanese keyboard with web clients that run Windows.</td>
<td></td>
</tr>
<tr>
<td>November 14, 2018</td>
<td>AppStream 2.0 agent update. Adds support for launching streaming sessions using the AppStream 2.0 Windows client.</td>
<td></td>
</tr>
<tr>
<td>September 14, 2018</td>
<td>Documentation update: Image sharing. Created &quot;Administer Your AppStream 2.0 Images&quot; and updated other content as needed.</td>
<td></td>
</tr>
<tr>
<td>September 5, 2018</td>
<td>Documentation update: Application settings persistence. Created &quot;Enable Application Settings Persistence for Your AppStream 2.0 Users&quot; and updated other content as needed.</td>
<td></td>
</tr>
<tr>
<td>August 29, 2018</td>
<td>AppStream 2.0 agent update. Adds support for application settings persistence.</td>
<td></td>
</tr>
<tr>
<td>July 31, 2018</td>
<td>Documentation update: OneDrive support. Created &quot;Enable and Administer OneDrive for Your AppStream 2.0 Users&quot; and updated other content as needed.</td>
<td></td>
</tr>
<tr>
<td>July 26, 2018</td>
<td>AppStream 2.0 agent update. Adds support for OneDrive persistent storage.</td>
<td></td>
</tr>
<tr>
<td>June 14, 2018</td>
<td>Documentation update: Regional settings. Created &quot;Enable Regional Settings for Your AppStream 2.0 Users&quot; and updated other content as needed.</td>
<td></td>
</tr>
<tr>
<td>June 14, 2018</td>
<td>Documentation update: Default application and Windows settings. Added &quot;Step 4: Create Default Application and Windows Settings&quot; to &quot;Tutorial: Create a Custom Image&quot; and updated other content as needed.</td>
<td></td>
</tr>
</tbody>
</table>

290
Earlier Updates

The following table describes important additions to the AppStream 2.0 service and the Amazon AppStream 2.0 Administration Guide documentation before June 4, 2018.

**Note**

Individual AppStream 2.0 base image and AppStream 2.0 agent releases are not included in this table. For information about these releases, see AppStream 2.0 Base Image Version History (p. 29) and AppStream 2.0 Agent Version History (p. 39).

<table>
<thead>
<tr>
<th>Change</th>
<th>Description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrative controls for data transfer</td>
<td>Updated the &quot;Create a Stack&quot; section in &quot;Create AppStream 2.0 Fleets and Stacks&quot; and updated other content as needed.</td>
<td>May 24, 2018</td>
</tr>
<tr>
<td>New region: Frankfurt</td>
<td>Updated the relay state endpoint table in &quot;Setting Up SAML&quot; and other content as needed.</td>
<td>March 28, 2018</td>
</tr>
<tr>
<td>Custom branding</td>
<td>Created &quot;Add Your Custom Branding to Amazon AppStream 2.0&quot; and updated other content as needed.</td>
<td>March 26, 2018</td>
</tr>
<tr>
<td>Image copy</td>
<td>Updated &quot;Tutorial: Create a Custom Image&quot; and updated other content as needed.</td>
<td>February 23, 2018</td>
</tr>
<tr>
<td>New regions: Singapore and Sidney</td>
<td>Updated the relay state endpoint table in &quot;Setting Up SAML&quot; and other content as needed.</td>
<td>January 24, 2018</td>
</tr>
<tr>
<td>Resource tagging</td>
<td>Created &quot;Tagging Your Amazon Amazon AppStream 2.0 Resources&quot; and updated other content as needed.</td>
<td>December 15, 2017</td>
</tr>
<tr>
<td>Managed AppStream 2.0 agent updates</td>
<td>Created &quot;Amazon AppStream 2.0 Agent Version History&quot; and updated other content as needed.</td>
<td>December 7, 2017</td>
</tr>
<tr>
<td>On-Demand fleets</td>
<td>Added &quot;Fleet Type&quot; section to “Amazon AppStream 2.0 Fleets and Stacks” and updated other content as needed.</td>
<td>September 19, 2017</td>
</tr>
<tr>
<td>Instance types</td>
<td>Created &quot;Amazon AppStream 2.0 Instance Families&quot; and updated other content as needed.</td>
<td>July 25, 2017</td>
</tr>
<tr>
<td>Change</td>
<td>Description</td>
<td>Date</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>Active Directory</td>
<td>Created &quot;Using Active Directory with Amazon AppStream 2.0&quot; and updated other content as needed.</td>
<td>July 24, 2017</td>
</tr>
<tr>
<td>User pool</td>
<td>Created &quot;Manage Access Using the AppStream 2.0 User Pool&quot; and updated other content as needed.</td>
<td>June 15, 2017</td>
</tr>
<tr>
<td>Security groups</td>
<td>Added &quot;Security Groups&quot; section to &quot;Network Settings for Amazon AppStream 2.0&quot; and updated other content as needed.</td>
<td>May 26, 2017</td>
</tr>
<tr>
<td>Home folders</td>
<td>Created &quot;Enable and Administer Home Folders for Your AppStream 2.0 Users&quot; and updated other content as needed.</td>
<td>May 18, 2017</td>
</tr>
<tr>
<td>Default internet access</td>
<td>Created &quot;Network Settings for Amazon AppStream 2.0&quot; and updated other content as needed.</td>
<td>April 21, 2017</td>
</tr>
<tr>
<td>Fleet automatic scaling</td>
<td>Created &quot;Fleet Auto Scaling for Amazon AppStream 2.0&quot; and updated other content as needed.</td>
<td>March 23, 2017</td>
</tr>
<tr>
<td>Fleet management</td>
<td>Created &quot;Amazon AppStream 2.0 Fleets and Stacks&quot; and updated other content as needed.</td>
<td>February 22, 2017</td>
</tr>
<tr>
<td>SAML 2.0 support</td>
<td>Created &quot;Single Sign-on Access to AppStream 2.0 Using SAML 2.0&quot; and updated other content as needed.</td>
<td>February 15, 2017</td>
</tr>
<tr>
<td>Image builders</td>
<td>Created &quot;AppStream 2.0 Image Builders&quot; and updated other content as needed.</td>
<td>January 19, 2017</td>
</tr>
<tr>
<td>Initial documentation release</td>
<td>Created the initial release of the Amazon AppStream 2.0 Administration Guide.</td>
<td>December 01, 2016</td>
</tr>
</tbody>
</table>