Amazon AppStream 2.0

Developer Guide

Amazon AppStream 2.0: Developer Guide

Table of Contents

What Is Amazon AppStream 2.0?

Amazon AppStream 2.0 is a fully managed, secure, application streaming service that allows you to stream desktop applications from AWS to any device running a web browser, without rewriting them. AppStream 2.0 provides users instant-on access to the applications they need, and a responsive, fluid user experience on the device of their choice.

With AppStream 2.0, you can easily add your existing desktop applications to AWS and instantly start streaming them to an HTML5 compatible browser. You can maintain a single version of each of your apps, 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 the AppStream 2.0 detail page, AppStream 2.0 Pricing or Amazon AppStream 2.0 FAQs.

The AppStream 2.0 API provides programmatic control over all streaming actions as an alternative to using the AWS Management Console. For more information, see Amazon AppStream 2.0 API Reference.

Topics

Features

Using Amazon AppStream 2.0 leverages the following advantages:

Run desktop applications on any device
With AppStream 2.0, your desktop applications can run securely in an HTML5 web browser on Windows and Linux PCs, Macs, and Chromebooks. You can add your applications without rewriting them, maintain a single version for all your users, and provide easy access to your users from anywhere.

Instant-on access
AppStream 2.0 provides users instant-on access to their desktop applications in a browser on the desktop device of their choice. There are no delays, no large files to download, and no time-consuming installations. Users get a responsive, fluid experience that is indistinguishable from natively installed apps.

Secure applications and data
With AppStream 2.0, applications and data remain on AWS — only encrypted pixels are streamed to end 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 VPC, and you can use Amazon VPC security features to control access. This allows you to isolate your applications and deliver them in a secure way.

Easily integrate with your IT environment
AppStream 2.0 can 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're running on AWS, reducing the movement of data between AWS and your location and providing a faster user experience. AppStream 2.0 supports 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.

Fully managed service
With AppStream 2.0, you don't need to plan, deploy, manage, or upgrade any application streaming infrastructure. AppStream 2.0 manages the AWS resources required to host and run your applications, scales automatically, and provides access to your end users on demand.

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 a new AWS region that is closest to your users, and start streaming with no incremental capital investment.

Key Concepts

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

stack
Set up an AppStream 2.0 stack to start streaming apps to user browsers. An AppStream 2.0 stack consists of a fleet of streaming instances, user access policies, and storage configurations.

fleet
The fleet in an AppStream 2.0 stack consists of streaming instances that can scale automatically based on demand. You can set the desired number of streaming instances in a fleet. The fleet runs the image that you specify.

image
An AppStream 2.0 image contains applications to be streamed to users accessing an AppStream 2.0 stack. The image is used to launch streaming instances that are part of an AppStream 2.0 fleet. Use an image builder to create or modify an image.

image builder
Install your apps and create an image by using an AppStream 2.0 image builder. You can launch and connect to an AppStream 2.0 image builder from the AWS Management Console. After you are connected to an image builder, you can install, add, and test your apps, and then use the image builder to publish an AppStream 2.0 image.

How to Get Started

If you are using AppStream 2.0 for the first time, follow the Getting Started with Amazon AppStream 2.0 (p. 8) tutorial in the console. When you go through the getting started experience for the first time, AppStream 2.0 creates an IAM role to create and manage AppStream 2.0 resources on your behalf.

To use the Try It Now feature

You can go directly to Try it now or follow these steps.

1. Open the AppStream 2.0 console at https://console.aws.amazon.com/appstream2.
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 interactive tutorial

1. Open the AppStream 2.0 console at https://console.aws.amazon.com/appstream2.
2. Choose **Get Started**.
3. Select the option to learn more about AppStream 2.0 resources.

Related Services

AppStream 2.0 is used in conjunction with the following AWS service:

AWS Identity and Access Management
IAM is a web service that helps you securely control access to AWS resources for your users. The AppStream 2.0 service uses IAM service roles to create and manage AppStream 2.0 resources on your behalf. You can use IAM to control who can use your AWS resources (authentication) and what resources they can use in which ways (authorization). For more information, see Using Identity Based Policies (p. 22).

Accessing AppStream 2.0

You can work with AppStream 2.0 in any of the following ways:

AWS Management Console
The console is a browser-based interface to manage AppStream 2.0 resources. For more information about using the console, see Getting Started with Amazon AppStream 2.0 (p. 8).

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. For more information, see the AWS Command Line Interface User Guide and AWS Tools for Windows PowerShell User Guide.

You can use the AWS command line tools to issue commands at your system's command line to perform AppStream 2.0 and AWS tasks. 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 about available SDKs, 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.

Topics

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

1. Open https://aws.amazon.com/, and then choose **Create an AWS Account**.
2. Follow the online instructions.

   Part of the sign-up procedure involves receiving a phone call and entering a PIN using the phone keypad.

Note your AWS account number, because you need it for the next task.

Create an IAM User

Services in AWS, such as AppStream 2.0, require that you provide credentials when you access them, so that the service can determine whether you have permission to access its resources. The console requires your password. You can create access keys for your AWS account to access the command line interface or API.

We recommend that you use IAM to access AWS instead of the credentials for your AWS account. Create an IAM user, and then add the user to an IAM group with administrative permissions or and grant this user administrative permissions. You can then access AWS using a special URL and the credentials for the IAM user.

If you signed up for AWS but have not created an IAM user for yourself, you can create one using the IAM console.

To create an IAM user for yourself and add the user to an Administrators group

1. Sign in to the Identity and Access Management (IAM) console at https://console.aws.amazon.com/iam/.
2. In the navigation pane, choose **Users**, and then choose **Add user**.
3. For **User name**, type a user name, such as `Administrator`. The name can consist of letters, digits, and the following characters: plus (+), equal (=), comma (,), period (.), at (@), underscore (_), and hyphen (-). The name is not case sensitive and can be a maximum of 64 characters in length.
4. Select the check box next to **AWS Management Console access**, select **Custom password**, and then type the new user's password in the text box. You can optionally select **Require password reset** to force the user to select a new password the next time the user signs in.
5. Choose **Next: Permissions**.
6. On the **Set permissions for user** page, choose **Add user to group**.
7. Choose **Create group**.
8. In the **Create group** dialog box, type the name for the new group. The name can consist of letters, digits, and the following characters: plus (+), equal (=), comma (,), period (.), at (@), underscore (_), and hyphen (-). The name is not case sensitive and can be a maximum of 128 characters in length.
9. For **Filter**, choose **Job function**.
10. In the policy list, select the check box for **AdministratorAccess**. Then choose **Create group**.
11. Back in the list of groups, select the check box for your new group. Choose **Refresh** if necessary to see the group in the list.
12. Choose **Next: Review** to see the list of group memberships to be added to the new user. When you are ready to proceed, choose **Create user**.

You can use this same process to create more groups and users, and to give your users access to your AWS account resources. To learn about using policies to restrict users' permissions to specific AWS resources, go to Access Management and Example Policies for Administering AWS Resources.

To sign in as this new IAM user, sign out of the AWS console, then use the following URL, where *your_aws_account_id* is your AWS account number without the hyphens (for example, if your AWS account number is `1234-5678-9012`, your AWS account ID is `123456789012`):

```
https://your_aws_account_id.signin.aws.amazon.com/console/
```

Enter the IAM user name and password that you just created. When you're signed in, the navigation bar displays "*your_user_name* @ *your_aws_account_id*".

If you don't want the URL for your sign-in page to contain your AWS account ID, you can create an account alias. From the IAM dashboard, choose **Create Account Alias** and enter an alias, such as your company name. To sign in after you create an account alias, use the following URL:

```
https://your_account_alias.signin.aws.amazon.com/console/
```

To verify the sign-in link for IAM users for your account, open the IAM console and check under **IAM users sign-in link** on the dashboard.

For more information about IAM, see the AWS Identity and Access Management User Guide.

(Optional) Install the AWS CLI

Note
This step is not required to use the first-run wizard and create your first stack.

The AWS Command Line Interface provides high-level commands to simplify creating, updating, and monitoring application streaming from a local development environment. For information about installing the AWS CLI or upgrading it to the latest version, see Installing the AWS Command Line Interface in the *AWS Command Line Interface User Guide*.

Getting Started with Amazon AppStream 2.0

To stream your applications, Amazon AppStream 2.0 requires an environment consisting of a stack and at least one application image. This topic walks through the steps needed to understand how to put together a sample AppStream 2.0 environment for application streaming and give users access to that stream.

Topics

Step 1: Set Up a Sample Stack

Before you can stream your applications, you need to create a stack. In this step, you create a new stack from the sample stack template to simplify the creation.

To set up the AppStream 2.0 sample stack

1. Open the AppStream 2.0 console at https://console.aws.amazon.com/appstream2.
2. Choose **Get Started** if you are visiting the service console for the first time, or **Quick Links** from the left navigation menu of the service console.
3. Select the option to create a new stack using sample apps from the list, choose **Next** to use the default sample stack name and details, or enter your own details and then choose **Next**.
4. Review the sample apps image details and choose **Next**. The sample apps image contains a few pre-installed open source applications for evaluation purposes.
5. Provide the details for your fleet. Note that most values are pre-populated for you.

   Instance Type — Choose an instance type that matches the performance requirements of your applications. All streaming instances in your fleet launch with the instance type that you select.

   Network Access — Provide two subnets that have access to the network resources with which your applications need to interact. If you don't have any subnets, create them using the help link provided and then refresh the subnets list. You can choose existing network settings or create

new settings for this fleet. For more information, see Network Settings for Fleet and Image Builder Instances (p. 14).

Disconnect Timeout — Select the time that a streaming instance should remain 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 the previous session. If users try to connect after this timeout interval, a session launches with a new instance.

Desired Capacity — Choose an initial capacity for your fleet. Desired capacity is defined in terms of number of instances to be launched in this fleet. Every unique user session is served by an instance. To have your stack support 100 concurrent users, enter the initial capacity as 100. If you are unsure about capacity, accept the default values and choose **Review**.

6. Review the details for the sample stack, choose **Edit** for any section to change, and choose **Create**.

After the service sets up some resources, the **Stacks** dashboard appears. Your new stack is listed as **Active** when it is available to work with from the console, but cannot be used for streaming sessions until the stack fleet is in **Running** status.

Step 2: Provide Access to Users

After you create a stack, each user needs an active URL for access. This step automatically creates a streaming URL that you can share with a user for access to apps.

To provide access to users

1. In the navigation pane, choose **Stacks**, select the stack to use, check that the fleet status is Running, and then choose **Actions**, **Create Streaming URL**.
2. For **UserID**, type the user ID and select an expiration time. This time determines how long the generated URL is valid.
3. To view the user ID and URL, choose **Get URL**.
4. To copy the link to the clipboard, choose **Copy Link**.
5. Choose **Exit**.

Next Steps

You can now create URLs and send them to users. At this point, you can monitor your stack and make decisions about managing it. For more information, see the following topics.

- Learn how to use the AppStream 2.0 image builder to add your own apps and create new images you can stream. For more information, see Tutorial: Using an AppStream 2.0 Image Builder (p. 10).
- Manage your AppStream 2.0 resources to optimize your streaming performance and cost structure. For more information, see Managing Amazon AppStream 2.0 Resources (p. 18).
- Control who has access to your AppStream 2.0 streaming instances. For more information, see Controlling Access to Amazon AppStream 2.0 (p. 21).
- Troubleshoot your AppStream 2.0 streaming experience. For more information, see Troubleshooting (p. 25).

Tutorial: Using an AppStream 2.0 Image Builder

Before you can stream your applications, AppStream 2.0 requires at least one image that you create using an image builder. This tutorial walks through the steps to create images using an image builder.

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

Topics

Step 1: Create an Image Builder

Create a new image builder so you can add apps and create images for streaming.

To create an image builder for adding applications

1. Open the AppStream 2.0 console at https://console.aws.amazon.com/appstream2.
2. You may see the welcome screen showing two choices: **Try it now** and **Get started**. Choose **Get started**, **Custom set up**. If you do not see the welcome screen, choose **Quick links** in the left navigation pane, then **Custom set up**.
3. Select a base image. If you are launching the image builder for the first time, you can use the sample base image provided by AWS. If you have created images before or to update applications in an existing image, you can choose one of your existing images. Choose **Next**.

4. Configure the image builder by accepting the default values or by providing your own inputs for the following fields:

   Name
   Provide a unique name identifier for the image builder.

   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.

   VPC and Subnet
   Choose a VPC subnet in which your image builder should be launched. Your image builder will have access to any of the network resources that are accessible from within this VPC subnet. For Internet access on the image builder, provide a subnet that is enabled for Internet traffic. For more information, see Network Settings for Fleet and Image Builder Instances (p. 14).

   After you have configured your image builder, choose **Review**.
5. Review the details for the image builder, choose **Edit** for any section to change, and choose **Launch**.

After the service sets up some resources, the image builder instance list appears. Your new image builder is listed as **Running** when it is ready to use (choose **Refresh** to update the status).

Step 2: Installing Applications to an Image

In this step, connect to the image builder that you created and launched, then install the applications to be included in the image.

To install applications

1. On the left navigation pane, choose **Images**, **Image Builder**.
2. Select the image builder to use, check to be sure it has a Running status, and choose **Actions**, **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. When you connect to your image builder, the service requests that you press "Ctrl + Alt + Delete" for first time sign-in. On the top right corner of the image builder session toolbar, choose **Admin Commands**, **Send Ctrl + Alt + Delete**.
4. Sign in by selecting one of thee following options:

   ImageBuilderAdmin
   This mode has full administrator privileges on the image builder instance. Use this mode to install your applications, add applications to the image, and create an image.

   ImageBuilderTest
   This mode has the same limited privileges as your end users have on their streaming instances. Use this mode to test applications for proper function as an end user.

   At any point after logging in, you can switch between admin and test modes by selecting the appropriate option from the **Admin Commands** menu.
5. If the image builder session requests you to enter a password, choose **Admin Commands**, **Log me in**.
6. Install apps by browsing to an application website or other download source and beginning the installation process as you normally would on a local physical computer. Complete the application's own installation process before moving to the next step.

Step 3: Adding Applications to an Image

In this step, you can add applications (*.exe*), batch scripts (*.bat*), and application shortcuts (*.lnk*) to the image.

To add applications

1. From the image builder desktop, launch the application named **Image Assistant**.
2. Choose **Add Application** and navigate to the location of the application, script, or shortcut to add. Choose **Open**.
3. In the **Application Properties** dialog box, enter a display name to be shown to the users in the catalog, change the icon, and enter launch parameters (additional arguments passed to the application when it is launched). Repeat for each application to add to the image.
4. When you are finished adding apps, choose **Next**.

To test your applications

a. On the web toolbar in the top right of your session, choose **Admin Commands**, **Switch to Image Builder Test**, **ImageBuilderTest**.
b. If a password is requested, choose **Admin Commands**, **Log me in**.
c. Use the **Image Assistant** to launch and test your apps.
d. To return to the admin mode, choose **Admin Commands**, **Switch to Image Builder Admin**.

- Verify that the apps you've added launch correctly by starting a Windows session as a test user.

Note
Do not exit the **Image Assistant** application, you need to use it in the next section.

Step 4: Optimizing Apps

In this step, you optimize your apps and create the image. The image builder optimizes your applications for start-up performance. This is a mandatory step that is performed on all applications in the list. All applications must be launched prior to optimization

To optimize your applications

1. .Choose **Launch** and the service automatically launches the first application in your list. When the app is completely started, choose **Continue**.
2. Provide any interactions or inputs that may be required by the application launched to bring it to a usable state. For example, a web browser may prompt you to import settings before it is completely up and running.
3. After you have brought the application to a usable state, choose **Continue** in the small dialog box. The app helper launches the next application automatically.
4. Repeat the previous step until all applications are launched, and leave them running. In the **Image Assistant** helper app, choose **Next**.

Step 5: Creating an Image

In this step, you choose a name and create the image.

To create the image

1. Enter a unique image name and image display name, with an optional description, and choose **Next**. The name you choose cannot begin with "Amazon", "AWS", or "AppStream".
2. Review the details and choose **Disconnect and Create Image**. Your new image is created and the session is disconnected. You can now close the session window.
3. Return to the console and navigate to **Images**, **Image Registry**. Verify your new image appears in the list.

The new image first appears with status **Pending** in the image registry of your console. After the image is successfully created, the status of the image changes to **Available**, which means you can now use the image to launch a stack and stream your applications. To continue to work with creating images, you can start the image builder and connect to it from the console, or create a new image builder. There is a limit of five image builders per account.

Step 7: 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**.

Next Steps

You can now create URLs and send them to users. At this point, you can monitor your stack and make decisions about managing it. For more information, see the following topics.

- Manage your AppStream 2.0 resources to optimize your streaming performance and cost structure. For more information, see Managing Amazon AppStream 2.0 Resources (p. 18).
- Control who has access to your AppStream 2.0 streaming instances. For more information, see Controlling Access to Amazon AppStream 2.0 (p. 21).
- Troubleshoot your AppStream 2.0 streaming experience. For more information, see Troubleshooting (p. 25).

Network Settings for Fleet and Image Builder Instances

The following sections contain information about configuring your AppStream 2.0 fleets and image builders to access network resources and the Internet.

When creating an AppStream 2.0 stack or image builder, you can provide Amazon VPC subnets. AppStream 2.0 sets up elastic network interfaces to the subnets provided so that AppStream 2.0 instances have access to your network resources through your VPC. For more information about Amazon VPC, see the Amazon VPC User Guide.

Network Setup Guidelines

There are some network setup guidelines to consider for fleets and image builders.

Image Builders

You can choose one subnet while launching an image builder. Ensure that the network resources, with which your applications may interact, are accessible through the chosen subnet. The typical resources required for successful execution of your apps may include licensing servers, database servers, file servers, and so on.

Fleets

You can provide subnets to establish network connections from your fleet instances to your VPC. We recommend that you specify two private subnets from different Availability Zones for high availability and fault tolerance. Also ensure that the network resources for your applications are accessible through both of the specified private subnets.

AppStream 2.0 creates as many elastic network interfaces as the maximum desired capacity of your fleet. The following guidelines will help you set up a VPC to support scaling behavior for your fleet.

- Make sure that your AWS account has sufficient elastic network interface (ENI) capacity to support the scale requirements of your fleet. If you are planning to launch a large fleet of streaming instances, contact AWS Support and request a higher ENI limit to match the maximum number of instances that you plan to launch.
- Specify subnets with a sufficient number of IP addresses to match the maximum desired capacity of your fleet.

Internet Connectivity

If your fleet instances and image builders require Internet access, use a VPC and a VPC NAT gateway, and launch these resources in private VPC subnets that provide a route to the Internet. You can use the instructions below to quickly create a network setup for enabling Internet access.

Topics

Image Builders

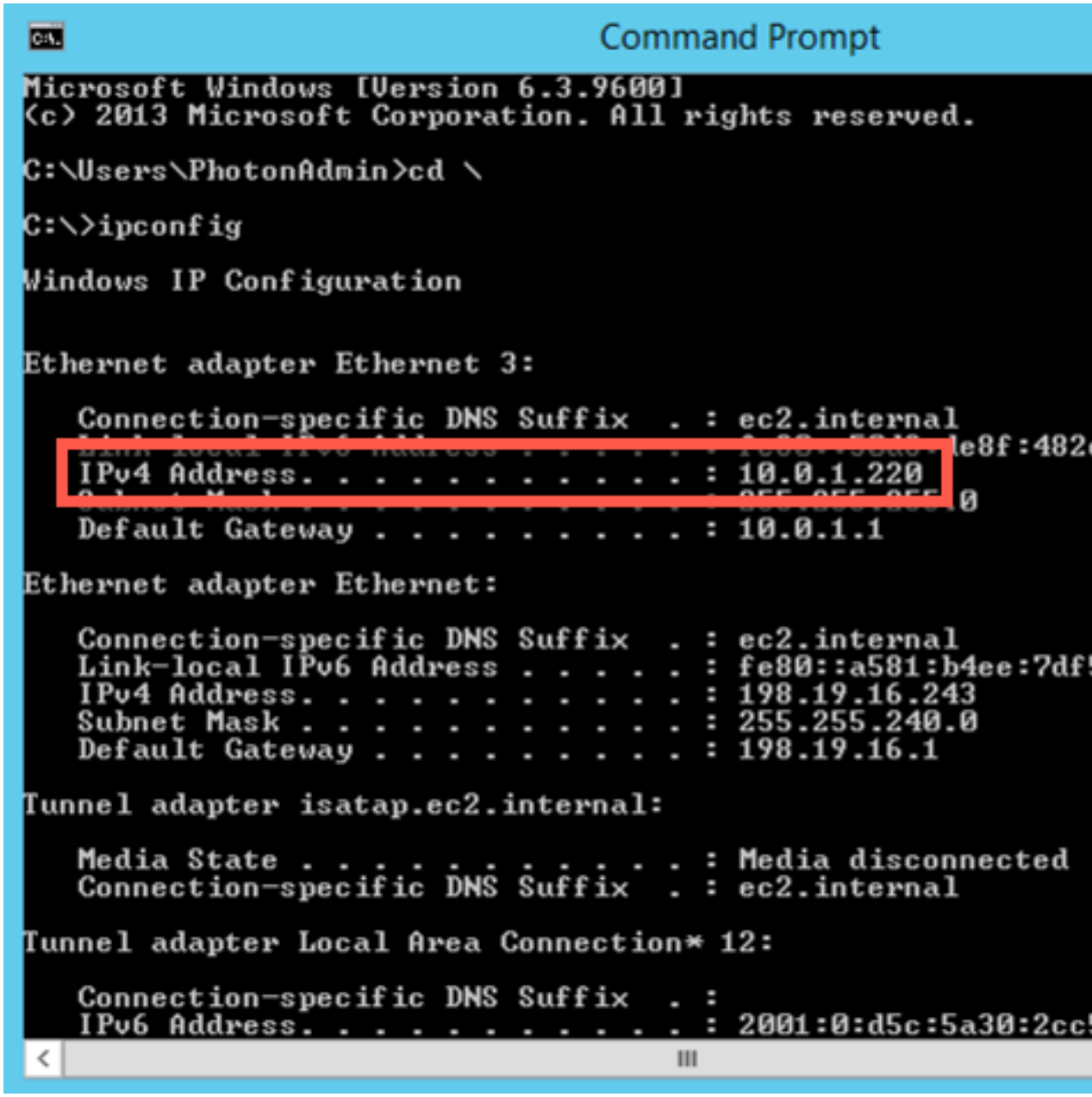
Choose one of the following connection methods: a VPC NAT gateway or a manual Elastic IP address (EIP). We recommend using a VPC NAT gateway because it is a one-time set up procedure that you can then use for launching all subsequent image builders. However, nominal charges apply to using a VPC NAT gateway. If you'd prefer a connection method that's free of charge, use a manual Elastic IP address to connect. This manual EIP procedure must be repeated for every new image builder you launch.

To enable Internet access using a VPC NAT gateway

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the dashboard, choose **Start VPC Wizard**, **VPC with Public and Private Subnets**, **Select**.
3. Provide suitable names for **VPC name**, **Public subnet name**, and **Private subnet name**.
4. For **Elastic IP Allocation ID**, choose an existing Elastic IP address. If you don’t have one, create an Elastic IP address from the **Elastic IPs** section on the Amazon VPC console.
5. Leave the other fields as their default values and choose **Create VPC**.
6. Open the AppStream 2.0 console at https://console.aws.amazon.com/appstream2.
7. Launch an image builder and choose the VPC and subnet names created earlier. To see the complete subnet name, hover over the list choices.
8. Connect to your image builder, launch a browser, and test the Internet connection.

To enable Internet access using manual EIP on a public subnet

1. Open the AppStream 2.0 console at https://console.aws.amazon.com/appstream2.
2. Choose **Images**, **Image Builders**.
3. Select your running image builder and choose **Actions**, **Connect**.
4. Open a Windows command prompt and type the command **ipconfig**.
5. In the output of the ipconfig command, take note of the IPv4 address of the Ethernet adapter that belongs to your VPC subnet. This is the only IPv4 address listed for any adapters shown.



6. Open the Amazon EC2 console and choose **Network & Security**, **Network Interfaces**.
7. Filter the network interfaces for the private IP address that you noted earlier. If you have more than one network interface with the same private IP address, add a filter for the VPC in which the image builder was launched.
8. Choose **Actions**, **Associate Address**, and enter an Elastic IP address. For more information, see Associating an EIP to a network interface.
9. Return to your image builder session and verify that you can connect to the Internet.

Fleets

Use the following procedure to set up Internet access for fleets.

To enable Internet access using a NAT gateway

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the dashboard, choose **Start VPC Wizard**, **VPC with Public and Private Subnets**, **Select**.
3. Provide suitable names for **VPC name**, **Public subnet name**, and **Private subnet name**.
4. For the **Availability Zone** fields, you can leave the public subnet zone as the default, and select a specific zone for the private subnet. Make a note of the zones you chose.
5. For **Elastic IP Allocation ID**, choose an existing Elastic IP address . If you don't have one, create an Elastic IP address from the **Elastic IPs** section on the Amazon VPC console.
6. Leave the other fields as their default values, making a note of the value in the **Private subnet's IPv4 CIDR** field, and then choose **Create VPC**. This may take some time to complete.
7. In the left navigation pane, choose **Subnets**, **Create Subnet**. Be sure to choose a different name than the one specified in step 3.
8. For **VPC**, enter the VPC that you created in step 6 , and for **Availability Zone**, enter a different value than the one you specified in step 4.
9. For **IPv4 CIDR block**, provide a unique for the new subnet. For example, if you noted the first subnet has a IPv4 CIDR block range of `10.0.1.0/24`, the new subnet could have a valid CIDR block range of `10.0.2.0/24`.
10. Choose **Yes, Create**.
11. Use the VPC and subnets created here when creating a new stack. You can test your Internet connectivity for this VPC using the sample stack creation steps provided in Getting Started with Amazon AppStream 2.0 (p. 8), specifying this VPC and subnets, and then connecting to your streaming instance and browsing to the Internet.

Managing Amazon AppStream 2.0 Resources

You can use the Amazon AppStream 2.0 console or API to manage stacks, fleets and images.

Topics

Image Builders

AppStream 2.0 provides virtual machines, or instances, that are used to install and add applications into and create your image. These instances are called *image builders*. You can launch an image builder from a base image provided by AWS, or from an image that you create. After your image builder instance is available (running), you can connect to the image builder to start a desktop session, install your applications, add your applications to an image, and create an image.

While launching a new image builder, you can choose from different instance types with various compute, memory, and graphics configurations. You also provide a VPC subnet so that AppStream 2.0 can establish a network interface to the image builder. This connection provides your image builder with access to resources that might be needed while you install and add applications; for example, file servers, licensing servers, database servers, and so on. For more information, see Tutorial: Using an AppStream 2.0 Image Builder (p. 10).

The following actions can be performed 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**
- **Snapshotting**
- **Stopping**
- **Starting**
- **Deleting**

Images

An Amazon AppStream 2.0 image contains applications that can be streamed to users. The image is used to launch streaming instances that are part of an AppStream 2.0 fleet. All images available to you are listed under the **Image Registry** section in the AWS Management Console. There are two types of images listed in your image registry that are differentiated by these visibility attributes:

- **Public Images** — Base images that are made available by AWS to help you create images with your own applications.
- **Private Images** — Images that are created and owned by you.

You can use either public or private images to launch an image builder and set up your AppStream 2.0 fleet. For more information, see Tutorial: Using an AppStream 2.0 Image Builder (p. 10).

You can also delete your private images. Note that a private image cannot be deleted if there are active fleets using it. You must stop all associated fleets before deleting the image.

Stacks and Fleets

An Amazon AppStream 2.0 stack consists of a fleet of streaming instances, user access policies, and storage configurations. For more information about creating a new stack and inviting users to stream your applications, see Getting Started with Amazon AppStream 2.0 (p. 8).

The fleet in an AppStream 2.0 stack consists of streaming instances that can scale automatically based on demand. You can set the desired capacity for the fleet.

The following actions can be performed on a fleet that is part of your stack.

Edit
Edit the fleet parameters. The parameters available for editing depends on the current status of the fleet.

Fleet Status	Editable Fields
RUNNING	Display Name, Desired Capacity

Fleet Status	Editable Fields
STOPPED	Display Name, Desired Capacity, Subnets, Image, Disconnect Timeout, Instance Type
STARTING	None
STOPPING	None

Stop
Stop all instances in the fleet.

Start
Start the desired number of instances in the fleet.

Delete
Delete all resources in a fleet. This action cannot be performed on a running fleet.

A stack that you no longer use can be deleted, but only if all fleets associated with it are deleted first. Deleting a stack is a three-step process. First, stop the running fleets associated with the stack, delete all the fleets associated with the stack, and then delete the stack.

Controlling Access to Amazon AppStream 2.0

By default, IAM users don't have permission to create or modify AppStream 2.0 resources, or perform tasks using the AppStream 2.0 API. To allow IAM users to create or modify resources and perform tasks, you must create IAM policies that grant permissions on specific resources and API actions, and then attach those policies to the IAM users or groups that require those permissions.

While creating AppStream 2.0 resources, AppStream 2.0 makes API calls to other AWS services on behalf of the user. This authentication is accomplished by the service assuming specific IAM roles available in the user's account. These IAM roles are created by the service when the user gets started with the service in an AWS region.

Topics

Using the IAM Service Role

The following service role is available for AppStream 2.0.

AmazonAppStreamServiceAccess
Allows AppStream 2.0 to create and manage AppStream 2.0 resources on the user's behalf. This role provides the permissions listed.

```
{
"Version": "2012-10-17",
"Statement": [
   {
      "Effect": "Allow",
      "Action": [
         "ec2:DescribeVpcs",
         "ec2:DescribeSubnets",
         "ec2:DescribeAvailabilityZones",
         "ec2:CreateNetworkInterface",
         "ec2:DescribeNetworkInterfaces",
         "ec2:DeleteNetworkInterface",
```

```
            "ec2:AssociateAddress",
            "ec2:DisassociateAddress",
            "ec2:DescribeRouteTables"
        ],
        "Resource": "*"
     }
  ]
}

Trust Relationship
{
"Version": "2012-10-17",
"Statement": [
   {
      "Effect": "Allow",
         "Principal": {
            "Service": "appstream.amazonaws.com"
         },
         "Action": "sts:AssumeRole"
   }
 ]
}
```

Using Identity Based Policies

AppStream 2.0 provides managed policies that you can attach to your IAM users and allow users to control your AppStream 2.0 resources or perform API actions through the AWS CLI, SDK, or console.

AmazonAppStreamFullAccess
Provides full administrator access to AppStream 2.0.

```
{
"Version": "2012-10-17",
"Statement": [
   {
      "Effect": "Allow",
         "Action": [
            "appstream:*"
         ],
         "Resource": "*"
   }
 ]
}
```

AmazonAppStreamReadOnlyAccess
Provides your IAM users with read-only access to AppStream 2.0 resources.

```
{
"Version": "2012-10-17",
"Statement": [
   {
      "Effect": "Allow",
         "Action": [
            "appstream:Describe*",
            "appstream:List*",
```

```
            "appstream:Get*"
         ],
         "Resource": "*"
   }
 ]
}
```

Amazon AppStream 2.0 Service Limits

By default, AWS limits the resources you can create and the number of users that can use the service. The following table lists the limits for each AppStream 2.0 resource.

Default Limits Per Region

Resource	Default Limit
Stacks	5 per account
Fleets	5 per account
Streaming instances	5 per account
Images	5 per account
Image builders	5 per account
Users	5 per account

For more information about limits, including how to increase limits for your account, see AWS Service Limits in *AWS General Reference*.

Troubleshooting

The following are possible problems you may have while trying to use Amazon AppStream 2.0.

Topics

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 Network Settings for Fleet and Image Builder Instances (p. 14).

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 can be added to an AppStream 2.0 image. Check if your application is supported on Microsoft Windows Server 2012 R2.

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.

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 the 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 need to 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 wallpaper, 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 need to 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 anti-virus application.

You can install any tools, including anti-virus 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.

Document History for Amazon AppStream 2.0

The following table describes the documentation for this release of Amazon AppStream 2.0.

- **API version:** 2016-12-01
- **Latest documentation update:** January 19, 2017

Change	Description	Date
Image builder support	Created image builder tutorial and other updates to support the launch of this feature.	January 19, 2017
Initial release	First publication of this guide.	December 01, 2016

AWS Glossary

For the latest AWS terminology, see the AWS Glossary in the *AWS General Reference*.