

Amazon AWS

- [Amazon AWS](#)
- [Access a Windows EC2 Machine Through Windows Firewall](#)
- [Amazon EC2 Instance has no public DNS assigned to it](#)
- [Attaching an Amazon EBS Volume to an Windows Instance](#)
- [Best Practices: Running Appvance IQ testing using EC2 cloud file](#)
- [Creating EC2 instances](#)
- [Use AWS Reserved Instances for AIQ Infrastructure](#)
- [Supported AWS Instance Types for AIQ](#)
- [Changing your IP in the security group with aws cli.](#)
- [AWS WorkSpace and Google Chrome](#)
- [Create an AWS IAM Access Key for AIQ Integrations](#)
- [Set Up IAM Role Permissions for an AWS EC2 AIQ Controller](#)
- [Amazon S3 Cross-Account Repository](#)

Amazon AWS

Contents:

- [Access windows EC2 machine which has windows firewall on from Local machine](#)
- [Amazon EC2 Instance has no public DNS assigned to it](#)
- [Attaching an Amazon EBS Volume to an Windows Instance](#)
- [Best Practices: Running Appvance IQ testing using EC2 cloud file](#)
- [Creating EC2 instances](#)
- [Reserving and using an AWS Reserved Instance](#)
- [Supported Instance Types](#)
- [Changing your IP in the security group with aws cli.](#)
- [AWS WorkSpace and Google Chrome](#)
- [How to Create an Access Key to your IAM User](#)
- [Setup IAM Role Permissions for the AWS EC2 Controller](#)

Access a Windows EC2 Machine Through Windows Firewall

Last reviewed: September 25, 2026

Use this article when a local machine needs to reach a Windows EC2 instance that has Windows Firewall enabled.

What changed in this article

- Corrected the title grammar and made the scope specific to Windows EC2 connectivity.
- Added checks for AWS security groups, Windows Firewall inbound rules, RDP/WinRM ports, and source IP restrictions.
- Added caution to avoid opening broad public access just to make a troubleshooting step pass.

Validation workflow

1. Confirm the EC2 security group allows the required port from the local source IP.
2. Confirm Windows Firewall allows the same service inside the instance.
3. Test connectivity with the approved access method and then narrow any temporary rule changes.

Review note: Kept the article as operational guidance but added security guardrails around firewall changes.

Last updated: September 25, 2026

If you have feedback on this topic, contact documentation@appvance.ai.

Amazon EC2 Instance has no public DNS assigned to it

Please follow the below steps to add dns hostnames to your EC2 Instances.

- Add Column VPC Id and check the VPC Id that your instance is assigned to.
- Go to console.aws.amazon.com
- Go To **Services -> VPC**
- Open **Your VPCs**
- select **your VPC** connected to your EC2 and
- Edit **Summary---** > **Change DNS hostnames: to YES**

Hope this helps.

Last updated: June 02, 2023

If you have any feedback on this topic, please contact us at documentation@appvance.ai

Attaching an Amazon EBS Volume to an Windows Instance

Attaching an Amazon EBS Volume to an Windows Instance.

You can attach an EBS volume to one of your instances that is in the same Availability Zone as the volume.

Prerequisites

- Determine the device names that you'll use.
- Determine how many volumes you can attach to your instance.
- If a volume is encrypted, it can only be attached to an instance that supports Amazon EBS encryption.
- If a volume has an AWS Marketplace product code:
 - The volume can only be attached to a stopped instance.
 - You must be subscribed to the AWS Marketplace code that is on the volume.
 - The configuration (instance type, operating system) of the instance must support that specific AWS Marketplace code. For example, you cannot take a volume from a Windows instance and attach it to a Linux instance.
 - AWS Marketplace product codes are copied from the volume to the instance.

To attach an EBS volume to an instance using the console

1. Open the Amazon EC2 console.
2. Click Volumes in the navigation pane.
3. Select a volume and then click Attach Volume.
4. In the Attach Volume dialog box, start typing the name or ID of the instance to attach the volume to in the Instance box, and select it from the list of suggestion options (only instances that are in the same Availability Zone as the volume are displayed).
5. You can keep the suggested device name, or enter a different supported device name.

Important The block device driver for the instance assigns the actual volume name when mounting the volume, and the name assigned can be different from the name that Amazon EC2 recommends.

6. Click Attach.
7. Connect to your instance and make the volume available.

Making an Amazon EBS Volume Available for Use

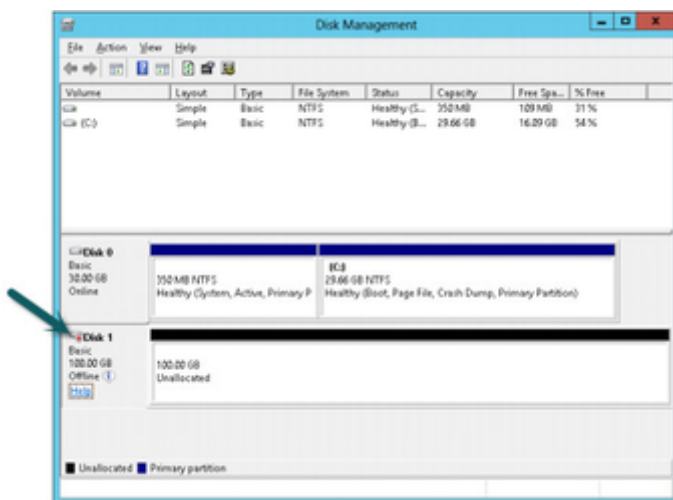
After you attach an Amazon EBS volume to your instance, it is exposed as a block device. You can format the volume with any file system and then mount it. After you make the EBS volume available for use, you can access it in the same ways that you access any other volume. Any data written to this file system is written to the EBS volume and is transparent to applications using the device.

Note that you can take snapshots of your EBS volume for backup purposes or to use as a baseline when you create another volume.

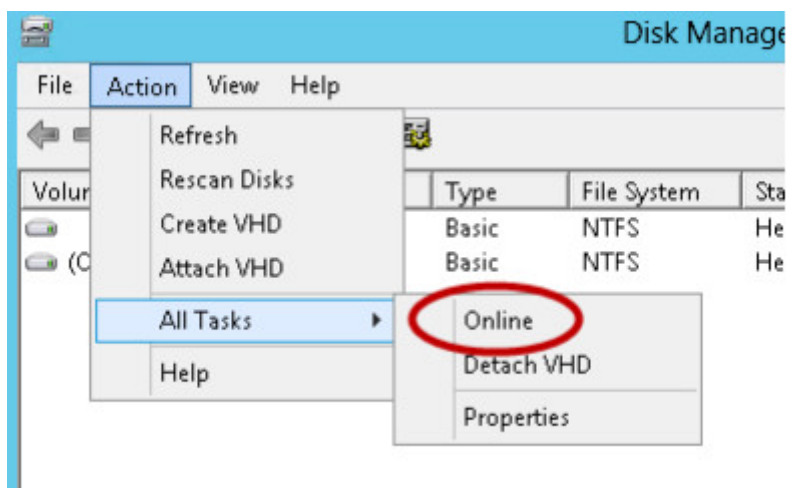
Making the Volume Available on Windows

Use the following procedure to make the volume available.

1. Log in to your Windows instance using Remote Desktop.
2. Start the Disk Management utility. On Windows Server 2012, on the taskbar, right-click the Windows logo, and then select Disk Management. On Windows Server 2003/2008, click Start, point to Administrative Tools, select Computer Management, and then select Disk Management.
3. Select the disk that represents the new EBS volume.



4. On the Disk Management menu, select Action - All Tasks - Online.



5. A new disk needs to be initialized before it can be used.

Caution:

If you're mounting a volume that already has data on it (for example, a public data set), make sure that you don't reformat the volume and delete the existing data.

To initialize a new disk:

- a. In the Disk Management utility, select the new EBS volume disk.
- b. On the Disk Management menu, select Action - All Tasks - Initialize Disk. Attaching Volume to the instance Page 4
- c. In the Initialize Disk dialog, select the disk to initialize, select the desired partition style, and press OK.

Restart your Amazon Instance.

Last updated: June 26, 2023

If you have any feedback on this topic, please contact us at documentation@appvance.ai

Best Practices: Running Appvance IQ testing using EC2 cloud file

The Best Practices that needs to be followed are below:

- Please make sure that Appvance IQ installed on the machine is using the License file that you have been provided with so that the new machines does not run into license issues while executing tests.
- Add Appvance IQ under Start up services so that once the machine starts, IQ is started automatically.
- Make sure the browsers are installed and are configured to have the correct paths.
- Once all the above changes are done create a new image and use this image to create machines while running tests.
- Do not run the test on Local machine instead EC2 machines to run these kind of tests which involves cloud file.
- Always click "Test Configuration" under cloud management to verify if the credentials entered are valid
- The Database URL should be changed from localhost URL to the name of the EC2 machine or use an RDS instance in Scenario Builder.
- Make sure to set "Terminate" as All and "Reboot" as Never under Locations in Scenario Builder or use these options depending on the needs.

Last updated: June 02, 2023

If you have any feedback on this topic, please contact us at documentation@appvance.ai

Creating EC2 instances

via Scenario Builder and running the tests, the first run's response times are on a higher side compared to consecutive runs by using the instances that are already running

Creating EC2 instances via Scenario Builder and running the tests, the first run's (where Appvance IQ creates requested instances from the Scenario builder using an AMI) response times are on a higher side compared to consecutive runs by using the instances that are already running.

AWS has a lazy start up, only required part of the hard disk is copied from the AMI to the run time hard disk.

Because of this, the machine starts up very fast, but accessing some parts of the hard disk could be very slow.

This affects Appvance IQ during the first scenario execution. The response times reported could be on the higher side.

Amazon provides a workaround which is reading the hard disk, in order to avoid this slowness during production.

Nevertheless the proposed solution is for Linux platform only.

Amazon documentation is located here.

<http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-initialize.html>

Appvance IQ now has the capability to read the hard disk the first time the machine is created (This is for windows platforms)

This will be done just the first time the AMI is created.

Note: Still there could be little difference in the response times when the test is started for the first time to create instance and run the scenario than compared to the subsequent scenario runs using the same started instance.

Last updated: June 02, 2023

If you have any feedback on this topic, please contact us at documentation@appvance.ai

Use AWS Reserved Instances for AIQ Infrastructure

Recommended guidance

AWS Reserved Instances can reduce infrastructure cost for stable AIQ workloads, but they should be planned with finance, cloud operations, and environment owners. Do not reserve capacity without confirming usage, region, instance family, term, and ownership.

Before reserving

- Confirm whether the AIQ workload is steady enough to justify a reservation.
- Review region, instance type, operating system, term, payment option, and expected utilization.
- Coordinate with cloud governance and budgeting owners.
- Compare Reserved Instances with Savings Plans or current cloud purchasing standards.

Operational guidance

1. Track which AIQ environments are intended to consume the reservation.
2. Review utilization regularly to avoid unused commitment.
3. Document renewal dates and ownership.
4. Re-evaluate reservations before AIQ upgrades, scaling changes, or cloud architecture changes.

Last updated: October 6, 2026

Supported AWS Instance Types for AIQ

Recommended guidance

Choose AWS instance types for AIQ based on the role of the host, expected workload, browser or API execution volume, storage needs, and monitoring results. Instance-type guidance can become outdated as AWS offerings and AIQ requirements change.

Selection factors

- AIQ application server, database, Test Node, reporting, or utility role.
- CPU, memory, disk throughput, network throughput, and browser concurrency.
- Operating system and dependency compatibility.
- Cloud region availability, cost model, and scaling policy.
- Performance test load profile and evidence retention needs.

Validation workflow

1. Start from the current AIQ system requirements and cloud architecture guidance.
2. Run representative smoke and load tests on the chosen instance family.
3. Monitor CPU, memory, disk, network, and browser stability.
4. Resize based on evidence instead of copying old environment sizes blindly.

Last updated: October 6, 2026

Changing your IP in the security group with aws cli.

First of all take install aws cli from this link:

<https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html>

Secondly, open the terminal and setup the default profile:

```
aws configure
```

insert AWS access Key ID:

insert AWS Secret Access Key:

Default region name:

Default output format: type <json>

After having to make the profile in config and credentials.

Instructions are here:

<https://docs.aws.amazon.com/cli/latest/userguide/cli-configure-profiles.html>

Download the script from the repository:

<https://gitlab.appvance.net/Appvance/devops/patch-rules-on-aws>

https://gitlab.appvance.net/Appvance/devops/patch-rules-on-aws/-/blob/master/patch_my_ruls.sh -
for Unix machine.

https://gitlab.appvance.net/Appvance/devops/patch-rules-on-aws/-/blob/master/patch_my_ruls.ps1 -
for Windows machine.

Go to the AWS console where your EC2 instance:

Open preferred security group:


```
1 #!/bin/bash
2
3 IP='curl -s http://whatismyip.akamai.com/'
4 aws ec2 modify-security-group-rules \
5   --group-id sg-03a8c97cdc9a1565b \
6   --security-group-rules "SecurityGroupRuleId=sgr-02aed38fba0f87801,SecurityGroupRule={Description='Mykola',IpProtocol='tcp',FromPort=3388,ToPort=3390,CidrIpv4=$IP/32}"
```

From now, when the IP is changed, and breaks ties with EC2 instance. Launch script it changes IP fast.

https://gitlab.appvance.net/Appvance/devops/patch-rules-on-aws/-/blob/master/patch_my_ruls.sh - for Unix.

https://gitlab.appvance.net/Appvance/devops/patch-rules-on-aws/-/blob/master/patch_my_ruls.ps1 - for Windows.

Last updated: June 26, 2023

If you have any feedback on this topic, please contact us at documentation@appvance.ai

AWS WorkSpace and Google Chrome

AWS WorkSpace is a cloud-based virtual desktop infrastructure (VDI) solution that allows users to access their desktop applications and data from any device with internet access.

There are considerations for third-party cookies and policy settings when using Amazon WorkSpace and AIQ.

Enabling Third-party Cookies

AIQ requires third-party cookies are enabled in the web browser being used for testing. Enabling third-party cookies is vital for AIQ tests because it allows the web browser to store and retrieve cookies from domains other than the one being visited. This is important because many websites rely on third-party cookies for tracking, advertising, and authentication purposes. Without the ability to access these cookies, AIQ tests may not be able to accurately simulate user behavior on the website being tested.

For example, suppose you are testing a website that uses a third-party authentication provider to log users in. If third-party cookies are not enabled, the browser will not be able to store the authentication cookie from the provider, and the login process may fail. Similarly, if a website relies on third-party tracking cookies to track user behavior, a test that does not enable third-party cookies may not accurately simulate user behavior and may produce incorrect results.

Here are some specific details why third-party cookies need to be enabled on the browsers that are used to record test scenarios for AIQ:

- Simulate user behavior: Cookies are used to store user-specific information, such as login credentials or session IDs.
- Preserve the session state: which means that users can return to the application later and continue where they left off. By allowing cookies, we can simulate this behavior and ensure that the application behaves correctly during test execution.
- Testing different user scenarios: Cookies can be used to simulate different user scenarios, such as testing the application as a new user or a returning user. With the cookies enabled we can simulate different user scenarios and test different parts of the application under different conditions.
- For performance: they can be used to speed up the test execution by avoiding the need to repeat certain steps, such as logging in, during each test run.

Chrome Policies

Using Amazon WorkSpaces supports the use of Google Chrome as a web browser, and there are several policies that can be enforced on Google Chrome.

Here are some of the policies that can be applied to Google Chrome in Amazon WorkSpaces:

- **Block access to the Chrome Web Store:** This policy prevents users from accessing the Chrome Web Store, which contains extensions, themes, and other apps for Google Chrome. This can be useful in environments where IT administrators want to control what extensions and apps are installed on users' devices.
- **Disable auto-updates:** This policy disables automatic updates for Google Chrome, which can be useful in environments where IT administrators want to control the version of Chrome that users are using.
- **Block pop-ups:** This policy blocks pop-up windows in Google Chrome, which can be useful in environments where IT administrators want to prevent users from accessing certain websites or ads.
- **Block unsafe downloads:** This policy prevents users from downloading files that Google Chrome has flagged as potentially unsafe. This can be useful in environments where IT administrators want to prevent users from downloading malware or other harmful files.
- **Enable Safe Browsing:** This policy enables Google Chrome's Safe Browsing feature, which helps protect users from phishing attacks and malware.

For more information on Chrome policies and Amazon WorkSpace see the Amazon AWS documentation at: [target=" blank">https://docs.aws.amazon.com/workspaces-web/latest/adminguide/browser-policies.html](https://docs.aws.amazon.com/workspaces-web/latest/adminguide/browser-policies.html).

Inspecting Chrome Policies

If you want to verify the current policies applied to the browser installed on the workspace, there are two methods to export the policies so they can be shared or studied.

Exporting Chrome Policies to a Text File

1. Open a new tab in the Chrome browser.
2. In the address bar, type "chrome://policy" and press Enter.
3. You should now see a list of policies that are currently applied to the Chrome browser. These policies will include both user and system policies that have been configured by your IT administrator.

4. To copy the policies, you can simply select the text on the page and copy it to your clipboard using the Ctrl+C (Windows) or Command+C (Mac) keyboard shortcut.
5. Once you have copied the policies to your clipboard, you can paste them into a text editor or other application for further analysis or sharing with your IT administrator.

Exporting Chrome Policies to a JSON File

To save Chrome policies to a JSON file in Amazon WorkSpaces, you can follow these steps:

1. Log in to the AWS Management Console and navigate to the Amazon WorkSpaces console.
2. Select the WorkSpaces directory that has the Chrome policies you want to export.
3. Click the **Group Policy** tab, and then click the name of the group policy that contains the Chrome policies you want to export.
4. Click on **Edit** button next to the "Chrome Policies" section.
5. In the Chrome policies editor, click **Download** button at the top of the editor window.
6. Choose a location on your computer where you want to save the JSON file, and then click **Save**.
7. The Chrome policies will be saved to a JSON file that you can use to import the policies into another Amazon WorkSpaces directory or share with other IT administrators.

Last updated: April 19, 2023

If you have any feedback on this topic, please contact us at documentation@appvance.ai

Create an AWS IAM Access Key for AIQ Integrations

Create an AWS IAM Access Key for AIQ Integrations

Create an AWS IAM access key only when an AIQ integration requires programmatic AWS access and a role-based approach is not available. Follow the customer's cloud security policy and use the minimum permissions required.

Security checklist

- Prefer IAM roles, temporary credentials, or workload identity when supported.
- Create keys for a dedicated integration user, not a personal administrator account.
- Attach a least-privilege policy scoped to the required AWS service and resource.
- Store the access key and secret in an approved secret manager or secured AIQ configuration.
- Rotate keys on a defined schedule and remove unused keys immediately.

Validation workflow

1. Test the integration with the minimum required permissions.
2. Confirm logs do not print the access key or secret.
3. Document key owner, purpose, creation date, rotation date, and AWS account.
4. Disable the key if the integration is retired or replaced by a role-based method.

Last updated: October 8, 2026

Set Up IAM Role Permissions for an AWS EC2 AIQ Controller

Set Up IAM Role Permissions for an AWS EC2 AIQ Controller

Use an IAM role for an EC2-hosted AIQ Controller when the controller must access AWS services without long-lived access keys. Scope permissions to the exact services and resources required by the integration.

Security checklist

- Prefer an EC2 instance profile over static IAM user keys.
- Create a least-privilege policy for the required AWS services, actions, and resources.
- Attach the role only to the AIQ Controller instances that need the access.
- Use tags, account boundaries, and environment separation to avoid cross-environment access.
- Review CloudTrail or equivalent logs after the first validation run.

Validation workflow

1. Attach the IAM role or instance profile to the EC2 Controller.
2. Restart or refresh the integration if required by the workflow.

3. Run a small AIQ operation that uses the AWS permission.
4. Confirm access succeeds and no broader permissions are required.
5. Document role name, policy owner, AWS account, and review date.

Last updated: October 8, 2026

Amazon S3 Cross-Account Repository

This document explains how a client sets up an Amazon S3 bucket in **their own AWS account** so that it can be used as an S3 repository by an AppvanceIQ (AIQ) Controller that is hosted in Appvance's AWS account, using AIQ's **cross-account S3 mode**.

Overview

In this setup:

- The **AIQ Controller** runs in the **Appvance AWS account** (`650030447088`).
- The **S3 bucket** used as the repository lives in the **client's AWS account**.
- The Controller reaches the client's bucket by **assuming an IAM role** in the client account via AWS STS (`sts:AssumeRole`), rather than using long-lived access keys.

This is the recommended pattern because the client never has to share IAM access keys/secrets with Appvance, access is scoped to a single role with only the S3 permissions needed, and the client retains full control (revocable at any time by deleting/disabling the role or its trust policy).

High-level flow

1. Client creates an S3 bucket in their account.
2. Client creates an IAM role in their account with a trust policy allowing the Appvance Controller's specific IAM role to assume the role, and a permissions policy scoping access to just that bucket.
3. Client shares the **Role ARN** (and, if used, the **External ID**) with Appvance.
4. Appvance configures the AIQ Controller's S3 repository settings to use cross-account mode with that Role ARN.
5. At runtime, the Controller's AWS integration layer calls `sts:AssumeRole`, receives temporary credentials, and uses those for all S3 operations against the bucket.

Prerequisites

- Access to the client's AWS Console / CLI with permissions to create S3 buckets and IAM roles/policies.
- The Appvance AWS account ID: **650030447088**, and the Appvance Controller's specific IAM role ARN for this engagement (e.g. `arn:aws:iam::650030447088:role/<clientname>-aiq-controller-role`) — Appvance provisions one such role per customer, so confirm the exact ARN with the Appvance team before starting.
- A name/tag convention for the bucket and role (e.g. `clientname-aiq-repo`, `clientname-aiq-crossaccount-role`).
- Decide whether to use an **External ID** for the trust policy (strongly recommended — see Security Recommendations below).

Step 1 — Create the S3 Bucket

1. In the client AWS account, go to **S3 → Create bucket**.
2. Choose a globally unique bucket name, e.g. `clientname-aiq-repo`.
3. Select the AWS region (ideally the same or nearest region to the Appvance Controller for latency).
4. Recommended settings:
 - **Block Public Access:** Keep all four options enabled (bucket should never be public).
 - **Bucket Versioning:** Enable (protects against accidental overwrite/delete of test artifacts).
 - **Default encryption:** Enable SSE-S3 (AES256), or SSE-KMS — either the AWS-managed default key (`alias/aws/s3`) or a customer-managed key — if the client prefers KMS-based encryption.
 - **Object Ownership:** Bucket owner enforced (ACLs disabled).
5. Create the bucket.

Step 2 — Create the Cross-Account IAM Role

This role is what the AIQ Controller in the Appvance account will assume.

1. Go to **IAM → Roles → Create role**.
2. Choose **Custom trust policy** and enter a trust policy that allows the Appvance Controller's specific IAM role to assume this role. This is the default recommendation — Appvance provisions a dedicated controller role per customer (e.g. `arn:aws:iam::650030447088:role/clientname-aiq-controller-role`), and scoping the trust policy to that exact role is tighter than trusting the whole Appvance account. Example (using an External ID, recommended):

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::650030447088:role/clientname-aiq-controller-role"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "sts:ExternalId": "REPLACE_WITH_AGREED_EXTERNAL_ID"
        }
      }
    }
  ]
}
```

Replace `clientname-aiq-controller-role` with the actual Controller role ARN Appvance provides for this customer. If Appvance instead indicates the integration should trust the account as a whole (less common), use `arn:aws:iam::650030447088:root` as the Principal instead. Confirm the exact principal with the Appvance team before finalizing.

3. Skip attaching AWS managed permission policies at this step — a custom least-privilege policy will be attached next.
4. Name the role, e.g. `clientname-aiq-crossaccount-role`.
5. Create the role.

Step 3 — Attach a Least-Privilege S3 Permissions Policy

Create and attach a customer-managed policy scoped to only the bucket created in Step 1.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowListBucket",
      "Effect": "Allow",
      "Action": [
        "s3:ListBucket",
        "s3:GetBucketLocation"
      ]
    }
  ]
}
```

```

    ],
    "Resource": "arn:aws:s3:::clientname-aiq-repo"
  },
  {
    "Sid": "AllowObjectReadWriteDelete",
    "Effect": "Allow",
    "Action": [
      "s3:GetObject",
      "s3:PutObject",
      "s3:DeleteObject",
      "s3:GetObjectVersion",
      "s3:AbortMultipartUpload",
      "s3:ListMultipartUploadParts"
    ],
    "Resource": "arn:aws:s3:::clientname-aiq-repo/*"
  }
]
}

```

Notes:

- Replace `clientname-aiq-repo` with the actual bucket name in both Resource lines.
- If the Controller writes into a specific prefix/folder only (e.g. `aiq/`), further scope Resource to `arn:aws:s3:::clientname-aiq-repo/aiq/*` for tighter isolation.
- If the bucket uses SSE-KMS with a **customer-managed key**, also attach `kms:Decrypt`, `kms:GenerateDataKey`, and `kms:DescribeKey` permissions scoped to that key's ARN, and add the role as a key user in the KMS key policy.
- If the bucket uses SSE-KMS with the **AWS-managed default key** (`alias/aws/s3`), the same three KMS actions (`kms:Decrypt`, `kms:GenerateDataKey`, `kms:DescribeKey`) are still required for cross-account access — a default key does not exempt a cross-account principal from needing explicit KMS permissions. IAM policies require the concrete key ARN rather than the alias, so resolve it first (e.g. `aws kms describe-key --key-id alias/aws/s3`) and use that ARN as the Resource. Also confirm the default key's key policy permits use by this role, since AWS-managed default keys have limited policy editability compared to customer-managed keys.

Attach this policy to the role created in Step 2.

Step 4 — (Optional but Recommended) Restrict the

Bucket Policy

In addition to the IAM role's permissions policy, add a bucket policy on the S3 bucket itself that only allows access from the specific cross-account role. This provides defense-in-depth: even if the IAM policy were ever broadened by mistake, the bucket policy still limits who can touch it.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAIQCrossAccountRoleOnly",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::CLIENT_ACCOUNT_ID:role/clientname-aiq-crossaccount-role"
      },
      "Action": [
        "s3:ListBucket",
        "s3:GetBucketLocation",
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject",
        "s3:GetObjectVersion"
      ],
      "Resource": [
        "arn:aws:s3:::clientname-aiq-repo",
        "arn:aws:s3:::clientname-aiq-repo/*"
      ]
    }
  ]
}
```

Replace `CLIENT_ACCOUNT_ID` with the client's own AWS account ID.

Step 5 — Share the Connection Details with Appvance

Once the bucket, role, and policies are created, share the following with the Appvance team so the AIQ Controller can be configured:

Item	Example	Notes
------	---------	-------

Bucket name	<code>clientname-aiq-repo</code>	
Bucket region	<code>us-east-1</code>	Must match what's configured in AIQ
Role ARN	<code>arn:aws:iam::CLIENT_ACCOUNT_ID:role/clientname-aiq-crossaccount-role</code>	Used by the Controller for <code>sts:AssumeRole</code>
External ID	<code>REPLACE_WITH_AGREED_EXTERNAL_ID</code>	Only if used in the trust policy
KMS Key ARN (if applicable)	<code>arn:aws:kms:...</code>	Required if bucket uses SSE-KMS — either the resolved default key ARN or a customer-managed key ARN

These details should be shared through a secure channel (not plaintext email), since the Role ARN plus External ID together control access to the bucket.

Step 6 — Configure the AIQ Controller (Appvance side)

On the AIQ Controller, the S3 repository is configured in **cross-account mode**, providing:

- Bucket name and region from the table above.
- Role ARN to assume in the client account.
- External ID, if the trust policy requires one.

Internally, AIQ's AWS integration layer performs an `sts:AssumeRole` call against the client's role, receives short-lived temporary credentials, and uses those credentials for all S3 operations (list, get, put, delete) against the bucket — no long-lived client credentials are ever stored in the Controller.

Step 7 — Validate the Setup

1. From the Controller, trigger a test scenario/run that writes to the S3 repo.
2. Confirm the object appears in the client's bucket (via S3 Console or CLI: `aws s3 ls s3://clientname-aiq-repo/`).
3. Confirm the Controller can list and read back existing objects from the bucket.

4. Check CloudTrail in the client account for `AssumeRole` and `s3:*` events from the Controller to confirm access is going through the intended role.

Security Recommendations

- Always use an **External ID** in the trust policy — mitigates the "confused deputy" problem and is an AWS-recommended best practice for third-party role assumption.
- Scope the trust policy Principal to the Appvance Controller's specific IAM role rather than the whole Appvance account root, where possible.
- Scope the IAM policy to the specific bucket (and prefix, if applicable) — never grant `s3:*` on `Resource: *`.
- Keep Block Public Access enabled on the bucket at all times.
- Enable bucket versioning so accidental deletes/overwrites can be recovered.
- Enable server-side encryption (SSE-S3 minimum, SSE-KMS if stricter compliance is required) — remember that SSE-KMS, even on the AWS-managed default key, requires explicit KMS permissions on the cross-account role.
- Enable S3 server access logging or CloudTrail data events on the bucket for auditability.
- Rotate the External ID and update the trust policy if the relationship with Appvance ever needs to be re-scoped or revoked.
- **Revocation:** access can be fully revoked at any time by deleting the IAM role, removing the trust policy statement, or removing the bucket policy statement — no coordination with Appvance is required.

Troubleshooting

Symptom	Likely Cause	Fix
<code>AccessDenied</code> on <code>AssumeRole</code>	Trust policy principal or External ID mismatch	Confirm the exact Principal ARN Appvance uses, and that the External ID matches exactly on both sides
<code>AccessDenied</code> on S3 operations after successful <code>AssumeRole</code>	IAM policy or bucket policy too narrow (wrong bucket name/prefix)	Verify Resource ARNs in both the role's permissions policy and the bucket policy match the actual bucket/prefix

Objects upload but can't be read back	KMS permissions missing on the cross-account role — this applies to both customer-managed keys and the AWS-managed default key (<code>alias/aws/s3</code>)	Add <code>kms:Decrypt</code> / <code>kms:GenerateDataKey</code> / <code>kms:DescribeKey</code> to the role's IAM policy scoped to the resolved key ARN, and for customer-managed keys also add the role as a key user in the KMS key policy
Intermittent <code>ExpiredToken</code> errors	Temporary credentials from AssumeRole expired mid-operation on long-running jobs	Confirm the Controller's AWS SDK layer is refreshing/re-assuming the role rather than caching credentials indefinitely

FAQ

Is IMDSv1 required on the client side for this setup?

No. Neither IMDSv1 nor IMDSv2 is required in the client's AWS account for this integration, and clients can safely leave IMDSv1 disabled.

This setup does not involve any EC2 instances running in the client's AWS account. It consists entirely of an IAM role, a trust policy, an S3 bucket policy, and STS `AssumeRole` calls initiated from the Appvance side. The AIQ Controller runs in Appvance's AWS account (`650030447088`) and calls `sts:AssumeRole` against the client's role to obtain temporary credentials — no compute resource in the client's account is ever involved, so there is no instance in the client's environment that would query the Instance Metadata Service (IMDS) as part of this flow.

Whatever IMDS credential-fetching occurs for the Controller itself happens on Appvance's own EC2 instance, in Appvance's own AWS account, and is entirely independent of and invisible to the client's environment.

For a client's security review: AWS's current guidance favors **IMDSv2-only (with IMDSv1 disabled)** as the more secure default, since it protects against SSRF-based credential theft. Leaving IMDSv1 off is aligned with AWS best practice, not against it.

This answer is scoped specifically to the S3 cross-account repository integration described in this document. If a client is also running other AIQ components inside their own AWS account — for

example, self-hosted TestNodes or agents on their own EC2 instances — those are separate compute resources with their own IMDS considerations, unrelated to this S3 cross-account setup.

Summary

The client only needs to create an S3 bucket, an IAM role with a scoped trust + permissions policy, and share the Role ARN (and External ID) with Appvance. No credentials ever leave the client's account — the AIQ Controller in Appvance's AWS account (650030447088) uses temporary, auditable STS credentials obtained by assuming that role, keeping the client in full control of their own data at all times.