JFrog's AWS Outpost Deployment Guide

JFrog products native hybrid design and versatility makes them a great fit for AWS Outpost. This guide covers the basics of installing into an AWS outpost and covers some best practices.

The (required) core of a JFrog platform is JFrog Artifactory. As such, we will cover its deployment and configuration. Once that is complete, the user can optionally add additional JFrog Products, such as JFrog Xray, JFrog Pipelines and JFrog Distribution.

Overview of the process

1. JFrog Artifactory
    a. Installation
        i. On an EC2 instance
        ii. On Amazon ECS (or other Docker services)
        iii. On Amazon EKS
    b. Set-up
        i. Picking your database
        ii. Picking your binary storage
        iii. HA or standalone
        iv. Putting it all together
        v. Securing your installation
    c. Connecting it to your CI/CD
2. (Optional) JFrog Xray (or any other JFrog Product)
    a. Installation
        i. On an EC2 instance
        ii. On Amazon ECS (or other Docker services)
        iii. On Amazon EKS
    b. Set-Up
        i. HA or standalone
        ii. Connect to JFrog Artifactory

JFrog Artifactory

Installation

It is possible to install JFrog Artifactory as an application on an EC2 instance. This can be done as a ZIP install, RPM, Debian or Docker. If you are using Docker, however, you are probably already using Amazon's ECS or EKS. In these cases, the Docker or Helm charts we offer will provide a more appropriate installation method.

Set-up

Database

Artifactory is quite pluggable and lets you bring your preferred tools and services. You will need a relational database that is JDBC compliant and a binary storage solution that is capable of handling large amounts of data. In AWS Outpost this will likely mean using Amazon RDS with your engine of choice. See our supported databases for more information, including how to configure the connection.

Binary Storage (filestore)

For the binary storage solution, the choice will depend on whether you are configuring High Availability or relying on a single server. In both cases, Amazon's S3 is often a natural fit since it can accommodate very large data sets without the need for manual resizing. If the configuration is HA, Amazon's EFS can be used as shared binary storage between the Artifactory nodes. EFS also scales well and offers great access speed. For instances that expect smaller loads, a simple Amazon EBS (or one per node in HA) may be enough. You can learn more about these options and how to configure them in our Configuring the Filestore wiki entry.

NOTE: You may choose to use AWS Outposts S3 or (for very large storage needs) a standard AWS S3 with a large local cache.

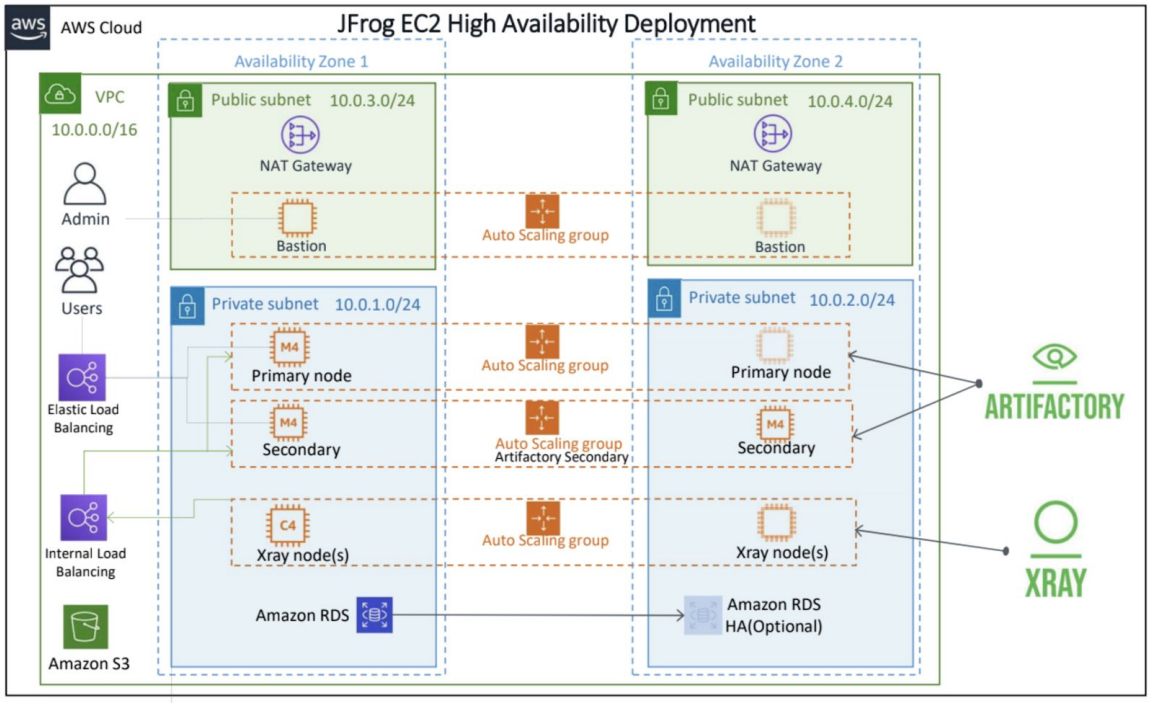
High Availability or Standalone

JFrog Artifactory is mission-critical. If it is down, your whole CI/CD pipeline is down with it. For this reason, we recommend the High Availability solution to everyone. That said, there are many scenarios where having a standalone system is fine. A standalone system can be used for smaller teams, testing, staging or as a disaster recovery site and one instance works well enough for those purposes.
If a High Availability configuration is chosen, the use of a load balancer (like AWS Outpost ALB) is required. Even in the case where a standalone is chosen, we generally recommend a load

balancer to provide SSL. Artifactory itself can be configured to handle SSL but we recommend using a load balancer (and/or reverse proxy) for that task.

Putting it all together

Once you know the type of installation you will be using, the type of database, the binary storage and whether you will be using HA or a standalone instance, you are ready to put everything together. The exact instructions will vary since different installations methods vary and how you provide service details (such as what IAM to use, the database details, etc) will also vary depending on the type of configuration and installation method. See the Installing Artifactory guide for a more complete set of instructions. As a customer, you can also count on the help of our 24/7 support team.



Securing your installation

Artifactory takes the security of the data it handles seriously. This includes encrypting data at rest and in transit (via SSL). There are additional steps you can take to ensure you are using Artifactory in a secure manner. These include using an IAM with the smallest amount of permissions to access your S3 service (if you are using it), making the bucket private and following other S3 best practices.
Artifactory, being a web application, is accessible via certain ports. We recommend blocking all ports but for those that are required. Access to those required ports should only be allowed via an SSL enabled ALB (or reverse proxy).

Aside from the actual installation and configuration, the application itself should be configured with correct groups, users and permissions to make sure only as much access is granted as is needed.
As always, when in doubt, feel free to reach out to our 24/7 support team.

Connecting it to your CI/CD

Artifactory is a universal binary repository. It plays an integral part in your CI/CD process. Its uses (among others) include:

1. Acting as a reliable proxy for your external dependencies
    a. Can point your build server to Artifactory instead of to public repositories like maven, DockerHub, etc
    b. Offers protection from external public repositories going down/removing packages (by caching packages)
    c. Allows you to determine who has access to what public repositories and to which packages in those repositories
    d. Can aggregate multiple public repositories and provide a single unified view
2. Acting as a Docker registry
    a. Can be used to push and pull images with fine-grained access controls
    b. Can proxy other Docker registries
3. Acting as a source/gateway for all packages
    a. This includes external ones and privately created packages (Maven, PyPi, RubyGems, GO, Docker, Vagrant, etc)
4. Acting as a source of truth
    a. Build information (what packages are used where)
    b. CI/CD pipeline metadata
        i. What packages are used for development, staging, testing, production

You will configure your clients (build servers, AWS CodeStar, etc) to both push and pull from Artifactory. This is done in a client-native way, as such, the instructions will vary depending on your usage. See the JFrog Artifactory wiki for more details.

(Optional) JFrog Xray

Installation

Just like JFrog Artifactory, it is possible to install JFrog Xray as an application on an EC2 instance. This can be done as a ZIP install, RPM, Debian or Docker Compose. If you prefer to use EKS, you may instead be interested in the helm chart.

This process is similar for JFrog Distribution, JFrog Mission Control and JFrog Pipelines. See the full JFrog Platform download page for more details.

Set-up

High Availability or Standalone

JFrog Xray is capable of running in either a standalone or HA mode. Unlike JFrog Artifactory, however, there is no need for a load balancer as all JFrog Xray nodes connect to JFrog Artifactory and JFrog Artifactory will load balance between the nodes as needed.
See the JFrog Xray installation wiki for full details.

Connect to JFrog Artifactory

JFrog Xray is a complementary product to JFrog Artifactory and must be connected to it to function. This will require that you get your JFrog Artifactory Join Key and the JFrog Platform URL.
NOTE: JFrog Artifactory and JFrog Xray must be capable of bidirectional communication. The machines should be in the same network for optimal performance.