**What is JFrog?**

JFrog is a **universal DevOps platform** that provides solutions for **continuous integration and delivery (CI/CD)** by automating software release management. Its key product is **JFrog Artifactory**, a binary repository manager that supports storing and managing all types of packages (Docker images, Maven, npm, PyPI, etc.).

**Key features:**

* **Universal package management**: Supports a wide range of software package types (Java, Python, Docker, etc.).
* **Integration with CI/CD tools**: Works seamlessly with Jenkins, GitLab, and other CI/CD tools.
* **High availability and scalability**: Suited for enterprise use with distributed teams.
* **Security**: Provides vulnerability scanning (JFrog Xray) and access control.
* **Repository management**: Stores, organizes, and manages artifacts (binary files, Docker images, etc.).

**Why JFrog?**

1. **Efficient Artifact Management**: JFrog helps manage and store binaries or packages used in your CI/CD pipeline, ensuring easy access to previous versions.
2. **Support for Multiple Technologies**: It supports Docker, Maven, npm, PyPI, and more.
3. **Seamless Integration**: JFrog integrates well with CI/CD tools like Jenkins, Bamboo, and GitLab, and with cloud platforms such as AWS, Azure, and GCP.
4. **Scalability**: JFrog is built to handle large-scale DevOps pipelines in distributed teams.
5. **Security**: With JFrog Xray, it can scan binaries and dependencies for vulnerabilities, improving security across your software supply chain.
6. **Faster Software Delivery**: Automated management of binary files and dependencies streamlines the process, making releases faster and more efficient.

**Installing and Configuring JFrog Artifactory on EC2 Instance**

Here’s a step-by-step guide on how to install and configure JFrog Artifactory on an AWS EC2 instance:

**Prerequisites:**

1. **AWS EC2 instance** running **Amazon Linux** or **Ubuntu** (t2.medium or higher is recommended).
2. **Access to the instance** via SSH.
3. **Java** installed on the instance, as JFrog requires Java to run.

**1. JFrog Artifactory OSS RPM Link**

**You can find the JFrog Artifactory OSS version RPM package at the following link:**

* **Official JFrog OSS RPM download: JFrog Artifactory OSS RPM**

**Make sure to download the latest stable version.**

**Steps to Install JFrog Artifactory OSS Using RPM on EC2**

**4. Download and Install the JFrog Artifactory OSS RPM:**

* Navigate to the JFrog Artifactory OSS RPM page from the link provided above. Find the latest version and copy the download link for the RPM file.
* Use wget to download the RPM directly onto your EC2 instance:

**wget https://releases.jfrog.io/artifactory/artifactory-pro/org/artifactory/oss/jfrog-artifactory-oss/7.x.x/jfrog-artifactory-oss-7.x.x.rpm**

* **Install the RPM package using the yum package manager:**

**sudo yum localinstall jfrog-artifactory-oss-7.x.x.rpm -y**

**5. Start JFrog Artifactory:**

**Once installed, start the Artifactory service:**

**bash**

**Copy code**

**sudo systemctl start artifactory**

* **Enable Artifactory to start on boot:**

**bash**

**Copy code**

**sudo systemctl enable artifactory**

**6. Check Artifactory Status:**

**To ensure that Artifactory is running properly, check its status:**

**bash**

**Copy code**

**sudo systemctl status artifactory**

**7. Access JFrog Artifactory:**

* **Open a browser and access JFrog Artifactory at:**

**vbnet**

**http://your-ec2-public-ip:8081**

* **Follow the setup wizard to configure the admin user and complete the installation.**

**Optional: Configure Firewall or Security Groups**

**If you need to adjust firewall or security group settings to allow access to port 8081, you can modify your EC2 security group to allow inbound traffic on that port.**

**7. Configure Artifactory:**

* After logging in, you can configure **local, remote, and virtual repositories** based on your project requirements.
  + **Local repository**: Stores artifacts that your team builds.
  + **Remote repository**: Proxies external repositories (e.g., Maven Central).
  + **Virtual repository**: Combines local and remote repositories into one URL for easier access