


# JFrog Artifactory Edge



























## Overview

JFrog Artifactory Edge (an "Edge node") is an edition of JFrog Artifactory whose available features have been customized to serve the primary purpose of distributing software to a runtime such as a data center, a point-of-sale or even a mobile device. All packages hosted in an Edge node are [part of a Release Bundle](#) which is a secure and immutable collection of software packages that make up a release to be provisioned or can be uploaded using direct file upload through the UI or REST API.

 Distributing Release Bundles [REST API can be found here.](#)

## Edge Node Features and Capabilities

An Artifactory Edge node is customized to only provide functionality needed for the distribution of packages specified in Release Bundles. The following table summarizes what an Edge node can and cannot do compared to a full Artifactory instance:

	Artifactory	Edge Node
<b>Uploading files</b> <ul style="list-style-type: none"><li>Upload from a source Artifactory instance during a Distribution transaction</li><li>Direct file upload through UI or REST API</li></ul>	 	  Uploading is only supported to the <b>artifactory-edge-uploads</b> generic repository.
<b>Downloading files</b>		
<b>Repositories</b> <ul style="list-style-type: none"><li>Local</li><li>Remote</li><li>Virtual</li></ul>	  	  <b>Smart Remote Repository</b> (Note: regular remote repositories are not supported) 
<b>Storage</b> All forms of storage including cloud storage providers such as S3, GCP and Azure		
<b>Filestore Sharding</b>		
<b>User Plugins</b>		
<b>Replication</b>		 Can only receive data specified in a distribution of a Release Bundle via the <a href="#">replicator</a>
<b>Builds</b>		
<b>Xray integration</b>		
<b>High Availability</b>		

## The Distribution Flow

The high-level distribution flow has two main flows:

### 1. Creating a Release Bundle

You can create a Release Bundle using the [Create Release Bundle Version](#) REST API call on JFrog Distribution and specify a variety of parameters including the files comprising the Release Bundle, and different properties associated with it or [directly in the UI](#).

JFrog Subscription Levels

CLOUD (SaaS) | SELF-HOSTED  
ENTERPRISE+

### Page Contents

- Overview
- Edge Node Features and Capabilities
- The Distribution Flow
- Installing an Edge Node
- Deploying Artifacts
- Pulling Artifacts
- Setting a GPG Key
- NGINX Configuration

### Quick Links

- [Get Started](#)
  - [Get Started: Cloud](#)
  - [Get Started: On-Prem](#)
- [Installing Artifactory](#)
- [Upgrading Artifactory](#)
- [Configuring Artifactory](#)
- [Artifactory REST API](#)
- [Artifactory Release Notes](#)

## 2. Distributing a Release Bundle

Distribution is responsible for triggering the replication process that happens from the source Artifactory to the Edge nodes. First, it replicates the Release Bundle info to each Edge Node, and then initiates the replication process in the source Artifactory.

## Behind the Scenes

### Distributing a Release Bundle

The distribution process includes these three steps:

1. **Start a distribution transaction:** Query JFrog Mission Control for details on the distribution target nodes, and provide the target nodes (Artifactory Edge Nodes), with information about distributed files such as their checksum.
2. **Transfer files with smart replication:** JFrog Distribution invokes the Replicator which distributes files by replicating them from the source Artifactory instance to the target Artifactory Edge nodes.
3. **End a distribution transaction:** JFrog Distribution notifies the Edge node that the transaction is complete. In turn, the Edge node validates the integrity of the transferred Release Bundle and the Release Bundle as a unit through their checksum and hosts them in the correct place as specified in the Release Bundle.

## Installing an Edge Node

The process of installing an Edge node is identical to installing any other Artifactory instance. For more information, see [Installing Artifactory Edge](#). The [Replicator](#) is a process that optimizes replication when distributing software with JFrog Distribution, dramatically reducing the load on the network and the time taken to synchronize Release Bundles from a source Artifactory instance to target instance or Edge node.



### Make sure to activate the Replicator

The primary purpose of an Edge node is to receive Release Bundles for deployment to a runtime which requires the use of the Replicator. Therefore, once your Edge node is installed, you need to ensure its Replicator has been activated and is ready for use as described in [Installation and Activation](#).



### Circle of Trust

An Edge node can only receive Release Bundles from an Artifactory service if they are both within the same circle of trust. Once you have completed the installation of the Edge node, make sure to add it to the circle of trust for any Artifactory service from which it should receive Release Bundles. To learn how to establish a circle of trust, please refer to [Establishing the Circle of Trust](#) in the JFrog Access User Guide.

## Deploying Artifacts

The *artifactory-edge-uploads* repository is automatically created and cannot be removed. Deploying artifacts to this repository can be done [using the UI](#) or [REST API](#).



All repositories in an Artifactory Edge are read-only. The *artifactory-edge-uploads* repository is the only repository to which your Release Bundles are deployed.

The screenshot shows the Artifactory Artifact Repository Browser interface. On the left, a tree view lists various repositories, with 'artifactory-edge-uploads' selected. The main panel displays the details for this repository, including its name, package type, repository path, layout, description, artifact count, and creation date.

General	Effective Permissions	Properties
<b>Info</b>		
Name:	artifactory-edge-uploads	
Package Type:	Generic	
Repository Path:	artifactory-edge-uploads/	
Repository Layout:	simple-default	
Description:	A generic repository for uploads	
Artifact Count / Size:	Show	
Created:	12-12-19 11:36:38 +00:00	

## Pulling Artifacts

From version 6.12, an Artifactory Edge node can be configured to pull artifacts using a [Smart Remote Repository](#). Artifacts can be pulled from other Artifactory instances (ones with Enterprise+ or Edge licenses, just like any remote repository).

When pulling data from a smart remote repository, the Edge node will first attempt to pull the data from cache. If it does not exist, it will continue to try and fetch it from the remote repository on the target instance.

Support for remote repositories (that are not Smart Remote) is not available. For example, creating a remote repository pointing to Docker hub is not supported.



Pull replication is not supported on Edge nodes.

## Setting a GPG Key

To enable a secure distribution flow, an Edge node must be able to validate the contents of any Release Bundles that it receives. This is done by providing the Edge node with the GPG signing key of any Distribution service that will be uploading Release Bundles as a one-time action before any Distribution flows are invoked.



### Updating GPG Keys

Note that if you change the GPG signing keys on any Distribution service that is uploading Release Bundles to an Edge node, the Edge node must be updated with the new public key.

To upload the GPG key of a Distribution service to an Edge node, use the [Set Signing Keys for Distribution](#) REST API endpoint.

## NGINX Configuration

As part of the distribution flow, Artifactory needs to interact with JFrog Distribution which relies on *HTTP/1.1* protocol functionalities, such as **chunked transfer encoding**. To support these functionalities, you need to add the following settings to your [NGINX configuration](#).

```
proxy_http_version 1.1;
chunked_transfer_encoding on;
```