HA Installation and Setup

Requires an **Enterprise license**

Page Contents

Overview

This page describes how to set up an Artifactory HA cluster with one or more nodes.

Requirements

Version

Artifactory HA is supported from Artifactory 3.1 and above. If you are running a previous version of Artifactory, you first need to upgrade as described in Upgrading Artifactory.

All nodes within the same Artifactory HA installation must be running the same Artifactory version and the same JVM version.

Licensing

Artifactory HA is supported with an Enterprise License. Each node in the cluster must be activated with a different license, however, this is transparently and automatically managed by the Artifactory Cluster License Manager.

Hardware

Artifactory HA requires the following hardware:

- Load balancer
- External database server with a single URL to the database

Network

- All the Artifactory HA components (Artifactory cluster nodes, database server and load balancer) must be within the same high-speed LAN
- All the HA nodes must communicate with each other through dedicated TCP ports
- Network communications between the cluster nodes must be enabled for each of the cluster nodes.

Database

Artifactory HA requires an external database and currently supports Oracle, MySQL, MS SQL, PostgreSQL and MariaDB. For details on how to configure any of these databases please refer to Configuring the Database.

> **Installing 5.6.x or below?**
>
> Refer to HA Installation and Setup Version 5.6.x and Below

Home Directory

When setting up Artifactory HA you need to configure the *$ARTIFACTORY_HOME* directory separately for each of the Artifactory cluster nodes in your system.

The general layout of these directories is as follows:

```
|- $ARTIFACTORY_HOME

     |- access

     |- etc/

          |- ha-node.properties
          |- logback.xml
          |- artifactory.cluster.lic
          |- mimetypes.xml
          |- cluster.id

          |- binarystore.xml
          |- db.properties
          |- plugins
          |- ui
          |- security
            |- access
               |-etc

     |- data/
          |- tmp/
          |- artifactory.properties

     |- logs/

     |- webapps/

     |- tomcat/
        |- lib/
           |- <jdbc driver>/

     |- bin/

     |- misc/

     |-backup/

     |-support
```

Installing Artifactory HA

An Artifactory HA node is first installed as an Artifactory Pro instance, and is then modified to operate as a node in the HA cluster by configuring the *$ARTIFACTORY_HOME/etc/ha-node.properties* file.

The Installation Process

The binary storage in an HA installation must be accessible to all nodes in the cluster. This is achieved either by mounting a Network File System (NFS) on each cluster node, using shared object storage, or by using the nodes' local file systems while using a mechanism that synchronizes the binaries between them.

The installation procedure involves two stages:

1. **Setting up your storage configuration**
   The storage configuration varies depending on your decision for two parameters of your setup:
   a. **Binary store:** Do you plan to use **Filesystem Storage** to store binaries on your nodes' filesystems, or a **Cloud Storage** provider such as S3, GCS or any other S3-compliant provider?
   b. **NFS:** Do you plan to use the Network File System (NFS) or not?
2. **Installing the cluster nodes**
   Once your storage is configured and set up, the rest of the installation process is identical

Setting Up Your Storage Configuration

Your choice for binary store and use of NFS or not leads to one of the following four options for setting up your storage configuration:

Using Filesystem Storage with the NFS

To set up your HA cluster to use filesystem storage with the NFS, follow these steps which are detailed below:

- Create and configure *$ARTIFACTORY_HOME/etc/ha-node.properties*
- Create an NFS mount
- Configure the binarystore.xml file

Once you have completed configuring your filestore, you are ready to complete the HA installation process by installing the cluster nodes.

Create ha-node.properties

Create the *$ARTIFACTORY_HOME/etc/ha-node.properties* file and populate it with the following parameters:

node.id	Unique descriptive name of this server. ⚠ **Uniqueness** Make sure that each node has an id that is unique on your whole network.
context.url	The context url that should be used to communicate with this server within the cluster. For example: `context.url=https://<ARTIFACTORY_URL>:<ARTIFACTORY_PORT>/artifactory` Applies to IPv4 and IPv6. There are two ways to specify the context.url field: • As an explicit IP address • As a host name. In this case, you need to specify the `hazelcast.interface` field with wildcards. For details, please refer to the description for `hazelcast.interface` field below.
membership.port	⚠ **Not used from version 6.2.0** From version 6.2.0 this parameter should be set to 0 since an internal locking mechanism was introduced in Artifactory removing the need for it. For more details please refer to the JFrog Knowledge Base. The port that should be used to communicate with this server within the cluster. If not specified, Artifactory will allocate a port automatically, however we recommend to set this to a fixed value to ensure that the port allocated is open to all of your organizations security systems such as firewalls etc.
primary	(true | false) Indicates if this is the primary server. There must be one (and only one) server configured in the cluster to be the primary server. For other servers this parameter is optional and its value defaults to "false".
artifactory.ha.data.dir	This property provides the full path to the root directory of your NFS binary storage.
artifactory.ha.backup.dir	This property provides the full path to the root directory of your Artifactory back-up data on the NFS.

<table>
<tr>
<td><b>hazelcast.interface</b></td>
<td>
<p>[Optional] When nodes in the same cluster are running on different networks (e.g. nodes on different docker hosts), set this value to match the server's internal IP address. Applies to IPv4 and IPv6.</p>
<p>If you have specified the <code>context.url</code> as a host name, you need to use to the wildcard character (i.e., an asterisk - '*') so as to include the server's internal IP address as well as that of all members in the cluster.</p>
<p>For example, if you have two nodes with the following parameters:</p>
<table>
<tr><th>Node</th><th>IP</th><th>Host name</th></tr>
<tr><td>A</td><td>10.1.<b>2.22</b></td><td>node.a</td></tr>
<tr><td>B</td><td>10.1.<b>3.33</b></td><td>node.b</td></tr>
</table>
<p>then the <code>hazelcast.interface</code> field should be set to <b>10.1.*.*</b></p>
<p>Another example, if you have two nodes with the following parameters:</p>
<table>
<tr><th>Node</th><th>IP</th><th>Host name</th></tr>
<tr><td>A</td><td>10.1.2.<b>22</b></td><td>node.a</td></tr>
<tr><td>B</td><td>10.1.2.<b>33</b></td><td>node.b</td></tr>
</table>
<p>then the <code>hazelcast.interface</code> field should be set to <b>10.1.2.*</b></p>
</td>
</tr>
</table>

ha-node.properties file permissions

On Linux, once the *ha-node.properties* file is created, the Artifactory user should be set as its owner and its permissions should be set to 644(-rw-r--r--)

The example below shows how an *ha-node.properties* file may be configured for using filesystem storage with the NFS

```
node.id=art1
context.url=http://10.0.0.121:8081/artifactory
membership.port=10001
primary=true
artifactory.ha.data.dir=/mnt/shared/artifactory/ha-data
artifactory.ha.backup.dir=/mnt/shared/artifactory/ha-backup
hazelcast.interface=192.168.0.2 (optional)
```

Escaping the backslash in Windows systems

Note that in Windows-based system the backslash characters in the paths to the ha-data and ha-backup directories need to be escaped with another backslash. For example:
artifactory.ha.data.dir = \\\\windows\\UNC\\path\\ha-data artifactory.ha.backup.dir = \\\\windows\\UNC\\path\\ha-backup

Create an NFS mount

When setting up Artifactory HA you need to configure the *$ARTIFACTORY_HOME* directory separately for each of the Artifactory cluster nodes in your system, and a common *$DATA_DIR* that is accessible to all nodes to host all your filestore binaries

Create an NFS mount which will be accessible to all nodes. This mount will serve as the *$DATA_DIR.*

In addition, you need to set up a *$BACKUP_DIR* that must be accessible by the master node. It may be located on the same NFS mount, however this is not compulsory.

Privileges

Each of the Artifactory cluster nodes must have full write privileges on the *$DATA_DIR* directory tree and the UID/GID for the artifactory user must match on all nodes.

Mounting the NFS from Artifactory HA nodes

When mounting the NFS on the client side, make sure to add the following option for the `mount` command:

```
lookupcache=none
```

This ensures that nodes in your HA cluster will immediately see any changes to the NFS made by other nodes.

Configure the binarystore.xml File

The default *`binarystore.xml`* that comes with Artifactory out-of-the-box contains the file-system template. Since this is exactly the configuration you need, there is no need to modify the `binarystore.xml` file.

In this configuration, Artifactory uses the `artifactory.ha.data.dir` as the location for all binaries.

You are now ready to complete the HA installation process by **installing the cluster nodes**.

Using Filesystem Storage Without the NFS

To set up your HA cluster to use filesystem storage without the NFS, follow these steps which are detailed below:

- Create and configure *`$ARTIFACTORY_HOME/etc/ha-node.properties`*
- Configure the binarystore.xml file

Create ha-node.properties

Create the *$ARTIFACTORY_HOME/etc/ha-node.properties* file and populate it with the following parameters:

node.id	Unique descriptive name of this server. **Uniqueness** Make sure that each node has an id that is unique on your whole network.
context.url	The context url that should be used to communicate with this server within the cluster. There are two ways to specify the context.url field: • As an explicit IP address. Applies to IPv4 and IPv6. • As a host name. In this case, you need to specify the `hazelcast.interface` field with wildcards. For details, please refer to the description for `hazelcast.interface` field below.
membership.port	**Not used from version 6.2.0** From version 6.2.0 this parameter should be set to 0 since an internal locking mechanism was introduced in Artifactory removing the need for it. For more details please refer to the JFrog Knowledge Base. The port that should be used to communicate with this server within the cluster. If not specified, Artifactory will allocate a port automatically, however we recommend to set this to a fixed value to ensure that the port allocated is open to all of your organizations security systems such as firewalls etc.
primary	(true | false) Indicates if this is the primary server. There must be one (and only one) server configured in the cluster to be the primary server. For other servers this parameter is optional and its value defaults to "false".

<table>
<tr>
<td><b>hazelcast.
interface</b></td>
<td>
<p>[Optional] When nodes in the same cluster are running on different networks (e.g. nodes on different docker hosts), set this value to match the server's internal IP address. Applies to IPv4 and IPv6.</p>
<p>If you have specified the <code>context.url</code> as a host name, you need to use to the wildcard character (i.e., an asterisk - '*') so as to include the server's internal IP address as well as that of all members in the cluster.</p>
<p>For example, if you have two nodes with the following parameters:</p>
<table>
<tr><th>Node</th><th>IP</th><th>Host name</th></tr>
<tr><td>A</td><td>10.1.<b>2.22</b></td><td>node.a</td></tr>
<tr><td>B</td><td>10.1.<b>3.33</b></td><td>node.b</td></tr>
</table>
<p>then the <code>hazelcast.interface</code> field should be set to <b>10.1.*.*</b></p>
<p>Another example, if you have two nodes with the following parameters:</p>
<table>
<tr><th>Node</th><th>IP</th><th>Host name</th></tr>
<tr><td>A</td><td>10.1.2.<b>22</b></td><td>node.a</td></tr>
<tr><td>B</td><td>10.1.2.<b>33</b></td><td>node.b</td></tr>
</table>
<p>then the <code>hazelcast.interface</code> field should be set to <b>10.1.2.*</b></p>
</td>
</tr>
</table>

ha-node.properties file permissions

On Linux, once the *ha-node.properties* file is created, the Artifactory user should be set as its owner and its permissions should be set to 644(-rw-r--r--)

The example below shows how the *ha-node.properties* file might be configured for your cluster nodes to use filesystem storage without the NFS:

```
node.id=art1
context.url=http://10.0.0.121:8081/artifactory
membership.port=10001
primary=true
hazelcast.interface=192.168.0.2 (optional)
```

Configure the binarystore.xml File

The default *binarystore.xml* that comes with Artifactory out-of-the-box contains the file-system template which uses the NFS. Therefore, to setup your filestore so that it doesn't use the NFS, you need to modify this file.

Take care when modifying binarystore.xml

Making changes to this file may result in losing binaries stored in Artifactory!

If you are not sure of what you are doing, please contact JFrog Support for assistance.

We recommend using the **cluster-file-system** template which is one of the built-in templates that come with Artifactory out-of-the-box. This configuration uses the default filestore location (under *$ARTIFACTORY_HOME/data*) to store binaries locally on the filesystem, unless specified otherwise. A mechanism connected to all other nodes in the cluster is used to keep binaries synchronized and accessible to all nodes, based on the required redundancy value (which is 2 by default).

How to use the cluster-file-system template

To learn how to configure your *binarystore.xml* to use the cluster-file-system template, please refer to Basic Configuration Elements under Configuring the Filestore.

If your cluster has only two nodes, we recommend modifying the lenientLimit from its default value of 0 which would prevent writes to Artifactory if one of the nodes goes down.

You are now ready to complete the HA installation process by **installing the cluster nodes**.

Using Cloud Storage With the NFS

To set up your HA cluster to use cloud storage with the NFS, follow these steps which are detailed below:

- Create and configure *$ARTIFACTORY_HOME/etc/ha-node.properties*
- Create an NFS mount
- Configure the binarystore.xml file

Create ha-node.properties

Create the ha-node.properties file and populate it with the following parameters:

<table>
<tr><td><b>node.id</b></td><td>Unique descriptive name of this server.
<b>Uniqueness</b>
Make sure that each node has an id that is unique on your whole network.</td></tr>
<tr><td><b>context.url</b></td><td>The context url that should be used to communicate with this server within the cluster.
There are two ways to specify the context.url field:
• As an explicit IP address. Applies to IPv4 and IPv6.
• As a host name. In this case, you need to specify the <code>hazelcast.interface</code> field with wildcards. For details, please refer to the description for <code>hazelcast.interface</code> field below.</td></tr>
<tr><td><b>membership.port</b></td><td><b>Not used from version 6.2.0</b>
From version 6.2.0 this parameter should be set to 0 since an internal locking mechanism was introduced in Artifactory removing the need for it. For more details please refer to the JFrog Knowledge Base.
The port that should be used to communicate with this server within the cluster.
If not specified, Artifactory will allocate a port automatically, however we recommend to set this to a fixed value to ensure that the port allocated is open to all of your organizations security systems such as firewalls etc.</td></tr>
<tr><td><b>primary</b></td><td>(true | false) Indicates if this is the primary server. There must be one (and only one) server configured in the cluster to be the primary server. For other servers this parameter is optional and its value defaults to "false".</td></tr>
<tr><td><b>artifactory.ha.data.dir</b></td><td>This property provides the full path to the root directory of your NFS binary storage.</td></tr>
<tr><td><b>artifactory.ha.backup.dir</b></td><td>This property provides the full path to the root directory of your Artifactory back-up data on the NFS.</td></tr>
<tr><td><b>hazelcast.interface</b></td><td>[Optional] When nodes in the same cluster are running on different networks (e.g. nodes on different docker hosts), set this value to match the server's internal IP address. Applies to IPv4 and IPv6.
If you have specified the <code>context.url</code> as a host name, you need to use to the wildcard character (i.e., an asterisk - '*') so as to include the server's internal IP address as well as that of all members in the cluster.
For example, if you have two nodes with the following parameters:
<table>
<tr><th>Node</th><th>IP</th><th>Host name</th></tr>
<tr><td>A</td><td>10.1.<b>2.22</b></td><td>node.a</td></tr>
<tr><td>B</td><td>10.1.<b>3.33</b></td><td>node.b</td></tr>
</table>
then the <code>hazelcast.interface</code> field should be set to <b>10.1.*.*</b>
Another example, if you have two nodes with the following parameters:
<table>
<tr><th>Node</th><th>IP</th><th>Host name</th></tr>
<tr><td>A</td><td>10.1.2.<b>22</b></td><td>node.a</td></tr>
<tr><td>B</td><td>10.1.2.<b>33</b></td><td>node.b</td></tr>
</table>
then the <code>hazelcast.interface</code> field should be set to <b>10.1.2.*</b></td></tr>
</table>

The example below shows how the *ha-node.properties* file might be configured for your cluster nodes to use cloud storage with the NFS:

```
node.id=art1
context.url=http://10.0.0.121:8081/artifactory
membership.port=10001
primary=true
artifactory.ha.data.dir=/mnt/shared/artifactory/ha-data
artifactory.ha.backup.dir=/mnt/shared/artifactory/ha-backup
hazelcast.interface=192.168.0.2
```

> **Escaping the backslash in Windows systems**
>
> Note that in Windows-based system the backslash characters in the paths to the ha-data and ha-backup directories need to be escaped with another backslash. For example:
>
> artifactory.ha.data.dir = \\\\windows\\UNC\\path\\ha-data
>
> artifactory.ha.backup.dir = \\\\windows\\UNC\\path\\ha-backup

> **ha-node.properties file permissions**
>
> On Linux, once the *ha-node.properties* file is created, the Artifactory user should be set as its owner and its permissions should be set to 644(-rw-r--r--)

Create an NFS mount

When setting up Artifactory HA you need to configure the *$ARTIFACTORY_HOME* directory separately for each of the Artifactory cluster nodes in your system, and a common *$DATA_DIR* that is accessible to all nodes to host all your filestore binaries

Create an NFS mount which will be accessible to all nodes. This mount will serve as the *$DATA_DIR.*

In addition, you need to set up a *$BACKUP_DIR* that must be accessible by the master node. It may be located on the same NFS mount, however this is not compulsory.

> **Privileges**
>
> Each of the Artifactory cluster nodes must have full write privileges on the *$DATA_DIR* directory tree.

> **Mounting the NFS from Artifactory HA nodes**
>
> When mounting the NFS on the client side, make sure to add the following option for the `mount` command:
>
> `lookupcache=none`
>
> This ensures that nodes in your HA cluster will immediately see any changes to the NFS made by other nodes.

Configure the *binarystore.xml* file

The default *binarystore.xml* that comes with Artifactory out-of-the-box contains the **file-system template**. Therefore, to setup your filestore so to use cloud storage with the NFS, you need to modify this file.

Warning: Take care when modifying the binarystore.xml file

Making changes to this file may result in losing binaries stored in Artifactory!

If you are not sure of what you are doing, please contact JFrog Support for assistance.

We recommend using either the **s3** chain or the **google-storage** chain which are among the built-in chain templates that come with Artifactory out-of-the-box. These chains use the shared filestore location (under $DATA_DIR) to store binaries in a staging area, before they are moved to the cloud storage.

Tip: To learn how to configure your *binarystore.xml* to use the s3 and google-storage chain templates, please refer to Basic Configuration Elements under Configuring the Filestore.

You are now ready to complete the HA installation process by **installing the cluster nodes**.

Using Cloud Storage Without the NFS

To set up your HA cluster to use cloud storage without the NFS, follow these steps which are detailed below:

- Create and configure *$ARTIFACTORY_HOME/etc/ha-node.properties*
- Configure the binarystore.xml file

Create ha-node.properties

Create the ha-node.properties file and populate it with the following parameters:

<table>
<tr><td><b>node.id</b></td><td>Unique descriptive name of this server.
<b>Uniqueness</b>
Make sure that each node has an id that is unique on your whole network.</td></tr>
<tr><td><b>context.url</b></td><td>The context url that should be used to communicate with this server within the cluster.
There are two ways to specify the context.url field:
• As an explicit IP address
• As a host name. In this case, you need to specify the <code>hazelcast.interface</code> field with wildcards. For details, please refer to the description for <code>hazelcast.interface</code> field below.</td></tr>
<tr><td><b>membership.port</b></td><td><b>Not used from version 6.2.0</b>
From version 6.2.0 this parameter should be set to 0 since an internal locking mechanism was introduced in Artifactory removing the need for it. For more details please refer to the JFrog Knowledge Base.
The port that should be used to communicate with this server within the cluster.
If not specified, Artifactory will allocate a port automatically, however we recommend to set this to a fixed value to ensure that the port allocated is open to all of your organizations security systems such as firewalls etc.</td></tr>
<tr><td><b>primary</b></td><td>(true | false) Indicates if this is the primary server. There must be one (and only one) server configured in the cluster to be the primary server. For other servers this parameter is optional and its value defaults to "false".</td></tr>
<tr><td><b>hazelcast.interface</b></td><td>[Optional] When nodes in the same cluster are running on different networks (e.g. nodes on different docker hosts), set this value to match the server's internal IP address.
If you have specified the <code>context.url</code> as a host name, you need to use to the wildcard character (i.e., an asterisk - '*') so as to include the server's internal IP address as well as that of all members in the cluster.
For example, if you have two nodes with the following parameters:
<table>
<tr><th>Node</th><th>IP</th><th>Host name</th></tr>
<tr><td>A</td><td>10.1.<b>2.22</b></td><td>node.a</td></tr>
<tr><td>B</td><td>10.1.<b>3.33</b></td><td>node.b</td></tr>
</table>
then the <code>hazelcast.interface</code> field should be set to <b>10.1.*.*</b>
Another example, if you have two nodes with the following parameters:
<table>
<tr><th>Node</th><th>IP</th><th>Host name</th></tr>
<tr><td>A</td><td>10.1.2.<b>22</b></td><td>node.a</td></tr>
<tr><td>B</td><td>10.1.2.<b>33</b></td><td>node.b</td></tr>
</table>
then the <code>hazelcast.interface</code> field should be set to <b>10.1.2.*</b></td></tr>
</table>

> **ha-node.properties file permissions**
>
> On Linux, once the *ha-node.properties* file is created, the Artifactory user should be set as its owner and its permissions should be set to 644(-rw-r--r--

The example below shows how the *ha-node.properties* file might be configured for your cluster nodes to use cloud storage without the NFS:

```
node.id=art1
context.url=http://10.0.0.121:8081/artifactory
membership.port=10001
primary=true
hazelcast.interface=192.168.0.2 (optional)
```

Configure the binarystore.xml File

The default *binarystore.xml* that comes with Artifactory out-of-the-box contains the file-system template. Therefore, to setup your filestore so to use cloud storage without the NFS, you need to modify this file.

> **Take care when modifying binarystore.xml**
>
> Making changes to this file may result in losing binaries stored in Artifactory!
>
> If you are not sure of what you are doing, please contact JFrog Support for assistance.

We recommend using either the **cluster-s3** chain or the **cluster-google-storage** chain which are among the built-in templates that come with Artifactory out-of-the-box. These templates use a mechanism connected to all other nodes in the cluster to keep binaries synchronized and accessible to all nodes according to the required redundancy (which is 2 by default). Binaries are first stored locally on each node (under *$ARTIFACTORY_HOME/data/eventual* by default), with additional copies on other nodes according to the redundancy configured, before moving on to persistent cloud storage.

> **How to use the s3 and google-storage chain templates**
>
> To learn how to configure your *binarystore.xml* to use the cluster-s3 and cluster-google-storage chain templates, please refer to Basic Configuration Elements under Configuring the Filestore.

You are now ready to complete the HA installation process by **installing the cluster nodes**.

Installing the Cluster Nodes

Once you have completed setting up your filestore configuration, the process for installing the cluster nodes is identical and described in the steps below:

1. Install the primary node
2. Create the Master Key
3. Add licenses
4. Set the cluster's URL Base
5. Add secondary nodes

Installing the Primary Node

Go through a regular installation of Artifactory Pro as described in Installing Artifactory, **and then convert it to be the HA primary node by adding the *ha-node.properties* file you created** when you set up your storage configuration to the *$ARTIFACTORY_HOME/etc*. **Do not start up the instance yet.**

Note that an external database must be configured for usage at this point, as mentioned in the Requirements section

You should also verify that your database JDBC driver is correctly located in *$ARTIFACTORY_HOME/tomcat/lib* **for each Artifactory cluster node**.

Create the Master Key

The Master key is an AES 128 bit secret key that's used by Artifactory to securely synchronize files between cluster nodes. It is responsible to encrypt and decrypt the shared data in the database.

By default the `master.key` file is automatically generated by Artifactory during the initial start up of the primary node. Optionally, you can generate one manually by running ‘`openssl rand -hex 16`’, which will output a 128 bit key size (hexadecimal encoded). Place this key in the *$ARTIFACTORY_HOME/etc/security/master.key* directory, before starting the primary node.

Add Licenses

There are several ways you can add licenses to the cluster:

- Using the Cluster License Manager UI or REST API as described in Adding Licenses
- As part of the onboarding wizard you will get when you start up Artifactory for the first time
- Using the Artifactory Bootstrap YAML File (NOTE: this will only work for the primary node)

Since currently, the only operative node is the primary node, you can install your licenses there. Once you add the secondary nodes to the cluster, they will be licensed automatically through the Cluster License Manager.

Requires an **Enterprise license**

All licenses used must be Enterprise licenses.

Set the URL Base

After you have installed the node and verified that your system is working correctly as an HA installation, you should configure the **Custom URL Base.**
In the **Admin** tab under **Configuration | General Configuration,** set the **Custom Base URL** field to the URL of the Load Balancer.

Add Secondary Nodes

You should also verify that your database JDBC driver is correctly located in *$ARTIFACTORY_HOME/tomcat/lib* **for each Artifactory cluster node**.

To add secondary nodes, for each node, follow these steps:

1. Create an *`ha-node.properties`* file according to how you want to set up your storage configuration.
2. Go through a new Artifactory Pro installation as described in Installing Artifactory. **Do not start up the instance yet.**
   Note that an external database **must** be configured for usage at this point, as mentioned in the Requirements section
3. Once the Artifactory Pro installation is complete, add the *`ha-node.properties`* file you created to the *`$ARTIFACTORY_HOME/etc`* folder.
4. On the secondary node, create an *$ARTIFACTORY_HOME/etc/security* folder, and copy to it the Master Key file (*$ARTIFACTORY_HOME/etc/security/master.key*) from one of the existing nodes in the cluster.

   Warning: Master Key and db.properties
   This is a critical step in the installation process. The *Master Key* and *db.properties* files must be installed in each secondary node before you start it up for it to operate correctly in the cluster.

5. Start up the cluster node. Upon starting up, the node is automatically allocated a license by the Cluster License Manager.

   Warning: Setting Enterprise+ Service Licensing in an HA Environment
   For Enterprise+ installations, licenses are added in JFrog Mission Control during the Add New Service process. When adding a new service, the license is extracted from the license bucket.
   During an HA installation, if you are adding a new node or swapping an existing one, you need to manually reconfigure the services with the licenses.

6. Test your HA configuration after each cluster node that you add to your system.

Warning: Ensure network communication

Make sure that network communication is enabled **between the cluster nodes** for each of the following:

- context.url
- hazelcast.interface and membership.port (used together. For example, `172.24.0.1:10001`)

> **membership.port not used from version 6.2.0**
>
> From version 6.2.0 this parameter should be set to 0 since an internal locking mechanism was introduced in Artifactory removing the need for it. Since the membership.port is no longer used by Artifactory you can also consider removing this parameter completely from the ha-node.properties file starting from version 6.2.0
>
> For more details please refer to the JFrog Knowledge Base.

Note: Synchronizing configuration files between nodes in an HA cluster

From version 5.0, Artifactory automatically propagates configuration files between the nodes of an HA cluster. These include both internal configuration files as well as *`binarystore.xml`* (which is responsible for the Artifactory filestore configuration), and the *`artifactory.system.properties`* file which may be modified by an Artifactory administrator.

Note that to modify the *`binarystore.xml`* or *`artifactory.system.properties`* you must make the changes on the **primary node and then restart** your HA cluster for the changes to take effect.

In addition, from version 5.7.0, propagation of configuration files **ignores the timestamp** of the files on each corresponding node and makes sure to mirror those on the primary node.

Database Locking

From version 6.0.0, the new database locking mechanism adds its own connection pool (defaults to the value of the pool.max.active value).

However, you may need to adjust your database connection limit to accept more connections. For example, if your database is set to accept up to 100 connections from each node, you may consider increasing the limit to 200 concurrent connections per-node, to accommodate the full utilization of the locking connection pool. Your database should accept the number of configured connections per-node multiplied by the number of the nodes in the cluster.

Session Management

From version 6.2, Artifactory offers different alternatives for session management between the Artifactory HA members when accessing one of the members.

The way to manage sessions is controlled via the `artifactory.map.provider.type` property (in the *$ARTIFACTORY_HOME/etc/artifactory.system.properties* file) which can take the following values:

db	(default) Sessions are managed by the database
distributed	Sessions are managed by Hazelcast
jvm	Sessions are managed by the JVM

If sessions are configured to be managed by the database, Artifactory will also schedule a cron job to cleanup old sessions.

The cron expression to trigger the sessions can be configured using the `artifactory.db.session.cleanup.cron` property in the *$ARTIFACTORY_HOME/etc/artifactory.system.properties* file.

During installation, this cron expression is set with a default value that triggers the cleanup at a set minute (determined randomly) past each hour.

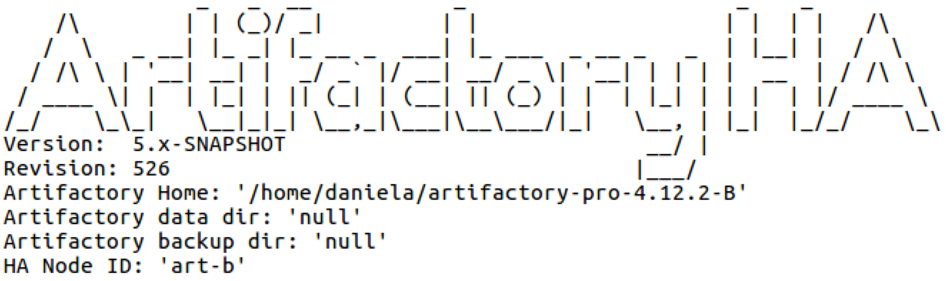
Upgrading Artifactory HA

Upgrading Artifactory HA depends on which version you are starting from. For detailed instructions, please refer to Upgrading an Enterprise HA Cluster.

Testing Your HA Configuration

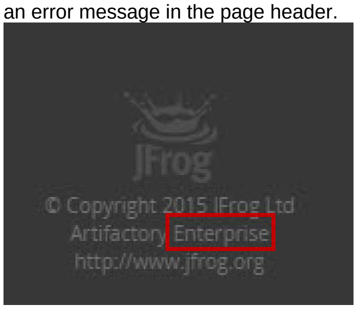
The following are a series of tests you can do to verify that your system is configured correctly as an HA installation:

1. Directly Access the Artifactory UI for the server you have just configured
2. In the **Admin** module go to **Advanced | System Logs** to view the log and verify that you see an entry for **HA Node ID.**

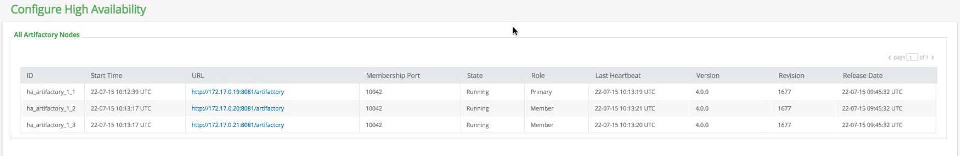


```
               _   _  __           _                       _    _
    /\        | | (_)/ _|         | |                     | |  | |   /\
   /  \   _ __| |_ _| |_ __ _  ___| |_ ___  _ __ _   _    | |__| |  /  \
  / /\ \ | '__| __| |  _/ _` |/ __| __/ _ \| '__| | | |   |  __  | / /\ \
 / ____ \| |  | |_| | || (_| | (__| || (_) | |  | |_| |   | |  | |/ ____ \
/_/    \_\_|   \__|_|_| \__,_|\___|\__\___/|_|   \__, |   |_|  |_/_/    \_\
Version:  5.x-SNAPSHOT                            __/ |
Revision: 526                                    |___/
Artifactory Home: '/home/daniela/artifactory-pro-4.12.2-B'
Artifactory data dir: 'null'
Artifactory backup dir: 'null'
HA Node ID: 'art-b'
```

3. The bottom of the module navigation bar should also indicate that you are running with an Enterprise license. In case of an error you will see an error message in the page header.



4. Access Artifactory through your load balancer and log in as **Admin**.
5. In the **Admin** module go to **Configuration.** There should be a section called **High Availability.** When selected you should see a table with details on all the Artifactory nodes in your cluster as displayed below.



Configure High Availability

All Artifactory Nodes

‹ page 1 of 1 ›

ID	Start Time	URL	Membership Port	State	Role	Last Heartbeat	Version	Revision	Release Date
ha_artifactory_1_1	22-07-15 10:12:39 UTC	http://172.17.0.19:8081/artifactory	10042	Running	Primary	22-07-15 10:13:19 UTC	4.0.0	1677	22-07-15 09:45:32 UTC
ha_artifactory_1_2	22-07-15 10:13:17 UTC	http://172.17.0.20:8081/artifactory	10042	Running	Member	22-07-15 10:13:21 UTC	4.0.0	1677	22-07-15 09:45:32 UTC
ha_artifactory_1_3	22-07-15 10:13:17 UTC	http://172.17.0.21:8081/artifactory	10042	Running	Member	22-07-15 10:13:20 UTC	4.0.0	1677	22-07-15 09:45:32 UTC

6. In the **Admin** module under **Configuration | General,** verify that the **Custom URL Base** field is correctly configured to the URL of the Load Balancer.

Cluster License Management

Artifactory 5.0 introduces an automated license management interface for HA clusters through which all licenses are allocated automatically to nodes as they are added to the cluster. A batch of licenses can be added through the UI and REST API to any node in a cluster.

A new node starting up will request an available license from the pool automatically, and will be allocated the license with the latest expiry date. The license is also automatically returned to the pool if the node is shut down or removed from the HA cluster.

> ⚠ **Which license is allocated?**
>
> Note that adding a license through a node does not necessarily mean that the license will be attached to that specific node. The license is added to the pool available and the available license with the latest expiry date will be allocated to the node.

Once you have purchased a set of licenses, they are provided to you as a space-separated or newline-separated list.

Adding Licenses

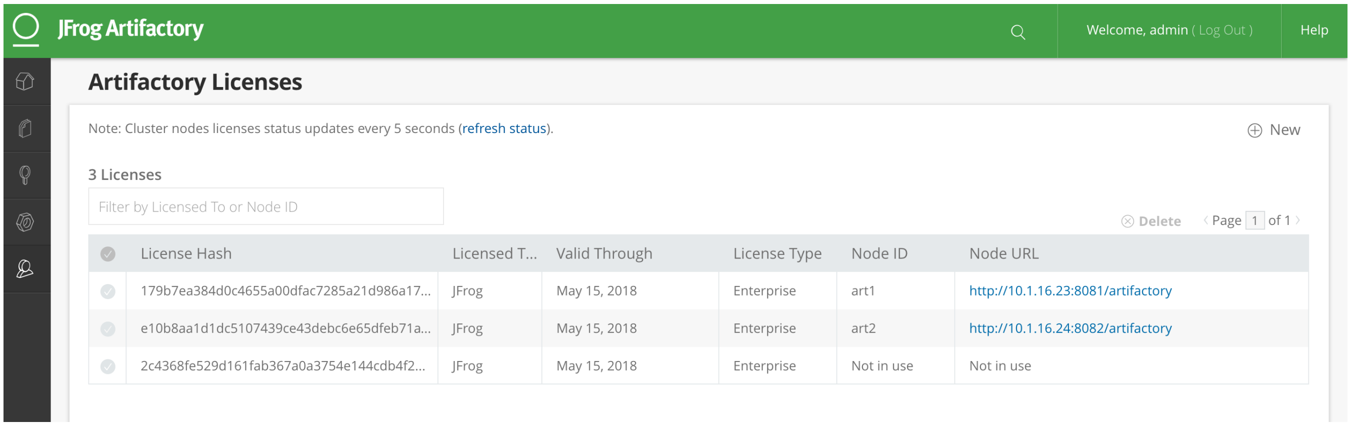
There are three ways that licenses can be added to an HA cluster:

- Through the UI
- Using the REST API
- Adding them to the primary node's filesystem (for automation).

> ✓ **Specifying multiple licenses**
>
> When specifying multiple licenses, whether in the Artifactory UI, using the REST API or in the `artifactory.cluster.license` file, make sure that the licenses are separated by a newline.

Using the UI

Through the UI, in the Admin module, under **Configuration** | **Artifactory Licenses**, you can view all licenses uploaded to your cluster.



Artifactory Licenses

Note: Cluster nodes licenses status updates every 5 seconds (refresh status).

⊕ New

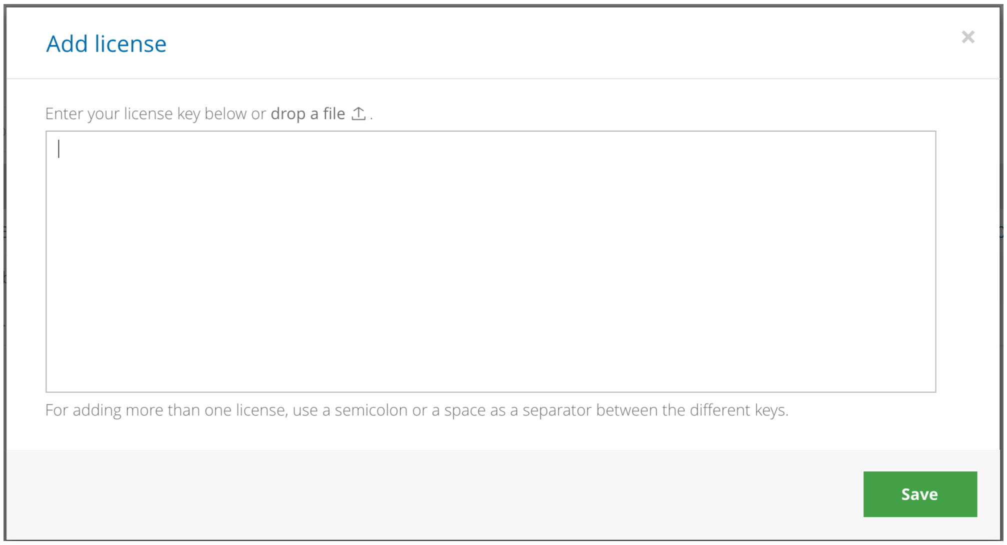
3 Licenses

Filter by Licensed To or Node ID

⊗ Delete ‹ Page 1 of 1 ›

	License Hash	Licensed T...	Valid Through	License Type	Node ID	Node URL
○	179b7ea384d0c4655a00dfac7285a21d986a17...	JFrog	May 15, 2018	Enterprise	art1	http://10.1.16.23:8081/artifactory
○	e10b8aa1d1dc5107439ce43debc6e65dfeb71a...	JFrog	May 15, 2018	Enterprise	art2	http://10.1.16.24:8082/artifactory
○	2c4368fe529d161fab367a0a3754e144cdb4f2...	JFrog	May 15, 2018	Enterprise	Not in use	Not in use

To add licenses to your cluster, click **New** and copy your license key(s) into the **License Key** entry field. You can also simply drag and drop the file containing the license key(s) into the same field. Make sure that each license is separated by a newline.



Add license

Enter your license key below or drop a file.

For adding more than one license, use a semicolon or a space as a separator between the different keys.

Using the REST API

You can also add licenses through the Install License REST API endpoint

Using the Primary Node's Filesystem

To accommodate spinning up Artifactory HA nodes using automation, **before booting up your primary node**, you can place the `artifactory.cluster.license` file in its *`$ARTIFACTORY_HOME/etc`* folder. Upon being booted up, the primary node automatically extracts one of the licenses.

Similarly, upon being started up, each secondary node also automatically extracts one of the remaining available licenses.

License Expiry

Nodes running with a license that is about to expire will automatically be updated with a new license available from the pool. Artifactory administrators can manually delete the expired license from within the UI or using REST API.

Deleting Licenses

A license can be deleted under one the following conditions:

- It is not currently being used,
- There is an alternative license available in the pool. In this case, the node to which the deleted license was attached will automatically be allocated with an alternative license.

> ⚠ **Perpetual License**
>
> Note that Artifactory licenses are perpetual and may continue to activate an Artifactory instance indefinitely, however, an instance running on an expired license may not be upgraded and is not eligible for support.

REST API

You can manage your Artifactory HA licenses using the HA License Information, Install HA Cluster Licenses and Delete HA Cluster License REST API endpoints.

Watch the Screencast