Configuration DSL

Overview

Mission Control comes with a comprehensive set of built-in configuration blocks that are intended as guides to make it easier for you to define configuration scripts using the allowed DSL.

> **Configuration blocks must be in service closures**
>
> Note that the script segments described on this page represent configuration blocks and cannot be used by themselves. Configuration blocks **must** be placed within a **service closure** in order to create an executable script.

Page Contents

- Overview
- Artifactory Configuration Blocks
    - Property Sets
    - Repository Layout
    - Proxies
    - LDAP Settings
    - LDAP Groups
    - Security Settings
- Repository Configuration Blocks
    - Local Repository
    - Remote Repository
    - Virtual Repository
    - Replication
    - Star Topology
        - Push Replication
        - Pull Replication
- Xray Configuration Blocks
    - Link to Artifactory - Binary Manager
    - Watches

Artifactory Configuration Blocks

This section presents configuration blocks that can be used to configure different administrative features of Artifactory services. As with any configuration block, these must be placed within an Artifactory **service closure** as shown below.

Artifactory service closure

```
artifactory('<Artifactory service name>'){

        <configuration blocks>

}
```

Property Sets

The parameters for a `propertySets` configuration block are described below. For more details on these parameters, please refer to Property Sets in the Artifactory User Guide.

propertySets block

```
propertySets {
  propertySet('property_set_name') {
    singleSelect('property_name') {
      defaultValue "value_1"
      value "value_1"
      value "value_2"
    }
    multiSelect('property_name_multi') {
      defaultValue "value_1"
      defaultValue "value_2"
      value "value_1"
      value "value_2"
    }
    anyValue('another_property_name') {
      defaultValue "value_1"
      value "value_1"
    }
  }
}
```

property_set_name	An identifier for this property set. The name must be unique in all Artifactory instances on which it is applied.
singleSelect parameters	
property_name	A unique identifier for the single selection property within this property set.
defaultValue	A default value for the single selection property.
value	The selection options for this property

multiSelect parameters	
property_name_multi	A unique identifier for the multiple selection property within this property set.
defaultValue	The default selected values for the multiple selection property.
value	The selection options for this property
anyValue parameters	
another_property_name	A unique identifier for the free-text property within this property set.
defaultValue	A default value for the free-text property.
value	The selection options for this property

Repository Layout

The parameters for a `repoLayout` configuration block are described below. For more details on these parameters, please refer to Repository Layouts in the Artifactory User Guide.

repoLayout block

```
repoLayouts {
  repoLayout ('repo_layout_name') {
    folderIntegrationRevisionRegExp "SNAPSHOT"
    fileIntegrationRevisionRegExp "SNAPSHOT|(?:(?:[0-9]{8}.[0-9]{6})-(?:[0-9]+))"
    distinctiveDescriptorPathPattern true
    artifactPathPattern "[orgPath]/[module]/[baseRev](-[folderItegRev])/[module]-[baseRev](-[fileItegRev])(-[classifier]).[ext]"
    descriptorPathPattern "[orgPath]/[module]/[baseRev](-[folderItegRev])/[module]-[baseRev](-[fileItegRev])(-[classifier]).pom"
  }
}
```

repo_layout_name	An identifier for this repository layout. The name must be unique in all Artifactory instances on which it is applied.
folderIntegrationRevisionRegExp	A regular expression for folder integration revision.
fileIntegrationRevisionRegExp	A regular expression for file integration revision.
distinctiveDescriptorPathPattern	A distinctive descriptor path pattern is used to recognize descriptor files.
artifactPathPattern	The typical structure in which all module artifacts are expected to be stored.
descriptorPathPattern	The pattern used to recognize descriptor files (such as `.pom` or `ivy.xml` files).

Proxies

The parameters for a `proxies` configuration block are described below. For more details on these parameters, please refer to Managing Proxies in the Artifactory User Guide.

```
proxies {
  proxy('proxy_key') {
    host 'proxy host' // mandatory
    port 8888 // mandatory
    username 'username'
    password 'password'
    defaultProxy false
    ntHost 'NT Host'
    ntDomain 'NT Domain'
    redirectedToHosts (['host1', 'host2', 'host3'])
  }
}
```

proxy_key	The ID of the proxy. Must be unique within the Artifactory instance.
host	The name of the proxy host.
port	The proxy port number
username	The proxy username when authentication credentials are required.
password	The proxy password when authentication credentials are required.
defaultProxy	When true, this proxy will be the default proxy for new remote repositories and for internal HTTP requests.
ntHost	The computer name of the machine (the machine connecting to the NTLM proxy).
ntDomain	The proxy domain/realm name.
redirectedToHosts	An optional list of newline or comma separated host names to which this proxy may redirect requests.

LDAP Settings

The parameters for an `ldap` configuration block are described below. For more details on these parameters, please refer to Managing Security with LDAP in the Artifactory User Guide.

```
ldap {
  settings('settings_name') {
    url 'ldap://myserver:myport/DC=sampledomain,DC=com' // mandatory
    userDnPattern 'uid={0},ou=People'
    emailAttribute 'mail'
    enabled true // default value - true
    autoCreateUser true // default value - true
    search {
      filter '(uid={0})'
      base 'OU=dev,DC=sampledomain,DC=com'
      searchSubTree true
      managerDn 'CN=admin,OU=ops,DC=sampledomain,DC=com'
      managerPassword 'password'
    }
  }
}
```

settings_name	The ID of the LDAP setting. Must be unique within the Artifactory instance being configured.
url	Location of the LDAP server in the following format: *`ldap://myserver:myport/dc=sampledomain,dc=com`*.
userDnPattern	A DN pattern used to log users directly in to the LDAP database. This pattern is used to create a DN string for "direct" user authentication, and is relative to the base DN in the LDAP URL.
emailAttribute	An attribute that can be used to map a user's email to a user created automatically by Artifactory.
enabled	When true, these settings are enabled.
autoCreateUser	When true, Artifactory will automatically create new users for those who have logged in using LDAP, and assign them to the default groups.
filter	A filter expression used to search for the user DN that is used in LDAP authentication.
base	The Context name in which to search relative to the base DN in the LDAP URL. This is parameter is optional.
searchSubTree	When true, enables deep search through the sub-tree of the LDAP URL + Search Base.
managerDn	The full DN of a user with permissions that allow querying the LDAP server.
managerPassword	The password of the user binding to the LDAP server when using "search" authentication.

LDAP Groups

The parameters for an `ldap groups` configuration block are described below. For more details on these parameters, please refer to LDAP Groups in the Artifactory User Guide.

```
ldap {
  groupSettings('static_group_settings_name') {
    settings // 'ldap settings ref'
    staticMapping {
      groupMemberAttribute 'uniqueMember' // mandatory
      groupNameAttribute 'cn' // mandatory
      descriptionAttribute 'description' // mandatory
      filter '(objectClass=groupOfNames)' // mandatory
      searchBase ''
      searchSubTree true
    }
  }
  groupSettings('dynamic_group_settings_name') {
    settings // 'ldap settings ref'
    dynamicMapping {
      groupMemberAttribute 'uniqueMember' // mandatory
      groupNameAttribute 'cn' // mandatory
      descriptionAttribute 'description' // mandatory
      filter '(objectClass=groupOfNames)' // mandatory
      searchBase ''
      searchSubTree true
    }
  }
  groupSettings('hierarchy_group_settings_name') {
    settings // 'ldap settings ref'
    hierarchyMapping {
      userDnGroupKey 'uniqueMember' // mandatory
      groupNameAttribute 'cn' // mandatory
      descriptionAttribute 'description' // mandatory
      filter '(objectClass=groupOfNames)' // mandatory
    }
  }
}
```

static_group_settings_name	A logical name for a **static** group mapping strategy.
dynamic_group_settings_name	A logical name for a **dynamic** group mapping strategy.
hierarchy_group_settings_name	A logical name for a **hierarchy** group mapping strategy.
settings	The LDAP settings reference.
groupMemberAttribute	The group membership attribute for this LDAP group.
groupNameAttribute	The group name attribute for this LDAP group.
descriptionAttribute	The description attribute for this LDAP group.
filter	A filter expression used to search for the user DN that is used in LDAP authentication.
searchBase	The Context name in which to search relative to the base DN in the LDAP URL.
searchSubTree	When true, enables deep search through the sub-tree of the LDAP URL + Search Base.

Security Settings

The configuration blocks for security settings related to users, groups and permissions are described below. For more details on these parameters, please refer to Configuring Security in the Artifactory User Guide.

```
security {
  users {
    conflictResolutionPolicy "OVERRIDE" // default
    user('name') {
      email 'login_1@jfrog.com'
      password 'passwd_1'
      admin false
      profileUpdatable false
      internalPasswordDisabled false
      groups (['groupA', 'groupB']) // values (['groupA', 'groupB']) are examples. Please set existing
values from the instance
    }
  }

  groups {
    conflictResolutionPolicy "OVERRIDE" // default
    group('name') {
      description 'desc_1'
      autoJoin false
    }
  }

  permissions {
    conflictResolutionPolicy "OVERRIDE" // default
    permission('name') {
      includesPattern '**'
      excludesPattern ''
      anyLocal false
      anyRemote false
      anyDistribution false
      repositories (["local-rep1", "local-rep2"]) // values (["local-rep1", "local-rep2", ...]) are
examples. Please set existing values from the instance
      users {
        userA (['manage', 'delete', 'deploy', 'annotate', 'read']) // value userA - is example. Please set
existing user names from the instance
      }
      groups {
        groupsG1 (['manage', 'delete', 'deploy', 'annotate', 'read']) // value groupsG1 - is example. Please
set existing group names from the instance
      }
    }
  }
}
```

Users block	Creates or updates usersin the instance
conflictResolutionPolicy	Default: OVERRIDE Specifies what to do if any setting in the users block of the configuration script conflicts with an existing value for the specified user. Currently, the only option available is the default OVERRIDE which means that values specified in the configuration script will override any existing values.
email	The user's email address.
password	The user's login password.
admin	When true, this user is an administrator with all the ensuing privileges
profileUpdatable	When true, this user can update their profile details (except for the password. Only an administrator can update the password).
internalPasswordDisabled	When true, disables the fallback of using an internal password when external authentication (such as LDAP) is enabled.
groups	Specifies the groups to which this user should belong

Groups block	Creates or updates groups in the instance
conflictResolutionPolicy	Default: OVERRIDE Specifies what to do if any setting in the groups block of the configuration script conflicts with an existing value for the specified user. Currently, the only option available is the default OVERRIDE which means that values specified in the configuration script will override any existing values.
description	A free text description for the group.
autoJoin	When true, any new users defined in the system are automatically assigned to this group.
Permissions block	Creates or updates permission targets in the instance.
conflictResolutionPolicy	Default: OVERRIDE Specifies what to do if any setting in the permissions block of the configuration script conflicts with an existing value for the specified user. Currently, the only option available is the default OVERRIDE which means that values specified in the configuration script will override any existing values.
includesPattern/ excludesPattern	"Ant-like" expressions that specify repositories and paths to be included or excluded from the permission target
anyLocal	When true, all local repositories are included in the permission target.
anyRemote	When true, all remote repositories are included in the persmission target.
anyDistribution	When true, all distribution repositories are included in the permission target.
repositories	Specific repositories on which to apply the permission target.
users	The users on whom to apply the permission target and the corresponding permissions they are given.
groups	The groups on which to apply the permission target and the corresponding permissions they are given.

Repository Configuration Blocks

This section presents all the configuration blocks that may be used to configure Artifactory repositories.

Local Repository

The parameters for a `localRepository` block are described below. For more details on these parameters, please refer to Common Settings and Local Repositories in the Artifactory User Guide.

localRepository block

```
localRepository('repository-key') {
  description "Public description"
  notes "Some internal notes"
  includesPattern "**/*" // default
  excludesPattern "" // default
  repoLayoutRef "maven-2-default"
  packageType "generic" // "maven" | "gradle" | "ivy" | "sbt" | "nuget" | "gems" | "npm" | "conan" | "helm" |
                                                // "bower" | "debian" | "pypi" | "docker" | "vagrant" |
"gitlfs" | "yum" | "generic"
  debianTrivialLayout false
  checksumPolicyType "client-checksums" // default | "server-generated-checksums"
  handleReleases true // default
  handleSnapshots true // default
  maxUniqueSnapshots  0 // default
  snapshotVersionBehavior "unique" // "non-unique" | "deployer"
  suppressPomConsistencyChecks false // default
  blackedOut false // default
  propertySets // (["ps1", "ps2"])
  archiveBrowsingEnabled false
  calculateYumMetadata false
  yumRootDepth 0
  xrayIndex false
  blockXrayUnscannedArtifacts false
  xrayMinimumBlockedSeverity "" // "Minor" | "Major" | "Critical"
  enableFileListsIndexing false //default
  yumGroupFileNames ""
  dockerApiVersion "V2" // default
  maxUniqueTags 0 // default
  optionalIndexCompressionFormats "" // "bz2", "lzma", "xz"
}
```

repository-key	The Repository Key is a mandatory identifier for the repository and must be unique within an Artifactory instance. It cannot begin with a number or contain spaces or special characters. For local repositories we recommend using a "-local" suffix (e.g. "libs-release-local").
description	A free text field that describes the content and purpose of the repository.
notes	A free text field to add additional notes about the repository. These are only visible to the Artifactory administrator and to Mission Control.
includesPattern and excludesPattern	These parameters provide a way to filter out specific repositories when trying to resolve the location of different artifacts.
repoLayoutRef	Sets the layout that the repository should use for storing and identifying modules. The layout should correspond to the value set in the packageType property.
packageType	The repository's package type.
debianTrivialLayout	Only valid if `packageType` is set to Debian. If true, the Debian repository will use the Trivial layout.
checksumPolicyType	Only valid if for Maven, Gradle, Ivy and SBT repositories. Determines how Artifactory behaves when a client checksum for a deployed resource is missing or conflicts with the locally calculated checksum.
handleReleases	If true, users will be able upload Release artifacts to this repository
handleSnapshots	If true, users will be able upload Snapshot artifacts to this repository
maxUniqueSnapshots	Specifies the maximum number of unique snapshots of the same artifact that should be stored. A value of 0 (default) indicates that there is no limit on the number of unique snapshots.
snapshotVersionBehavior	Artifactory supports centralized control of how snapshots are deployed into a repository, regardless of end user-specific settings. This can be used to guarantee a standardized format for deployed snapshots within your organization.

suppressPomConsistencyChecks	If true, Artifactory will reject a deployment in which the `groupId:artifactId:version` set in the path conflicts with the deployed path.
blackedOut	If true, Artifactory will ignores this repository when trying to resolve, download or deploy artifacts.
propertySets	Defines the property sets that will be available for artifacts stored in this repository.
archiveBrowsingEnabled	If true, allows uses to view archive file contents (e.g., Javadoc browsing, HTML files) directly from Artifactory.
calculateYumMetadata	Only valid for YUM repositories. If true, YUM metadata calculation will be automatically triggered for the events described in Triggering RPM Metadata Updates.
yumRootDepth	Only valid for YUM repositories. Informs Artifactory under which level of directory to search for RPMs and save the `repodata` directory.
xrayIndex	If true, this repository should be indexed by JFrog Xray connected to the Artifactory instance.
blockXrayUnscannedArtifacts	If true, artifacts that have not yet been scanned by the connected JFrog Xray will be blocked for download.
xrayMinimumBlockedSeverity	The minimum severity of an issue detected for an artifact to be blocked for download.
enableFileListsIndexing	For an RPM repository, this field specifies if the RPM file lists metadata file should be indexed by Artifactory or not.
dockerApiVersion	
maxUniqueTags	
optionalIndexCompressionFormats	Cannot be an empty string

Remote Repository

The parameters for a remote`Repository` block are described below. For more details on these parameters, please refer to Common Settings and Remote Repositories in the Artifactory User Guide.

remoteRepository block

```
remoteRepository('repository-key') {
  url "http://host:port/some-repo"
  username "remote-repo-user"
  password "pass"
  proxy "proxy1"
  description "Public description"
  notes "Some internal notes"
  includesPattern "**/*" // default
  excludesPattern "" // default
  packageType "generic" // "maven" | "gradle" | "helm" | "ivy" | "sbt" | "nuget" | "gems" | "npm" | "bower"
| "debian" | "pypi" | "docker" | "yum" | "vcs" | "p2" | "generic"
  remoteRepoChecksumPolicyType "generate-if-absent" // default | "fail" | "ignore-and-generate" | "pass-thru"
  handleReleases true // default
  handleSnapshots true // default
  maxUniqueSnapshots  0 // default
  suppressPomConsistencyChecks false // default
  offline false // default
  blackedOut false // default
  storeArtifactsLocally true // default
  socketTimeoutMillis 15000
  localAddress "123.123.123.123"
  retrievalCachePeriodSecs 43200 // default
  failedRetrievalCachePeriodSecs 30 // default
  missedRetrievalCachePeriodSecs 7200 // default
  unusedArtifactsCleanupEnabled false // default
  unusedArtifactsCleanupPeriodHours 0 // default
  fetchJarsEagerly false // default
  fetchSourcesEagerly false // default
  shareConfiguration false // default
  synchronizeProperties false // default
  propertySets // (["ps1", "ps2"])
  allowAnyHostAuth false // default
  enableCookieManagement false // default
  xrayIndex false
  blockXrayUnscannedArtifacts false
  xrayMinimumBlockedSeverity "" // "Minor" | "Major" | "Critical"
  enableFileListsIndexing false //default
  blockMismatchingMimeTypes false // default
  bowerRegistryUrl "https://registry.bower.io" //default
  bypassHeadRequests false // default
  clientTlsCertificate "" // default
  composerRegistryUrl "https://packagist.org" // default
  assumedOfflinePeriodSecs 300 // default
  hardFail false // default
  repoLayoutRef // "maven-2-default"
  vcsGitDownloadUrl "" // default
  vcsGitProvider "GITHUB" // default "BITBUCKET" | "OLDSTASH" | "STASH" | "ARTIFACTORY" | "CUSTOM"
  vcsTpe "GIT" // default
}
```

repository-key	Please refer to the description for this parameter in a Local Repository block.
url	The URL for the remote repository. Currently only HTTP and HTTPS URLs are supported.
username	The username that should be used for HTTP authentication when accessing the remote proxy.
password	The password that should be used for HTTP authentication when accessing the remote proxy.
proxy	If the organization in which the Artifactory instance is hosted requires users to go through a proxy to access a remote repository, this parameter lets you select the corresponding **Proxy Key.**
description	Please refer to the description for this parameter in a Local Repository block.
notes	Please refer to the description for this parameter in a Local Repository block.

includesPattern	Please refer to the description for this parameter in a Local Repository block.
excludesPattern	Please refer to the description for this parameter in a Local Repository block.
packageType	Please refer to the description for this parameter in a Local Repository block.
remoteRepoChecksumPolicyType	Specifies how the Artifactory instance should behave when a client checksum for a remote resource is missing or conflicts with the locally calculated checksum.
handleReleases	Please refer to the description for this parameter in a Local Repository block.
handleSnapshots	Please refer to the description for this parameter in a Local Repository block.
maxUniqueSnapshots	Please refer to the description for this parameter in a Local Repository block.
suppressPomConsistencyChecks	Please refer to the description for this parameter in a Local Repository block.
offline	If true, the repository will be considered offline and no attempts will be made to fetch artifacts from it.
blackedOut	Please refer to the description for this parameter in a Local Repository block.
storeArtifactsLocally	If true, artifacts from the repository will be cached locally. If not set, direct repository-to-client streaming is used.
socketTimeoutMillis	The time that the Artifactory instance should wait for both a socket and a connection before giving up on an attempt to retrieve an artifact from a remote repository.
localAddress	When working on multi-homed systems, this parameter lets you specify which specific interface (IP address) should be used to access the remote repository.
retrievalCachePeriodSecs	Defines how long before the Artifactory instance should check for a newer version of a requested artifact in a remote repository. A value of 0 means that Artifactory will always check for a newer version.
missedRetrievalCachePeriodSecs	If a remote repository is missing a requested artifact, Artifactory will return a "404 Not found" error. This response is cached for the period of time specified by this parameter. During that time, Artifactory will not issue new requests for the same artifact. A value of 0 means that the response is not cached and Artifactory will always issue a new request when demanded.
unusedArtifactsCleanupPeriodHours	Specifies how long an unused artifact will be stored in the Artifactory instance before it is removed. A value of 0 means that the artifact is stored indefinitely.
fetchJarsEagerly	If true, if a POM is requested, the Artifactory instance attempts to fetch the corresponding jar in the background. This will accelerate first access time to the jar when it is subsequently requested.
fetchSourcesEagerly	If true, if a binaries jar is requested, the Artifactory instance attempts to fetch the corresponding source jar in the background. This will accelerate first access time to the source jar when it is subsequently requested.
synchronizeProperties	Only valid for Smart Remote Repositories. If true, properties for artifacts that have been cached in the repository will be updated if they are modified in the artifact hosted at the remote Artifactory instance.
propertySets	Please refer to the description for this parameter in a Local Repository block.
allowAnyHostAuth	If true, allows using the repository credentials on any host to which the original request is redirected.
enableCookieManagement	If true, the repository will allow cookie management to work with servers that require them.

xrayIndex	Please refer to the description for this parameter in a Local Repository block.
blockXrayUnscannedArtifacts	Please refer to the description for this parameter in a Local Repository block.
xrayMinimumBlockedSeverity	Please refer to the description for this parameter in a Local Repository block.
enableFileListsIndexing	For an RPM repository, this field specifies if the RPM file lists metadata file should be indexed by Artifactory or not.
allowAnyHostAuth	
bowerRegistryUrl	
blockMismatchingMimeTypes	
clientTlsCertificate	
composerRegistryUrl	
assumedOfflinePeriodSecs	
byPassHeadRequests	
hardFail	
repoLayoutRef	
vcsGitDownloadUrl	
vcsGitProvider	
vcsType	

Virtual Repository

The parameters for a virtual`Repository` block are described below. For more details on these parameters, please refer to Common Settings and Virtual Repositories in the Artifactory User Guide.

virtualRepository block

```
virtualRepository('repository-key') {
  repositories (["local-rep1", "local-rep2"]) // values (["local-rep1", "local-rep2", ...]) are examples. Please set existing values from the instance
  description "Public description"
  notes "Some internal notes"
  includesPattern "**/*" // default
  excludesPattern "" // default
  packageType "generic" // "maven" | "gradle" | "helm" | "ivy" | "sbt" | "nuget" | "gems" | "npm" | "bower" | "pypi" | "p2" | "generic"
  debianTrivialLayout false
  artifactoryRequestsCanRetrieveRemoteArtifacts false
  keyPair "keypair1" //value "keypair1" is example. Please set existing value from the instance
  pomRepositoryReferencesCleanupPolicy "discard_active_reference" // default | "discard_any_reference" | "nothing"
  defaultDeploymentRepo "local-rep1"
  forceMavenAuthentication false // default
}
```

repository-key	Please refer to the description for this parameter in a Local Repository block.
repositories	The list of repositories that should be aggregated in this virtual repository.
description	Please refer to the description for this parameter in a Local Repository block.
notes	Please refer to the description for this parameter in a Local Repository block.
includesPattern	Please refer to the description for this parameter in a Local Repository block.
excludesPattern	Please refer to the description for this parameter in a Local Repository block.
packageType	Please refer to the description for this parameter in a Local Repository block.
debianTrivialLayout	Please refer to the description for this parameter in a Local Repository block.
artifactoryRequestsCanRetrieveRemoteArtifacts	If true, the virtual repository should search through remote repositories when trying to resolve an artifact requested by another Artifactory instance.
keyPair	A named key-pair to use for automatically signing artifacts.
pomRepositoryReferencesCleanupPolicy	This setting gives you the ability to ensure Artifactory is the sole provider of Artifacts in your system by automatically cleaning up the POM file.
forceMavenAuthentication	

Replication

A replication block is used for creating push/pull replication. This block should always be nested inside a repository block: `localRepository` for push replication, `remoteRepository` for pull replication or a `repository` block for both.

replication block

```
localRepository("example") {
  replication(<REPLICATION_TARGET>) {
      cronExp "0 0/9 14 * * ?"
      socketTimeoutMillis 15000
      username "remote-repo-user"
      password "pass"
      proxy //"proxy-ref"
      enableEventReplication true
      enabled true
      syncDeletes false
      syncProperties true
      clientTlsCertificate ""
    }
}
```

REPLICATION_TARGET	Target local repository for push replication. There are two ways to provide a target repository: 1. **Target repository URL** - the URL of the target local repository (String) 2. **Target Artifactory object** - the service object the target repository belongs to. This applies **only** when creating a new repository. Mission control will select the repository about to be created on the target instance within the current configuration context (current configuration action). An instance object can be obtained from the context variables or by asking for user input of type ARTIFACTORY. For pull replication there is no need to set the target repository
cronExp	Define the replication task schedule using a valid cron expression
socketTimeoutMillis	The network timeout in milliseconds to use for remote operations
username	The HTTP authentication username
password	The HTTP authentication password
proxy	The key of a proxy configuration to use when communicating with the remote instance
enableEventReplication	When set, event-based push replication is enabled
enabled	When set to true, this replication will be enabled
syncDeletes	When set, items that were deleted remotely should also be deleted locally
syncProperties	When set, the task also synchronizes the properties of replicated artifacts
clientTlsCertificate	The SSL/TLS certificate used for authentication

Star Topology

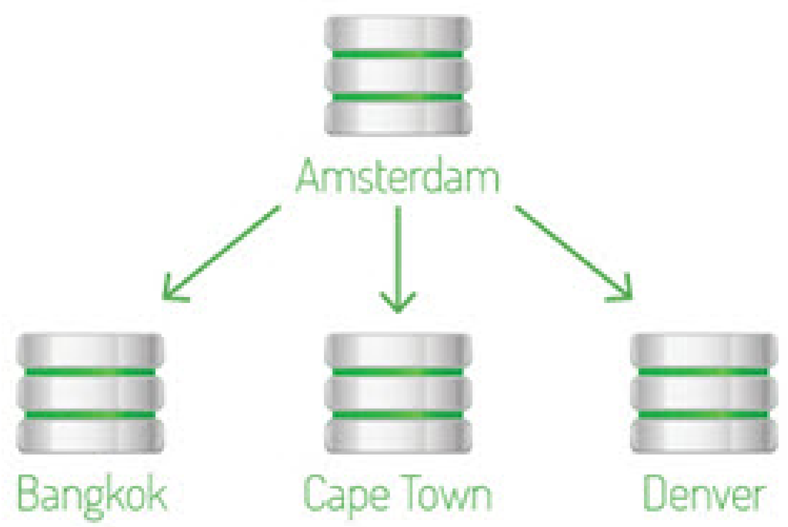
Mission Control provides built-in configuration blocks that make it very easy to create replication relationships between different Artifactory services in a star topology. This is done using the following two key words to create push replication and pull replication configurations respectively:

- starPush
- starPull

In each case the DSL may include a Replication block through which you can configure the parameters governing the replication process.

Push Replication

The example below shows how to create a star topology using push replication.



Star topology using multi-push replication

This example shows:

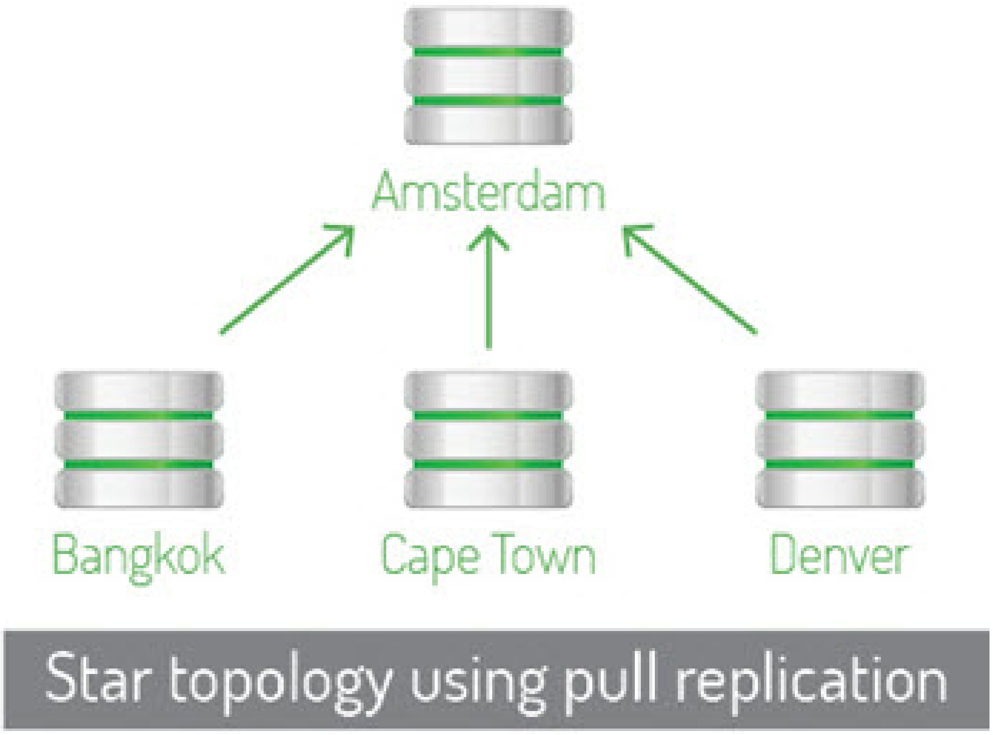
- An Artifactory service named "Amsterdam" with two local repositories, "maven-local-1" and "maven-local-2"
- The configuration block will create or update a push replication relationship between "maven-local-1" and "maven-local-2" in "Amsterdam" to the corresponding repositories in each of the Artifactory services named "Bangkok", "Cape Town" and "Denver".
- If "maven-local-1" or "maven-local-2" do not already exist in any of the target Artifactory services, Mission Control will create them
- The replication will be governed by the parameters specified (`socketTimeoutMillis, enableEventReplication`, etc.)

starPush replication block

```
artifactory('Amsterdam') {
        repository("maven-local-1", "maven-local-2") {
                  starPush('Bangkok', 'Cape Town', 'Denver'){
                      socketTimeoutMillis 60000
                      enableEventReplication true
                      cronExp '0 0/9 14 * * ?'
                      syncDeletes false
                      syncProperties false
                     syncStatistics false
            }
        }
}
```

Pull Replication

The example below shows how to create a star topology using pull replication.



This example shows:

- An Artifactory service named "Amsterdam" with two repositories, "maven-local" and "maven-virtual"
- The configuration block will create or update a pull replication relationship in corresponding repositories in each of the Artifactory services named "Bangkok", "Cape Town" and "Denver" to pull replicate from the "maven-local" and "maven-virtual" repositories in "Amsterdam" .
- If "maven-local" or "maven-remote" don't exist in "Bangkok", "Cape Town" or "Denver", Mission Control will create them
- The replication will be governed by the parameters specified (`socketTimeoutMillis, enableEventReplication`, etc.)

starPull replication block

```
artifactory('Amsterdam') {
        repository("maven-local", "maven-virtual") {
                  starPull('Bangkok', 'Cape Town', 'Denver'){
                      socketTimeoutMillis 60000
                      enableEventReplication true
                      cronExp '0 0/9 14 * * ?'
                      syncDeletes false
                      syncProperties false
                     syncStatistics false
            }
        }
}
```

Xray Configuration Blocks

This section presents configuration blocks that can be used to configure different administrative features of Xray services. As with any configuration block, these must be placed within an Xray **service closure** as shown below.

Artifactory service closure

```
xray('<Xray service name>'){

        <configuration blocks>

}
```

Link to Artifactory - Binary Manager

The following is an example of a configuration block that adds an Artifactory service as a binary manager to an Xray service.

If `Artifactory-prod` is already registered as a service in Mission Control, the admin login credentials specified when registering will be provided to Xray when configuring it as the binary manager.

```
{
  binaryManager('Artifactory-prod')
}
```

If `Artifactory-prod` is not registered as a service in Mission Control, you need to provide all properties of the service as follows:

```
{
  binaryManager('Artifactory-prod') {
    url 'http://artifactory.com/artifactory'
    login 'login'
    password 'password'
  }
}
```

In this case, Mission Control will create an Artifactory service with these properties and then configure the Xray service in the enclosing **service closure** with it as its binary manager.

Watches

Below is an example of a `watch` configuration block. For more details on these parameters, please refer to Watches in the JFrog Xray User Guide.

```
{
  watch('watch') {
    binaryManagerId 'binaryManagerId'
    targetType 'repository'
    description 'description'
    active true
    postActions {
      emails(['email1@email.com', 'email2@email.com'])
      slacks 'slacks'
      webhooks(['webhook1', 'webhook2'])
      failBuild true
    }
    filters {
      filter {
        type 'license_black'
        value 'value1'
      }
      filter {
        type 'regex'
        value 'value1'
      }
    }
    repoType 'repoType'
    severity 'severity'
    system true
    targetName 'targetName'
    temp true
  }
}
```