

Nexus anonymous access removal- Developer guideline

CHANGE LOG

Date	Author	Change
2022-11-02	Méndez Guerrero, Juan	After Deployment Changes
2022-10-11	Méndez Guerrero, Juan	Add Dockerfile instructions
2022-10-03	Méndez Guerrero, Juan	Change Date Extension Update
2022-09-27	Méndez Guerrero, Juan	Initial Document

Contents

INTRODUCTION	4
NEXUS RESOURCES	5
JENKINS SHARED LIBRARIES	6
CHANGE REQUIRED	8
JAVA	8
MAVEN	8
GRADLE	8
LINUX/MAC	9
WINDOWS	9
PYTHON	9
NPM, YARN, SFDX AND VLOCITY	9
HELM	9
NUGET	10
DOCKER.....	10
DOCKERFILE	10
KUBERNETES	12
SUPPORT	13

Introduction

To ensure application security and compliance with Lumen internal policies, Enterprise Internal Tools Team (EITT) will **remove anonymous access to Nexus** ([Nexus Repository Manager \(corp.intranet\)](#)) **on November 2nd, 2022**, under **GCR0178715**. This will affect internal applications that require authentication or RBAC (Role Base Access Control) where deployment information to production is stored.

This document aims to provide a solid guideline for teams to help them prepare for this change.

***Important Note:** The anonymous access removal was originally scheduled for October 18th, 2022. However, an extension has been approved to better accommodate team's needs, especially those working in the BrightSpeed effort. We want to guarantee that these teams have enough time to check, review, implement and test changes before the target date. Please note that no further extensions will be approved. We encourage all teams to take immediate action before November 2nd to avoid issues after the access is removed.

Nexus Resources

All Nexus docs and resources (including access and tool basics) can be found on the following link:

Nexus onboarding and basics:

https://mysupportdesk.service-now.com/msd/?id=kb_article&sys_id=36a75f9b1b7254d411de0f69cc4bcb1b

Jenkins Shared Libraries

Jenkins shared libraries will be prepared for the change, ensuring that all the libraries that include usage of Nexus will have credentials preloaded with global credentials used in Jenkins. For the build libraries on different languages, further instructions are included within this document.

The DevOps Enablement Team strongly recommends the use of the **stable release**. For more information please read: <https://github.com/CenturyLink/jsl-jenkins-shared-library#how-to-use-jenkins-shared-libraries>

In case you use a dated release, please ensure that your pipelines are updated to release/20221015.0.0 after October 15th, 2022. Libraries are updated as part of release/20220928.0.0 for those that want to perform any testing prior to October 15th release.

Since October 1st changes have been included as part of stable release. Additional fixes will also be (automatically) included in the release.

***Note:** The DevOps Enablement Team cannot guarantee that all authentication features will be included without bugs in releases prior to date October 15th or stable release. Teams are accountable for updating them. All changes and updates will be posted in the [DevOps Community of Practice General Channel](#). Please use it as well to report any issues in case you are testing new changes.

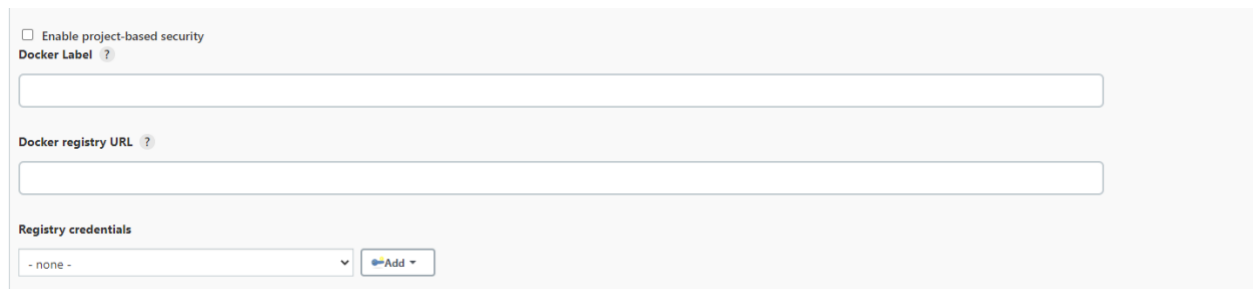
Only Jenkins Prod 01 and 02 are supported under this change. In case you haven't migrated, please create a Jenkins support request in the following link: [HOW TO Request access or support to tools - CenturyLink/jsl-jenkins-shared-library Wiki \(github.com\)](#)

Please note that Docker will be authenticated to nexusprod.corp.intranet:4567 by default. Therefore, you must change all repository references to port 4567 (e.g., 4566 to 4567), included but not limited to pipelines, Dockerfiles, Kubernetes templates.

Since the moment of the change, any reference to Docker will go to Nexus. Thus, in case you want to use a different registry you need to use the following structure:

```
agent {
  docker {
    image IMAGE
    label 'Docker-enabled'
    registryUrl REGISTRY_URL
    registryCredentialsId LOGIN_CREDENTIALS_IF_REQUIRED
  }
}
```

Additionally, you can set your own configuration at folder level, which will apply to all your pipelines



The screenshot shows the Jenkins configuration interface for Docker settings. It includes a checkbox for "Enable project-based security", a "Docker Label" field with a help icon, a "Docker registry URL" field with a help icon, and a "Registry credentials" section with a dropdown menu showing "- none -" and an "Add" button.

The default values are:

Docker Label: Docker-enabled

Docker registry URL: <https://nexusprod.corp.intranet:4567>

Registry Credentials: CICD Nexus Creds

We strongly recommend you configure them, even in the case they are the global ones.

In any case, if you change this to other registry, please remember that your pipelines need to have the Nexus information to work:

```
agent {  
  docker {  
    image nexusprod.corp.intranet:4567/test:latest  
    label 'Docker-enabled'  
    registryUrl https://nexusprod.corp.intranet:4567  
    registryCredentialsId 'nexus-cicd-creds'  
  }  
}
```

Last, please check the below document for more information on onboarding basics.

DevOps Pipeline Onboarding Manual

https://centurylink.sharepoint.com/sites/XLR8/SitePages/Standardized_Pipeline_Onboarding_Process.aspx

Change Required

***Note:** For any Nexus account/access request please check [Nexus Resources](#)

JAVA

For General Information please check [CenturyLink/jsl-catalog-java-example \(github.com\)](#)

Maven

Maven users who are currently using the below Nexus repositories or any other target inside Nexus, must ensure that the settings include the needed credentials

<https://nexusprod.corp.intranet:8443/repository/maven-public>

<https://nexusprod.corp.intranet:8443/repository/maven-snapshots>

<https://nexusprod.corp.intranet:8443/repository/maven-releases>

The example settings are [jsl-catalog-java-example/settings.xml at master · CenturyLink/jsl-catalog-java-example \(github.com\)](#). To load the credentials please use the config file provider:

<https://jenkinsprod.corp.intranet:8443/job/DEVOPS-CICD/job/CATALOG/job/JAVA/configfiles/editConfig?id=MAVEN-SETTINGS>:

Also note that any relation in your pom to the repositories must include the credentials as well. For instance, if the id of the repositories in your pom is NEXUS, you need to include the server NEXUS with the respective credentials in the config created for any given purpose. Global credentials are created for this purpose.

Last, local build also requires authentication. For that reason, you need to check your settings and update the information with your credentials.

Gradle

For Gradle users, we have preloaded the credentials variables in all the related JSL. If you are using a direct sh command, you might need to move to the official libraries.

On the other hand, your Gradle configuration needs to be updated for all the Maven repositories used, e.g.:

```
maven {  
    url = "https://nexusprod.corp.intranet:8443/repository/maven-snapshots/"  
    credentials {  
        username = System.getenv("DOCKER_REGISTRY_USR") ?: "MISSING_USER"  
        password = System.getenv("DOCKER_REGISTRY_PSW") ?:  
"MISSING_PASSWORD"  
    }  
    authentication {  
        basic(BasicAuthentication)  
    }  
}
```

In case you need to perform a local compilation, you need to load your credentials as an environment variable, e.g.:

Linux/MAC

```
export DOCKER_REGISTRY_USR = 'YOUR_USER'
export DOCKER_REGISTRY_PSW = 'YOUR_TOKEN'
```

Windows

```
SET DOCKER_REGISTRY_USR = 'YOUR_USER'
SET DOCKER_REGISTRY_PSW = 'YOUR_TOKEN'
```

All executions accept the pass of environment variables as any JAVA standard execution which can be used as well.

PYTHON

For general information please read [CenturyLink/jsl-catalog-python-django-example \(github.com\)](https://github.com/CenturyLink/jsl-catalog-python-django-example)

Python users must create the pipconf. Please see: https://jenkinsprod.corp.intranet:8443/job/DEVOPS-CICD/job/CATALOG/job/DJANGO_PYTHON/configfiles/editConfig?id=PIPCONF and https://jenkinsprod.corp.intranet:8443/job/DEVOPS-CICD/job/CATALOG/job/DJANGO_PYTHON/configfiles/editConfig?id=PYPIRC

Likewise Maven, you will need to keep your local pypirc updated. Here's an example:

```
[distutils]
index-servers =
    ctlpypi
    pypi

[pypi]
repository: https://nexusprod.corp.intranet:8443/repository/pypi-proxy/
username: <NEXUS_CREDS_USR>
password: <NEXUS_CREDS_PSW>

[ctlpypi]
repository: https://nexusprod.corp.intranet:8443/repository/CTLPyPI/
username: <NEXUS_CREDS_USR>
password: <NEXUS_CREDS_PSW>
```

NPM, YARN, SFDX and VLOCITY

For users of these tools, or related, no change is required. We will preload the authentication via "npm config set" for npm-group and CTLNPM repository.

For local compilation it is recommended to set your credentials via npm config set

```
npm config set //nexusprod.corp.intranet:8443/repository/npm-group/:_auth
<TOKEN>
npm config set //nexusprod.corp.intranet:8443/repository/CTLNPM/:_auth <TOKEN>
```

Helm

For Helm, our wrapper already supports credentials jslHelmWrapper, in case you use sh directly with Helm, you might need to migrate to the wrapper.

For your local configuration, you need to add your repo:

```
helm repo add nexus https://nexusprod.corp.intranet:8443/repository/CTLHELM/ --
username <USER> --password <PASSWORD>
```

NUGET

Nuget users may already be using API TOKEN. If that is not the case, please check the Nexus resources at the beginning of this document.

For more information on how to add your credentials please read <https://learn.microsoft.com/en-us/nuget/consume-packages/consuming-packages-authenticated-feeds>

Docker

All Docker libraries (as well as Jenkins) will have a mechanism to authenticate to nexusprod.corp.intranet:4567

Please note that all the rest of ports won't be preauthenticated

For your local configuration please use:

- docker login: nexusprod.corp.intranet:4567
- Enter your username and password.

Dockerfile

If you are using calls inside your Dockerfile to Nexus, such as:

1. AppDynamics
2. Tools Download
3. Configuration Download

You require to change the different calls to provide the user and password. These ones will be preloaded as part of your docker build with BUILDKIT and secrets. All needed libraries are included in stable release.

Please update your Dockerfiles according to the following instructions:

ADD

In case you are calling ADD to include a file from nexus:

```
ADD
https://nexusprod.corp.intranet:8443/repository/CTLRaw/appdynamics/AppServerAgent-
${APPDYNAMICS_VERSION}.zip /opt/appdynamics.zip
```

You need to include an ARG with the credentials and specifically include the user and password

```
ARG DOCKER_REGISTRY_USR
ARG DOCKER_REGISTRY_PSW
ADD
https://${DOCKER_REGISTRY_USR}:${DOCKER_REGISTRY_PSW}@nexusprod.corp.intranet:8443/re
pository/CTLRaw/appdynamics/AppServerAgent-${APPDYNAMICS_VERSION}.zip
/opt/appdynamics.zip
```

RUN

For run you can mount the secret with netrc and use it for your calls when using curl or specifically add credentials as explained before.

If you have

```
RUN curl -sL
https://nexusprod.corp.intranet:8443/repository/CTLRaw/appdynamics/AppServerAgent-
20.11.1.31618.zip --output /tmp/appdynamics.zip
```

Or

```
RUN wget --progress=dot:giga --no-check-certificate
'https://nexusprod.corp.intranet:8443/repository/CTLRaw/appdynamics/AppServerAgent-
20.11.1.31618.zip' -O /tmp/appdynamics2.zip
```

Change to

```
RUN --mount=type=secret,id=nexus_netrc curl --netrc-file /run/secrets/nexus_netrc -sL
https://nexusprod.corp.intranet:8443/repository/CTLRaw/appdynamics/AppServerAgent-
20.11.1.31618.zip --output /tmp/appdynamics.zip
```

```
ARG DOCKER_REGISTRY_USR
ARG DOCKER_REGISTRY_PSW
RUN wget --user ${DOCKER_REGISTRY_USR} --password ${DOCKER_REGISTRY_PSW} --
progress=dot:giga --no-check-certificate
'https://nexusprod.corp.intranet:8443/repository/CTLRaw/appdynamics/AppServerAgent-
20.11.1.31618.zip' -O /tmp/appdynamics2.zip
```

Other uses

In case you need to include authentication to internal registries such as node or include configuration files, please follow previous instructions for local access and include inside your container.

The environment variables for user and password are DOCKER_REGISTRY_USR and DOCKER_REGISTRY_PSW which by substituted in your configuration file or be mentioned as part of a command as previously stated.

For the cases where you want to include the secrets, you need to have the DOCKER_BUILDKIT environment variable set to 1

Please note that the secret can't be injected in agents built by Dockerfiles. In this case we recommend the use of ARGS, in order to use them please set in environment block:

```
BUILD_DOCKER_CREDENTIALS = credentials('nexus-cicd-creds')
```

Then you can use ARGS as BUILD_DOCKER_CREDENTIALS_USR and BUILD_DOCKER_CREDENTIALS_PSW e.g:

```
agent {
  dockerfile {
    filename 'cicd/docker/Dockerfile'
    label 'docker-enabled'
    additionalBuildArgs "--build-arg DOCKER_CREDENTIALS_USR=${env.BUILD_DOCKER_CREDENTIALS_USR} --
build-arg DOCKER_CREDENTIALS_PSW=${env.BUILD_DOCKER_CREDENTIALS_PSW}"
  }
}
```

Kubernetes

In general, Kubernetes does not require this access. However, deployments require the download of images. By default, our libraries inject the needed secrets to download the Docker images. If you use a custom image (including those in the JSL but maintained by other teams), please ensure you generate your needed secrets or you use the standard JSL.

In case you want to enable the credentials globally for your service accounts you can follow:

1. Create your general secret with your nexus credentials (substitute the needed variables)

```
kubectl --namespace=${mal_namespace} create secret docker-registry docker-registry-secrets --docker-
server="nexusprod.corp.intranet:4567" --docker-username="${DOCKER_REGISTRY_USR}" --docker-
password="${DOCKER_REGISTRY_PSW}"
```

2. Configure your service accounts to use the newly created secrets

```
kubectl --namespace=${mal_namespace} patch serviceaccount default -p '{"imagePullSecrets": [{"name":
"docker-registry-secrets"}]}'
kubectl --namespace=${mal_namespace} patch serviceaccount ${mal_namespace} -p '{"imagePullSecrets":
[{"name": "docker-registry-secrets"}]}'
```

This will enable a default secret in all the namespaces

Support

Should you require further support or assistance, please contact the DevOps Community of Practice MS Teams Channel: <https://teams.microsoft.com/l/channel/19%3abca94bd5af8e458f8c29c4e30ec42605%40thread.skype/General?groupId=285569f6-f759-4426-bc9d-6410a520fbcf&tenantId=72b17115-9915-42c0-9f1b-4f98e5a4bcd2>

If you need specific Nexus Support please create an issue here: https://mysupportdesk.service-now.com/msd?id=sc_cat_item&sys_id=b19b93e41bf21c5838edfe651a4bcb8a&sysparm_category=e8945f011b0a98501718fe6edd4bcb7b