Discovery 5: Use NSO in Docker

Introduction

In this activity, you will learn how to set up the NSO in Docker environment with GitLab. The main reason behind using Docker for NSO lies in ensuring a consistent, customizable, portable environment, which can be used for development, testing, and production purposes.

After completing this activity, you will be able to meet these objectives:

- Create and build NSO Docker images.
- Start NSO Docker container.

Job Aid

The following job aid is available to help you complete the lab activities:

- This Lab Guide

The following table contains passwords that you might need.

Device	Username	Password
Student-VM	student	1234QWer
NSO application	admin	admin
GitLab	student	1234QWer

Required Resources

The following resources and equipment are required for completing the activities in this lab guide:

- PC or laptop with a web browser
- Access to the internet

Command List

The following are the most common commands that you will need:

Linux Shell:

Command	Comment
source /opt/ncs/ncs-6.1/ncsrc	Source NSO environmental variable in Docker container.
ls|ll	Display contents of the current directory.
cd	Move directly to user home directory.

Command	**Comment**
cd ..	Exit out of current directory.
cd test	Move into the "test" folder, which is a subfolder of the current directory.
cd /home/student	Move into the "nso300" folder by specifying the direct path to it starting from the root of the directory system.
ncs_cli -C	Log in to NSO CLI directly from local server.

NSO CLI:

Command	**Comment**
switch cli	Change CLI style.
show ?	Display all command options for current mode.
configure	Enter configuration mode.
commit	Commit new configuration (configuration mode only command).
show configuration	Display new configuration that has not yet been committed (configuration mode only command).

Makefile commands for Docker environment:

Command	**Comment**
make build	Builds the main NSO Docker image.
make testenv-start	Starts the NSO Docker environment.
make testenv-stop	Stops the NSO Docker environment.
make testenv-build	Recompiles and reloads the NSO packages.
make testenv-cli	Enters the NSO CLI of the NSO Docker container.
make testenv-shell	Enters the Linux shell of the NSO Docker container.
make dev-shell	Enters the Linux shell of the NSO Docker development container.

Command Syntax Reference

This lab guide uses the following conventions for command syntax:

Formatting	**Description and Examples**
show running config	Commands in steps use this formatting.
Example	Type **show running config**
Example	Use the **name** command.
`show running`	Commands in CLI outputs and configurations use this formatting.

Formatting	Description and Examples
`config`	
highlight	CLI output that is important is highlighted.

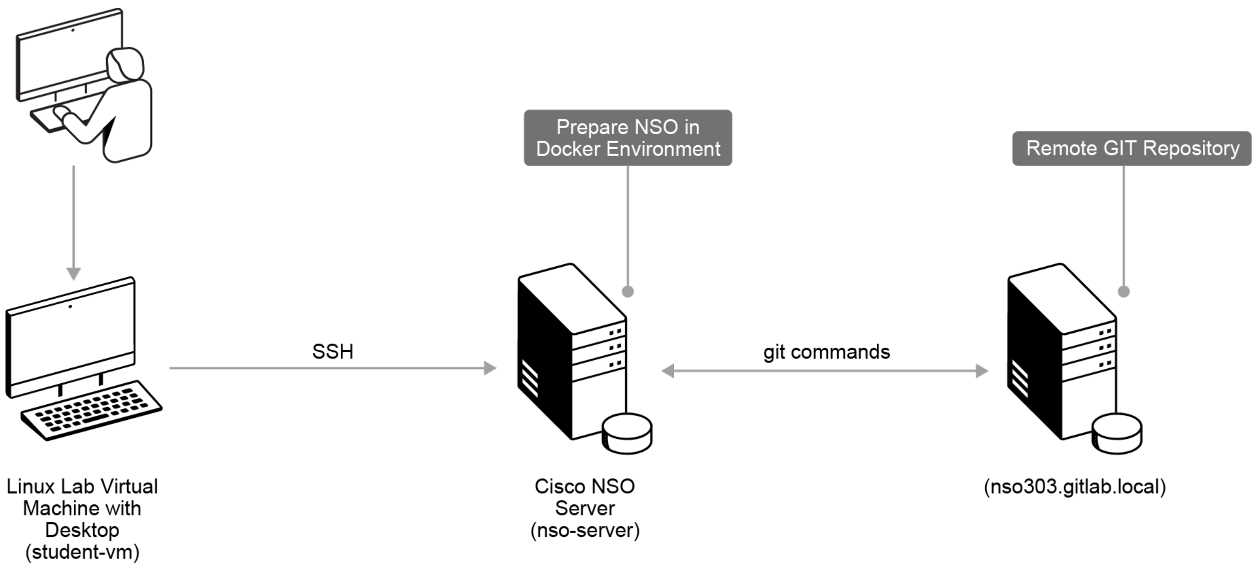
Example	
	```
	student@student-vm:~$ ncs -version
	          6.1
	```

Example	Save your current configuration as the default **startup config**. `Router Name# copy running startup`
brackets ([ ])	Indicates optional element. You can choose one of the options.
Example:	`(config-if)# frame-relay lmi-type {ansi|cisco|q933a}`
italics font	Arguments for which you supply values.
Example	Open file **ip tcp window-size** *bytes*
angle brackets (<>)	In contexts that do not allow italics, arguments for which you supply values are enclosed in angle brackets [<>]. Do not type the brackets when entering the command.
Example	If the command syntax is **ping** <*ip_address*>, you enter ping *10.0.0.102*
string	A non-quoted set of characters. Type the characters as-is.
Example	(config)# **hostname MyRouter**
vertical line (|)	Indicates that you enter one of the choices. The vertical line separates choices. Do not type the vertical line when entering the command.
Example	If the command syntax is **show ip route|arp**, you enter either **show ip route** or **show ip arp**, but not both.

Lab Topology Information

Your lab session is your own personal sandbox. Whatever you do in your session will not be reflected in anyone else's session. There are two topologies. Your lab environment is a Linux server (Student-VM) acting as a jumphost, and a Linux server (NSO-server) acting as an NSO and GitLab server within a Docker container.

Topology

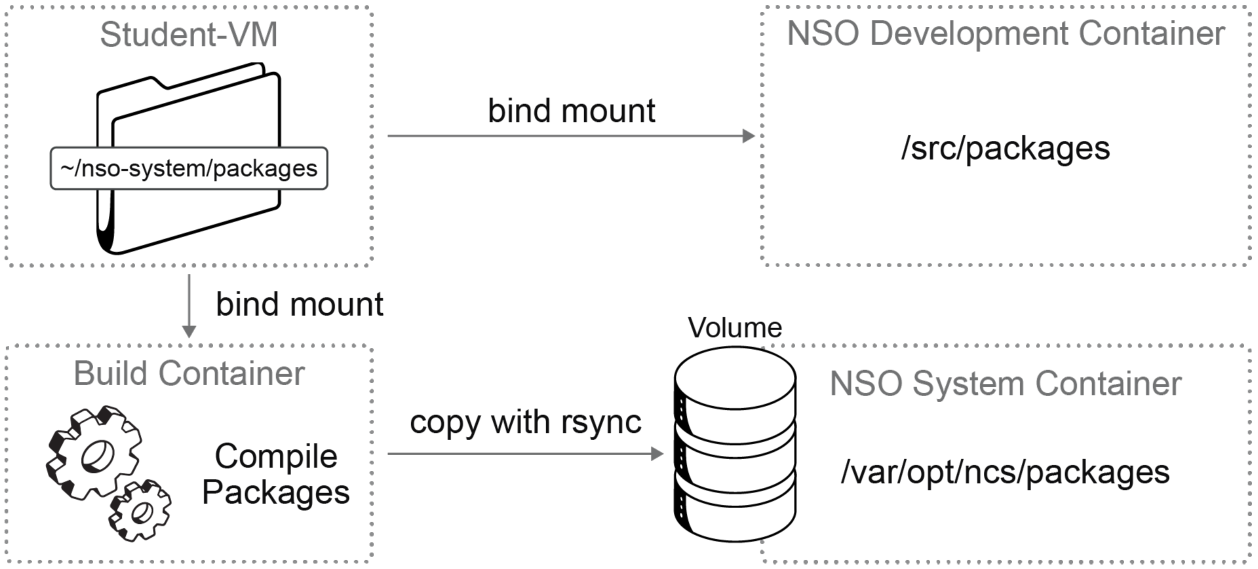

Prepare NSO in Docker Environment
Remote GIT Repository
SSH
git commands
Linux Lab Virtual Machine with Desktop (student-vm)
Cisco NSO Server (nso-server)
(nso303.gitlab.local)

Docker Topology

The following image illustrates the topology of NSO Docker containers in relation to the build process, triggered by the **make testenv-build** command.

The NSO Development container bind mounts the ~/nso-system/packages folder from the nso-server machine to its /src/packages folder.

The NSO System container bind mounts the packages to an intermediate build container, where the packages are recompiled. The recompiled packages are then copied to a volume, mounted on the NSO System container with the rsync tool, which is used to incrementally copy only the files that have been changed.



Task 1: Set Up the NSO in Docker Project and Images

In this task, you will inspect the nso-docker project on GitLab and pull the produced NSO Docker images. These images will then be used as a base for an NSO project.

Activity

Complete these steps:

Step 1

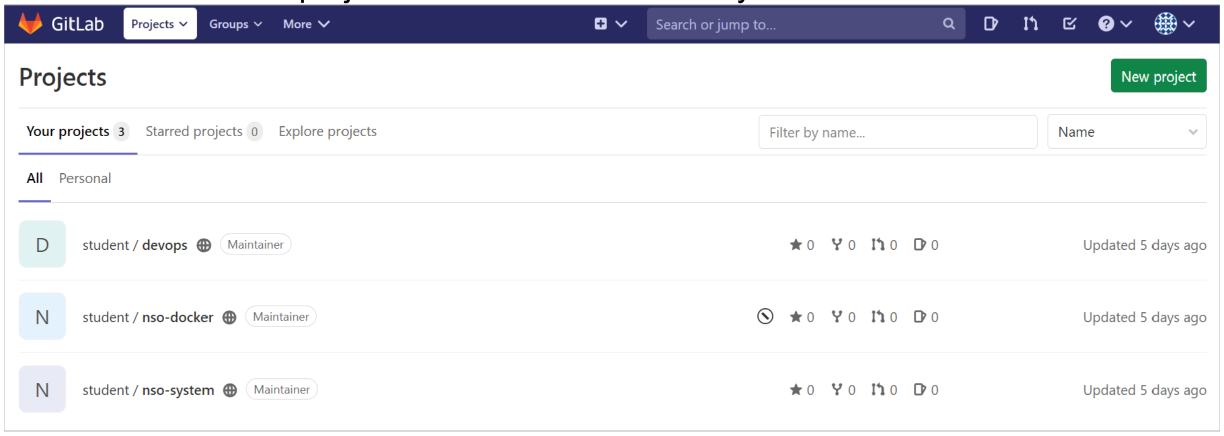
Connect to the Student-VM.

You can connect to the server either by choosing the **Student-VM** from the device list or by clicking the **Student-VM** icon in the topology map.

Step 2

Open a web browser, navigate to the GitLab WebUI, and sign in with the **student** username and **1234QWer** password. The address of this user interface is **http://nso303.gitlab.local**.

Under Projects, you will see a couple of already created projects. In this activity, you will use two of those projects: nso-docker and nso-system.



Projects

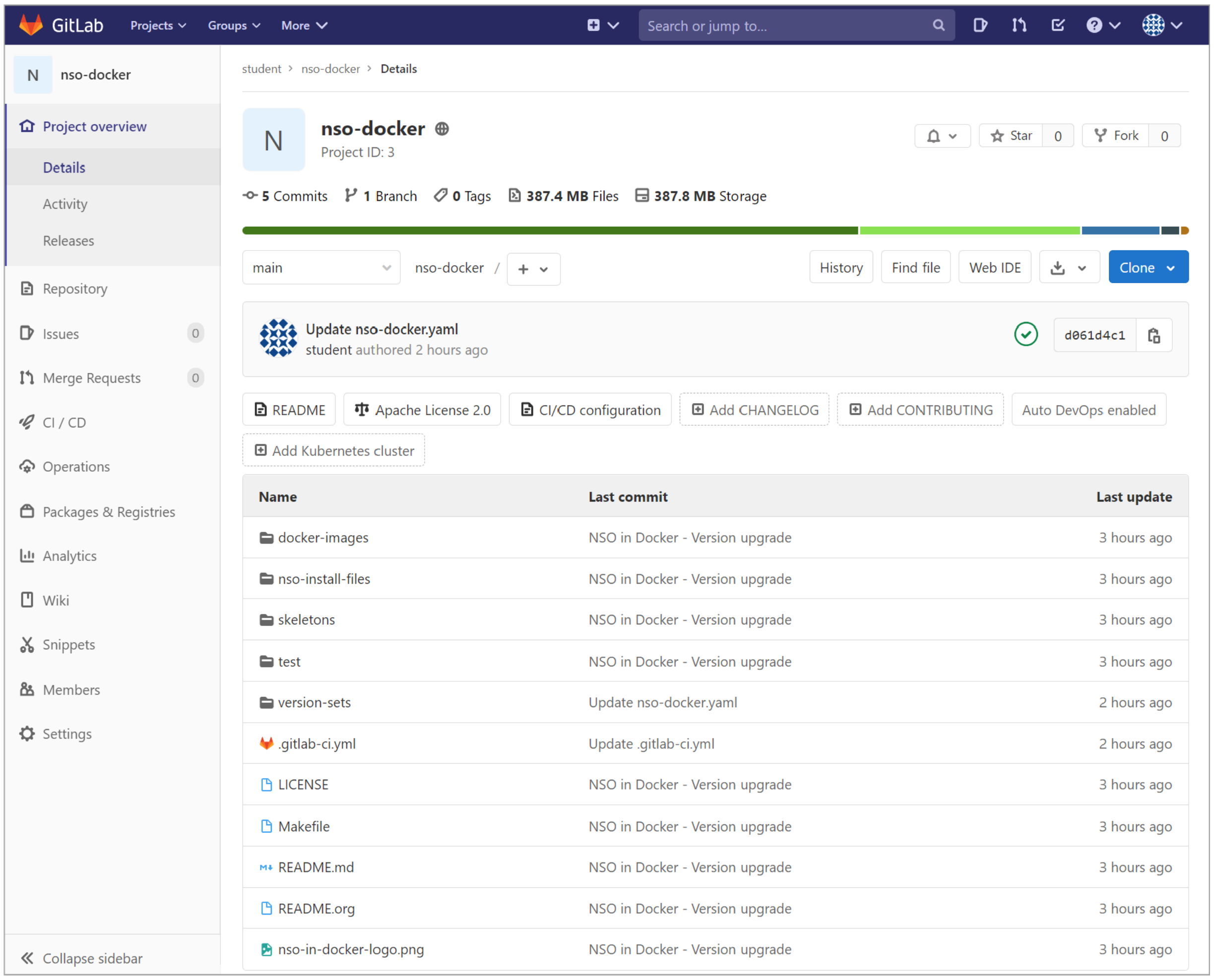
student / devops	Maintainer	★ 0	0	0	0	Updated 5 days ago
student / nso-docker	Maintainer	★ 0	0	0	0	Updated 5 days ago
student / nso-system	Maintainer	★ 0	0	0	0	Updated 5 days ago

The nso-docker project is already included in this lab but can otherwise be found at: https://github.com/NSO-developer/nso-docker. It is a repository run by Cisco and contains the files that are required to set up an NSO in Docker project. It also contains several project skeletons that can be used for different types of NSO projects: system, NED, and package.

Step 3

Click the **nso-docker** project to open the project page.

This project allows you to create Docker images that are tailored to the various NSO use cases. It produces two Docker images—base and dev. The former is used as a production NSO image and contains the bare necessities that the NSO needs and the latter is used as a development image and contains various tools and compilers. They are produced as a result of one of the steps of the CI/CD pipeline of the nso-docker project.



student › nso-docker › Details

nso-docker
Project ID: 3

5 Commits | 1 Branch | 0 Tags | 387.4 MB Files | 387.8 MB Storage

main | nso-docker /

Update nso-docker.yaml
student authored 2 hours ago
d061d4c1

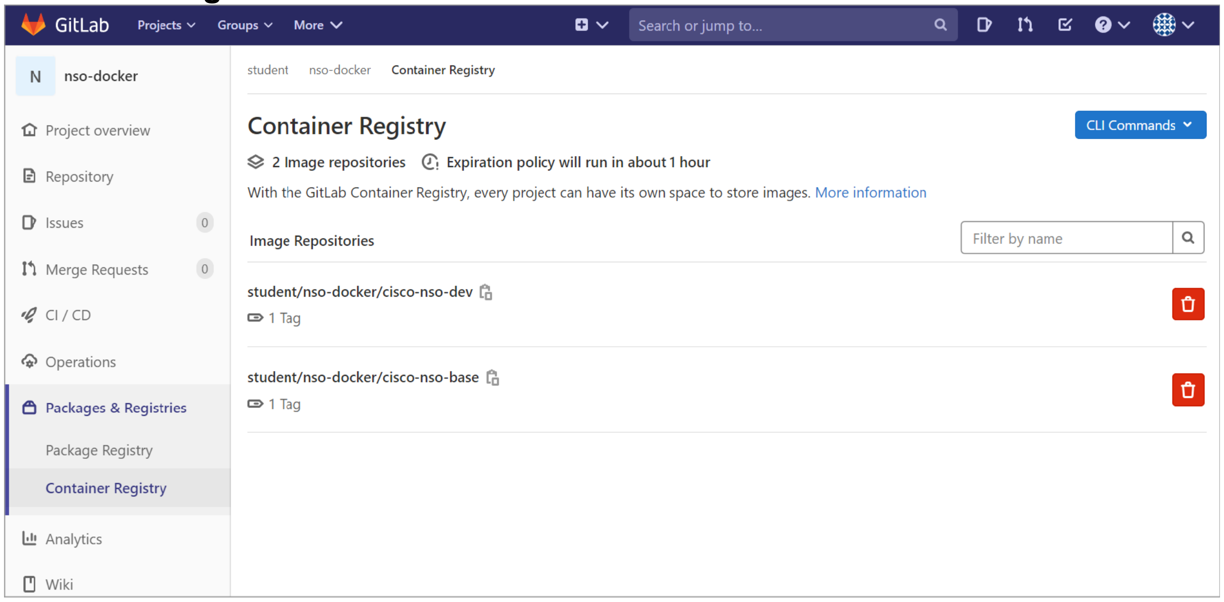
README | Apache License 2.0 | CI/CD configuration | Add CHANGELOG | Add CONTRIBUTING | Auto DevOps enabled | Add Kubernetes cluster

Name	Last commit	Last update
docker-images	NSO in Docker - Version upgrade	3 hours ago
nso-install-files	NSO in Docker - Version upgrade	3 hours ago
skeletons	NSO in Docker - Version upgrade	3 hours ago
test	NSO in Docker - Version upgrade	3 hours ago
version-sets	Update nso-docker.yaml	2 hours ago
.gitlab-ci.yml	Update .gitlab-ci.yml	2 hours ago
LICENSE	NSO in Docker - Version upgrade	3 hours ago
Makefile	NSO in Docker - Version upgrade	3 hours ago
README.md	NSO in Docker - Version upgrade	3 hours ago
README.org	NSO in Docker - Version upgrade	3 hours ago
nso-in-docker-logo.png	NSO in Docker - Version upgrade	3 hours ago

Step 4

Locate the Docker images that are produced by the nso-docker project.

Click the **Packages & Registries** on the left-side navigation bar, and then choose **Container Registries**.



student nso-docker Container Registry

Container Registry

2 Image repositories | Expiration policy will run in about 1 hour

With the GitLab Container Registry, every project can have its own space to store images. More information

Image Repositories

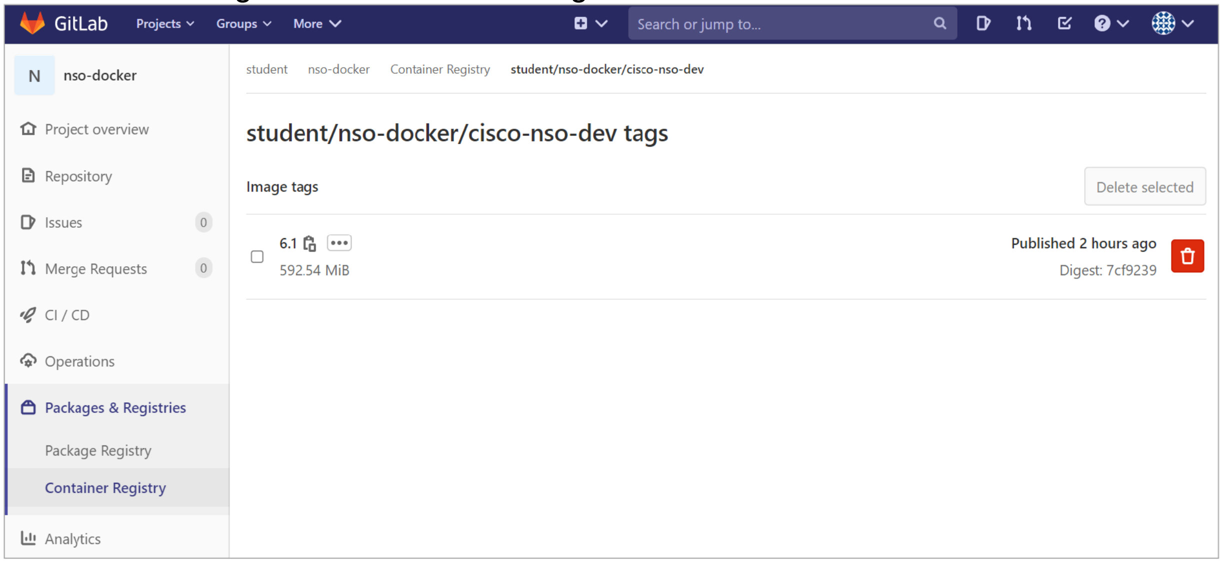
student/nso-docker/cisco-nso-dev
1 Tag

student/nso-docker/cisco-nso-base
1 Tag

Step 5

Click one of the image repositories.

All available image versions with their tags are listed here.



student/nso-docker/cisco-nso-dev tags

Image tags

6.1 592.54 MiB Published 2 hours ago Digest: 7cf9239

The images that are produced by the nso-docker project are already built in this activity because this process can take some time. You do not need to build them yourself.

Step 6

Open the terminal window.

Open the terminal window by clicking the **Terminal** icon on the bottom bar.

```
student@student-vm:~$
```

Step 7

Connect to the NSO server *nso-server*.

Connect to NSO server *nso-server* with the **student** user by using the SSH client. The authentication is already preconfigured with the public key authentication, therefore the password is not needed. The prompt will change stating you are now connected to the nso-server.

```
student@student-vm:/$ ssh student@nso-server
Last login: Mon Oct 16 14:18:51 2023 from 10.0.0.102
student@nso-server:~$
```

Step 8

Make sure that Docker is installed on the system by using the **docker -v** command.

The command should display your Docker version.

```
student@nso-server:~$ docker -v
Docker version 24.0.6, build ed223bc
student@nso-server:~$
```

Step 9

Log in to the Docker image registry.

The registry is located at the **registry.nso303.gitlab.local** URL. If asked for credentials, use the same credentials as you do for GitLab (**student** and **1234QWer**).

```
student@nso-server:~$ docker login registry.nso303.gitlab.local
Authenticating with existing credentials...

WARNING! Your password will be stored unencrypted in /home/
student/.docker/config.json.

Configure a credential helper to remove this warning. See

https://docs.docker.com/engine/reference/commandline/login/
#credentials-store



Login Succeeded






student@nso-server:~$
```

Step 10

Pull the two images to your local image registry with the **docker pull** command.

When specifying image names, do not forget to append the registry URL and an image tag.

```
student@nso-server:~$ docker pull registry.nso303.gitlab.local/student/
nso-docker/cisco-nso-base:6.1
6.1: Pulling from student/nso-docker/cisco-nso-base
99db33be616a: Pull complete
739b0e6c0051: Pull complete
cd37a2b881e2: Pull complete
7857116c2d32: Pull complete
8e4d24887a6d: Pull complete
a432a0b265e6: Pull complete
6d2ca1bef080: Pull complete
3ce9d5d6dfcb: Pull complete
c5a8bf40e998: Pull complete
12e20bfb1759: Pull complete
Digest:
sha256:948fcc8e3e7759bce299ddcb7973c32ccab969e1a1cdaeebca0767a4577bb106
```

```
Status: Downloaded newer image for registry.nso303.gitlab.local/
student/nso-docker/cisco-nso-base:6.1
registry.nso303.gitlab.local/student/nso-docker/cisco-nso-base:6.1

student@nso-server:~$ docker pull registry.nso303.gitlab.local/student/
nso-docker/cisco-nso-dev:6.1
6.1: Pulling from student/nso-docker/cisco-nso-dev
99db33be616a: Already exists
739b0e6c0051: Already exists
8ab15a3f6ece: Pull complete
0157d604ef55: Pull complete
b1b7090b8757: Pull complete
1f18615acad1: Pull complete
ae2865803401: Pull complete
d973a78581c5: Pull complete
9fa34ea7d410: Pull complete
12e20bfb1759: Pull complete
Digest:
sha256:7cf923936faafc721d55c5ee7cce0ed8a2dc78dc31382809fc40f6c2e7abb767
Status: Downloaded newer image for registry.nso303.gitlab.local/
student/nso-docker/cisco-nso-dev:6.1
registry.nso303.gitlab.local/student/nso-docker/cisco-nso-dev:6.1

student@nso-server:~$
```

Step 11

List the Docker images with the **docker images** command and locate the two images you pulled from the remote registry.

Use the **docker images** command to list the images.

```
student@nso-server:~$ docker images
REPOSITORY                                                        TAG
IMAGE ID       CREATED       SIZE
registry.nso303.gitlab.local/student/nso-docker/cisco-nso-base   6.1
9e4e0efe64ee   3 hours ago   686MB

registry.nso303.gitlab.local/student/nso-docker/cisco-nso-dev    6.1
bfcd7713b4dd   3 hours ago   1.45GB

gitlab/gitlab-ee
13.4.0-ee     16742366e880   3 years ago   2.18GB

gitlab/gitlab-ee
13.4.0-ee.0   16742366e880   3 years ago   2.18GB
```

Step 12

In your home folder, create and navigate to the Git directory.

In the terminal, use the **mkdir** and **cd** commands.

```
student@nso-server:~$ cd
student@nso-server:~$ mkdir Git
```

```
student@nso-server:~$ cd Git
student@nso-server:~/Git$
```

Step 13

Clone the nso-docker and nso-system projects from GitLab.

To clone the remote repository, use the **git clone** command.

```
student@nso-server:~/Git$ git clone http://nso303.gitlab.local/student/
nso-system.git
Cloning into 'nso-system'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.

student@nso-server:~/Git$ git clone http://nso303.gitlab.local/student/
nso-docker.git
Cloning into 'nso-docker'...
remote: Enumerating objects: 424, done.
remote: Counting objects: 100% (424/424), done.
remote: Compressing objects: 100% (252/252), done.
remote: Total 424 (delta 133), reused 389 (delta 106), pack-reused 0
Receiving objects: 100% (424/424), 387.10 MiB | 28.98 MiB/s, done.
Resolving deltas: 100% (133/133), done.
student@nso-server:~/Git$
```

Step 14

Navigate to the nso-docker directory and list the contents of the **skeletons** folder.

The nso-docker project contains project "skeletons" for multiple types of NSO projects. NED and package skeletons are used to build, deploy, and test network element drivers and NSO packages, and the system skeleton is used as a base for NSO projects.

```
student@nso-server:~/Git$ cd nso-docker/
student@nso-server:~/Git/nso-docker$ ls skeletons/
ned  nso-docker-composition.excalidraw  nso-docker-composition.png
package  README.org  system  test
student@nso-server:~/Git/nso-docker$
```

Step 15

Copy the system skeleton into the **~/Git/nso-system** folder.

By copying the system skeleton, you will prepare the base for your final NSO docker image.

```
student@nso-server:~/Git/nso-docker$ cp -r skeletons/system/* ../nso-
system/
```

Step 16

Navigate to the **~/Git/nso-system** folder and list the contents.

Make sure the output resembles the following output:

```
student@nso-server:~/Git/nso-docker$ cd ~/Git/nso-system
student@nso-server:~/Git/nso-system$ ls -al
total 96
drwxrwxr-x 8 student student  4096 Oct 17 08:30 .
drwxrwxr-x 4 student student  4096 Oct 17 08:14 ..
-rw-rw-r-- 1 student student  5004 Oct 17 08:30 Dockerfile.in
drwxrwxr-x 2 student student  4096 Oct 17 08:30 extra-files
drwxrwxr-x 8 student student  4096 Oct 17 08:14 .git
drwxrwxr-x 2 student student  4096 Oct 17 08:30 includes
-rw-rw-r-- 1 student student   261 Oct 17 08:30 Makefile
drwxrwxr-x 2 student student  4096 Oct 17 08:30 nid
-rw-rw-r-- 1 student student 11761 Oct 17 08:30 nidcommon.mk
-rw-rw-r-- 1 student student  3811 Oct 17 08:30 nidsystem.mk
-rw-rw-r-- 1 student student   650 Oct 17 08:30 nidvars.mk
drwxrwxr-x 2 student student  4096 Oct 17 08:30 packages
-rw-rw-r-- 1 student student    25 Oct 17 08:14 README.md
-rw-rw-r-- 1 student student 24681 Oct 17 08:30 README.nid-system.org
drwxrwxr-x 3 student student  4096 Oct 17 08:30 testenvs
student@nso-server:~/Git/nso-system$
```

Step 17

Open the **nidvars.mk** file.

The nidvars.mk file contains environment-specific variables, such as NSO version. Most of the complexity is hidden in the pre-built nidsystem.mk library.

The following output shows how the file should appear when you open it for the first time:

```
student@student-vm:~/nso-system$ nano nidvars.mk
# You can set the default NSO_IMAGE_PATH & PKG_PATH to point to your
docker
# registry so that developers don't have to manually set these
variables.
# Similarly for NSO_VERSION you can set a default version. Note how the
?=
# operator only sets these variables if not already set, thus you can
easily
# override them by explicitly setting them in your environment and they
will be
# overridden by variables in CI.
# TODO: uncomment and fill in values for your environment
# Default variables:
#export NSO_IMAGE_PATH ?= registry.example.com:5000/my-group/nso-
docker/
#export PKG_PATH ?= registry.example.com:5000/my-group/
#export NSO_VERSION ?= 5.4
```

Step 18

Set environmental variables that are needed for this project.

Open the **nidvars.mk** file. Set NSO_VERSION to the NSO version you are using (**6.1**) and set NSO_IMAGE_PATH and IMAGE_PATH to that of your Docker image registry.

```
# You can set the default NSO_IMAGE_PATH & PKG_PATH to point to your
docker
# registry so that developers don't have to manually set these
variables.
# Similarly for NSO_VERSION you can set a default version. Note how the
?=
# operator only sets these variables if not already set, thus you can
easily
# override them by explicitly setting them in your environment and they
will be
# overridden by variables in CI.
# TODO: uncomment and fill in values for your environment
# Default variables:

export NSO_IMAGE_PATH = registry.nso303.gitlab.local/student/nso-
docker/
#export PKG_PATH ?= registry.example.com:5000/my-group/
export NSO_VERSION = 6.1

export IMAGE_PATH = registry.nso303.gitlab.local/student/
```

Make sure you only use TAB characters when indenting commands inside Makefiles. If you use regular space characters, you will encounter errors.

Step 19

Save the file and exit the file editor.

Use the **CTRL+X** keys on the keyboard and confirm saving the modification to save and exit the editor.

Step 20

Build the NSO project image using the **make build** command.

Make sure that the image is successfully built. The output should resemble the following output.

```
student@nso-server:~/Git/nso-system$ make build
Checking NSO in Docker images are available...
INFO: registry.nso303.gitlab.local/student/nso-docker/cisco-nso-
base:6.1 exists, attempting pull of latest version
6.1: Pulling from student/nso-docker/cisco-nso-base
Digest:
sha256:948fcc8e3e7759bce299ddcb7973c32ccab969e1a1cdaeebca0767a4577bb106
Status: Image is up to date for registry.nso303.gitlab.local/student/
```

```
nso-docker/cisco-nso-base:6.1
registry.nso303.gitlab.local/student/nso-docker/cisco-nso-base:6.1
6.1: Pulling from student/nso-docker/cisco-nso-dev
Digest:
sha256:7cf923936faafc721d55c5ee7cce0ed8a2dc78dc31382809fc40f6c2e7abb767
Status: Image is up to date for registry.nso303.gitlab.local/student/
nso-docker/cisco-nso-dev:6.1
registry.nso303.gitlab.local/student/nso-docker/cisco-nso-dev:6.1
-- Generating Dockerfile
cp Dockerfile.in Dockerfile
for DEP_NAME in $(ls includes/); do export DEP_URL=$(awk '{ print
"echo", $0 }' includes/${DEP_NAME} | /bin/sh -); awk "/DEP_END/ { print
\"FROM ${DEP_URL} AS ${DEP_NAME}\" }; /DEP_INC_END/ { print \"COPY --
from=${DEP_NAME} /var/opt/ncs/packages/ /includes/\" }; 1" Dockerfile >
Dockerfile.tmp; mv Dockerfile.tmp Dockerfile; done
docker build --target build -t registry.nso303.gitlab.local/student/
nso-system/build:6.1-student  --build-arg http_proxy --build-arg
https_proxy --build-arg no_proxy --build-arg
NSO_IMAGE_PATH=registry.nso303.gitlab.local/student/nso-docker/ --
build-arg NSO_VERSION=6.1 --build-arg
PKG_FILE=registry.nso303.gitlab.local/student/nso-system/package:6.1-
student --progress=plain  .
#0 building with "default" instance using docker driver

#1 [internal] load build definition from Dockerfile
#1 transferring dockerfile: 5.04kB done
#1 DONE 0.0s

#2 [internal] load .dockerignore
#2 transferring context: 2B done
#2 DONE 0.0s

#3 [internal] load metadata for registry.nso303.gitlab.local/student/
nso-docker/cisco-nso-dev:6.1
#3 DONE 0.0s

#4 [internal] load build context
#4 transferring context: 108.87kB 0.0s done
#4 DONE 0.0s

#5 [build 1/6] FROM registry.nso303.gitlab.local/student/nso-docker/
cisco-nso-dev:6.1
#5 DONE 0.0s

...
...
...

#15 [nso 4/5] COPY extra-files /
#15 DONE 0.0s

#16 [nso 5/5] RUN for PKG in $(find /var/opt/ncs/packages -mindepth 1 -
maxdepth 1 -type d); do   if make -C ${PKG}/src -n compose >/dev/null
2>&1; then     echo "Found 'compose' target in ${PKG}/src, executing
...";     make -C ${PKG}/src compose || exit 1;   fi; done
#16 DONE 0.3s

#17 exporting to image
```

```
#17 exporting layers 0.1s done
#17 writing image
sha256:7f70506046b4fb968f99b1245149e6bbebd7ce762ece2a29cc2a9fcbcc33ad89
done
#17 naming to registry.nso303.gitlab.local/student/nso-system/nso:6.1-
student 0.0s done
#17 DONE 0.1s
docker tag registry.nso303.gitlab.local/student/nso-system/nso:6.1-
student registry.nso303.gitlab.local/student/nso-system/nso:MM_6.1-
student
student@nso-server:~/Git/nso-system$
```

Step 21

Start the NSO Docker environment using the **make testenv-start** command.

This command starts the NSO System container. This container includes the NSO installation and is based on the base (production) NSO image.

```
student@student-vm:~/Git/nso-system$ make testenv-start
make -C testenvs/quick start
make[1]: Entering directory '/home/student/Git/nso-system/testenvs/
quick'
docker network inspect testenv-nso-system-quick-6.1-student >/dev/null
2>&1 || docker network create testenv-nso-system-quick-6.1-student
999e86c4120e1a36a0c8eece3b12b0d38d1e87f3cd0679f4a1ba036afddea2ed
docker run -td --name testenv-nso-system-quick-6.1-student-nso --
network-alias nso  --network testenv-nso-system-quick-6.1-student --
label com.cisco.nso.testenv.name=testenv-nso-system-quick-6.1-student
--label com.cisco.nso.testenv.type=nso --volume /var/opt/ncs/packages -
e DEBUGPY= --expose 5678 --publish-all registry.nso303.gitlab.local/
student/nso-system/nso:6.1-student
a773f5a18b89464d702dfc80e9dad3152b4fff4b9944dfffe3d0f6cc89411bba
[ "false" = "true" ] && echo $(docker network inspect --format '{{range
.IPAM.Config}}{{- if (gt (split .Subnet ":"|len) 1) -}}{{.Subnet}}{{-
end}}{{end}}' testenv-nso-system-quick-6.1-student) | egrep "^[23]...:"
|| (echo "Removing IPv6 default route" && docker ps -aq --filter
label=com.cisco.nso.testenv.name=testenv-nso-system-quick-6.1-student |
xargs --no-run-if-empty -I CNT -n1 docker run --rm --net=container:CNT
--cap-add=NET_ADMIN registry.nso303.gitlab.local/student/nso-docker/
cisco-nso-base:6.1 ip -6 route del default >/dev/null 2>&1 || true)
Removing IPv6 default route
make wait-started-nso
make[2]: Entering directory '/home/student/Git/nso-system/testenvs/
quick'
NSO instance testenv-nso-system-quick-6.1-student-nso has started
All NSO instance have started
make[2]: Leaving directory '/home/student/Git/nso-system/testenvs/
quick'
make[1]: Leaving directory '/home/student/Git/nso-system/testenvs/
quick'
student@student-vm:~/Git/nso-system$
```

Step 22

Verify that the NSO System container is running.

Use the **docker ps -a** command.

```
student@nso-server:~/Git/nso-system$ docker ps -a
CONTAINER ID   IMAGE
COMMAND                CREATED          STATUS                     PORTS
NAMES
a773f5a18b89   registry.nso303.gitlab.local/student/nso-system/nso:6.1-
student   "/run-nso.sh"       4 minutes ago   Up 4 minutes (healthy)
0.0.0.0:32781->22/tcp, 0.0.0.0:32780->80/tcp, 0.0.0.0:32779->443/tcp,
0.0.0.0:32778->830/tcp, 0.0.0.0:32777->4334/tcp, 0.0.0.0:32776->4570/
tcp, 0.0.0.0:32775->5678/tcp   testenv-nso-system-quick-6.1-student-nso

b4fdf15c0206   gitlab/gitlab-ee:13.4.0-ee
"/assets/wrapper"   25 hours ago    Up 19 hours (healthy)
0.0.0.0:80->80/tcp, 0.0.0.0:443->443/tcp, 0.0.0.0:8022->22/tcp
gitlab_web_1
student@nso-server:~/Git/nso-system$
```

Step 23

Enter the NSO CLI by using the **make testenv-cli** command.

This command takes you into the test NSO System container and executes the **ncs_cli** command, which takes you directly to the NSO CLI.

```
student@nso-server:~/Git/nso-system$ make testenv-cli
make -C testenvs/quick cli
make[1]: Entering directory '/home/student/Git/nso-system/testenvs/
quick'
docker exec -it testenv-nso-system-quick-6.1-student-nso bash -lc
'ncs_cli -u admin'

admin connected from 127.0.0.1 using console on a773f5a18b89
admin@ncs>
```

To enter the NSO System container Linux shell instead of the NSO CLI, use the **make testenv-shell** command.

Step 24

Exit the NSO CLI.

Use the **exit** command. This action also exits the Docker container.

```
admin@ncs> exit
make[1]: Leaving directory '/home/student/Git/nso-system/testenvs/
quick'
student@nso-server:~/Git/nso-system$
```

Activity Verification

You have completed this task when you attain these results:

- You can enter the NSO CLI within the System container.

What is the Makefile target command that builds NSO Docker images?

- ○ **make build**
- ○ **make testenv-build**
- ○ **make testenv-run**
- ○ **make testenv-start**