Discovery 2: Implement Role-Based Access and PAM

Introduction

In this activity, you will work with different users and groups and apply different permissions based on their roles.

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

- Create different user groups.
- Implement role-based access.

Job Aid

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

- Lab Guide

The following table contains passwords that you might need.

Device	Username	Password
student-vm	student	1234QWer
nso-server	student	1234QWer
NSO application	oper	oper123
NSO application	monitor	monitor123

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.
cd ..	Exit out of current directory.

Command	Comment
cd test	Move into the "test" folder, which is a subfolder of the current directory.
cd /home/student	Move into the "student" 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.

Formatting	Description and Examples
`show running config`	Commands in CLI outputs and configurations use this formatting.
highlight	CLI output that is important is highlighted.

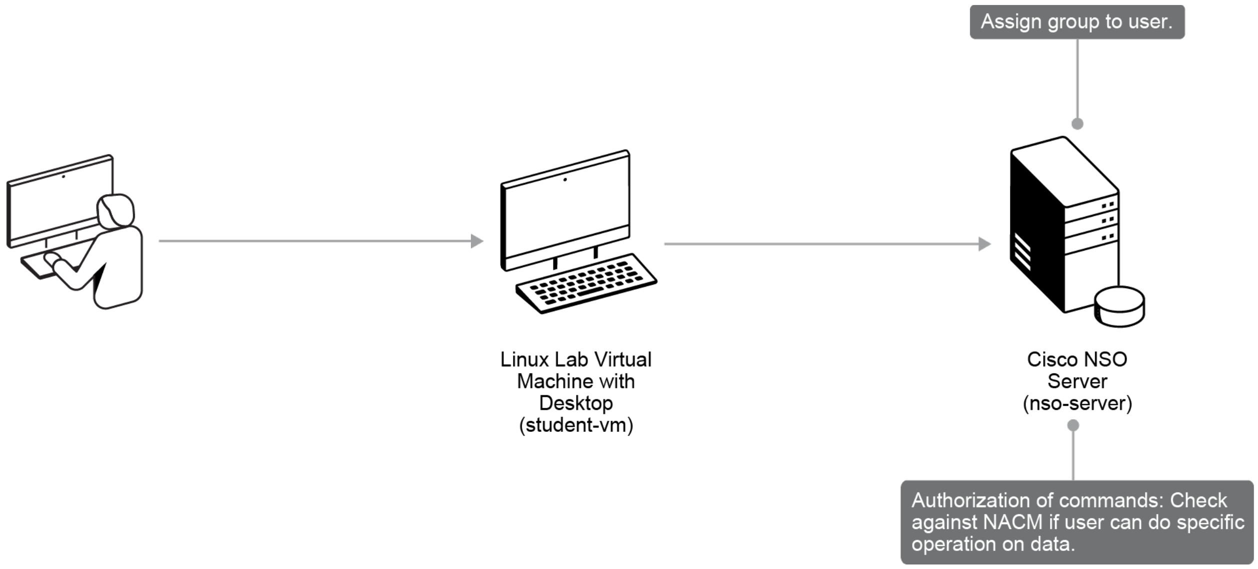
Example	`student@student-vm:~$ **ncs --version**`
	`        6.1`

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. Your lab environment is a Linux server (Student-VM) acting as a jumphost and a Linux server (NSO-server) acting as a NSO server. This is where you will be installing NSO.

Topology


Assign group to user.
Linux Lab Virtual Machine with Desktop (student-vm)
Cisco NSO Server (nso-server)
Authorization of commands: Check against NACM if user can do specific operation on data.

Task 1: Prepare the Test Environment

In this task, you will create a NetSim lab with some simulated devices and add those devices to NSO. You will also install the already developed l3vpn package that you will use for testing different permissions based on the assigned group.

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 on the Student-VM icon in the topology map.

Step 2

Open the terminal window.

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

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

Step 3

Connect to the NSO server **nso-server**.

Connect to the NSO server **nso-server** with the **student** user using the SSH client. The authentication is already preconfigured with 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: Tue Oct 3 09:14:42 2023 from 10.0.0.102
student@nso-server:~$
```

Step 4

Make sure that the already prepared NetSim lab is running.

Change the directory to **lab** and make sure that the NetSim lab is running. Use the **ncs-netsim is-alive** command. If the lab is not running for any reason (such as VM reboot), use the **ncs-netsim start** command to start it.

```
student@nso-server:~$ cd lab/
student@nso-server:~/lab$ ncs-netsim is-alive
DEVICE CE11 OK
DEVICE CE12 OK
DEVICE CE21 OK
DEVICE CE22 OK
DEVICE PE11 OK
DEVICE PE22 OK
student@nso-server:~/lab$
```

Step 5

Export devices from the NetSim lab in the NSO-compatible XM.

Use the **ncs-netsim ncs-xml-init** command. Use the **~/lab** directory as the destination.

```
student@nso-server:~/lab$ ncs-netsim ncs-xml-init > devices.xml
student@nso-server:~/lab$
```

Step 6

Connect to NSO CLI, create a **default** authorization group and load previously exported devices to NSO.

Connect to NSO CLI using **ncs_cli -C** command. You can also access the CLI using SSH to port 2024. Create a **default** authorization group.

```
student@nso-server:~/lab$ ncs_cli -C

student connected from 10.0.0.102 using ssh on nso-server
student@ncs# config
Entering configuration mode terminal
student@ncs(config)# devices authgroups group default default-map
remote-name admin remote-password admin
student@ncs(config-group-default)# top
```

Step 7

Load exported devices using the **load merge** command.

The **load merge** command will load the XML directly from the file system.

```
student@ncs(config)# load merge /home/student/lab/devices.xml
Loading.
6.71 KiB parsed in 0.03 sec (218.17 KiB/sec)
student@ncs(config)# commit
Commit complete.
student@ncs(config)# exit
student@ncs#
```

Step 8

Perform a sync from devices.

Execute **devices sync-from** to synchronize the NSO CDB with lab devices configurations.

```
student@ncs# devices sync-from
sync-result {
    device CE11
    result true
}
sync-result {
    device CE12
    result true
}
sync-result {
    device CE21
    result true
}
sync-result {
    device CE22
    result true
}
sync-result {
    device PE11
    result true
}
sync-result {
    device PE22
    result true
}
student@ncs#
```

Step 9

Copy the **l3vpn** service package to the NSO packages folder.

Exit the NSO CLI, switch back to the home directory and copy the **l3vpn** service package that you will use to test RBAC.

```
student@ncs# exit
```

```
student@nso-server:~/lab$ cd
student@nso-server:~$ cp -r packages/l3vpn /var/opt/ncs/packages/
student@nso-server:~$
```

Step 10

Build the package in the target directory.

Build the package with **make**. You can use the **-C** flag so that you do not have to enter the corresponding src directory.

```
student@nso-server:~$ make -C /var/opt/ncs/packages/l3vpn/src/
make: Entering directory '/var/opt/ncs/packages/l3vpn/src'
mkdir -p ../load-dir
/opt/ncs/current/bin/ncsc  `ls l3vpn-ann.yang  > /dev/null 2>&1 && echo
"-a l3vpn-ann.yang"` \
              -c -o ../load-dir/l3vpn.fxs yang/l3vpn.yang
make: Leaving directory '/var/opt/ncs/packages/l3vpn/src'
student@nso-server:~$
```

Step 11

Log back in to the NSO CLI and reload the packages.

Log back in to the NSO CLI using the **ncs_cli -C** command, and perform **packages reload** so that the l3vpn service package becomes available and its result is set to true (result true).

```
student@nso-server:~$ ncs_cli -C

User student last logged in 2023-10-03T13:10:16.190806+00:00, to nso-
server, from 10.0.0.102 using cli-ssh
student connected from 10.0.0.102 using ssh on nso-server
student@ncs# packages reload

>>> System upgrade is starting.
>>> Sessions in configure mode must exit to operational mode.
>>> No configuration changes can be performed until upgrade has
completed.
>>> System upgrade has completed successfully.
reload-result {
    package cisco-ios-cli-6.85
    result true
}
reload-result {
    package cisco-iosxr-cli-7.41
    result true
}
reload-result {
    package l3vpn
    result true
}
student@ncs#
System message at 2023-10-03 13:13:35...
```

```
    Subsystem stopped: ncs-dp-1-cisco-ios-cli-6.85:IOSDp
student@ncs#
System message at 2023-10-03 13:13:35...
    Subsystem started: ncs-dp-2-cisco-ios-cli-6.85:IOSDp
student@ncs# exit
student@nso-server:~$
```

Activity Verification

You have completed this task when you created a NetSim lab with some simulated devices and add those devices to NSO. You also installed the already developed l3vpn package.

Task 2: Define User Roles and Access Rules

In this task, you will configure role-based access. You will have two different groups—**ncsoper** for operators and **ncsadmin** for admins. They will have different access to data and commands that they can execute. This will be defined with NACM rules.

Activity

Complete these steps:

Step 1

On your NSO server, create an **oper** user with the password oper123.

Using the terminal window, create an **oper** user with the **oper123** password.

```
student@nso-server:~$ sudo useradd oper
student@nso-server:~$ sudo passwd oper
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
passwd: password updated successfully
student@nso-server:~$
```

You will get a bad password warning which you can ignore. For simplicity, a weak password is used; in your production environment, use strong passwords.

Step 2

Add the new **oper** user to the **ncsoper** group.

Use the **usermod** command to add the new **oper** user to the **ncsoper** group that you have created in the previous lab.

```
student@nso-server:~$ sudo usermod -aG ncsoper oper
student@nso-server:~$
```

Since you are using PAM as the authentication method, the **oper** user will be logged into NSO with the **ncsoper** group assigned.

Step 3

Connect to the NSO CLI as the **student** user.

Connect to the NSO CLI as user **student** using **ncs_cli -C**, where **-C** stands for the Cisco style of NSO CLI. First, examine the NACM groups that are created by NSO at install.

```
student@nso-server:~$ ncs_cli -C

User student last logged in 2023-10-03T13:11:24.452529+00:00, to nso-
server, from 10.0.0.102 using cli-ssh
student connected from 10.0.0.102 using ssh on nso-server
student@ncs# show running-config nacm groups
nacm groups group ncsadmin
 user-name [ private ]
!
nacm groups group ncsoper
 user-name [ public ]
!
student@ncs#
```

Step 4

Create a rule that allows you to create and update actions on device configurations for operators.

You will limit permissions that the oper group has to create and read actions only by adding a rule.

```
student@ncs# config
Entering configuration mode terminal
student@ncs(config)# nacm rule-list oper rule allow-create-update-on-
device path /devices access-operations create,update action permit
student@ncs(config-rule-allow-create-update-on-device)# top
student@ncs(config)# commit
Commit complete.
student@ncs(config)#
```

There is an implicit deny rule in place when NACM is configured.

Step 5

Create another rule that will allow operators to create, update, and delete configurations using the **l3vpn** service package that you have installed.

To create the rule, use the following commands in the configuration mode:

```
student@ncs(config)# nacm rule-list oper rule allow-l3vpn path /
services/l3vpn access-operations create,update,delete action permit
student@ncs(config-rule-allow-l3vpn)# top
student@ncs(config)# commit
Commit complete.
student@ncs(config)# exit
student@ncs# exit
student@nso-server:~$
```

Step 6

Log out as **student** and connect back to NSO CLI as the **oper** user to test the rule that you have created.

Configure a loopback interface with an IP address to test if the create operation is allowed. Committing the changes should be successful.

```
student@nso-server:~$ ssh oper@nso-server -p 2024
oper@127.0.0.1's password:

oper connected from 127.0.0.1 using ssh on nso-server
oper@ncs> switch cli
oper@ncs# config
Entering configuration mode terminal
oper@ncs(config)# devices device PE11 config interface Loopback 0
ipv4 address 1.2.3.4 /32
oper@ncs(config-if)# commit
Commit complete.
oper@ncs(config-if)#
```

Step 7

Now try to remove the configuration that you have just created. The operation should fail.

Since you have the rule that only allows create and update on devices in place, the operation should fail.

```
oper@ncs(config-if)# top
oper@ncs(config)# no devices device PE11 config interface Loopback 0
Warning: Some elements could not be removed due to NACM rules
prohibiting access.

oper@ncs(config)#
```

Step 8

Test service provisioning with the **l3vpn** service.

Test the service provisioning that you have allowed for the operators to see if NACM rules are in effect.

```
oper@ncs(config)# services l3vpn test link 1 device PE11 interface
0/0/0/1 ip-address 10.1.1.1 mask 255.255.255.252
oper@ncs(config-link-1)# exit
oper@ncs(config-l3vpn-test)# services link 2 device PE22 interface
0/0/0/1 ip-address 10.1.2.1 mask 255.255.255.252
oper@ncs(config-link-2)# top
oper@ncs(config)# commit
Commit complete.
oper@ncs(config)# exit
oper@ncs#
```

Step 9

Confirm that commands **sync-from** or **check-sync** are not available in the CLI.

Use the question mark or tab completion under the root of the CLI. Observe the options that are listed; you will see that there are no devices-related commands available, including **sync-from** or **check-sync** commands, since these operations are not permitted in NACM for operators.

```
oper@ncs# ?
Possible completions:
autowizard - Automatically query for mandatory elements
call-home - Set satellite URL for Smart Licensing
cd - Change working directory
clear - Clear parameter
compare - Compare running configuration to another configuration or a
file
complete-on-space - Enable/disable completion on space
config - Manipulate software configuration information
debug - Commands for debugging
describe - Display transparent command information
devtools - Enable/disable development tools
display-level - Configure show command display level
exit - Exit the management session
file - Perform file operations
help - Provide help information
...
```

Step 10

Define rules to allow **ncsoper** users to execute the **check-sync** and **sync-to** operation.

Exit as the **oper** user and log back in with **student**. Define rules to allow **ncsoper** users to execute the **check-sync** and **sync-to** operation. In case there was out-of-band configuration change done on the device, you are able to override the configuration on a device with the NSO version of configuration.

```
oper@ncs# exit
Connection to nso-server closed.
student@nso-server:~$ ncs_cli -C
```

```
User student last logged in 2023-10-03T13:16:59.376591+00:00, to nso-
server, from 10.0.0.102 using cli-ssh
student connected from 10.0.0.102 using ssh on nso-server
student@ncs# config
Entering configuration mode terminal
student@ncs(config)# nacm rule-list oper rule allow-check-sync path /
devices/device/check-sync access-operations exec action permit
student@ncs(config-rule-allow-check-sync)# top
student@ncs(config)# nacm rule-list oper rule allow-sync-to path /
devices/device/sync-to access-operations exec action permit
student@ncs(config-rule-allow-sync-to)# top
student@ncs(config)# commit
Commit complete.
student@ncs(config)# exit
student@ncs# exit
student@nso-server:~$
```

Step 11

Switch back to the oper user, and test if you can execute **check-sync** and **sync-to** to device now. After that exit the NSO CLI.

Test the actions using **check-sync** commands. Finally, exit the NSO CLI session.

```
student@nso-server:~$ ssh oper@nso-server -p 2024
oper@nso-server's password:

User oper last logged in 2023-10-03T13:19:02.928592+00:00, to nso-
server, from 127.0.0.1 using cli-ssh
oper connected from 127.0.0.1 using ssh on nso-server
oper@ncs> switch cli
oper@ncs# devices device PE11 check-sync
result in-sync
oper@ncs# devices device PE22 sync-to
result true
oper@ncs# exit
Connection to nso-server closed.
student@nso-server:~$
```

Step 12

Login to NSO CLI as student user and verify the current NACM configuration for the oper user.

Use the **show running-config nacm rule-list oper** command. The output should resemble the output below.

```
student@nso-server:~$ ncs_cli -C

User student last logged in 2023-10-03T13:22:56.760469+00:00, to nso-
server, from 10.0.0.102 using cli-ssh
student connected from 10.0.0.102 using ssh on nso-server
student@ncs# show running-config nacm rule-list oper
```

```
nacm rule-list oper
 group [ ncsoper ]
 rule tailf-aaa-user
  module-name       tailf-aaa
  path              /user[name='$USER']
  access-operations create,read,update,delete
  action            permit
 !
 rule tailf-webui-user
  module-name       tailf-webui
  path              /webui/data-stores/user-profile[username='$USER']
  access-operations create,read,update,delete
  action            permit
 !
 rule tailf-aaa-alluser
  module-name tailf-aaa
  path        /user
  action      deny
 !
 rule tailf-aaa-aaa
  module-name tailf-aaa
  path        /aaa
  action      deny
 !
 rule nacm
  module-name ietf-netconf-acm
  path        /
  action      deny
 !
 rule read-only
  path              /
  access-operations read
  action            permit
 !
 rule allow-create-update-on-device
  path              /devices
  access-operations create,update
  action            permit
 !
 rule allow-l3vpn
  path              /services/l3vpn
  access-operations create,update,delete
  action            permit
 !
 rule allow-check-sync
  path              /devices/device/check-sync
  access-operations exec
  action            permit
 !
 rule allow-sync-to
  path              /devices/device/sync-to
  access-operations exec
  action            permit
 !
 cmdrule c-logout
  command logout
  action  deny
 !
```

```
  cmdrule j-logout
   command "request system logout"
   action  deny
  !
  cmdrule any-command
   action permit
  !
 !
student@ncs# exit
student@nso-server:~$
```

Activity Verification

You have completed this task when you have verified that the **oper** user is allowed to create and update device configurations but cannot delete them. The **oper** user is allowed to create, update, and delete configurations through the **l3vpn** service. The **oper** user is allowed to execute **check-sync** and **sync-to** actions towards devices.

Task 3: Define Read-Only User Group

In this task, you will create a new NACM group with read-only access to the l3vpn service while all other data access will be denied.

Activity

Complete these steps:

Step 1

Create a new **monitor** user with the **monitor123** password.

On your NSO server, create a new **monitor** user with the **monitor123** password. You will get a bad password warning which you can ignore. For simplicity, a weak password is used; in your production environment, use strong passwords.

```
student@nso-server:~$ sudo useradd monitor
student@nso-server:~$ sudo passwd monitor
New password:
BAD PASSWORD: The password contains the user name in some form
Retype new password:
passwd: password updated successfully
student@nso-server:~$
```

Step 2

Create a new group **ncsmonitor** and add the **monitor** user to it.

Use the **groupadd** command to add the new **ncsmonitor** group. Use the **usermod** command to add a new monitor user to the **ncsmonitor** group.

```
student@nso-server:~$ sudo groupadd ncsmonitor
student@nso-server:~$ sudo usermod -aG ncsmonitor monitor
student@nso-server:~$
```

Step 3

In the NSO CLI, create a new NACM user group called **ncsmonitor**.

Connect to the NSO CLI with the **student** user. Create a new NACM user group called **ncsmonitor**, commit the changes and exit the CLI.

```
student@nso-server:~$ ncs_cli -C

student connected from 10.6.0.19 using ssh on student-vm
student@ncs# config
Entering configuration mode terminal
student@ncs(config)# nacm groups group ncsmonitor
student@ncs(config-group-ncsmonitor)# top
student@ncs(config)# commit
Commit complete.
student@ncs(config)# exit
student@ncs# exit
student@nso-server:~$
```

Step 4

Log back in the NSO CLI with the new **monitor** user and verify user permissions.

Log back in the NSO CLI with the new **monitor** user. You can see that, by default, the **ncsmonitor** group to which the **monitor** user is assigned has no permission at all.

```
student@nso-server:~$ ssh monitor@127.0.0.1 -p 2024
monitor@127.0.0.1's password:

monitor connected from 127.0.0.1 using ssh on student-vm
monitor@ncs> ?
Possible completions:
exit - Exit the management session
quit - Exit the management session
monitor@ncs> exit
Connection to 127.0.0.1 closed.
student@nso-server:~$
```

The **show running-config nacm** command displays NACM rules that are specified. You can see default deny rules on top, which prevents you to do any action with the monitor user.

Step 5

Configure a rule that will allow the monitor user to display the **l3vpn** service instances configuration.

Switch back to the **student** user and configure a rule that will allow the **monitor** user to display the **l3vpn** service instances configuration.

```
student@nso-server:~$ ncs_cli -C

User student last logged in 2023-09-07T10:22:20.460758+00:00, to
student-vm, from172.23.80.1 using cli-ssh
student connected from 172.23.80.1 using ssh on student-vm
student@ncs# config
Entering configuration mode terminal
student@ncs(config)# nacm rule-list monitor group ncsmonitor rule
permit-l3vpn-read access-operations read path /services/l3vpn action
permit
student@ncs(config-rule-permit-l3vpn-read)# top
student@ncs(config)# commit
Commit complete.
student@ncs(config)#
```

Step 6

Configure a rule that will allow the **monitor** user to execute actions.

In the same configuration session, add a new rule that will allow the monitor user to execute actions. Commit the changes.

```
student@ncs(config)# nacm rule-list monitor cmdrule any-command action
permit
student@ncs(config-cmdrule-any-command)# top
student@ncs(config)# commit
Commit complete.
student@ncs(config)# exit
student@ncs# exit
student@nso-server:~$
```

Step 7

Test new behavior for the monitor user.

Log in the NSO CLI with the new **monitor** user. Verify the running configuration. Only the **l3vpn** service instances configuration should be visible to the monitor user.

```
student@nso-server:~$ ssh monitor@nso-server -p 2024
monitor@nso-server's password:

User monitor last logged in 2023-10-03T13:30:03.926057+00:00, to nso-
server, from 127.0.0.1 using cli-ssh
monitor connected from 127.0.0.1 using ssh on nso-server
monitor@ncs> switch cli
monitor@ncs# show running-config
services l3vpn test
link 1
```

```
device PE11
interface 0/0/0/1
ip-address 10.1.1.1
mask 255.255.255.252
!
link 2
device PE22
interface 0/0/0/1
ip-address 10.1.2.1
mask 255.255.255.252
!
!
monitor@ncs# exit
Connection to nso-server closed.
student@nso-server:~$
```

Step 8

Verify the current NACM configuration for the **monitor** user.

Log in the NSO CLI with the **student** user. Verify the current NACM configuration for the **monitor** user.

```
student@nso-server:~$ ncs_cli -C

User student last logged in 2023-10-03T13:33:05.736614+00:00, to nso-
server, from 10.0.0.102 using cli-ssh
student connected from 10.0.0.102 using ssh on nso-server
student@ncs# show running-config nacm rule-list monitor
nacm rule-list monitor
group [ ncsmonitor ]
rule permit-l3vpn-read
path /services/l3vpn
access-operations read
action permit
!
cmdrule any-command
action permit
!
!
student@ncs# exit
student@nso-server:~$
```

Activity Verification

You have completed this task when you have verified that: The **monitor** user is allowed to read only the **l3vpn** service instances configuration.

What does the **show running-config nacm** display in NSO CLI?

- ○ Currently logged in users
- ○ Log of the requests denied by the NACM rules

- ○ NACM rules that are specified
- ○ Operation status of the NACM