Discovery 3: Configure and Troubleshoot Devices

Introduction

In this lab, you will practice configuring multiple network devices at once in a single transaction. You will experience an out-of-sync issue with one of the managed routers in your lab network and resolve it by synchronizing the CDB from the device. Then, you will perform the investigation and troubleshooting for a device that was accidentally configured directly by someone who could not use Cisco NSO to accomplish their job. Finally, you will learn how to create and apply the templates to the individual devices and the device groups and how to troubleshoot NSO to device communications by using the NED southbound traffic tracing feature.

Scenario

You have been hired as a junior network engineer at a company that uses Cisco NSO to automate device configuration management and network service activation tasks. Your manager requires that you learn how to configure and troubleshoot devices in Cisco NSO by accomplishing simple activities, such as creating and committing multiple device configurations, synchronizing the devices and the CDB, using the device templates, and tracing the NED to device communications. You will perform all the tasks in a lab with an already installed local NSO instance and a few virtual routers mimicking a small segment of the production network.

Activity Objective

To accomplish the task that your manager assigned to you, you will use a Linux-based virtual machine with the NSO software installed on it in your home directory.

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

- Configure multiple features on multiple devices at once
- Synchronize the device configurations from and to the network
- Troubleshoot the unexpected direct device configuration changes
- Inspect the configuration differences between the devices and the CDB
- Configure and apply device templates with variables
- Troubleshoot NED to device communications

Topology

Job Aid

Device Information

Device	Description	IP Address	Credentials
Student DevBox	Linux Ubuntu VM	10.10.20.50	developer, C1sco12345
R1 (dist-rtr01)	Virtual Router, IOS XE - Amsterdam-17.3.4	10.10.20.175	cisco, cisco
R2 (dist-rtr02)	Virtual Router, IOS XE - Amsterdam-17.3.4	10.10.20.176	cisco, cisco
R3 (internet-rtr01)	Virtual Router, IOS XE - Amsterdam-17.3.4	172.21.1.181	cisco, cisco

Task 1: Configure Multiple Devices at Once

In this task, you will experience an out-of-sync issue with one of the managed routers in your lab network. You will resolve the issue by synchronizing the CDB from the device and completing a networkwide configuration change of creating multiple LAN subinterfaces on multiple devices at once in a single transaction.

Activity

For this activity, complete the following steps:

Step 1 Connect to the Student DevBox.

On the topology diagram, click the **Student DevBox**.

Step 2 On the Student DevBox, launch the NSO CLI as admin in the Cisco-style mode.

Open the Terminal application on the Student DevBox and launch the **ncs_cli** as the NSO local user **admin** indicating that you want to use the Cisco style immediately.

```
developer@devbox:~$ ncs_cli -Cu admin


User admin last logged in 2022-06-27T16:28:25.553422+00:00, to student-vm, from 100.65.0.11 using cli-ssh admin connected from 100.65.0.11 using ssh on devbox
admin@ncs#
```

Step 3 In the NSO CLI config mode, create the dot1Q VLAN subinterface configurations in all your managed lab devices simultaneously. Use the parameters from the following table:

Device Configuration Attribute	Value
Interface	GigabitEthernet4.110
Description	LAN
802.1Q VLAN Number	110
IP Subnet Mask	255.255.255.0
R1 IP Address	10.100.21.1
R2 IP Address	10.100.22.1
R3 IP Address	10.100.23.1

In the same Terminal application window or tab, in NSO CLI, switch to the configuration mode by using the **config** command and then enter the following commands line by line:

```
devices device R1
 config
  interface GigabitEthernet4.110
   description LAN
   encapsulation dot1Q 110
   ip address 10.100.21.1 255.255.255.0

devices device R2
 config
  interface GigabitEthernet4.110
   description LAN
   encapsulation dot1Q 112
   ip address 10.100.22.1 255.255.255.0

devices device R3
 config
  interface GigabitEthernet4.110
   description LAN
   encapsulation dot1Q 110
   ip address 10.100.23.1 255.255.255.0
```

In NSO CLI, the configuration must look exactly as in the following example:

```
admin@ncs# config
Entering configuration mode terminal
admin@ncs(config)#
admin@ncs(config)# devices device R1
admin@ncs(config-device-R1)#  config
admin@ncs(config-config)#   interface GigabitEthernet4.110
admin@ncs(config-if)#    description LAN
admin@ncs(config-if)#    encapsulation dot1Q 110
admin@ncs(config-if)#    ip address 10.100.21.1 255.255.255.0
admin@ncs(config-if)#
admin@ncs(config-if)# devices device R2
admin@ncs(config-device-R2)#  config
admin@ncs(config-config)#   interface GigabitEthernet4.110
admin@ncs(config-if)#    description LAN
admin@ncs(config-if)#    encapsulation dot1Q 110
admin@ncs(config-if)#    ip address 10.100.22.1 255.255.255.0
admin@ncs(config-if)#
admin@ncs(config-if)# devices device R3
admin@ncs(config-device-R3)#  config
admin@ncs(config-config)#   interface GigabitEthernet4.110
admin@ncs(config-if)#    description LAN
admin@ncs(config-if)#    encapsulation dot1Q 110
admin@ncs(config-if)#    ip address 10.100.23.1 255.255.255.0
admin@ncs(config-if)#
```

To make changes on the devices in NSO CLI, you must switch to the config mode, then navigate to each device *config* section, and create or modify the desired items.

Note The configurations you enter in the config mode are not immediately sent to the devices. The configuration commands are stored in the candidate config instance, which Device Manager applies to the network once you commit.

Step 4 In the NSO CLI config mode, commit the entered subinterface configurations all at once.

In the same Terminal application window or tab, in the NSO CLI config mode, execute the **commit** command:

```
admin@ncs(config-if)#
admin@ncs(config-if)# commit
Aborted: Network Element Driver: device R3: out of sync
admin@ncs(config-if)# *** ALARM out-of-sync: Device R3 is out of sync
admin@ncs(config-if)#
```

Your transaction will be aborted with the Alarm message from the system notifying you about the router R3 being out of sync. It is expected in this exercise because the configuration from the device R3 has not been synchronized into the CDB after adding that device to the inventory.

Note The default Device Manager behavior is to avoid making inaccurate changes in the network, so it aborts and reverts the transactions that include the nonsynchronized devices.

Step 5 In the NSO CLI config mode, from the top level of the CDB data tree, check the router R3 synchronization state.

In the same Terminal application window or tab, in the NSO CLI config mode, execute the **top** command, and then the **devices device R3 check-sync** command:

```
admin@ncs(config-if)# top
admin@ncs(config)#
admin@ncs(config)# devices device R3 check-sync
result unknown
admin@ncs(config)#
```

The result *unknown* displayed for the device R3 is an indication that the device has never been synchronized after its onboarding.

Note Synchronization between the configuration data in the CDB and the network device configurations is important because of the Device Manager logic, which implies calculating the minimal config diff in every transaction. To calculate a relevant diff, the Device Manager must have a precise copy of the active network config before the transaction.

Step 6 In the NSO CLI config mode, resolve the R3 out-of-sync issue by copying the configuration data *from* the device to the CDB.

In the same Terminal application window or tab, in the NSO CLI config mode, execute the **devices device R3 sync-from** command:

```
admin@ncs(config)# devices device R3 sync-from
result true
admin@ncs(config)#
```

The R3 router is in sync with the CDB now, and you can proceed to complete the original task of configuring multiple subinterfaces on multiple network devices at once.

Step 7 In the NSO CLI config mode, make sure that the subinterface configurations are still present in the candidate config.

In the same Terminal application window or tab, in the NSO CLI config mode, at the top level of the data tree, execute the **show configuration** command:

```
admin@ncs(config)# show configuration
devices device R1
 config
  interface GigabitEthernet4.110
   description LAN
   encapsulation dot1Q 110
   no switchport
   ip address 10.100.21.1 255.255.255.0
   no shutdown
  exit
 !
!
devices device R2
 config
  interface GigabitEthernet4.110
   description LAN
   encapsulation dot1Q 112
   no switchport
   ip address 10.100.22.1 255.255.255.0
   no shutdown
  exit
 !
!
devices device R3
 config
```

```
  interface GigabitEthernet4.110
   description LAN
   encapsulation dot1Q 110
   no switchport
   ip address 10.100.23.1 255.255.255.0
   no shutdown
  exit
 !
!
admin@ncs(config)#
```

The previous transaction attempt has been aborted, and any changes in the network that might have been made in it have been reverted, but the candidate configuration stays in your config mode session.

Note If you cannot see the subinterface configurations in the candidate config or if any of the configurations is incomplete, use the instructions from **Step 3** to add the missing device config items.

It is recommended that you review the candidate configuration and ensure it is correct before committing it and sending it to the network.

Step 8 In the NSO CLI config mode, commit the networkwide configuration change.

In the same Terminal application window or tab, in the NSO CLI config mode, execute the **commit** command with the **LANSUBIF** label and the **"Set All LAN Subinterfaces"** comment:

```
admin@ncs(config)# commit label LANSUBIF comment "Set All LAN Subinterfaces"
Commit complete.
admin@ncs(config)#
```

The recommendation is to always give your transactions meaningful comments and labels.

You have successfully resolved an out-of-sync issue with the router R3 and completed a networkwide configuration change of creating multiple LAN subinterfaces on multiple devices in a single transaction.

Task 2: Troubleshoot Device Out-of-Band Changes

In this task, you will simulate an accidental out-of-sync issue with one of your managed devices. You will log into a router directly and configure it through the router's CLI console, as if you could not use Cisco NSO to accomplish the job. You will then perform the investigation and troubleshooting of the issue.

Activity

For this activity, complete the following steps:

Step 1 From the Student DevBox, open the Terminal application and connect to the **R3** router.

From the Student DevBox, open the Terminal application. Connect to the R3 router (internet-rtr01) by using the **ssh cisco@10.10.20.181** command. When the terminal asks you for the password, use password **cisco**:

```
[developer@devbox nso-run]$ ssh cisco@10.10.20.181
Warning: Permanently added '10.10.20.181' (RSA) to the list of known
hosts.
cisco@10.10.20.181's password: cisco


internet-rtr01#
```

Step 2 In the router **R3** (internet-rtr01) CLI, switch to the configuration mode and add a description on the interface GigabitEthernet 4.

In the router **R3** CLI console, enter the configuration mode by using the **configure terminal** command, and then on the interface **GigabitEthernet 4**, add the description **## TROUBLESHOOTING ##**:

```
internet-rtr01#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Internet-rtr01(config)#interface GigabitEthernet 4
Internet-rtr01(config-if)#description ## TROUBLESHOOTING ##
Internet-rtr01(config-if)#
```

If you make a direct change to a device, it becomes out-of-sync. Because Cisco NSO never actively tracks the configuration status of the devices for out-of-sync changes, you will not know about it until you try and fail to commit a configuration update to the device or when you request a check-sync action.

Step 3 Disconnect from the R3 CLI and connect to NSO CLI.

```
internet-rtr01(config-if)#exit
internet-rtr01(config)#exit
internet-rtr01#exit
Connection to 10.10.20.181 closed by remote host.
Connection to 10.10.20.181 closed.
(py3venv) (py3venv) [developer@devbox nso-run]$ ncs_cli -Cu admin

User admin last logged in 2022-08-23T11:13:29.736232-07:00, to
devbox, from 192.168.254.11 using cli-ssh
admin connected from 192.168.254.11 using ssh on devbox
admin@ncs#
```

Step 4 In the NSO CLI operational mode, check the router R3 synchronization state.

In the same Terminal application window or tab that is left open from the previous Task, in the NSO CLI operational mode, execute the **devices device R3 check-sync** command:

```
admin@ncs# devices device R3 check-sync
result out-of-sync
info got: e36d84c613ddcc413fd05042487f8dc9 expected:
d4fb1874a0fa68edda0890fdb9b89a03
```

Note To successfully complete this and all the proceeding steps, make sure that you have switched from the router R3 console to the **Student DevBox**.

The Cisco NSO **check-sync** action reveals the *fact* of discrepancy by displaying the expected transaction id, which the Device Manager remembers from the most recent NSO transaction, and the actual transaction id, calculated by the check-sync action based on the actual device config state. To see the *difference* between CDB and the device, you need to request another specific action for that—the *compare-config* action.

Step 5 In the NSO CLI operational mode, compare the router R3 actual configuration against its CDB copy.

In the same Terminal application window or tab, in the NSO CLI operational mode, execute the **devices device R3 compare-config** command:

```
admin@ncs# devices device R3 compare-config
diff
```

```
 devices {
     device R3 {
         config {
             interface {
                 GigabitEthernet 4 {
-                    description to;
+                    description "## TROUBLESHOOTING ##";
                 }
             }
         }
     }
 }

admin@ncs#
```

The **compare-config** action reads and parses the entire active configuration from the device and compares it against the CDB device config. The output from the compare-config command in NSO CLI is a diff-like structure that shows the potential outcome of synchronizing *from* the device. The *plus* (+) lines are the items that would be added to the CDB, and the *minus* (-) lines are the items that would be removed from the CDB, should you request the devices device R3 *sync-from* action.

After inspecting the compare-config output, you may decide that the "out-of-band" device change was undesired, and the correct copy of the device config is still the configuration stored in CDB. Therefore, you will synchronize it *to* the device from NSO.

Step 6 In the NSO CLI operational mode, study the expected outcome of synchronizing from the CDB *to* the device.

In the same Terminal application window or tab, in the NSO CLI operational mode, execute the **devices device R3 sync-to dry-run** command:

```
admin@ncs# devices device R3 sync-to dry-run
cli  config {
          interface {
              GigabitEthernet 4 {
     -            description "## TROUBLESHOOTING ##";
     +            description to;
              }
          }
      }
```

The sync-to *dry-run* diff is the towards-device diff, and the *minus* (**-**) lines in it mark the items that will be removed from the device as a result of *sync-to*. If you are satisfied with the expected sync-to outcome, you may then execute it without any flags.

Step 7 In the NSO CLI operational mode, synchronize the configuration from the CDB *to* the device.

In the same Terminal application window or tab, in the NSO CLI operational mode, execute the **devices device R3 sync-to** command:

```
admin@ncs# devices device R3 sync-to
result true
admin@ncs#
```

Note Although it is possible to request sync-to for device ranges, groups, and even the entire device inventory, you must be extremely careful. You should *never* do a **sync-to** from a freshly installed Cisco NSO instance towards a production network because this will effectively wipe your devices blank without an easy way to recover them.

Generally, after adding a device to Cisco NSO, you should immediately synchronize the device configuration. Typically, you would stop making changes to the device directly and switch to using the orchestrator for most device-related tasks.

You have successfully performed the investigation and troubleshooting for a device that was accidentally configured directly by someone who could not use Cisco NSO to accomplish their task.

Task 3: Configure and Apply a Device Template

In this task, you will create a device template configuration to standardize the basic DNS and SNMP settings across the network and learn how to apply the templates to the individual devices and device groups.

Activity

For this activity, complete the following steps:

Step 1 In the NSO CLI config mode, create the device template named **u_DNS_SNMP** that contains unified basic DNS and SNMP configurations for your lab network.

In the same Terminal application window or tab, in NSO CLI, switch to the configuration mode by using the **config** command and then enter the following commands line by line:

```
devices template u_DNS_SNMP
 ned-id cisco-ios-cli-6.67
  config
   ip name-server name-server-list {$DNS_SRV}
   access-list access-list 55
    rule "remark SNMP ACCESS"
    rule "permit {$SNMP_SRV}"
    exit
   snmp-server community c0mmun1t7 RO
    access-list-name 55
```

In NSO CLI, the configuration must look exactly as in the following example, with a single device type section referencing the NED ID **cisco-ios-cli-6.67**:

```
admin@ncs# config
Entering configuration mode terminal
admin@ncs(config)#
admin@ncs(config)#
admin@ncs(config)# devices template u_DNS_SNMP
admin@ncs(config-template-u_DNS_SNMP)#  ned-id cisco-ios-cli-6.67
admin@ncs(config-ned-id-cisco-ios-cli-6.67)# config
admin@ncs(config-config)# ip name-server name-server-list {$DNS_SRV}
admin@ncs(config-name-server-list-{$DNS_SRV})# access-list access-
list 55
admin@ncs(config-access-list-55)# rule "remark SNMP ACCESS"
admin@ncs(config-rule-remark SNMP ACCESS)# rule "permit {$SNMP_SRV}"
admin@ncs(config-rule-permit {$SNMP_SRV})# exit
admin@ncs(config-access-list-55)# snmp-server community c0mmun1t7 RO
admin@ncs(config-community-c0mmun1t7)# access-list-name 55
admin@ncs(config-community-c0mmun1t7)#
```

The DNS and SNMP server IP addresses in the template are configured as variables, **$DNS_SRV**, and **$SNMP_SRV** accordingly. Mind the notation: the variable names must be prefixed with the dollar sign character ($) and put in curly braces {}. The convention is to write variable names in all capitals for better visibility, but there is no formal requirement.

Step 2 In the NSO CLI config mode, at the top level of the CDB data tree, add the **replace** tag to the access-list configuration in the device template.

In the same Terminal application window or tab, in the NSO CLI config mode, execute the **top** command, and then the **tag add** command exactly as in the following example:

```
admin@ncs(config-community-c0mmun1t7)# top
admin@ncs(config)#
admin@ncs(config)# tag add devices template u_DNS_SNMP ned-id cisco-ios-cli-6.67 config access-list access-list 55 replace
```

Throughout the life of the template, you may use it multiple times with the same device but different parameters, for example, when you need to change the DNS and SNMP settings in the network quickly and consistently. Be especially careful with reconfiguring access lists and similar iterable instances so that you do not constantly add new items to them without removing the old, irrelevant values. The **replace** tag in the template is used to avoid that. It is essentially the instruction to replace the entire access list configuration on the device whenever the template is applied.

Step 3 In the NSO CLI config mode, verify that the **replace** tag is present in the device template candidate configuration.

In the same Terminal application window or tab, in the NSO CLI config mode, use the **show configuration** command to make sure that the configuration line **access-list access-list 55** is tagged with the **replace** tag:

```
admin@ncs(config)# show configuration
devices template u_DNS_SNMP
 ned-id cisco-ios-cli-6.67
  config
   ip name-server name-server-list {$DNS_SRV}
   !
   ! Tags: replace
   access-list access-list 55
    rule "remark SNMP ACCESS"
    !
    rule "permit {$SNMP_SRV}"
    !
   !
   snmp-server community c0mmun1t7
    RO
    access-list-name 55
   !
```

```
  !
 !
!
admin@ncs(config)#
```

Step 4 In the NSO CLI config mode, commit the device template configuration.

In the same Terminal application window or tab, in the NSO CLI config mode, execute the **commit** command with the **TDNSNMP** label and the **"Template to Unify DNS/SNMP"** comment:

```
admin@ncs(config)#
admin@ncs(config)# commit label TDNSNMP comment "Template to Unify DNS/SNMP"
Commit complete.
admin@ncs(config)#
```

Step 5 In the NSO CLI config mode, apply the template to the router **R1**. Use the parameters from the following table:

Device Template Variable	Value
DNS_SRV	'10.100.21.251'
SNMP_SRV	'10.100.21.253'

In the same Terminal application window or tab, in the NSO CLI config mode, enter the **devices device R1 apply-template** command exactly as in the following example:

```
admin@ncs(config)# devices device R1 apply-template template-name u_DNS_SNMP variable { name DNS_SRV value '10.100.21.251' } variable { name SNMP_SRV value '10.100.21.253' }
apply-template-result {
    device R1
    result ok
}
```

Note The values that you assign to the template variables must be enclosed in single quotes ' '.

The apply-template action does *not* immediately send the template-generated config to the network, it only adds the configuration commands to your current config session. You are free to further edit that config together with any other items you may have entered or loaded there already. You may also add extra config items on top of anything generated from templates.

Step 6 In the NSO CLI config mode, apply the template to the device group **Branches**. Use the parameters from the following table:

Device Template Variable	Value
DNS_SRV	'10.100.21.251'
SNMP_SRV	'10.100.21.253'

In the same Terminal application window or tab, in the NSO CLI config mode, enter the **devices device-group Branches apply-template** command exactly as in the following example:

```
admin@ncs(config)# devices device-group Branches  apply-template
template-name u_DNS_SNMP variable { name DNS_SRV value
'10.100.21.251' } variable { name SNMP_SRV value '10.100.21.253' }
apply-template-result {
    device R2
    result ok
}
apply-template-result {
    device R3
    result ok
}
```

The most powerful method of applying the templates is using them with the device-groups. That way you can easily and elegantly configure standard features precisely where required.

Step 7 In the NSO CLI config mode, verify the device configurations resulting from the templates.

In the same Terminal application window or tab, in the NSO CLI config mode, use the **show configuration** command to make sure that the DNS and SNMP configurations have been created for the routers R1, R2, and R3 exactly as per the template specification and the provided values:

```
admin@ncs(config)# show configuration
devices device R1
 config
```

```
  ip name-server 10.100.21.251
  access-list 55 remark SNMP ACCESS
  access-list 55 permit 10.100.21.253
  snmp-server community c0mmun1t7 RO 55
 !
!
devices device R2
 config
  ip name-server 10.100.21.251
  access-list 55 remark SNMP ACCESS
  access-list 55 permit 10.100.21.253
  snmp-server community c0mmun1t7 RO 55
 !
!
devices device R3
 config
  ip name-server 10.100.21.251
  access-list 55 remark SNMP ACCESS
  access-list 55 permit 10.100.21.253
  snmp-server community c0mmun1t7 RO 55
 !
!
admin@ncs(config)#
```

Note The access-list CDB configurations generated for the devices are slightly different in structure from how they are defined in the template. This is because the device data models in specific cases work differently for the templates than for the managed devices.

After the templates are applied to devices and groups, and all the extra configuration edits are completed, you can commit the changes to the network.

Step 8 In the NSO CLI config mode, commit the device configurations resulting from the templates.

In the same Terminal application window or tab, in the NSO CLI config mode, execute the **commit** command with the **DEVTMPT** label and the **"Unify DNS/SNMP from Templates"** comment:

```
admin@ncs(config)#
admin@ncs(config)# commit label DEVTMPT comment "Unify DNS/SNMP from Templates"
```

```
Commit complete.
admin@ncs(config)#
```

Note The templates are applied in the **config** mode of the CLI, which generates additional items in the current config session. You must **commit** the changes to modify the configuration of your network according to the template definitions.

You have successfully created a device template configuration and learned how to apply the templates to the individual devices and the device groups.

Task 4: Troubleshoot NSO to Device Communications

In this task, you will simulate an issue in the NSO to device communication and resolve it using the NED southbound traffic tracing feature.

Activity

For this activity, complete the following steps:

Step 1 In the NSO CLI config mode, turn on the NED southbound traffic tracing for the router **R1** and commit the system configuration change.

In the same Terminal application window or tab, in the NSO CLI config mode, enter the **trace pretty** command for the device R1, and commit the configuration with the **R1TRC** label and the **"Enable Tracing for the Router R1"** comment:

```
admin@ncs(config)# devices device R1
admin@ncs(config-device-R1)# trace pretty
admin@ncs(config-device-R1)#
admin@ncs(config-device-R1)# commit label R1TRC comment "Enable
Tracing for the Router R1"
Commit complete.
admin@ncs(config-device-R1)#
```

The **trace pretty** command tells the Device Manager to write all the communications between the NED and the managed device in an enhanced human-friendly format into a plain text file named after the NED and the device itself and located in the *logs* directory.

Step 2 In the NSO CLI config mode, force close any sessions that may still be open from NSO towards the device **R1**.

In the same Terminal application window or tab, in the NSO CLI config mode, enter the **disconnect** command for the device **R1:**

```
admin@ncs(config)# devices device R1
admin@ncs(config-device-R1)# disconnect
```

After being configured and committed, the trace setting only affects new NED sessions towards devices. To ensure getting the trace output from the device R1, you must disconnect any potentially open connections to the device and only proceed to the testing or troubleshooting activities.

Step 3 In the NSO CLI config mode, create a Layer 2 VLAN configuration for the router **R1**.

In the same Terminal application window or tab, in the NSO CLI configuration mode, enter the following commands line by line:

```
devices device R1
 config
  vlan 110
   name LAN
```

In NSO CLI, your candidate configuration must look as follows:

```
admin@ncs(config)# devices device R1
admin@ncs(config-device-R1)# config
admin@ncs(config-config)# vlan 110
admin@ncs(config-vlan)# name LAN
```

Step 4 In the NSO CLI config mode, ensure that the configuration is valid and fits into the CDB schema.

In the same Terminal application window or tab, in the NSO CLI config mode, execute the **commit check** command:

```
admin@ncs(config-vlan)#
admin@ncs(config-vlan)# commit check
```

```
Validation complete
admin@ncs(config-vlan)#
```

The *Validation complete* message confirms that the candidate configuration is in line with the NED data model specifications. The configuration looks valid for the Device Manager just because the Cisco IOS NED is universal, and its data model includes many features that belong to different types of IOS-driven devices, such as routers and switches. No actual device is supposed to support all the features modeled in the NED.

Step 5 In the NSO CLI config mode, verify the device-native command sequence that the NED generates for the managed device based on the candidate configuration.

In the same Terminal application window or tab, in the NSO CLI config mode, execute the **commit dry-run outformat native** command**:**

```
admin@ncs(config-vlan)#
admin@ncs(config-vlan)# commit dry-run outformat native
native {
    device {
        name R1
        data vlan 110
              name LAN
             !
    }
}
admin@ncs(config-vlan)#
```

The **commit dry-run outformat native** command reveals the configuration lines that will be sent to the router R1. You may already get concerned because the virtual *router* R1 in the lab is not supposed to understand and accept Layer 2 switch-like VLAN configurations.

Step 6 In the NSO CLI config mode, try to commit the configuration and inspect the response from the system.

In the same Terminal application window or tab, in the NSO CLI config mode, execute the **commit** command with the **R1VLAN** label and the **"Try Setting VLAN on the Router R1"** comment:

```
admin@ncs(config-vlan)# commit label R1VLAN comment "Try Setting VLAN
on the Router R1"
```

```
Aborted: External error in the NED implementation for device R1:
command: vlan 110
: vlan 110
     ^
% Invalid input detected at '^' marker.
admin@ncs(config-vlan)#
```

Your transaction will be aborted with a brief error message from the NED.

Step 7 In the Linux shell on the Student DevBox, inspect the messages written to the trace file for the router **R1**.

In the same Terminal application window or tab, exit the NSO CLI config mode by using the **end** command or **Ctrl+Z** key combination, and close the NSO CLI session by using the **exit** command or **Ctrl+D** key combination:

```
admin@ncs(config-vlan)# end
admin@ncs# exit
developer@devbox:~$
```

In the Linux shell, navigate to the *logs* directory of your local NSO instance by using the **cd /home/student/nso-run/logs** command. Then view the trace file generated for the router R1 by using the **less ned-cisco-ios-cli-6.67-R1.trace** command:

```
developer@devbox:~$ cd /home/student/nso-run/logs
developer@devbox: ~/nso-run/logs$ less ned-cisco-ios-cli-6.67-R1.trace


>> 26-Jun-2022::04:46:25.333 user: admin/78 thandle 603 hostname nso device R1 CLI NOCONNECT
to R1
<< 26-Jun-2022::04:46:25.653 user: admin/78 thandle 603 hostname nso device R1 CONNECTED 0
h
>> 26-Jun-2022::04:49:37.819 user: admin/78 thandle 603 hostname nso device R1 CLI CONNECT to
R1-172.21.1.21:22 as cisco (Trace=false)
...
<< 26-Jun-2022::04:50:00.107 user: admin/78 thandle 603 hostname nso device R1 INITIALIZED
b7323f058af9518d2dd20d600d542390
>> 26-Jun-2022::04:50:00.108 user: admin/78 thandle 603 hostname nso device R1 PREPARE 0:
vlan 110
 name LAN
!
```

```
<< 26-Jun-2022::04:50:00.110 user: admin/78 thandle 603 hostname nso device R1 SET_TIMEOUT

...

<< 26-Jun-2022::04:50:00.245 user: admin/78 thandle 603 hostname nso device R1 ERROR: External
error in the NED implementation for device R1: command: vlan 110

: vlan 110

     ^

% Invalid input detected at '^' marker.

>> 26-Jun-2022::04:50:00.246 user: admin/78 thandle 603 hostname nso device R1 CLOSE 0: (Pool:
false)

<< 26-Jun-2022::04:50:00.247 user: admin/78 thandle 603 hostname nso device R1 CLOSED

...

<< 26-Jun-2022::04:50:01.728 user: admin/78 thandle 603 hostname nso device R1 SHOW

>> 26-Jun-2022::04:50:01.734 user: admin/78 thandle 603 hostname nso device R1 ABORT 0:


<< 26-Jun-2022::04:50:01.735 user: admin/78 thandle 603 hostname nso device R1 ABORT OK

>> 26-Jun-2022::04:50:01.735 user: admin/78 thandle 603 hostname nso device R1 GET_TRANS_ID

...

<< 26-Jun-2022::04:50:02.304 user: admin/78 thandle 603 hostname nso device R1 TRANS_ID
b7323f058af9518d2dd20d600d542390

>> 26-Jun-2022::04:50:02.311 user: admin/78 thandle 603 hostname nso device R1 CLOSE 0: (Pool:
false)

<< 26-Jun-2022::04:50:02.312 user: admin/78 thandle 603 hostname nso device R1 CLOSED

(END)
```

More details about each stage of NED to device communications, with timestamps, user session and transaction IDs, communication direction, device commands, and responses to them, are available in the device trace file **ned-cisco-ios-cli-6.67-R1.trace**. With all that information, you can precisely pinpoint the issue and see which commands could not be processed by the device and for what reason.

You have successfully resolved the issue in the NSO to device communications using the NED southbound traffic tracing feature, and you have completed all the tasks of this Discovery Lab.

Review Questions

Question 1:

Which NSO action command displays the potential outcome of overwriting the whole device config with the data stored in the CDB?

a) compare-config
b) outformat native
c) sync-from dry-run
d) sync-to dry-run

Answers: The correct answer is **sync-to dry-run**.

Question 2:

Which statement is true about the apply-template action command in Cisco NSO?

a) It configures all the compatible devices in a device group.
b) It immediately modifies the device configurations in the managed network.
c) It requires all devices to be managed by the same network element driver.
d) It requires all devices to be of the same device type.

Answers: The correct answer is **It configures all the compatible devices in a device group**.