Discovery 9: Perform Service Backup and Restore

Introduction

In this activity, you will learn how to create and restore a backup of a service and its configuration. There are different ways to back up a service. Some of the options are as follows:

- Compress the service package directory and copy it to a backup server. This only preserves the service package and not the configuration. Therefore, you also need to export the service configuration data and save it. The activity will focus on this option.
- Use the NSO backup tool to back up the entire CDB and all the packages. This does not allow you to selectively back up individual service models.

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

- Back up a service package.
- Back up service configuration data.
- Restore service configuration data.

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-server	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 Syntax Reference

Command Syntax Reference

This lab guide uses the following conventions for command syntax:

Formatting	Description and Examples
show running	Commands in steps use this formatting.

Formatting	Description and Examples
config	
Example	Type **show running config**
Example	Use the **name** command.
`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.

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.
cd test	Move into folder "test" which is a subfolder of the current directory.
cd /home/student	Move into folder "nso300" 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).

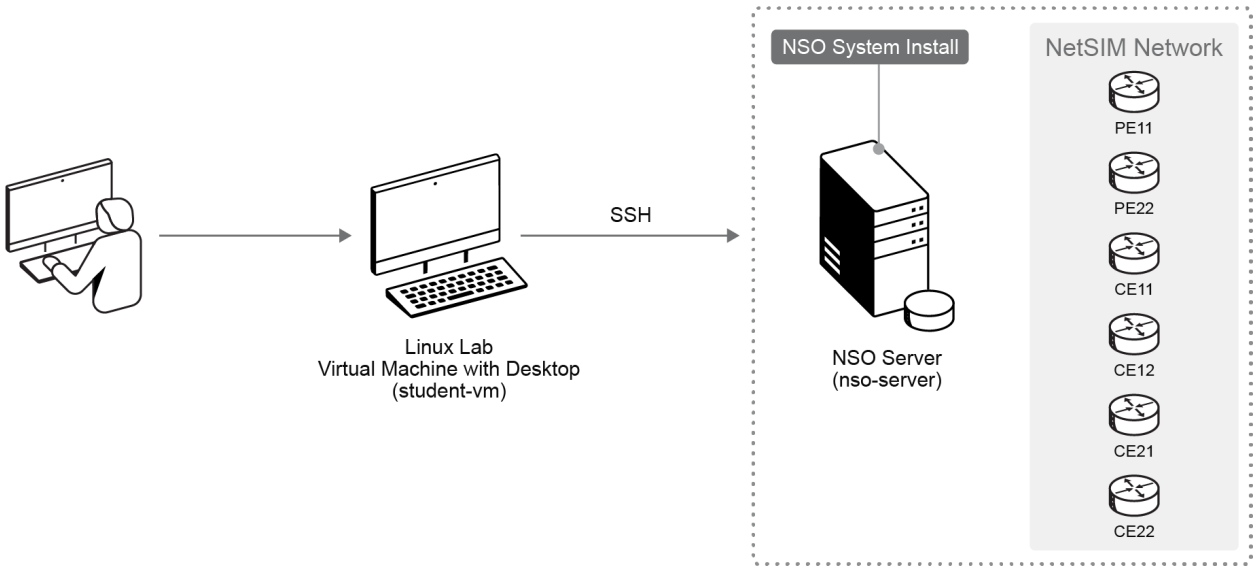
Makefile commands for Docker environment:

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.

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 an NSO server. NSO server includes NetSIM routers and switches. This will be the network that you will orchestrate with your NSO.

Topology



Task 1: Perform Service Package Backup

In this task, you will perform a service package backup by compressing the static-route service for backup purposes.

Activity

Step 1

Open the terminal window and SSH to the NSO server.

You can open the terminal window by using the **Terminal** icon in the bottom bar.

Connect to **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 2

List the currently installed packages.

Currently installed packages are located in the */var/opt/ncs/packages* folder. Use the **ls** command to view the packages (located in folders).

```
student@nso-server:~$ ls /var/opt/ncs/packages/
cisco-ios-cli-6.85  cisco-iosxr-cli-7.41  static-route
student@nso-server:~$
```

Step 3

Create a **~/backup** folder and compress the static-route package into it.

Use the **mkdir** command to create a folder named **backup**. Use the **tar -zxcf <destination> <source>** command to compress the package.

```
student@nso-server:~$ mkdir backup
student@nso-server:~$ tar -zcvf backup/static_route.tar.gz /var/opt/
ncs/package
s/static-route
tar: Removing leading `/' from member names
/var/opt/ncs/packages/static-route/
/var/opt/ncs/packages/static-route/README
/var/opt/ncs/packages/static-route/templates/
/var/opt/ncs/packages/static-route/templates/static-route-template.xml
/var/opt/ncs/packages/static-route/load-dir/
/var/opt/ncs/packages/static-route/load-dir/static-route.fxs
/var/opt/ncs/packages/static-route/package-meta-data.xml
/var/opt/ncs/packages/static-route/python/
/var/opt/ncs/packages/static-route/python/static_route/
/var/opt/ncs/packages/static-route/python/static_route/static_route.py
/var/opt/ncs/packages/static-route/python/static_route/__init__.py
/var/opt/ncs/packages/static-route/src/
/var/opt/ncs/packages/static-route/src/yang/
/var/opt/ncs/packages/static-route/src/yang/static-route.yang
/var/opt/ncs/packages/static-route/src/Makefile
student@nso-server:~$
```


You can create a periodic cron job to perform individual service package backups regularly to ensure you always have a backup of a production package.

Activity Verification

You have completed this task when you attain these results:

- You have backed up the static-route package.

Task 2: Perform Service Configuration Backup

In this task, you will perform a backup of the **static-route** service configuration with the **ncs_load** tool.

Activity

Step 1

Enter the NSO CLI.

Use the **ncs_cli** command.

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

```
User student last logged in 2024-01-26T14:54:35.577522+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#
```

Step 2

Display the service configuration for **static-route** service

Use the **show running-config services static-route** command.

```
student@ncs# show running-config services static-route
services static-route CE11
 route 10.100.0.0/24
  next-hop 10.0.0.1
 !
!
services static-route CE21
 route 10.200.0.0/24
  next-hop 10.0.0.2
 !
!
student@ncs#
```

Step 3

Exit the NSO CLI and study the help page for the **ncs_load** command.

Pay attention to the following flags:

- The **–M** option ensures that the service metadata is also exported.
- The **–F p** option formats the XML file with new-lines and indentations for easier reading.
- The **–U** option unhides all hidden nodes (this is the default option).
- The **–p** option is used to specify the path in the configuration tree to export.

```
student@ncs# exit
student@nso-server:~$ ncs_load -help
A utility that saves and loads the running configuration, usage:
    ncs_load [options] [filename]
    ncs_load -l [options] [filename...]
 Valid options are (for further details, see the manpage):
    -d            debug flag
    -l            load config into NCS [default is to save]
    -j            disable NCS Fastmap during load
    -n            no networking, just apply the data to NCS
    -M            include NCS service-meta-data attributes
    -F  <format>  format can be one of:
                    x  XML [default]
                    p  Pretty XML
                    o  JSON
                    j  J-style CLI
```

```
                    c  C-style CLI
                    i  I-style CLI
                    t  C-style using turbo parser. Only applicable
                       for load config into NCS
    -W              include default values
    -S              include default values as comments
    -m              when loading config, merge [default is delete and
replace]
    -r              when loading config, replace [default is delete and
replace]
    -H              hide all hidden nodes [default depends on
transaction]
    -U              unhide all hidden nodes [default depends on
transaction]
    -a              when loading 'c' or 'i' config, commit after each
line
    -e              when loading, do not abort on errors
    -p  <path>      when saving the path to save
                    when loading this path will first be deleted
    -N              when saving, do Not include parents to the path
    -P  <xpath>     when saving apply this xpath filter
    -D              do maapi_delete_all(MAAPI_DEL_ALL) before loading
    -o              when saving include operational data
                    when loading ignore operational data
    -A              when saving, only include nodes for which the user
                    has read_write access
    -u  <user>      use this user
    -g  <group>     use this group (may be repeated)
    -c  <context>   use this context [default is 'system']
    -i              attach to init session instead of starting new
session
    -s              start transaction towards startup [default is
running]
    -O              when saving, include only operational data, not
config
                    when loading, start operational transaction (load
oper
                    data via a transaction instead of via CDB)
    -b              when saving, include only data stored in CDB
    -t              measure how long the requested command takes
    -R              generate CDB operational subscription notifications
    -w              enable "delayed when" mode of transaction
student@nso-server:~$
```

Step 4

Export the **static-route** service configuration with the **ncs_load** command to the backup folder. Use the flags that you saw in the previous step.

Use the flags that you saw in the previous step.

```
student@nso-server:~$ ncs_load -M -F p -U -p /services/static-route -u
student backup/static-route.xml
student@nso-server:~$
```

Step 5

Check the content of the backup/static-route.xml file.

Use the **cat backup/static-route.xml** command.

```
student@nso-server:~$ cat backup/static-route.xml
<config xmlns="http://tail-f.com/ns/config/1.0">
  <services xmlns="http://tail-f.com/ns/ncs">
  <static-route xmlns="http://example.com/static-route">
    <device>CE11</device>
    <private>
      <diff-
set>g1AAAAD0eAFlj8EKwjAMhjOdOvYE3j13YwdFdhWfY5RmtZGuHW2nj69td/BgAj8/
H+FPoiqE
GknKwQceRg0AxRSlvN277pkqoySV2qqS1xw5cIjukzu5OhFeIBxpZs4uYWTSujd3SObBNPmA
sMs8hyE0gyAvLCPrmdDELs313P+j0+JMH2cQNjQj7IU1kh7qd2BkOL5IrHfH/SqEuW/
bwEkz
2Qg7tca3RsSEwzro40df4AVHGg==</diff-set>
      <forward-diff-
set>g1AAAAD5eAFljs0OgjAQhJcfRfEFvHsuhIPGcDU+B2m6rV0CLWmLPr4WOJq9zH6ZzIxOEU4K
SanOBx7kAACJzhAK4WT8cQEHnemclxw5cIjqu96iyoXwBOFME3N2DpIp6z7cIZkXG8gHhN3K
12CEqhPkhWVkPRMDsVt1v7b/6DI700YPQkoTwl5Yo+ilkzGuyR/
PpokM5ZvENjj26xCmtq4D
p4GpStixNr42IiYUm9H3CMfZoFRkJPY/AKlLBA==</forward-diff-set>
      <device-list>CE11</device-list>
      <ned-id-list>cisco-ios-cli-6.85:cisco-ios-cli-6.85</ned-id-list>
      <latest-commit-
params>g2wAAAABaAJkAAh0cmFjZV9pZG0AAAAkNGMzODkzOWEtMzlkZC00YjYzLWI0NzEtOGFkZmJl
Nzk2MzY4ag==</latest-commit-params>
      <latest-u-
info>g2gJZAANbGF0ZXN0X3VfaW5mb2QAA2NsaWQAA3NzaG0AAAAHc3R1ZGVudGgEYQphAGEAYWZi
AADorm0AAAAAZAAJdW5kZWZpbmVkbAAAAAxtAAAAA2FkbW0AAAAFY2Ryb21tAAAAA2RpcG0A
AAAGZG9ja2VybQAAAAdscGFkbWlubQAAAANseGRtAAAACG5jc2FkbWlubQAAAAdwbHVnZGV2
bQAAAApzYW1iYXNoYXJlbQAAAAdzdHVkZW50bQAAAARzdWRvbQAAAAl3aXJlc2hhcmtq</
latest-u-info>
    </private>
    <route>
      <ip-prefix>10.100.0.0/24</ip-prefix>
      <next-hop>10.0.0.1</next-hop>
    </route>
  </static-route>
  <static-route xmlns="http://example.com/static-route">
    <device>CE21</device>
    <private>
      <diff-set>g1AAAAD0eAFlj8EKwjAMhuOcOvYE3j13RUGRXcXnGKVZ18jWjrXTV/
SttO0OHkzg5+cj/El0
gVAiKdU4L3zbA8BqCJLf7qfjI1ZCUQq91rkoxVuAgOA+qaMrIxEZwp5GNtnZt0zZ6SUmJNOx
npxH2CSewhCqRpKTlpF1TPbELtX1XP+jwzyZOswgZDQibKU1ijr9OzAwbJ8kl7vDfu39WHPu
BfVMVdIO3DhuZEjYLYMufPQFLTJHgQ==</diff-set>
      <forward-diff-set>g1AAAAD5eAFljs0KwjAQhLfW3/
oC3j2nRUGRXsXnkCWbNCttUpJUX9G30rQeZS+zH8PMmBnB
VhNrfQ8Ro2oBIDM5wUp6lX4awdrkZo4FvhEQkvpMN6piJJgydtwL74aohHb+hZ7YNqLlEAkW
E5+CCcq75CCdYBeEbFmcy8up/
kf7wds6eQhm3BMspbOaG5N1ac38ejseEiP1ZPkbnPpNjH1d
VRG5FbqUrqtsqKxMCaufMTwINoMlpdkqenwBUVBLag==</forward-diff-set>
      <device-list>CE21</device-list>
      <ned-id-list>cisco-ios-cli-6.85:cisco-ios-cli-6.85</ned-id-list>
```

```
      <latest-commit-
params>g2wAAAABaAJkAAh0cmFjZV9pZG0AAAAkNGMzODkzOWEtMzlkZC00YjYzLWI0NzEtOGFkZmJl
Nzk2MzY4ag==</latest-commit-params>
      <latest-u-
info>g2gJZAANbGF0ZXN0X3VfaW5mb2QAA2NsaWQAA3NzaG0AAAAHc3R1ZGVudGgEYQphAGEAYWZi
AADorm0AAAAZAAJdW5kZWZpbmVkbAAAAAxtAAAAA2FkbW0AAAAFY2Ryb21tAAAAA2RpcG0A
AAAGZG9ja2VybQAAAAdscGFkbWlubQAAAANseGRtAAAACG5jc2FkbWlubQAAAAdwbHVnZGV2
bQAAAApzYW1iYXNoYXJlbQAAAAdzdHVkZW50bQAAAARzdWRvbQAAAAl3aXJlc2hhcmtq</
latest-u-info>
    </private>
    <route>
      <ip-prefix>10.200.0.0/24</ip-prefix>
      <next-hop>10.0.0.2</next-hop>
    </route>
  </static-route>
  </services>
</config>
student@nso-server:~$
```

Activity Verification

You have completed this task when you attain these results:

- You have backed up the static-route service configuration.

Task 3: Restore the Service Configuration Data

In this task, you will first delete all the **static-route** service instances to simulate a loss of data. Then, you will test the service configuration restore procedure.

Activity

Step 1

Enter the NSO CLI.

Use the **ncs_cli** command.

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

User student last logged in 2024-01-26T17:37:17.815428+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#
```

Step 2

Enter the configuration mode and delete all the **static-route** service instances without removing the corresponding configuration on managed devices.

You can use the commands that are used in the following output.

```
student@ncs# config
Entering configuration mode terminal
```

```
student@ncs(config)# no services static-route
student@ncs(config)# commit ?
Possible completions:
  and-quit                Exit configuration mode
  check                   Validate configuration
  comment                 Add a commit comment
  commit-queue            Commit through commit queue
  label                   Add a commit label
  no-confirm              No confirm
  no-deploy               Commit without invoking the service create
method
  no-lsa                  Commit and do not use LSA
  no-networking           Send nothing to the devices
  no-out-of-sync-check    Commit even if out of sync
  no-overwrite            Do not overwrite modified data on the device
  no-revision-drop        Fail if device has too old data model
  reconcile               Reconcile existing configuration data
  rollback-id             Display rollback-id for commit
  save-running            Save running to file before performing the
commit
  trace-id                Use the given trace-id while performing the
commit.
  use-lsa                 Commit and use LSA where applicable
  wait-device             Wait for devices before entering critical
section
  ---
  dry-run                 Show the diff but do not perform commit
  <cr>
student@ncs(config)# commit no-networking
Commit complete.
student@ncs(config)#
```

Step 3

Exit the configuration mode and verify that the service configuration is gone.

To verify that the service configuration is gone, use the **show running-config** command.

```
student@ncs(config)# exit
student@ncs# show running-config services static-route
% No entries found.
student@ncs#
```

Step 4

Exit the NSO CLI and study the help page for **ncs_load** command again.

To display the help page, use the **ncs_load** command with the **-help** flag.

Pay attention to the following flags this time:

- The **-n** flag uses the no-networking option for commit.
- The **-l** option loads the configuration into the CDB.
- The **-m** option merges the XML configuration with the existing data in the CDB.

```
student@ncs# exit
student@nso-server:~$ ncs_load -help
A utility that saves and loads the running configuration, usage:
    ncs_load [options] [filename]
    ncs_load -l [options] [filename...]
 Valid options are (for further details, see the manpage):
    -d            debug flag
    -l            load config into NCS [default is to save]
    -j            disable NCS Fastmap during load
    -n            no networking, just apply the data to NCS
    -M            include NCS service-meta-data attributes
    -F  <format>  format can be one of:
                    x  XML [default]
                    p  Pretty XML
•       JSON
                    j  J-style CLI
                    c  C-style CLI
                    i  I-style CLI
                    t  C-style using turbo parser. Only applicable
                       for load config into NCS
    -W            include default values
    -S            include default values as comments
    -m            when loading config, merge [default is delete and
replace]
    -r            when loading config, replace [default is delete and
replace]
    -H            hide all hidden nodes [default depends on
transaction]
    -U            unhide all hidden nodes [default depends on
transaction]
    -a            when loading 'c' or 'i' config, commit after each
line
    -e            when loading, do not abort on errors
    -p  <path>    when saving the path to save
                  when loading this path will first be deleted
    -N            when saving, do Not include parents to the path
    -P  <xpath>   when saving apply this xpath filter
    -D            do maapi_delete_all(MAAPI_DEL_ALL) before loading
    -o            when saving include operational data
                  when loading ignore operational data
    -A            when saving, only include nodes for which the user
                  has read_write access
    -u  <user>    use this user
    -g  <group>   use this group (may be repeated)
    -c  <context> use this context [default is 'system']
    -i            attach to init session instead of starting new
session
    -s            start transaction towards startup [default is
running]
    -O            when saving, include only operational data, not
config
                  when loading, start operational transaction (load
oper
                  data via a transaction instead of via CDB)
    -b            when saving, include only data stored in CDB
    -t            measure how long the requested command takes
```

```
    -R            generate CDB operational subscription notifications
    -w            enable "delayed when" mode of transaction

student@nso-server:~$
```

Step 5

Load the backed-up static-route configuration into the CDB.

Use the **ncs_load** command to perform the load of the backed-up **static-route** configuration data.

It is very important to make sure that you use the "**-m**" flag to merge the configuration instead of replacing it. If you forget to use this option, you remove the entire CDB configuration data and replace it with only the service configuration data. This, of course, breaks the NSO system because it loses all devices and other configuration data.

```
student@nso-server:~$ ncs_load -l -m -j -M -U -n -u student backup/
static-route.xml
student@nso-server:~$
```

Step 6

Verify that the configuration has been imported.

Enter the NSO CLI again and verify that the configuration has been imported with the **show running-config** command.

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

student connected from 10.30.0.102 using ssh on student-vm
student@ncs# show running-config services static-route
services static-route CE11
  route 10.100.0.0/24
    next-hop 10.0.0.1
  !
!
services static-route CE21
  route 10.200.0.0/24
    next-hop 10.0.0.2
  !
!
student@ncs#
```

Step 7

Check services sync status with the **services check-sync** command.

As you can see, the service and device configuration are not in-sync. The deletion of the service resulted in the deletion of the corresponding device configuration in the CDB even though you used the no-networking option (this only preserved the real

configuration on the managed devices).

```
student@ncs# services check-sync
sync-result {
    service-id /ncs:services/static-route:static-route[static-
route:device='CE11']
    in-sync false
}
sync-result {
    service-id /ncs:services/static-route:static-route[static-
route:device='CE21']
    in-sync false
}
student@ncs#
```

Step 8

Re-deploy the imported services with the no-networking option because you want to restore service configuration without disrupting the actual configuration on the managed devices.

To perform the re-deploy, use the **re-deploy** action.

```
student@ncs# services static-route CE11 re-deploy no-networking
student@ncs#
System message at 2024-01-26 18:05:32...
Commit performed by student via ssh using cli.
student@ncs# services static-route CE21 re-deploy no-networking
student@ncs#
System message at 2024-01-26 18:05:38...
Commit performed by student via ssh using cli.
student@ncs#
```

Step 9

Check the sync status of the services again.

The services should now be in-sync.

```
student@ncs# services check-sync
sync-result {
    service-id /ncs:services/static-route:static-route[static-
route:device='CE11']
    in-sync true
}
sync-result {
    service-id /ncs:services/static-route:static-route[static-
route:device='CE21']
    in-sync true
}
student@ncs#
```

Step 10

Check the sync status of the devices.

The devices are out-of-sync too, due to the no-networking commits.

```
student@ncs# devices check-sync
sync-result {
    device CE11
    result out-of-sync
    info Out of sync due to no-networking or failed commit-queue
commits
}
sync-result {
    device CE12
    result in-sync
}
sync-result {
    device CE21
    result out-of-sync
    info Out of sync due to no-networking or failed commit-queue
commits
}
sync-result {
    device CE22
    result in-sync
}
sync-result {
    device PE11
    result in-sync
}
sync-result {
    device PE22
    result in-sync
}
student@ncs# *** ALARM out-of-sync: Out of sync due to no-networking or
failed commit-queue commits
student@ncs# *** ALARM out-of-sync: Out of sync due to no-networking or
failed commit-queue commits
student@ncs#
```

Step 11

Sync the devices.

Use the **devices sync-from** command.

```
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#
```


To preview what configuration will be altered before you actually sync the devices, you can use the **devices sync-from dry-run** command. Similarly, **dry-run** can also be applied to the **devices sync-to dry-run** command.

Activity Verification

You have completed this task when you attain these results:

- You have successfully restored the service configuration.

Which flag of the **ncs_load** command will merge the configuration you are importing with the CDB?

- **ncs_load -e**
- **ncs_load -j**
- **ncs_load -m**
- **ncs_load -n**