Discovery 8: Deploy LSA Services

Introduction

In this activity, you will deploy an *l3vpn* service that uses a layered services architecture. You will also learn how to configure a basic LSA environment that consists of a customer-facing NSO and two additional resource (device)-facing NSO servers, in this case, one for each geographical region (*am*—Americas, *emea*—Europe, Middle East, and Africa).

After completing this activity, you will be able to:

- Configure a basic NSO LSA environment.
- Deploy an LSA service.
- Understand LSA implications.

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 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.

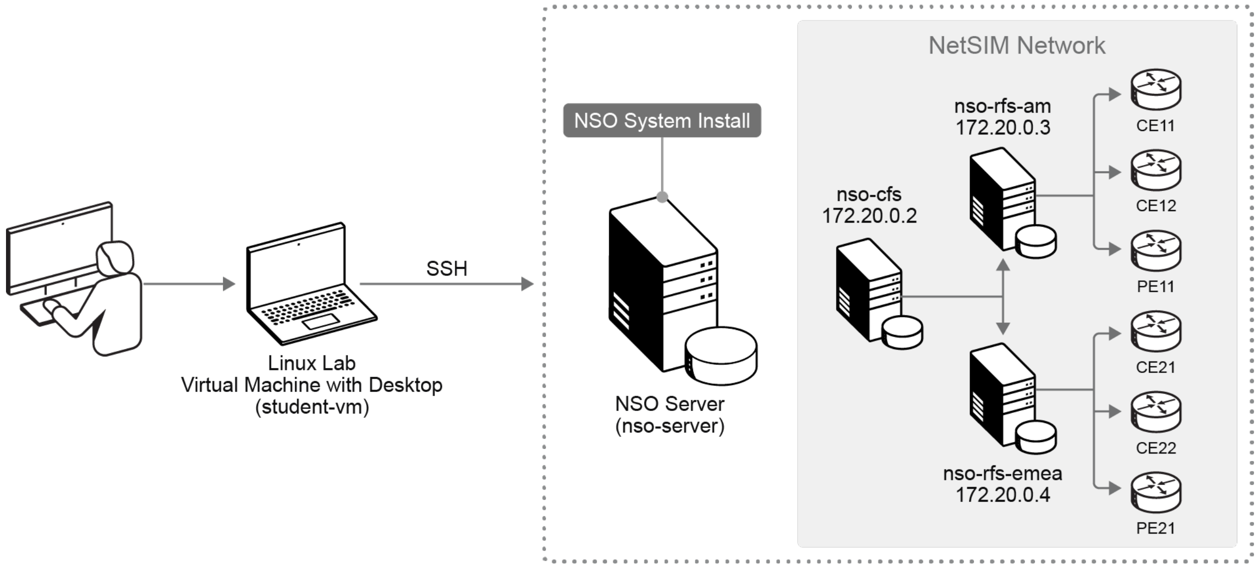
Lab Topology Information

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. The general topology is your lab environment with a Linux server (Student-VM). On the Linux server within that topology is your second topology, a Docker environment, which consists of an NSO Docker image with your NSO installation, together with numerous Docker images of NetSim routers and switches that are logically grouped into a network topology. This is the network that you will orchestrate with your NSO.

The network topology is designed in a way to cover both Service Provider and Enterprise use cases. It is a simulated NetSim network—devices have no Control or

Data Plane. Devices will, however, accept or reject configuration sent by the NSO, just as real devices would.

Topology



Task 1: Configure RFS Nodes

In this task, you will configure and explore the resource-facing NSO nodes. These nodes are responsible for deploying the device configuration to the actual network devices.

The final solutions for this lab are in the ~/packages directory. You can use it for copy-pasting longer pieces of code and as a reference point for troubleshooting your packages.

Activity

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 the terminal window.

Open the terminal window by clicking the **Terminal** icon in 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 that 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

Navigate to the **~/nso-lsa/nso-cfs** directory.

This is the location of the Docker project and is used to start the containers.

```
student@nso-server:~/ $ cd nso-lsa/nso-cfs
student@nso-server:~/nso-lsa/nso-cfs$
```

Step 5

Start the Docker test environment.

The **make testenv-start** command will create docker containers for LSA environment.

There are three NSO containers, together with six NetSim device containers. The **testenv-nso-cfs-6.1-student-nso** container hosts the upper, customer-facing node, and the **testenv-nso-rfs-6.1-student-nso-am** and **testenv-nso-rfs-6.1-student-nso-emea** containers host the lower, resource-facing nodes. Each RFS node is designated to a specific geographical region (*am*—Americas, *emea*—Europe, Middle East, and Africa).

```
student@nso-server:~/nso-lsa/nso-cfs$ make testenv-start
docker network inspect testenv-nso-cfs-6.1-student >/dev/null 2>&1 ||
docker network create lsa --subnet=172.21.0.0/24
f3cc6fe48b6459be2de1b959a0032ad0a30aa265adbb4165e8efe32802efb6e1
docker run -td --name testenv-nso-cfs-6.1-student-nso --network-alias
nso-cfs --label lsa --network lsa --label testenv-nso-cfs-6.1-student
--label nidtype=nso --volume /var/opt/ncs/packages -e SSH_PORT=8822 -p
8022:22 -e ADMIN_PASSWORD=admin ${NSO_EXTRA_ARGS} --ip=172.21.0.2
nso303.gitlab.local/nso-cfs/nso:6.1-student
93e07615affb6620f229658e1e91dcb0e453bf5d4363c66572ff88207349709f
make testenv-start-extra
make[1]: Entering directory '/home/student/nso-lsa/nso-cfs'

== Starting repository specific testenv
docker run -td --name testenv-nso-rfs-6.1-student-nso-am --network-
alias nso-rfs-am --network lsa --ip=172.21.0.3 --label lsa --label
testenv-nso-rfs-6.1-student --label nidtype=nso --volume /var/opt/ncs/
packages -e ADMIN_PASSWORD=admin -e SSH_PORT=8022 nso303.gitlab.local/
nso-rfs/nso:6.1-student
```

```
70dab8e3303e1593e402ccd0ffc533a14144bd06bcd680357b735e55ff47dc62
docker run -td --name testenv-nso-rfs-6.1-student-nso-emea --network-
alias nso-rfs-emea --network lsa --ip=172.21.0.4 --label lsa --label
testenv-nso-rfs-6.1-student --label nidtype=nso --volume /var/opt/ncs/
packages -e ADMIN_PASSWORD=admin -e SSH_PORT=8022 nso303.gitlab.local/
nso-rfs/nso:6.1-student
671979e03cfdf3cce0626e6c80b7ecb6c895e324e8b8bfb140498dc5a5c7878e
docker run -td --name testenv-CE11-am-6.1-student --network-alias CE11-
am --network lsa --label lsa nso303.gitlab.local/ned-ios/netsim:6.1-
student
397f3975dc51231ad95ac07d20cb279bef3acd1dae46626101fe79286e2ac302

...
...

docker exec -t testenv-nso-rfs-6.1-student-nso-emea bash -lc 'echo -e
"request devices fetch-ssh-host-keys" | ncs_cli -Ju admin'
fetch-result {
    device CE21-EMEA
    result updated
    fingerprint {
        algorithm ssh-ed25519
        value 98:ee:0c:b6:9d:87:62:96:d3:99:73:b9:2d:b2:3c:5e
    }
}
fetch-result {
    device CE22-EMEA
    result updated
    fingerprint {
        algorithm ssh-ed25519
        value 98:ee:0c:b6:9d:87:62:96:d3:99:73:b9:2d:b2:3c:5e
    }
}
fetch-result {
    device PE21-EMEA
    result updated
    fingerprint {
        algorithm ssh-ed25519
        value 98:ee:0c:b6:9d:87:62:96:d3:99:73:b9:2d:b2:3c:5e
    }
}
docker exec -t testenv-nso-rfs-6.1-student-nso-emea bash -lc 'echo -e
"request devices sync-from" | ncs_cli -Ju admin'
sync-result {
    device CE21-EMEA
    result true
}
sync-result {
    device CE22-EMEA
    result true
}
sync-result {
    device PE21-EMEA
    result true
}
make[2]: Leaving directory '/home/student/nso-lsa/nso-cfs'
make[1]: Leaving directory '/home/student/nso-lsa/nso-cfs'
docker exec -t testenv-nso-cfs-6.1-student-nso bash -lc 'ncs --wait-
```

```
started 600'
docker exec -t testenv-nso-cfs-6.1-student-nso bash -lc 'service ssh
start'
[ ok ] Starting OpenBSD Secure Shell server: sshd.
student@nso-server:~/nso-lsa/nso-cfs$
```

Step 6

Navigate to the **nso-rfs** directory.

This is the location of the Docker project and is used to manage RFS containers.

```
student@student-vm:~/nso-lsa$ cd ../nso-rfs
student@student-vm:~/nso-lsa/nso-rfs$
```

Step 7

Enter the **am** Docker container shell.

Use the **make testenv-shell NSO=am** command.

```
student@nso-server:~/nso-lsa/nso-rfs$ make testenv-shell NSO=am
docker exec -it testenv-nso-rfs-6.1-student-nso-am bash -l
root@70dab8e3303e:/#
```

Step 8

Make sure that the northbound NETCONF communication is enabled.

Open the NSO configuration, located at **/etc/ncs/ncs.conf**. Make sure that the northbound NETCONF communication is enabled.

```
root@70dab8e3303e:/# vim /etc/ncs/ncs.conf
...
  <netconf-north-bound>
    <enabled>true</enabled>
    <transport>
      <ssh>
        <enabled>true</enabled>
        <ip>0.0.0.0</ip>
        <port>830</port>
        ...
      </ssh>
      ...
    </transport>
  </netconf-north-bound>
...
```

Step 9

Change the CLI prompt for this AM NSO instance.

Change the CLI by editing the CLI portion of the configuration. Add the **am** suffix to the CLI prompt strings.

```
...
   <cli>
    <enabled>true</enabled>
    <!-- Use the builtin SSH server -->
    <ssh>
      ...
    </ssh>
    <prompt1>\u@ncs-am&gt; </prompt1>
    <prompt2>\u@ncs-am% </prompt2>
    <c-prompt1>\u@ncs-am# </c-prompt1>
    <c-prompt2>\u@ncs-am(\m)# </c-prompt2>
    <restricted-file-access>true</restricted-file-access>
    <show-commit-progress>true</show-commit-progress>
    <suppress-commit-message-context>maapi</suppress-commit-message-
context>
    <suppress-commit-message-context>system</suppress-commit-message-
context>
  </cli>
...
```

Step 10

Save the file and exit the file editor.

To exit the edit mode, press the **Esc** key, and then use the **:wq** command and press the **Enter** key to save the file and exit the file editor.

Step 11

Apply the configuration changes.

Use the **ncs --reload** command.

This command applies the configuration changes done to the **ncs.conf** file that you just edited. This configuration change only applies to the current NSO Docker environment since a configuration file does not persist outside of it. If the environment is stopped, the configuration changes get lost. Persistent configuration changes are done through startup scripts and are outside of the scope of the current activity.

```
root@70dab8e3303e:/# ncs --reload
root@70dab8e3303e:/#
```

Step 12

Exit the NSO container shell.

Use the **exit** command.

```
root@70dab8e3303e:/# exit
logout
```

```
student@nso-server:~/nso-lsa/nso-rfs$
```

Step 13

Review the prompt of the **am** container.

Enter the NSO CLI of the **am** container. Observe how the NSO CLI prompt has changed. Then, exit the NSO CLI.

```
student@nso-server:~/nso-lsa/nso-rfs$ make testenv-cli NSO=am
docker exec -it testenv-nso-rfs-6.1-student-nso-am bash -lc 'ncs_cli -
Cu admin'

User admin last logged in 2024-02-09T04:45:27.817897+00:00, to
70dab8e3303e, from 127.0.0.1 using cli-console
admin connected from 127.0.0.1 using console on 70dab8e3303e
admin@ncs-am# exit
student@nso-server:~/nso-lsa/nso-rfs$
```

Step 14

Repeat the steps for the **emea** container.

Repeat steps 8 to 12 for the **emea** container.

Step 15

Enter the NSO CLI of the AM RFS node and display the devices that are added to NSO.

As the name tells you, these devices are in the *am* (Americas) region. This NSO node will manage only the devices located in this region. LSA nodes are not necessarily divided geographically. Different nodes can take care of different customers, vendors, versions, and so on. They can also simply be used to divide the load between several nodes.

```
student@nso-server:~/nso-lsa/nso-rfs$ make testenv-cli NSO=am
docker exec -it testenv-nso-rfs-6.1-student-nso-am bash -lc 'ncs_cli -
Cu admin'

User admin last logged in 2024-02-09T04:55:10.661502+00:00, to
70dab8e3303e, from 127.0.0.1 using cli-console
admin connected from 127.0.0.1 using console on 70dab8e3303e
admin@ncs-am# show devices brief
NAME     ADDRESS  DESCRIPTION  NED ID
---------------------------------------------------
CE11-AM  CE11-AM  -            cisco-ios-cli-6.85
CE12-AM  CE12-AM  -            cisco-ios-cli-6.85
PE11-AM  PE11-AM  -            cisco-iosxr-cli-7.41
admin@ncs-am# exit
student@nso-server:~/nso-lsa/nso-rfs$
```

Step 16

Display the devices added to the emea RFS node.

Use the commands provided in the following output.

```
student@nso-server:~/nso-lsa/nso-rfs$ make testenv-cli NSO=emea
docker exec -it testenv-nso-rfs-6.1-student-nso-emea bash -lc 'ncs_cli
-Cu admin'

User admin last logged in 2024-02-09T04:39:47.256721+00:00, to
671979e03cfd, from 127.0.0.1 using cli-console
admin connected from 127.0.0.1 using console on 671979e03cfd
admin@ncs-emea# show devices brief
NAME       ADDRESS    DESCRIPTION  NED ID
-------------------------------------------------------
CE21-EMEA  CE21-EMEA  -            cisco-ios-cli-6.85
CE22-EMEA  CE22-EMEA  -            cisco-ios-cli-6.85
PE21-EMEA  PE21-EMEA  -            cisco-iosxr-cli-7.41
admin@ncs# exit
student@nso-server:~/nso-lsa/nso-rfs$
```

Activity Verification

You have completed this task when you attain these results:

- You have configured both RFS nodes.

Task 2: Configure CFS Node

In this task, you will configure the customer-facing node for the LSA architecture. This is the "upper" node that an NSO administrator interacts with and is responsible for deploying service configuration to the resource-facing nodes below.

Activity

Step 1

Navigate to the **~/nso-lsa/nso-cfs** directory.

Use the **cd ../nso-cfs** command.

```
student@nso-server:~/nso-lsa/nso-rfs$ cd ../nso-cfs
student@nso-server:~/nso-lsa/nso-cfs$
```

Step 2

Open and study the YANG model for the *l3vpn-cfs* service.

Examine the model. Note how it is very similar to the RFS model and how the device parameter is a leaf and not a leafref. This is because the network devices are added to RFS nodes and therefore cannot be referenced from the CFS node.

Also, a *dispatch-map* list is present, which is used to map the lower RFS nodes to the geographical region.

```
student@nso-server:~/nso-lsa/nso-cfs$ cat packages/l3vpn-cfs/src/yang/
l3vpn-cfs.yang
module l3vpn-cfs {
  namespace "http://cisco.com/example/l3vpn-cfs";
  prefix l3vpn-cfs;

  import ietf-inet-types { prefix inet; }
  import tailf-common { prefix tailf; }
  import tailf-ncs { prefix ncs; }

  list dispatch-map {
    key region;
    tailf:cli-suppress-mode;

    leaf region {
      type string;
    }
    leaf rfs-node {
      tailf:cli-drop-node-name;
      type leafref {
        path "/ncs:devices/ncs:device/ncs:name";
      }
    }
  }

  list l3vpn-cfs {
    description "This is an L3VPN cfs service model.";

    key vpn-name;
    leaf vpn-name {
      tailf:info "Unique L3 VPN cfs service instance id";
      tailf:cli-allow-range;
      type string;
    }

    uses ncs:service-data;
    ncs:servicepoint l3vpn-cfs-servicepoint;

    list link {
      key id;
      leaf id {
        tailf:info "Unique L3VPN link id";
        tailf:cli-allow-range;
        type string;
      }

      leaf device {
        tailf:info "Device";
        type string;
      }

      leaf interface {
        tailf:info "Customer Facing Interface";
        type string;
      }
      leaf ip-address {
         tailf:info "Remote IP Address";
         type inet:ipv4-address;
      }

      leaf mask {
        tailf:info "Subnet mask for Remote IP Address";
        type inet:ipv4-address;
      }
    }
  }
}
```

Step 3

Display the XML template for the **l3vpn-cfs** service.

The file should appear like the following example when you open it for the first time. In addition to the normal service parameters, you will need to set the correct RFS-NODE to which the service configuration will be dispatched. To see how the RFS node is mapped, you can check the Python code, located in the **packages/l3vpn-cfs/python** folder.

```
student@nso-server:~/nso-lsa/nso-cfs$ cat packages/l3vpn-cfs/templates/
l3vpn-cfs-template.xml
<config-template xmlns="http://tail-f.com/ns/config/1.0">
  <devices xmlns="http://tail-f.com/ns/ncs">
    <device>
      <name>{$RFS-NODE}</name>
      <config>
        <l3vpn-rfs xmlns="http://cisco.com/example/l3vpn-rfs">
          <vpn-name>{string(../vpn-name)}</vpn-name>
          <link>
            <id>{$ID}</id>
            <device>{$DEVICE}</device>
            <interface>{$INTERFACE}</interface>
            <ip-address>{$IP-ADDRESS}</ip-address>
            <mask>{$MASK}</mask>
          </link>
        </l3vpn-rfs>
      </config>
    </device>
  </devices>
</config-template>
student@nso-server:~/nso-lsa/nso-cfs$
```

Step 4

Enter the NSO CLI.

Use the **make testenv-cli** command.

```
student@nso-server:~/nso-lsa/nso-cfs$ make testenv-cli
docker exec -it testenv-nso-cfs-6.1-student-nso bash -lc 'ncs_cli -Cu
```

```
admin'

User admin last logged in 2024-02-09T05:05:47.608517+00:00, to
93e07615affb, from 127.0.0.1 using cli-console
admin connected from 127.0.0.1 using console on 93e07615affb
admin@ncs#
```

Step 5

Add both RFS nodes as devices.

Make sure you add additional ssh-algorithms to avoid SSH key echange problems. You can use the **lsa** authentication group for remote authentication.

```
admin@ncs# config
admin@ncs(config)# devices global-settings ssh-algorithms public-key [
ssh-ed25519 ecdsa-sha2-nistp256 ecdsa-sha2-nistp384 ecdsa-sha2-nistp521
rsa-sha2-512 rsa-sha2-256 ssh-rsa ssh-dss ]
admin@ncs(config)# devices device NSO-RFS-AM address NSO-RFS-AM port
830 authgroup lsa device-type netconf ned-id lsa-netconf
admin@ncs(config-device-NSO-RFS-AM)# state admin-state unlocked
admin@ncs(config-device-NSO-RFS-AM)# top
admin@ncs(config)# devices device NSO-RFS-EMEA address NSO-RFS-EMEA
port 830 authgroup lsa device-type netconf ned-id lsa-netconf
admin@ncs(config-device-NSO-RFS-EMEA)# state admin-state unlocked
admin@ncs(config-device-NSO-RFS-EMEA)# top
admin@ncs(config)# commit
Commit complete.
admin@ncs(config)#
```

Step 6

Exit the configuration mode, fetch device SSH keys, and perform a sync-from devices.

Make sure that the synchronization is successful.

```
admin@ncs(config)# exit
admin@ncs# devices fetch-ssh-host-keys
fetch-result {
    device NSO-RFS-AM
    result updated
    fingerprint {
        algorithm ssh-rsa
        value 82:0f:4d:d9:79:7d:dc:84:89:c0:8f:48:a1:bb:a8:75
    }
}
fetch-result {
    device NSO-RFS-EMEA
    result updated
    fingerprint {
        algorithm ssh-rsa
        value b1:47:e3:26:e5:f3:53:22:8a:00:a4:88:03:15:9a:9d
    }
}
```

```
admin@ncs# devices sync-from
sync-result {
    device NSO-RFS-AM
    result true
}
sync-result {
    device NSO-RFS-EMEA
    result true
}
admin@ncs#
```

Activity Verification

You have completed this task when you attain these results:

- You have added both RFS nodes as devices to the CFS node.

Task 3: Deploy an LSA Service Instance

In this task, you will create a dispatch map and deploy an LSA service instance, using the CFS and RFS nodes you configured in the previous tasks.

Activity

Step 1

Create a dispatch map entry for each of the regions.

In NSO CLI, enter the config mode and create a dispatch map entry for each of the regions. Use the commands provided in the output.

```
admin@ncs# config
Entering configuration mode terminal
admin@ncs(config)# dispatch-map AM NSO-RFS-AM
admin@ncs(config)# dispatch-map EMEA NSO-RFS-EMEA
admin@ncs(config)# commit
Commit complete.
admin@ncs(config)#
```

Step 2

Configure a l3vpn service instance.

Create a **dry-run** of the commit and observe on which LSA RFS node the request has been dispatched, then **commit** the transaction.

```
admin@ncs(config)# l3vpn-cfs vpn1 link CustomerA device CE11-AM
interface 0/1 ip-address 10.100.0.1 mask 255.255.255.255
admin@ncs(config-link-CustomerA)# commit dry-run
cli {
    local-node {
        data  devices {
                  device NSO-RFS-AM {
                      config {
```

```
         +            l3vpn-rfs vpn1 {
         +                link CustomerA {
         +                    device CE11-AM;
         +                    interface 0/1;
         +                    ip-address 10.100.0.1;
         +                    mask 255.255.255.255;
         +                }
         +            }
                  }
              }
          }
         +l3vpn-cfs vpn1 {
         +    link CustomerA {
         +        device CE11-AM;
         +        interface 0/1;
         +        ip-address 10.100.0.1;
         +        mask 255.255.255.255;
         +    }
         +}
         +dispatch-map AM {
         +    rfs-node NSO-RFS-AM;
         +}
         +dispatch-map EMEA {
         +    rfs-node NSO-RFS-EMEA;
         +}
}
lsa-node {
    name NSO-RFS-AM
    data +l3vpn-rfs vpn1 {
         +    link CustomerA {
         +        device CE11-AM;
         +        interface 0/1;
         +        ip-address 10.100.0.1;
         +        mask 255.255.255.255;
         +    }
         +}
          devices {
              device CE11-AM {
                  config {
                      vrf {
         +                definition vpn1 {
         +                    rd 65000:9867;
         +                    route-target {
         +                        export 65000:9867;
         +                        import 65000:9867;
         +                    }
         +                }
                      }
                      interface {
                          GigabitEthernet 0/1 {
                              vrf {
         +                        forwarding vpn1;
                              }
                              ip {
                                  no-address {
         -                            address false;
                                  }
                                  address {
                                        primary {
            +                               address 10.100.0.1;
            +                               mask 255.255.255.255;
                                        }
                                    }
                                }
                            }
                        }
                        router {
            +               bgp 65000 {
            +                   address-family {
            +                       with-vrf {
            +                           ipv4 unicast {
            +                               vrf vpn1 {
            +                                   neighbor 10.100.0.1 {
            +                                       remote-as 65001;
            +                                   }
            +                               }
            +                           }
            +                       }
            +                   }
            +               }
                        }
                    }
                }
            }
    }
}
admin@ncs(config-link-CustomerA)# commit
Commit complete.
admin@ncs(config-link-CustomerA)# top
admin@ncs(config)#
```

Step 3

Configure an additional l3vpn service instance.

This time, use an EMEA device and verify the commit with a **dry-run**.

```
admin@ncs(config)# l3vpn-cfs vpn1 link CustomerB device CE21-EMEA
interface 0/1 ip-address 10.200.0.1 mask 255.255.255.255
admin@ncs(config-link-CustomerB)# commit dry-run
cli {
    local-node {
        data  devices {
                  device NSO-RFS-EMEA {
                      config {
            +             l3vpn-rfs vpn1 {
            +                 link CustomerB {
            +                     device CE21-EMEA;
            +                     interface 0/1;
            +                     ip-address 10.200.0.1;
            +                     mask 255.255.255.255;
            +                 }
            +             }
                      }
                  }
```

```
          }
          l3vpn-cfs vpn1 {
         +    link CustomerB {
         +        device CE21-EMEA;
         +        interface 0/1;
         +        ip-address 10.200.0.1;
         +        mask 255.255.255.255;
         +    }
          }
}
lsa-node {
    name NSO-RFS-EMEA
    data +l3vpn-rfs vpn1 {
         +    link CustomerB {
         +        device CE21-EMEA;
         +        interface 0/1;
         +        ip-address 10.200.0.1;
         +        mask 255.255.255.255;
         +    }
         +}
          devices {
              device CE21-EMEA {
                  config {
                      vrf {
         +                definition vpn1 {
         +                    rd 65000:32843;
         +                    route-target {
         +                        export 65000:32843;
         +                        import 65000:32843;
         +                    }
         +                }
                      }
                      interface {
                          GigabitEthernet 0/1 {
                              vrf {
         +                        forwarding vpn1;
                              }
                              ip {
                                  no-address {
         -                            address false;
                                  }
                                  address {
                                      primary {
         +                                address 10.200.0.1;
         +                                mask 255.255.255.255;
                                      }
                                  }
                              }
                          }
                      }
                      router {
         +                bgp 65000 {
         +                    address-family {
         +                        with-vrf {
         +                            ipv4 unicast {
         +                                vrf vpn1 {
         +                                    neighbor 10.200.0.1 {
         +                                        remote-as 65001;
```

```
            +                                    }
            +                                }
            +                            }
            +                        }
            +                    }
            +                }
                         }
                     }
                 }
             }
    }
}
admin@ncs(config-link-CustomerB)# commit
Commit complete.
admin@ncs(config-link-CustomerB)#
```

Step 4

Exit the CFS NSO CLI and connect to CLI of the RFS-AM node.

Use the following commands.

```
admin@ncs(config-link-CustomerB)# top
admin@ncs(config)# exit

admin@ncs# exit
student@nso-server:~/nso-lsa/nso-cfs$ cd ../nso-rfs
student@nso-server:~/nso-lsa/nso-rfs$ make testenv-cli NSO=am
docker exec -it testenv-nso-rfs-6.1-student-nso-am bash -lc 'ncs_cli -
Cu admin'

User admin last logged in 2024-02-09T05:21:15.527492+00:00, to
70dab8e3303e, from 172.21.0.2 using netconf-ssh
admin connected from 127.0.0.1 using console on 70dab8e3303e
admin@ncs-am#
```

Step 5

Verify that the configuration from the *l3vpn-cfs* service has been dispatched and applied to the correct devices.

To verify, use the **show running-config devices device CE11-AM config router bgp** command.

```
admin@ncs-am# show running-config devices device CE11-AM config router
bgp
devices device CE11-AM
 config
  router bgp 65000
   address-family ipv4 unicast vrf vpn1
    neighbor 10.100.0.1 remote-as 65001
    exit-address-family
   !
  !
 !
```

```
!
admin@ncs-am#
```

Step 6

Exit the NSO CLI and enter the container shell.

Use the following commands.

```
admin@ncs-am# exit
student@nso-server:~/nso-lsa/nso-rfs$ make testenv-shell NSO=am
docker exec -it testenv-nso-rfs-6.1-student-nso-am bash -l
root@70dab8e3303e:/#
```

Step 7

Use SSH to connect to CE11-AM device with the **admin** username and **admin** password credentials

Use the **ssh admin@CE11-AM** command.

```
root@70dab8e3303e:/# ssh admin@CE11-AM
The authenticity of host 'ce11-am (172.17.0.5)' can't be established.
RSA key fingerprint is
SHA256:zhkMRdv3HHyrS2jE2vxBW0CSAC0r45Kznjrjtt3tjD8.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'ce11-am,172.17.0.5' (RSA) to the list of
known hosts.
admin@ce11-am's password:

admin connected from 172.17.0.3 using ssh on c65d004564a0
dev>
```

Step 8

Verify that the configuration is present on the device.

Use the **show running-config router bgp** to verify that the configuration is present.

```
dev> enable
dev# show running-config router bgp
router bgp 65000
 address-family ipv4 unicast vrf vpn1
  neighbor 10.100.0.1 remote-as 65001
  exit-address-family
 !
!
dev#
```

Step 9

Exit the SSH session and NSO container.

Use the **exit** command.

```
dev# exit
Connection to ce11-am closed.
root@70dab8e3303e:/# exit
logout
student@nso-server:~/nso-lsa/nso-rfs$
```

Step 10

You may repeat the steps for the EMEA container and the CE21-EMEA device.

You may repeat steps 4 to 9 for the EMEA container and the CE21-EMEA device.

Activity Verification

You have completed this task when you attain these results:

- You have deployed and verified several service instances using LSA.

How can you verify if your l3vpn service on CFS node really deployed the configuration on the network device?

- ○ By inspecting the actual network device configuration.
- ○ By inspecting the device configuration on the CFS node.
- ○ By inspecting the l3vpn service instance configuration on the CFS node.
- ○ By inspecting the l3vpn service instance configuration on the RFS node.