# Discovery 4: Create a Service Package

## Introduction

In this lab, you will practice creating a service package. Your service package will apply provided standardized configuration to one or more devices at once. You will first create a service package. Then, you will create a service model and a service template. Finally, you will compile and deploy the package to your local NSO instance, and provision a service instance.

## 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 you to learn how to design a service package based on a provided configuration to standardize the basic DNS and SNMP settings across the network and learn how to design a service package based on a provided configuration to standardize the basic DNS and SNMP settings across the network and learn how to apply this configuration to the individual devices. 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:

- Create a service package
- Create a service model
- Create a service template
- Compile a service package
- Deploy a service package
- Configure and apply a service instance to multiple devices at once

## 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: Create a Service Package

In this task, you will create a service package using the **ncs-make-package** tool. The tool will generate a service package skeleton. You will use this skeleton to develop a service package that will generate a standardized configuration for one or more devices.

## *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 Terminal application and go to a directory where service packages are located.

```
developer@devbox:~$ cd nso-run
developer@devbox:~/nso-run$ cd packages
developer@devbox:~/nso-run/packages$
```

**Step 3** Generate a new service package named "**dns-snmp-service**."

Generate a new template-based service package using the **ncs-make-package** tool.

```
developer@devbox:~/nso-run/packages$ ncs-make-package --service-
skeleton template dns-snmp-service
```

After issuing the command, verify if the service package was created successfully. There should be two service packages: Cisco IOS NED and a service package called **dns-snmp-service** with skeleton code.

```
developer@devbox:~/nso-run/packages$ ls
cisco-ios-cli-6.67  dns-snmp-service
developer@devbox:~/nso-run/packages$
```

**Step 4** Modify the contents of the **package-meta-data.xml** file in the service package file so that the attributes will more accurately describe your service package.

Go to the service package directory and open the **package-meta-data.xml** file for editing by using your favorite editing tool.

```
developer@devbox:~/nso-run/packages$ cd dns-snmp-service
```

```
developer@devbox:~/nso-run/packages/dns-snmp-service$ code package-meta-data.xml
```

| Note | The **code package-meta-data.xml** command will open the file for editing in Visual Studio Code. You can use any editing tool of your choice. |
|---|---|

Change only the attributes from the following table:

| Package Metadata Attribute | Value |
|---|---|
| package-version | 1.0.0.0 |
| description | This is a DNS SNMP configuration service. |

After editing, the file contents should match the following output:

```
<ncs-package xmlns="http://tail-f.com/ns/ncs-packages">
  <name>dns-snmp-service</name>
  <package-version>1.0.0.0</package-version>
  <description>This is a DNS SNMP configuration service.</description>
  <ncs-min-version>5.4</ncs-min-version>

  <!-- It's possible to add more components to the -->
  <!—same package, multiple services, data providers etc -->

</ncs-package>
```

**Step 5** Save the file contents and close the editor.

You have successfully created a new service package and edited the version and the description of the service package.

# Task 2: Create a Service Model

In this task, you will edit the YANG service model to support the DNS and SNMP service. The YANG service model skeleton is located in the **src/yang/dns-snmp-service.yang** file, created by the **ncs-make-package** tool.

The following standardized configuration needs to be applied to one or more Cisco IOS XE devices:

```
ip name-server 172.21.1.10
access-list 55 remark SNMP ACCESS
access-list 55 permit 172.21.1.20
snmp-server community c0mmun1t7 RO 55
```

After analysis, the following parameters were identified as dynamic and need to be supported in the service model:

| Service Parameter | Description | Node Name | Default Value | Restrictions |
|---|---|---|---|---|
| Service Instance Name | Service instance name | name | | Mandatory |
| List of Devices | List of devices where the configuration needs to be applied. | devices | | |
| DNS Server IP | IP Address of the DNS Server | dns-server | | Mandatory |
| SNMP Server IP | IP Address of the SNMP Server | snmp-server | | Mandatory |
| ACL Number | Standard Access List Number | acl-number | 55 | Range: 1 – 99 |
| Community String | SNMP Community String Value | community-string | c0mmun1t7 | Length: 6 – 32 |

## *Activity*

For this activity, complete the following steps:

**Step 1** Connect to the Student DevBox.

On the lab page, in the device list or on the topology diagram, choose and click the **Student DevBox**.

**Step 2** On the Student DevBox, launch the Terminal application and go to your newly created service package.

```
developer@devbox:~$ cd $HOME/nso-run/packages
developer@devbox:~/nso-run/packages$ cd dns-snmp-service
```

**Step 3** Navigate to the YANG service model file and open it for editing.

```
developer@devbox:~/nso-run/packages/dns-snmp-service$ code
src/yang/dns-snmp-service.yang
```

**Step 4** Change the namespace of the module to a unique distinctive value "http://cisco.com/examples/dnssnmpservice" that can be used to easily identify the service package.

```
module dns-snmp-service {
  namespace "http://cisco.com/examples/dnssnmpservice";
  prefix dns-snmp-service;

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

  ...
  ...

}
```

**Step 5** Add a reference to the **tailf-common** module.

Add a reference to the **tailf-common** module by importing that module. The **tailf-common** module contains NSO specific YANG extensions and statements. You will use the module's **info** statement for the NSO CLI syntax help that you will implement.

```
module dns-snmp-service {
  namespace "http://cisco.com/examples/dnssnmpservice";
  prefix dns-snmp-service;

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

  // add support for NSO syntax help
  import tailf-common {
    prefix tailf;
  }

  list dns-snmp-service {
    ...
  }
}
```

**Step 6** Add additional header information to provide organization, contact, description, and revision information.

```
module dns-snmp-service {
  namespace "http://cisco.com/examples/dnssnmpservice";
  prefix dns-snmp-service;

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

  // add support for NSO syntax help
  import tailf-common {
    prefix tailf;
```

```
  }

  organization "CISCO SYSTEMS";
  contact "CISCO SYSTEMS; support@cisco.com";
  description "The module to showcase NSO capabilities.";
  revision 2022-06-22 {
    description "version 1.0.0.0, see README file.";
  }

  list dns-snmp-service {
    ...
  }
}
```

**Step 7** Rename the "device" leaf to "devices."

Renaming the "device" leaf to "devices" will simplify the usage of the service by immediately indicating to the operator that multiple devices can be configured.

```
module dns-snmp-service {
  namespace "http://com/example/dnssnmpservice";
  prefix dns-snmp-service;

  ...

  list dns-snmp-service {
    key name;

    uses ncs:service-data;
    ncs:servicepoint "dns-snmp-service";

    leaf name {
      type string;
    }

    leaf-list devices {
      type leafref {
        path "/ncs:devices/ncs:device/ncs:name";
```

```
      }
    }

    // replace with your own stuff here
    leaf dummy {
      type inet:ipv4-address;
    }
  }
}
```

**Step 8** Delete the dummy leaf.

```
module dns-snmp-service {

  ...

  list dns-snmp-service {
    key name;

    uses ncs:service-data;
    ncs:servicepoint "dns-snmp-service";

    leaf name {
      type string;
    }

    leaf-list devices {
      type leafref {
        path "/ncs:devices/ncs:device/ncs:name";
      }
    }

    // replace with your own stuff here
    leaf dummy {
      type inet:ipv4-address;
    }
  }
}
```

**Step 9** Add additional nodes according to the service model parameters: **dns-server**, **snmp-server**, **acl-number**, and **community string**.

You will now add additional nodes after the "devices" node. As per the service model, you must add the following nodes: **dns-server**, **snmp-server**, **acl-number**, and **community-string**.

```
module dns-snmp-service {
  namespace "http://com/example/dnssnmpservice";
  prefix dns-snmp-service;

  ...

  list dns-snmp-service {
    key name;

    uses ncs:service-data;
    ncs:servicepoint "dns-snmp-service";

    leaf name {
      type string;
    }

    leaf-list devices {
      type leafref {
        path "/ncs:devices/ncs:device/ncs:name";
      }
    }

    leaf dns-server {
      type inet:ipv4-address;
    }

    leaf snmp-server {
      type inet:ipv4-address;
    }

    leaf acl-number {
```

```
      type uint16;
    }

    leaf community-string {
      type string;
    }

  }
}
```

**Step 10** Add node descriptions to your service model.

Add node descriptions to your service model. You will use the **tailf:info** statement from the previously imported **tailf-common module** to implement the descriptions and NSO CLI syntax help.

```
module dns-snmp-service {
  namespace "http://cisco.com/examples/dnssnmpservice";
  prefix dns-snmp-service;

  ...

  list dns-snmp-service {
    tailf:info "DNS and SNMP configuration service";
    key name;

    uses ncs:service-data;
    ncs:servicepoint "dns-snmp-service";

    leaf name {
      tailf:info "Unique service instance name";
      type string;
    }

    leaf-list devices {
      tailf:info "Devices for this service configuration";
      type leafref {
        path "/ncs:devices/ncs:device/ncs:name";
```

```
      }
    }

    leaf dns-server {
      tailf:info "DNS server";
      type inet:ipv4-address;
    }

    leaf snmp-server {
      tailf:info "SNMP server";
      type inet:ipv4-address;
    }

    leaf acl-number {
      tailf:info "IP Standard Access List";
      type uint16;
    }

    leaf community-string {
      tailf:info "SNMP Community string";
      type string;
    }

  }
}
```

**Step 11** Add range and length constraints to specific leafs, define the mandatory leafs, and finally, define default values for the **acl-number** and the **community-string** leafs.

When modifying leafs **acl-number** and **community-string**, make sure to remove the semicolon after the data type, because you are adding a block of substatements enclosed within braces ("{ }"). In YANG, each statement ends either by a semicolon (";") or a block of substatements.

```
module dns-snmp-service {
  namespace "http://cisco.com/examples/dnssnmpservice";
  prefix dns-snmp-service;

  ...
```

```
list dns-snmp-service {
  tailf:info "DNS and SNMP configuration service";
  key name;

  uses ncs:service-data;
  ncs:servicepoint "dns-snmp-service";

  leaf name {
    ...
  }

  leaf-list devices {
    ...
  }

  leaf dns-server {
    tailf:info "DNS server";
    type inet:ipv4-address;
    mandatory true;
  }

  leaf snmp-server {
    tailf:info "SNMP server";
    type inet:ipv4-address;
    mandatory true;
  }

  leaf acl-number {
    tailf:info "IP Standard Access List";
    type uint16
    {
       range "1..99";
    }
    default 55;
  }

  leaf community-string {
    tailf:info "SNMP Community string";
```

```
      type string
      {
         length "6..32";
      }
      default "c0mmun1t7";
    }

  }
}
```

| Note | You have limited the **acl-number** parameter to reflect the Standard ACL numbering range. Similarly, you have limited the minimum and maximum allowed length of the **community-string** between 6 and 32 characters for security and compatibility purposes. |
|---|---|

**Step 12** Verify the final version of the service model, and then save the file and exit the editor.

```
module dns-snmp-service {
  namespace "http://cisco.com/examples/dnssnmpservice";
  prefix dns-snmp-service;

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

  // add support for NSO syntax help
  import tailf-common {
    prefix tailf;
  }

  organization "CISCO SYSTEMS";
  contact "CISCO SYSTEMS; support@cisco.com";
  description "The module to showcase NSO capabilities.";
  revision 2022-06-22 {
    description "version 1.0.0.0, see README file.";
```

```
}

list dns-snmp-service {
  tailf:info "DNS and SNMP configuration service";
  key name;

  uses ncs:service-data;
  ncs:servicepoint "dns-snmp-service";

  leaf name {
    tailf:info "Unique service instance name";
    type string;
  }

  leaf-list devices {
    tailf:info "Devices for this service configuration ";
    type leafref {
      path "/ncs:devices/ncs:device/ncs:name";
    }
  }

  leaf dns-server {
    tailf:info "DNS server";
    type inet:ipv4-address;
    mandatory true;
  }

  leaf snmp-server {
    tailf:info "SNMP server";
    type inet:ipv4-address;
    mandatory true;
  }

  leaf acl-number {
    tailf:info "IP Standard Access List";
    type uint16
    {
       range "1..99";
```

```
      }
      default 55;
    }

    leaf community-string {
      tailf:info "SNMP Community String";
      type string
      {
         length "6..32";
      }
      default "c0mmun1t7";
    }

  }
}
```

As you can see from the bolded text, you have added additional header information, context help that will be available in NSO CLI (tailf:info), and additional range and length restrictions.

You have successfully created a YANG service model for your service.

# Task 3: Create a Service Template

In this task, you will create a service template configuration to standardize the basic DNS and SNMP settings across the network and learn how to map the service parameters, defined in the YANG service model, to device configuration.

## *Activity*

For this activity, complete the following steps:

**Step 1** On the Student DevBox, launch the Terminal Application. Navigate to the service template skeleton file **dns-snmp-service-template.xml** and open it for editing.

```
developer@devbox:~$ cd $HOME/nso-run/packages/dns-snmp-service
developer@devbox:~/nso-run/packages/dns-snmp-service$ cd templates
developer@devbox:~/nso-run/packages/dns-snmp-service/templates$ ls
dns-snmp-service-template.xml
```

```
developer@devbox:~/nso-run/packages/dns-snmp-service/templates$ code
dns-snmp-service-template.xml
```

| **Note** | The **code dns-snmp-service-template.xml** command will open the file for editing in Visual Studio Code. You can use any editing tool of your choice. |
|---|---|

The service template skeleton should resemble the following output:

```
<config-template xmlns="http://tail-f.com/ns/config/1.0"
        servicepoint="dns-snmp-service">

  <devices xmlns="http://tail-f.com/ns/ncs">

    <device>

      <!--

          Select the devices from some data structure in the service

          model. In this skeleton the devices are specified in a
        leaf-list.

          Select all devices in that leaf-list:

      -->

      <name>{/device}</name>

      <config>

        <!--

            Add device-specific parameters here.

            In this skeleton the, java code sets a variable DUMMY,
        use it

            to set something on the device e.g.:

            <ip-address-on-device>{$DUMMY}</ip-address-on-device>

        -->

      </config>

    </device>

  </devices>

</config-template>
```

You will now modify the template with the DNS and SNMP configuration.

**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 2022-07-05T16:28:25.553422+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** Use NSO CLI config mode to clear any configuration from previous discovery.

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
   ip name-server 172.21.1.10
   access-list 55 remark SNMP ACCESS
   access-list 55 permit 172.21.1.20
   snmp-server community c0mmun1t7 RO 55
```

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

```
admin@ncs# config
Entering configuration mode terminal
admin@ncs(config)# no devices device R1 config snmp-server
admin@ncs(config)# no devices device R2 config snmp-server
admin@ncs(config)# no devices device R3 config snmp-server
admin@ncs(config)# no devices device R1 config ip access-list
admin@ncs(config)# no devices device R2 config ip access-list
admin@ncs(config)# no devices device R3 config ip access-list
admin@ncs(config)# no devices device R1 config ip name-server
admin@ncs(config)# no devices device R2 config ip name-server
admin@ncs(config)# no devices device R3 config ip name-server
admin@ncs(config)#
Commit complete.
admin@ncs(config)# end
```

```
admin@ncs#
```

**Step 4** Use NSO CLI config mode to render the desired DNS and SNMP configuration for a Cisco IOS XE device R1.

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
   ip name-server 172.21.1.10
   access-list 55 remark SNMP ACCESS
   access-list 55 permit 172.21.1.20
   snmp-server community c0mmun1t7 RO 55
```

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

```
admin@ncs# config
Entering configuration mode terminal
admin@ncs(config)# devices device R1 config
admin@ncs(config-config)# ip name-server 172.21.1.10
admin@ncs(config-config)# access-list 55 remark SNMP ACCESS
admin@ncs(config-config)# access-list 55 permit 172.21.1.20
admin@ncs(config-config)# snmp-server community c0mmun1t7 RO 55
admin@ncs(config-config)#
```

**Step 5** In the NSO CLI config mode, preview the device configuration in the XML format by issuing the **commit dry-run outformat xml** command.

In the same Terminal application window or tab, in the NSO CLI config mode, first execute the **rop command,** followed by the **commit dry-run outformat xml** command:

```
admin@ncs(config-config)# top
admin@ncs(config)# commit dry-run outformat xml
result-xml {
    local-node {
        data <devices xmlns="http://tail-f.com/ns/ncs">
               <device>
                 <name>R1</name>
```

```
                    <config>
                      <ip xmlns="urn:ios">
                        <name-server>
                          <name-server-list>
                            <address>172.21.1.10</address>
                          </name-server-list>
                        </name-server>
                      </ip>
                      <access-list xmlns="urn:ios">
                        <access-list>
                          <id>55</id>
                          <rule>
                            <rule>remark SNMP ACCESS</rule>
                          </rule>
                          <rule>
                            <rule>permit 172.21.1.20</rule>
                          </rule>
                        </access-list>
                      </access-list>
                      <snmp-server xmlns="urn:ios">
                        <community>
                          <name>c0mmun1t7</name>
                          <RO/>
                          <access-list-name>55</access-list-name>
                        </community>
                      </snmp-server>
                    </config>
                  </device>
                </devices>
    }
}
admin@ncs(config)#
```

**Step 6** Switch back to your service template skeleton file that you have previously opened in Visual Studio Code and delete all the content between the **<config-template>** tags.

After removing the content, your service template skeleton file must look exactly as in the following example:

```
<config-template xmlns="http://tail-f.com/ns/config/1.0"
       servicepoint="dns-snmp-service">



</config-template>
```

**Step 7** Insert the contents of the XML output between the **<config-template>** tags in the service template skeleton file.

Make sure that you copy only the XML contents of the output. After copying, your service template **dns-snmp-service-template.xml** file must look exactly as in the following example:

```
<config-template xmlns="http://tail-f.com/ns/config/1.0"
       servicepoint="dns-snmp-service">

  <devices xmlns="http://tail-f.com/ns/ncs">
    <device>
      <name>R1</name>
      <config>
        <ip xmlns="urn:ios">
          <name-server>
            <name-server-list>
              <address>172.21.1.10</address>
            </name-server-list>
          </name-server>
        </ip>
        <access-list xmlns="urn:ios">
          <access-list>
            <id>55</id>
            <rule>
              <rule>remark SNMP ACCESS</rule>
            </rule>
            <rule>
              <rule>permit 172.21.1.20</rule>
            </rule>
          </access-list>
        </access-list>
        <snmp-server xmlns="urn:ios">
          <community>
            <name>c0mmun1t7</name>
            <RO/>
```

```
            <access-list-name>55</access-list-name>
          </community>
        </snmp-server>
      </config>
    </device>
  </devices>
</config-template>
```

**Step 8** Replace the hardcoded values with the service model parameters from the following table:

| Device Template Variable | Current Value | Service Parameter |
|---|---|---|
| Name of the device | R1 | {/devices} |
| ACL name / number | 55 | {/acl-number} |
| DNS Server IP Address | 172.21.1.10 | {/dns-server} |
| SNMP Server IP Address | 172.21.1.20 | {/snmp-server} |
| SNMP Community String | c0mmun1t7 | {/community-string} |

After replacing the hardcoded values with the service model parameters, your service template **dns-snmp-service-template.xml** file must look exactly as in the following example:

```
<config-template xmlns="http://tail-f.com/ns/config/1.0"
       servicepoint="dns-snmp-service">

  <devices xmlns="http://tail-f.com/ns/ncs">
    <device>
      <name>{/devices}</name>
      <config>
        <ip xmlns="urn:ios">
          <name-server>
            <name-server-list>
              <address>{/dns-server}</address>
            </name-server-list>
          </name-server>
        </ip>
        <access-list xmlns="urn:ios">
```

```
          <access-list>
            <id>{/acl-number}</id>
            <rule>
              <rule>remark SNMP ACCESS</rule>
            </rule>
            <rule>
              <rule>permit {/snmp-server}</rule>
            </rule>
          </access-list>
        </access-list>
        <snmp-server xmlns="urn:ios">
          <community>
            <name>{/community-string}</name>
            <RO/>
            <access-list-name>{/acl-number}</access-list-name>
          </community>
        </snmp-server>
      </config>
    </device>
  </devices>
</config-template>
```

**Step 9** Add additional instruction elements to replace the device configuration subsection in case it already exists on the device.

Replacing the device configuration is necessary in cases where the configuration subsection can have multiple items and you want to make sure that the configuration only contains your configuration. The default system behavior is merging the new configuration with the existing one. Be 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 configuration subsection on the device whenever the template is applied. Such an example is an access list entry.

In your service template, add the **tags="replace"** attribute to your **access-list** XML element.

```
<config-template xmlns="http://tail-f.com/ns/config/1.0"
       servicepoint="dns-snmp-service">

  <devices xmlns="http://tail-f.com/ns/ncs">
    <device>
      <name>{/devices}</name>
      <config>
```

```
        <ip xmlns="urn:ios">
          <name-server>
            <name-server-list>
              <address>{/dns-server}</address>
            </name-server-list>
          </name-server>
        </ip>
        <access-list xmlns="urn:ios">
          <access-list tags="replace">
            <id>{/acl-number}</id>
            <rule>
              <rule>remark SNMP ACCESS</rule>
            </rule>
            <rule>
              <rule>permit {/snmp-server}</rule>
            </rule>
          </access-list>
        </access-list>
        <snmp-server xmlns="urn:ios">
          <community>
            <name>{/community-string}</name>
            <RO/>
            <access-list-name>{/acl-number}</access-list-name>
          </community>
        </snmp-server>
      </config>
    </device>
  </devices>
</config-template>
```

This is now your service template. This service template will create a standardized configuration for all the devices that you will specify in the service instance. That way you can easily and elegantly configure standard features precisely where required and keep it unified across the complete network.

**Step 10** Save the file and exit the editor.

**Step 11** Switch to the Terminal application and exit NSO CLI.

Switch to your Terminal application, which is running NSO CLI. Abort the NSO config mode by issuing the **abort** command and then exit NSO CLI by issuing the **exit** command.

```
admin@ncs(config)# abort
admin@ncs# exit
developer@devbox:~$
```

**Step 12** Save the file and exit the editor.

You have successfully created a service template configuration and learned how to apply the YANG service model parameters to the elements of the configuration.

# Task 4: Compile and Deploy the Package

In this task, you will compile your service package and deploy the package to NSO.

## *Activity*

For this activity, complete the following steps:

**Step 1** On the Student DevBox, launch or reuse the Terminal application and go to your service package.

```
developer@devbox:~$ cd $HOME/nso-run/packages
developer@devbox:~/nso-run/packages$ cd dns-snmp-service
```

**Step 2** Compile the package by issuing the **make** command.

Navigate to the **src** directory of the package where the **Makefile** file is located and issue the **make** command.

```
developer@devbox:~/nso-run/packages/dns-snmp-service$ cd src
developer@devbox:~ /nso-run/packages/dns-snmp-service/src$ make
```

You should expect the following output after issuing the **make** command:

```
developer@devbox:~ /nso-run/packages/dns-snmp-service/src$ make
```

```
mkdir -p ../load-dir
/home/developer/nso/bin/ncsc   `ls   dns-snmp-service-ann.yang      >
/dev/null 2>&1 && echo "-a dns-snmp-service-ann.yang"` \
        --fail-on-warnings \
         \
        -c    -o    ../load-dir/dns-snmp-service.fxs    yang/dns-snmp-
service.yang
developer@devbox:~/nso-run/packages/dns-snmp-service/src$
```

If the output is different and ends with one or more errors, there is a syntax error in the YANG service model. You can use the error output to determine in which line there is an issue, and use this information to review the service model content. For example, consider a missing semicolon in the service model in the line 25:

```
23  ...
24  list dns-snmp-service {
25    tailf:info "DNS and SNMP configuration service"
26    key name;
27
28    uses ncs:service-data;
29    ...
```

When compiling the service package, the compiler will produce the following output:

```
...
/home/developer/nso/bin/ncsc `ls dns-snmp-service-ann.yang  >
/dev/null 2>&1 && echo "-a dns-snmp-service-ann.yang"` \
        --fail-on-warnings \
         \
        -c -o ../load-dir/dns-snmp-service.fxs yang/dns-snmp-
service.yang
yang/dns-snmp-service.yang:26:5: error: unterminated statement for
keyword "info"
make: *** [Makefile:26: ../load-dir/dns-snmp-service.fxs] Error 1
developer@devbox:~/nso-run/packages/dns-snmp-service/src$
```

In the last part of the output, you can observe the "unterminated statement" error. This is because the line 25 is not properly terminated with a semicolon, which generates an error in line 26. After making corrections to the service model, the service package must be recompiled once again and should end with success.

When the package is successfully compiled, you are ready to reload all your packages in NSO CLI for the changes to take effect.

**Step 3** Launch the NSO CLI as admin in the Cisco-style mode.

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 2022-07-05T16:28:25.553422+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 4** Issue the **packages reload** command.

```
admin@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-6.67

    result true

}

reload-result {

    package dns-snmp-service

    result true

}

admin@ncs#

System message at 2022-07-05 17:03:02...
```

```
    Subsystem stopped: ncs-dp-1-cisco-ios-cli-6.67:IOSDp

admin@ncs#

System message at 2022-07-05 17:03:02...

    Subsystem started: ncs-dp-2-cisco-ios-cli-6.67:IOSDp

admin@ncs#
```

Make sure that the package reload result of your service package dns-snmp-service is set to **true**. The **result true** confirms that your package is active and ready to accept service instance configuration.

**Step 5** Verify the package attributes by issuing the **show packages package dns-snmp-service** command.

```
admin@ncs# show packages package dns-snmp-service
packages package dns-snmp-service
 package-version       1.0.0.0
 description           "This is a DNS SNMP configuration service."
 ncs-min-version       [ 5.4 ]
 directory             ./state/packages-in-use/1/dns-snmp-service
 templates             [ dns-snmp-service-template ]
 template-loading-mode strict
 oper-status up
admin@ncs#
```

You have successfully deployed the package in NSO. You are now ready to configure a service instance.

# Task 5: Create a Service Instance

In this task, you will create a **dns-snmp-service** service instance and deploy the configuration to multiple devices.

## *Activity*

For this activity, complete the following steps:

**Step 1** On the Student DevBox, launch or reuse the Terminal application.

**Step 2** Launch the NSO CLI as admin in the Cisco-style mode.

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 2022-07-05T16:28:25.553422+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** Use the NSO CLI config mode to create a **dns-snmp-service** service instance.

In NSO CLI, switch to the configuration mode by using the **config** command. Then, create a service instance and provide the following parameters from the table:

| Service Instance Parameter | Service Parameter | Value |
|---|---|---|
| Service Instance Name | name | InternalNetwork |
| Devices | devices | R1, R2, R3 |
| DNS Server IP Address | dns-server | 172.21.1.10 |
| SNMP Server IP Address | snmp-server | 172.21.1.20 |

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

```
admin@ncs# config
Entering configuration mode terminal
admin@ncs(config)# dns-snmp-service InternalNetwork
admin@ncs(config-dns-snmp-service-InternalNetwork)# dns-server 172.21.1.10
admin@ncs(config-dns-snmp-service-InternalNetwork)# snmp-server 172.21.1.20
admin@ncs(config-dns-snmp-service-InternalNetwork)# devices R1
admin@ncs(config-dns-snmp-service-InternalNetwork)# devices R2
admin@ncs(config-dns-snmp-service-InternalNetwork)# devices R3
```

Because only the service instance name **InternalNetwork** was initially provided, the system automatically asked you for mandatory parameters **dns-server** and **snmp-server**. After specifying the devices to which you want to push the configuration, the service instance has enough information to produce the configuration. The ACL Number and the SNMP community string service parameters are set to default values, "55" and "c0mmun1t7," respectively.

**Step 4** Preview the changes by issuing the **commit dry-run** command.

In NSO CLI, the configuration must resemble the following example:

```
admin@ncs(config-dns-snmp-service-InternalNetwork)# commit dry-run
cli {
    local-node {
        data  devices {
                  device R1 {
                      config {
                          ip {
                              name-server {
            +                     # first
            +                     name-server-list 172.21.1.10;
                              }
                          }
                          access-list {
            +                 access-list 55 {
            +                     rule "remark SNMP ACCESS";
            +                     rule "permit 172.21.1.20";
            +                 }
                          }
                          snmp-server {
            +                 community c0mmun1t7 {
            +                     RO;
            +                     access-list-name 55;
            +                 }
                          }
                      }
                  }
                  device R2 {
                      config {
                          ip {
                              name-server {
            +                     # first
            +                     name-server-list 172.21.1.10;
```

```
                    }
                }
                access-list {
+                   access-list 55 {
+                       rule "remark SNMP ACCESS";
+                       rule "permit 172.21.1.20";
+                   }
                }
                snmp-server {
+                   community c0mmun1t7 {
+                       RO;
+                       access-list-name 55;
+                   }
                }
            }
        }
        device R3 {
            config {
                ip {
                    name-server {
+                       # first
+                       name-server-list 172.21.1.10;
                    }
                }
                access-list {
+                   access-list 55 {
+                       rule "remark SNMP ACCESS";
+                       rule "permit 172.21.1.20";
+                   }
                }
                snmp-server {
+                   community c0mmun1t7 {
+                       RO;
+                       access-list-name 55;
+                   }
                }
            }
        }
```

```
             }
            +dns-snmp-service InternalNetwork {
            +    devices [ R1 R2 R3 ];
            +    dns-server 172.21.1.10;
            +    snmp-server 172.21.1.20;
            +}
    }
}
```

**Step 5** Go to the topmost level of config mode and commit the changes by issuing the **commit** command.

In NSO CLI, issue the following commands:

```
admin@ncs(config-dns-snmp-service-InternalNetwork)# top
admin@ncs(config)# commit
Commit complete.
```

Changes are now pushed to your specified devices.

**Step 6** While still in the configuration mode, **edit** the configuration of your service instance **InternalNetwork** by changing the community string value.

In the NSO CLI, issue the following commands:

```
admin@ncs(config)# dns-snmp-service InternalNetwork
dmin@ncs(config-dns-snmp-service-InternalNetwork)# community-string
?
Description: SNMP Community String
Possible completions:
  <string, min: 6 chars, max: 32 chars>[c0mmun1t7]
```

Observe how restrictions from the service model for the **community-string** leaf are in place:

- the minimum length is 6 characters
- the maximum length is 32 characters
- the default value is set to **c0mmun1t7**

Change the default value of the **community-string** leaf to a different value.

```
admin@ncs(config-dns-snmp-service-InternalNetwork)# community-string
c0mmunit789
```

**Step 7** Preview the changes by issuing the **commit dry-run** command.

```
admin@ncs(config-dns-snmp-service-InternalNetwork)# commit dry-run
cli {
    local-node {
        data  devices {
                  device R1 {
                      config {
                          snmp-server {
              -               community c0mmun1t7 {
              -                   RO;
              -                   access-list-name 55;
              -               }
              +               community c0mmunit789 {
              +                   RO;
              +                   access-list-name 55;
              +               }
                          }
                      }
                  }
                  device R2 {
                      config {
                          snmp-server {
              -               community c0mmun1t7 {
              -                   RO;
              -                   access-list-name 55;
              -               }
              +               community c0mmunit789 {
              +                   RO;
              +                   access-list-name 55;
              +               }
                          }
                      }
                  }
                  device R3 {
                      config {
                          snmp-server {
              -               community c0mmun1t7 {
              -                   RO;
              -                   access-list-name 55;
```

```
-                   }
+                   community c0mmunit789 {
+                       RO;
+                       access-list-name 55;
+                   }
                 }
             }
         }
         dns-snmp-service InternalNetwork {
+    community-string c0mmunit789;
         }
    }
}
admin@ncs(config-dns-snmp-service-InternalNetwork)#
```

Observe how NSO recalculates the configuration based on the service instance changes and prepares the configuration change for all the devices. The service approach makes it easy to apply changes in the network configuration in the service life cycle and simplifies the management of the network devices.

In this case, you will not push the changes to the network but instead, decide to abort the configuration changes.

**Step 8** Commit the changes by issuing the **commit** command.

In NSO CLI, issue the following commands:

```
admin@ncs(config-dns-snmp-service-InternalNetwork)# commit
Commit complete.
admin@ncs(config-dns-snmp-service-InternalNetwork)#
```

Changes are now pushed to your specified devices.

You have successfully completed the service instance deployment and you have completed all the tasks of this Discovery Lab.