Understand NSO Service Ownership

Contents

Introduction

This document describes the general concept, common pitfalls, and solutions of service ownership in Cisco® Network Service Orchestrator (NSO).

Prerequisites

Requirements

This document applies to all currently available NSO versions, including NSO 6. The behavior described is only applicable to NSO setups using a combination of services and non-service configuration. While the specific commands used in examples throughout this document only apply to the Network Element Driver (NED) used, the underlying logic applies to any device managed by NSO.

Components Used

- NSO 6.1.6
- NED: test-ned 1.0 , a custom built netconf NED using this yang model.

```
NED yang model:
module test-ned{
    namespace "http://example.org/ned/service-ownership";
    prefix ownership;
    import ietf-inet-types{ prefix inet;}

    list interface {
        key interface-name;
        leaf interface-name{
            type string;
        }

        leaf ip-address {
            type inet:ipv4-address;
        }

        leaf description {
            type string;
        }
    }
}
```

- Service: example-service 1.0 : a custom built service using this yang model and template

```
module example-service {
  namespace "http://com/example/exampleservice";
  prefix example-service;

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

  list example-service {
    key name;

    uses ncs:service-data;
    ncs:servicepoint "example-service";

    leaf name {
      type string;
    }
    leaf-list device {
      type leafref {
        path "/ncs:devices/ncs:device/ncs:name";
      }
    }
    leaf ipaddress {
      type inet:ipv4-address;
    }
  }
```

```
}

<config-template xmlns="http://tail-f.com/ns/config/1.0"
                 servicepoint="example-service">
  <devices xmlns="http://tail-f.com/ns/ncs">
    <device>
      <name>{/device}</name>
      <config>
        <interface xmlns="http://example.org/ned/service-ownership">
          <interface-name>FE1</interface-name>
          <ip-address>{/ipaddress}</ip-address>
        </interface>
      </config>
    </device>
  </devices>
</config-template>
```

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.

Basics

The purpose of service ownership is to allow NSO to track what configuration is related to which service. When you delete a service, NSO needs to delete the configuration related to it and it uses service ownership to determine what configuration to delete. When configuration is owned by more than one service, deleting one of the services simply removes that ownership reference; The configuration itself remains in the NSO database (CDB) and on the devices of the network.

Ownership is displayed through refcounts and backpointers. A refcount shows how many entities own that part of the configuration. A refcount equals the amount of service instances plus 1 if the configuration is also device owned. A backpointer shows the path of these service instances. There is no backpointer to display "device owned". Backpointers are only shown in CDB for lists and containers. Individual leafs do not show their backpointers but they inherit from their parent.

Device owned

In addition to configuration being owned by a service, it can also be owned by the device. This is sometimes called "device owned" or "non-service owned". While this document uses "device-owned", note that while this is easier to understand, non-service ownership does not have to include devices. LSA setups or stacked services can have non-service owned configuration without devices.

Configuration is device-owned when the configuration is added to CDB without the use of a service deployment but instead used a method such as sync-from, load merge, or ncs_cli to set the configuration. When a service instance takes ownership of configuration which was already device owned, the refcount is set to 2 to reflect the shared ownership. When the service instance gets deleted, the configuration is not deleted despite the fact only one service instance owns the configuration prior to deletion. Additionally, device-owned configuration adds an "original value" tag. If a service instance overwrites the device owned configuration and the service is later deleted, the configuration restores to the original value.

Device ownership is only assigned if the configuration did not already exist in CDB when you add it through non-service means. Service-owned configuration does not become device owned after sync-from. But Device-owned configuration becomes both device owned and service owned if you deploy a service on

top.

Ownership in action

Showcase: Only Service Owned

When you deploy a service while the target configuration is empty, the service creates the configuration and takes ownership of it. The user can check ownership by using a **show running-config** command and append **| display service-meta-data**. While not mandatory, it is recommended to also append **| display xml** as the default CLI style output does not always correctly reflect how the data is modeled in CDB.

```
admin@ncs(config)# do show running-config devices device mydevice0 config
% No entries found.
admin@ncs(config)# example-service s1 device mydevice0 ipaddress 192.0.2.1
admin@ncs(config-example-service-s1)# commit dry-run
cli {
    local-node {
        data  devices {
                  device mydevice0 {
                      config {
             +            interface FE1 {
             +                ip-address 192.0.2.1;
             +            }
                      }
                  }
              }
             +example-service s1 {
             +    device [ mydevice0 ];
             +    ipaddress 192.0.2.1;
             +}
    }
}
admin@ncs(config-example-service-s1)# commit
Commit complete.
admin@ncs(config-example-service-s1)# do show running-config devices device mydevice0 config | display
<config xmlns="http://tail-f.com/ns/config/1.0">
  <devices xmlns="http://tail-f.com/ns/ncs">
    <device>
      <name>mydevice0</name>
      <config>
        <interface xmlns="http://example.org/ned/service-ownership" refcounter="1" backpointer="[ /exam
          <interface-name>FE1</interface-name>
          <ip-address refcounter="1">192.0.2.1</ip-address>
        </interface>
      </config>
    </device>
  </devices>
</config>
```

Additionally, if you add a second service instance targetting the same configuration, ownership is shared by the 2 service instances. The Refcount is 2 and there are 2 backpointers.

```
admin@ncs(config-example-service-s1)# example-service s2 device mydevice0 ipaddress 192.0.2.2
admin@ncs(config-example-service-s2)# commit dry-run
```

```
cli {
    local-node {
        data  devices {
                  device mydevice0 {
                      config {
                          interface FE1 {
             -                ip-address 192.0.2.1;
             +                ip-address 192.0.2.2;
                          }
                      }
                  }
              }
             +example-service s2 {
             +    device [ mydevice0 ];
             +    ipaddress 192.0.2.2;
             +}
    }
}
admin@ncs(config-example-service-s2)# commit
Commit complete.
admin@ncs(config-example-service-s2)# do show running-config devices device mydevice0 config | display
<config xmlns="http://tail-f.com/ns/config/1.0">
  <devices xmlns="http://tail-f.com/ns/ncs">
    <device>
      <name>mydevice0</name>
      <config>
        <interface xmlns="http://example.org/ned/service-ownership" refcounter="2" backpointer="[ /exam
          <interface-name>FE1</interface-name>
          <ip-address refcounter="2">192.0.2.2</ip-address>
        </interface>
      </config>
    </device>
  </devices>
</config>
```

Showcase: Only device owned

When you add data to CDB using load merge, ncs_cli or sync-from, without the use of a service, this data becomes device-owned. The refcount and backpointer are hidden.

```
admin@ncs(config)# no example-service
admin@ncs(config)# commit
Commit complete.
admin@ncs(config)# do show running-config devices device mydevice0 config
% No entries found.
admin@ncs(config)# load merge merge-config.xml
Loading.
386 bytes parsed in 0.00 sec (137.22 KiB/sec)
admin@ncs(config)# commit dry-run
cli {
    local-node {
        data  devices {
                  device mydevice0 {
                      config {
             +            interface FE1 {
             +                ip-address 192.0.2.1;
             +            }
                      }
```

```
                }
            }
    }
}
admin@ncs(config)# commit
Commit complete.
admin@ncs(config)# do show running-config devices device mydevice0 config | display service-meta-data |
<config xmlns="http://tail-f.com/ns/config/1.0">
  <devices xmlns="http://tail-f.com/ns/ncs">
    <device>
      <name>mydevice0</name>
      <config>
        <interface xmlns="http://example.org/ned/service-ownership">
          <interface-name>FE1</interface-name>
          <ip-address>192.0.2.1</ip-address>
        </interface>
      </config>
    </device>
  </devices>
</config>
```

Showcase: Device Owned + Service Owned

This example demonstrates how to easily create combined device and service ownership using a service and sync-from.

You deploy the service, then delete the service only in CDB using **Commit no-networking**. This way, the configuration still exists on the end-device. When you perform sync-from, the configuration is added back in CDB but it is device owned instead of service owned. Remember that **show running-config** when run in NSO shows you CDB data, not data currently on the devices.

```
admin@ncs(config)# no devices device mydevice0 config
admin@ncs(config)# commit
admin@ncs(config)# do show running-config devices device mydevice0 config
% No entries found.
admin@ncs(config)# example-service s1 device mydevice0 ipaddress 192.0.2.1
admin@ncs(config-example-service-s1)# commit
admin@ncs(config-example-service-s1)# top
admin@ncs(config)# no example-service s1
admin@ncs(config)# commit no-networking
Commit complete.
admin@ncs(config)# devices device mydevice0 sync-from
result true
admin@ncs(config)# do show running-config devices device mydevice0 config | display service-meta-data |
<config xmlns="http://tail-f.com/ns/config/1.0">
  <devices xmlns="http://tail-f.com/ns/ncs">
    <device>
      <name>mydevice0</name>
      <config>
        <interface xmlns="http://example.org/ned/service-ownership">
          <interface-name>FE1</interface-name>
          <ip-address>192.0.2.1</ip-address>
        </interface>
      </config>
    </device>
  </devices>
</config>
```

After sync-from, the configuration is only owned by the device. Refcounter is hidden. When deploying the service again, the refcount becomes 2, but backpointer is only displaying a single service instance. The second refcounter represents device ownership. The same rules apply as with shared service-ownership, if the service gets deleted, the configuration is not removed because the device also partially owns the configuration. Additionally, if the service data did not match the "original-value" as stored in the service-meta-data, NSO reverts the value back to the "original-value" if the service is removed.

```
admin@ncs(config)# example-service s1 device mydevice0 ipaddress 192.0.2.2
admin@ncs(config-example-service-s1)# commit dry-run
cli {
    local-node {
        data  devices {
                  device mydevice0 {
                      config {
                          interface FE1 {
            -                 ip-address 192.0.2.1;
            +                 ip-address 192.0.2.2;
                          }
                      }
                  }
             }
            +example-service s1 {
            +    device [ mydevice0 ];
            +    ipaddress 192.0.2.2;
            +}
    }
}
admin@ncs(config-example-service-s1)# commit
Commit complete.
admin@ncs(config-example-service-s1)# do show running-config devices device mydevice0 config | display
<config xmlns="http://tail-f.com/ns/config/1.0">
  <devices xmlns="http://tail-f.com/ns/ncs">
    <device>
      <name>mydevice0</name>
      <config>
        <interface xmlns="http://example.org/ned/service-ownership" refcounter="2" backpointer="[ /exam
          <interface-name>FE1</interface-name>
          <ip-address refcounter="2" original-value="192.0.2.1">192.0.2.2</ip-address>
        </interface>
      </config>
    </device>
  </devices>
</config>
```

Reconcile

Syntax: **<path-to-service instance> re-deploy reconcile**

Optional flags: **{ keep-non-service-config } dry-run { outformat native }**

Purpose

The core purpose for the reconcile functionality is to allow users to get rid of unwanted device-ownership

and transfer ownership fully to the services. When users already have a functioning network and they are trying to transfer ownership to NSO, they typically first introduce the configuration through sync-from operations. Once CDB has all of the network configuration, the user deploys service instances on top of the existing configuration. At this point, the configuration is still device owned which limits the service's ability to delete configuration. When users want to give their services full ownership of the configuration, they can use the reconcile functionality which does 3 things.

1) 1) Transfer ownership to services

2) 2) Remove "original value" flags

3) 3) Correcting service ownership

Transfer Ownership to Services

Reconcile evaluates all the config owned by a service, and if it finds any config which is owned by both this service and the device or is otherwise non-service owned, it removes this device ownership, making the service the exclusive owner. Reducing the refcount by 1.

Note: If 2 services own a piece of config, and that config is also non-service owned, the refcount is 3. Reconciling either one of the services removes that non-service ownership, reducing the refcount to 2 to reflect both services.

Remove original value flags

When a service is deployed and overwrites non-service owned data, refcount becomes 2 and NSO adds an "original value" tag. If the service instance is ever deleted, NSO tries to revert to that original value which existed before the service.

During reconcile, not only is the refcount reduced, but the original value is removed. Deleting the service now makes that value empty or change it to a default value as defined by the yang model.

Correcting service ownership

In some situations, ownership can be incorrectly assigned. Either device-owned configuration is incorrectly owned by the service, or configuration is incorrectly owned by both service and device, while it is expected to only be owned by the service. Reconcile can correct these misalignment. This is important to avoid issues where the service deletes non-service owned configuration.

Reconcile as re-deploy

Reconcile is a sub-function of service re-deploy. If a service is not in sync with CDB, the reconcile operation also executes a re-deploy in addition to performing the reconcile function.

Reconcile Inner Workings

While exact details of how reconcile operates are only known to the developers, this document provides a simplified understanding:

1) NSO Corrects service ownership

2) NSO deletes all configuration owned by this service instance from CDB, even if it is also owned by other services or is device-owned

3) NSO re-deploys the service instance

4) NSO restores service refcounts and backpointers from other services

Issues involving Service Ownership

Reconcile fails

In case you find "re-deploy" works but "re-deploy reconcile" fails: This can indicate that you have encountered a conflict between their service-design and the way the reconcile function operates.

The problem arises from service code which tries to read configuration from CDB which the service then later deploys. You can deploy this service only because the configuration already partially exists in CDB before deployment. But during reconcile, NSO temporarily deletes all configuration which is owned by this service, including the configuration the service is trying to read during the service re-deploy in the next step. This usually results in a java or python error reporting the failure to read the data.

Service is deleting non-service owned configuration

In this scenario, you encounter NSO deleting non-service owned configuration during the service instance delete or re-deploy. This is because the service instance created and owns the original configuration, and you later manually (through ncs_cli, sync-from or other method) added a part of the configuration inside a service owned container or list.

This new piece of configuration is not supposed to be owned by the service, but because the service has full ownership of the container or list, the service ends up indirectly owning it.

The way to resolve this is by using re-deploy reconcile { keep-non-service-config } to correct service ownership. When doing this reconcile, the refcount of the container or list increases to reflect that this container or list has both service-owned and non-service owned child nodes. Inside the parent node, only the service-owned nodes have a refcount and backpointer.

Showcase: Service is deleting non-service owned configuration

Starting with a single service instance deployed with full ownership, manually add a description to the interface using ncs_cli.

```
admin@ncs(config-example-service-s1)# do show running-config devices device mydevice0 config | display s
<config xmlns="http://tail-f.com/ns/config/1.0">
  <devices xmlns="http://tail-f.com/ns/ncs">
    <device>
      <name>mydevice0</name>
      <config>
        <interface xmlns="http://example.org/ned/service-ownership" refcounter="1" backpointer="[ /exam
          <interface-name>FE1</interface-name>
          <ip-address refcounter="1">192.0.2.1</ip-address>
        </interface>
      </config>
    </device>
  </devices>
</config>
admin@ncs(config-example-service-s1)# top
admin@ncs(config)# devices device mydevice0 config interface FE1 description "This is an example descri
admin@ncs(config-interface-FE1)# commit
```

```
Commit complete.
admin@ncs(config-interface-FE1)# do show running-config devices device mydevice0 config | display servi
<config xmlns="http://tail-f.com/ns/config/1.0">
  <devices xmlns="http://tail-f.com/ns/ncs">
    <device>
      <name>mydevice0</name>
      <config>
        <interface xmlns="http://example.org/ned/service-ownership" refcounter="1" backpointer="[ /exam
          <interface-name>FE1</interface-name>
          <ip-address refcounter="1">192.0.2.1</ip-address>
          <description>This is an example description</description>
        </interface>
      </config>
    </device>
  </devices>
</config>
```

Note how the refcount on <interface> remains 1 even though device-owned configuration was added. When trying to delete the service instance, the description is also deleted even though it is not supposed to be part of the service instance. To avoid this you can run the reconcile command.

```
admin@ncs(config-interface-FE1)# top
admin@ncs(config)# no example-service s1
admin@ncs(config)# commit dry-run
cli {
    local-node {
        data  devices {
                  device mydevice0 {
                      config {
             -            interface FE1 {
             -                ip-address 192.0.2.1;
             -                description "This is an example description";
             -            }
                      }
                  }
              }
             -example-service s1 {
             -    device [ mydevice0 ];
             -    ipaddress 192.0.2.1;
             -}
    }
}
admin@ncs(config)# abort
admin@ncs# config
Entering configuration mode terminal
admin@ncs(config)# example-service s1 re-deploy reconcile { keep-non-service-config }
admin@ncs(config)# do show running-config devices device mydevice0 config | display service-meta-data |
<config xmlns="http://tail-f.com/ns/config/1.0">
  <devices xmlns="http://tail-f.com/ns/ncs">
    <device>
      <name>mydevice0</name>
      <config>
        <interface xmlns="http://example.org/ned/service-ownership" refcounter="2" backpointer="[ /exam
          <interface-name>FE1</interface-name>
          <ip-address refcounter="1">192.0.2.1</ip-address>
          <description>This is an example description</description>
        </interface>
      </config>
```

```
    </device>
  </devices>
</config>
```

After the reconcile, the refcount for list interface has increased to 2. Meanwhile, refcount on leaf ip-address, has remained 1. The list entry "interface FE1" contains both service-owned and non-service owned data. By using reconcile, NSO re-evaluates the ownership and assign refcounts accordingly. Deletion now only targets the areas which are fully owned by the service instance. Neither the description or the list entry gets deleted.

```
admin@ncs(config)# no example-service s1
admin@ncs(config)# commit dry-run
cli {
    local-node {
        data  devices {
                  device mydevice0 {
                      config {
                          interface FE1 {
             -                ip-address 192.0.2.1;
                          }
                      }
                  }
              }
             -example-service s1 {
             -    device [ mydevice0 ];
             -    ipaddress 192.0.2.1;
             -}
    }
}
```

Keep-non-service-config vs Discard-non-service-config

Users sometimes misunderstand the use of discard-non-service-config.

In the reconcile example, "keep-non-service-config" was used. If discard is used it looks like this:

```
admin@ncs(config)# example-service s1 re-deploy reconcile { discard-non-service-config } dry-run
cli {
    local-node {
        data  devices {
                   device mydevice0 {
                       config {
                           interface FE1 {
              -                description "This is an example description";
                           }
                       }
                   }
               }

    }
}
```

The default is "keep-non-service-config". If neither option is defined NSO defaults to keep. Discard is rarely used as most users prefer to keep what is on their network, even if it is not managed by NSO. Reconcile { discard-non-service-config } dry-run can be used to find out which points of data exist in CDB which are not part of the service config but would get deleted if the service got deleted or re-deployed.

No Create

An alternative to using re-deploy reconcile to correct service-ownership when mixed with non-service owned data, is to prevent the conflict using the nocreate tag.

This is a tag which can be used in your XML service template. The documentation states “nocreate: the merge only affects configuration items that already exist in the template. It never creates the configuration with this tag. It only modifies existing configuration structures.”

Use of this tag has an interesting side-effect: Because the service does not create the node, it takes no ownership of it.

This is commonly used to prevent situations where NSO attempts to delete configuration which the device does not allow to be deleted.

Note that this tag gets inherited by child nodes, this means that if you add the nocreate tag to the interface, it also applies to any nodes inside that interface unless you mark them with a different tag such as merge"

Add a nocreate tag to the service template. If interface FE1 does not exist, nothing is configured.

```
<config-template xmlns="http://tail-f.com/ns/config/1.0"
                 servicepoint="example-service">
  <devices xmlns="http://tail-f.com/ns/ncs">
    <device>
      <name>{/device}</name>
      <config>
        <interface xmlns="http://example.org/ned/service-ownership" tags="nocreate">
          <interface-name>FE1</interface-name>
          <ip-address>{/ipaddress}</ip-address>
        </interface>
      </config>
    </device>
  </devices>
</config-template>
```

Recompile and reload the package, then test.

```
admin@ncs(config)# do show running-config devices device mydevice0 config | display service-meta-data |
No entries found.
admin@ncs(config)# example-service s1 device mydevice0 ipaddress 192.0.2.1
admin@ncs(config-example-service-s1)# commit dry-run
cli {
    local-node {
        data +example-service s1 {
             +    device [ mydevice0 ];
             +    ipaddress 192.0.2.1;
             +}
    }
```

```
}
```

Even though the same parameters were defined as before, the interface or any underlying configuration are not created in the device configuration.

Merge

Add a merge tag on the configuration inside of the interface. Do not add a tag to "interface-name" as this is the key of list interface. The key must always be allowed to inherit the behavior of the list. Recompile and reload the package.

```
<config-template xmlns="http://tail-f.com/ns/config/1.0"
                 servicepoint="example-service">
  <devices xmlns="http://tail-f.com/ns/ncs">
    <device>
      <name>{/device}</name>
      <config>
        <interface xmlns="http://example.org/ned/service-ownership" tags="nocreate">
          <interface-name>FE1</interface-name>
          <ip-address tags="merge">{/ipaddress}</ip-address>
        </interface>
      </config>
    </device>
  </devices>
</config-template>
```

Manually configure the interface FE1 prior to deploying the service.

```
admin@ncs(config)# no example-service
admin@ncs(config)# commit
admin@ncs(config)# do show running-config devices device mydevice0 config | display service-meta-data |
No entries found.
admin@ncs(config)# devices device mydevice0 config interface FE1
admin@ncs(config-interface-FE1)# commit dry-run
cli {
    local-node {
        data  devices {
                  device mydevice0 {
                      config {
             +            interface FE1 {
             +            }
                      }
                  }
              }
    }
}
admin@ncs(config-interface-FE1)# commit
Commit complete.
admin@ncs(config-interface-FE1)# top
admin@ncs(config)# example-service s1 device mydevice0 ipaddress 192.0.2.1
admin@ncs(config-example-service-s1)# commit dry-run
cli {
    local-node {
```

```
        data  devices {
                  device mydevice0 {
                      config {
                          interface FE1 {
             +                ip-address 192.0.2.1;
                          }
                      }
                  }
               }
              +example-service s1 {
              +    device [ mydevice0 ];
              +    ipaddress 192.0.2.1;
              +}
    }
}
admin@ncs(config-example-service-s1)# commit
Commit complete.
admin@ncs(config-example-service-s1)# do show running-config devices device mydevice0 config | display
<config xmlns="http://tail-f.com/ns/config/1.0">
  <devices xmlns="http://tail-f.com/ns/ncs">
    <device>
      <name>mydevice0</name>
      <config>
        <interface xmlns="http://example.org/ned/service-ownership">
          <interface-name>FE1</interface-name>
          <ip-address refcounter="1">192.0.2.1</ip-address>
        </interface>
      </config>
    </device>
  </devices>
</config>
```

Interface has a hidden refcount 1 : interface was deployed using ncs_cli, but it has a nocreate tag in the service package; the service did not take ownership. It is device-owned.

Primary has refcount 1: It is only owned by the service

If the service instance gets deleted, it would only delete ipaddress since that is the only part which is entirely owned by the service.