NSOサービスの所有権について

内容

はじめに

このドキュメントでは、Cisco® Network Service Orchestrator(NSO)の一般的な概念、一般的な落とし穴、およびサービス所有権のソリューションについて説明します。

前提条件

要件

このドキュメントは、NSO 6を含む現在利用可能なすべてのNSOバージョンに適用されます。ここで説明する動作は、サービスと非サービスの設定を組み合わせて使用するNSOセットアップにのみ適用されます。このドキュメント全体の例で使用されている特定のコマンドは、使用されているNetwork Element Driver（NED；ネットワーク要素ドライバ）にのみ適用されますが、基礎

となるロジックはNSOによって管理されるすべてのデバイスに適用されます。

使用するコンポーネント

- NSO 6.1.6
- NED:test-ned 1.0、このyangモデルを使用してカスタム構築されたnetconf NED。

```
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;
        }
    }
}
```

- サービス：example-service 1.0：このyangモデルとテンプレートを使用したカスタムビルドサービス

```
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;
    }
  }
}
```

```
        {/device}
```

```
          FE1
```

```
          {/ipaddress}
```

このドキュメントの情報は、特定のラボ環境にあるデバイスに基づいて作成されました。このドキュメントで使用するすべてのデバイスは、クリアな（デフォルト）設定で作業を開始しています。本稼働中のネットワークでは、各コマンドによって起こる可能性がある影響を十分確認してください。

基本

サービス所有権の目的は、NSOがどの設定がどのサービスに関連しているかを追跡できるようにすることです。サービスを削除する場合、NSOは関連する設定を削除する必要があり、削除する設定を判断するためにサービス所有権を使用します。構成が複数のサービスによって所有されている場合、サービスの1つを削除すると、その所有権参照が削除されます。構成自体はNSOデータベース(CDB)およびネットワークのデバイスに残ります。

所有権は、参照とバックポインタによって表示されます。refcountは、設定のその部分を所有するエンティティの数を示します。refcountは、サービスインスタンスの数に、設定がデバイス所有でもある場合は1を加えた数です。バックポインタは、これらのサービスインスタンスのパスを示します。「デバイス所有」を表示するバックポインタはありません。バックポインタは、リストおよびコンテナのCDBにのみ表示されます。個々のリーフはバックポインタを示しませんが、親から継承します。

デバイス所有

構成は、サービスによって所有されるだけでなく、デバイスによって所有されることもあります。これは、「デバイス所有」または「サービス非所有」と呼ばれることがあります。このドキュメントでは「デバイス所有」を使用していますが、これは理解しやすいものですが、サービス所有以外はデバイスを含める必要がないことに注意してください。LSAセットアップまたはスタックされたサービスは、デバイスなしでサービス非所有構成を持つことができます。

サービス導入を使用せずに設定をCDBに追加し、代わりにsync-from、load merge、ncs_cliなどの方法を使用して設定を設定すると、設定はデバイス所有になります。すでにデバイスが所有していた設定がサービスインスタンスに所有権を与えられると、共有の所有権を反映するためにrefcountが2に設定されます。サービスインスタンスが削除されても、削除前に1つのサービスインスタンスだけが設定を所有していたにもかかわらず、設定は削除されません。さらに、デバイスが所有する設定には「元の値」タグが追加されます。サービスインスタンスがデバイス所有の設定を上書きし、その後サービスが削除されると、設定は元の値に復元されます。

デバイスの所有権は、サービス以外の手段で追加したときにCDBに設定が存在していなかった場合にのみ割り当てられます。サービス所有の設定は、同期後にデバイス所有にはなりません。ただし、サービスを最上位に展開すると、デバイス所有の設定はデバイス所有とサービス所有の両方になります。

実施中の所有権

ショーケース：サービスのみ所有

ターゲット構成が空の状態でサービスを展開すると、サービスによって構成が作成され、その構成の所有権が取得されます。ユーザは、show running-configコマンドを使用して所有権を確認し、| service-meta-dataを表示します。必須ではありませんが、追加することを推奨します | display xml as the default CLI style output does not always correct how the data is modeled in CDB.(デフォルトのCLIスタイル出力としてxmlを表示すると、データがCDBでどのようにモデル化されているかが必ずしも正しく反映されない)

```
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
```

```
         mydevice0
```

```
            FE1




            192.0.2.1
```

さらに、同じ設定を対象とする2つ目のサービスインスタンスを追加すると、所有権は2つのサービスインスタンスで共有されます。Refcountは2で、2つのバックポインタがあります。

```
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 s
```

```
        mydevice0
```

```
          FE1
```

```
          192.0.2.2
```

ショーケース：所有デバイスのみ

サービスを使用せずに、ロードマージ、ncs_cli、またはsync-fromを使用してCDBにデータを追加すると、このデータはデバイス所有になります。refcountとbackpointerは隠しコマンドです。

```
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 |
```

```
         mydevice0
```

```
            FE1




            192.0.2.1
```

ショーケース：デバイス所有+サービス所有

この例では、サービスと同期元を使用して、デバイスとサービスの所有権を簡単に組み合わせる方法を示します。

サービスを導入し、Commit no-networkingを使用してCDBのみからサービスを削除します。このように、設定はエンドデバイス上に残ります。sync-fromを実行すると、設定はCDBに再び追加されますが、サービス所有ではなくデバイス所有になります。NSOでshow running-configを実行すると、現在デバイス上にあるデータではなく、CDBデータが表示されることに注意してください。

```
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 |
```

```
mydevice0
```

```
  FE1
```

```
  192.0.2.1
```

同期元の後、設定はデバイスによってのみ所有されます。Refcounterは非表示です。サービスを再び展開すると、refcountは2になりますが、backpointerは1つのサービスインスタンスのみを表示します。2つ目のrefcounterはデバイスの所有権を表します。共有サービス所有権の場合と同じ

ルールが適用されます。サービスが削除されても、デバイスが設定の一部を所有するため、設定は削除されません。 また、サービスデータがservice-meta-dataに保存されている「original-value」と一致しなかった場合、サービスが削除されると、NSOは値を「original-value」に戻します。

```
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
```

```
         mydevice0
```

```
FE1



192.0.2.2
```

調整

構文： <path-to-service instance> re-deploy reconcile

オプションのフラグ：{ keep-non-service-config } dry-run { outformat native }

目的

調整機能の主な目的は、ユーザが不要なデバイス所有権を取り除き、所有権をサービスに完全に移行できるようにすることです。ユーザがすでに機能しているネットワークを所有していて、所有権をNSOに転送しようとする場合、通常は最初に同期元の操作を通じて設定を導入します。CDBにすべてのネットワーク設定が完了すると、ユーザは既存の設定に加えてサービスインスタンスを展開します。この時点では、設定はまだデバイス所有であるため、サービスによる設定の削除機能は制限されています。ユーザがサービスに設定の完全な所有権を与えたい場合は、3つの処理を行う調整機能を使用できます。

1) 1)サービスへの所有権の移転

2) 2) 「元の値」フラグを削除する

3) 3)サービス所有権の修正

所有権のサービスへの移転

調整は、サービスが所有するすべての構成を評価します。また、このサービスとデバイスの両方によって所有されている構成、またはサービスによって所有されていない構成を見つけた場合、

そのデバイスの所有権を削除し、サービスを排他的な所有者にします。参照カウントを1減らします。

注：2つのサービスが構成の一部を所有し、その構成もサービス所有でない場合、refcountは3になります。いずれかのサービスを調整すると、そのサービス以外の所有権が削除され、両方のサービスを反映するためにrefcountが2に減ります。

元の値フラグの削除

サービスが展開され、サービスが所有していないデータを上書きすると、refcountは2になり、NSOは「original value」タグを追加します。サービスインスタンスが削除されると、NSOはサービスの前に存在していた元の値に戻そうとします。

調整時に、参照回数が減るだけでなく、元の値も削除されます。サービスを削除すると、その値は空になるか、yangモデルで定義されているデフォルト値に変更されます。

サービス所有権の修正

場合によっては、所有権が誤って割り当てられる可能性があります。デバイスが所有する設定がサービスによって誤って所有されているか、設定がサービスとデバイスの両方によって誤って所有されており、サービスによってのみ所有されることが想定されています。 調整を行うと、このような位置合わせの誤りを修正できます。これは、サービスが所有するサービス以外の設定を削除するという問題を回避するために重要です。

再展開として調整

調整は、サービスの再導入のサブ機能です。サービスがCDBと同期していない場合、調整操作では、調整機能の実行に加えて再配置も実行されます。

内部処理の調整

調整の動作方法の詳細は開発者にしか知られていませんが、このドキュメントでは次の項目について簡単に説明します。

1)NSOによるサービス所有権の修正

2)NSOは、他のサービスによっても所有されている場合や、デバイスが所有している場合でも、このサービスインスタンスによって所有されているすべての設定をCDBから削除します

3)NSOがサービスインスタンスを再導入

4)NSOは、他のサービスからのサービスの参照およびバックポインタを復元します。

サービス所有権に関する問題

調整が失敗しました

「再配備」は動作しているものの、「再配備」による調整が失敗した場合：これは、サービス設計と調整機能の動作方法との間に競合が発生したことを示している可能性があります。

この問題は、サービスが後で展開するCDBから設定を読み取ろうとするサービスコードから発生します。このサービスを展開できるのは、展開前にCDBに構成の一部が既に存在しているためです。ただし、調整中に、NSOは一時的にこのサービスが所有するすべての設定を削除します。これには、次のステップでサービスを再展開する際に読み取ろうとしている設定も含まれます。この場合、通常はJavaまたはPythonのエラーが発生し、データの読み取り工ラーが報告されます。

サービスが所有する構成を削除しています

このシナリオでは、サービスインスタンスの削除または再配置中に、NSOがサービス所有以外の構成を削除しています。これは、サービスインスタンスが作成して元の設定を所有し、後で手動で（ncs_cli、sync-from、またはその他の方法を使用して）サービス所有のコンテナまたはリスト内に設定の一部を追加したためです。

この新しい設定はサービスが所有するものではありませんが、サービスがコンテナまたはリストの完全な所有権を持っているため、サービスは間接的にコンテナまたはリストを所有することになります。

これを解決するには、re-deploy reconcile { keep-non-service-config }を使用してサービス所有権を修正します。この調整を行うと、コンテナまたはリストの参照カウントが増加し、このコンテナまたはリストにサービス所有およびサービス非所有の両方の子ノードがあることを反映します。親ノードの内部では、サービス所有ノードだけがrefcountとbackpointerを持ちます。

ショーケース：サービスによって所有されていない構成がサービスによって削除されている

完全な所有権で導入された単一のサービスインスタンスから開始し、ncs_cliを使用してインターフェイスに説明を手動で追加します。

```
admin@ncs(config-example-service-s1)# do show running-config devices device mydevice0 config | display s
```

```
        mydevice0
```

```
            FE1
```

```
            192.0.2.1
```

```
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
```

```
          mydevice0
```

```
        FE1




        192.0.2.1




        This is an example description
```

デバイスが所有する設定が追加されても、<interface>のrefcountが1のままであることに注意してください。サービスインスタンスを削除しようとすると、説明はサービスインスタンスの一部ではなくても削除されます。これを回避するには、reconcileコマンドを実行します。

```
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 |
```

mydevice0

FE1

192.0.2.1

```
            This is an example description
```

調整後、リストインターフェイスのrefcountが2に増加しました。一方、リーフIPアドレスのrefcountは1のままです。リストエントリ「interface FE1」には、サービス所有データとサービス非所有データの両方が含まれています。調整を使用すると、NSOは所有権を再評価し、それに応じて参照カウントを割り当てます。これで、サービスインスタンスによって完全に所有されているエリアだけが削除されます。説明もリストエントリも削除されません。

```
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とDiscard-non-service-configの比較

ユーザがdiscard-non-service-configの使用について誤解することがあります。

調整の例では、「keep-non-service-config」が使用されました。discardを使用すると、次のようになります。

```
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";
                            }
                        }
                    }
                }

    }
}
```

デフォルトは「keep-non-service-config」です。どちらのオプションも定義されていない場合、NSOはデフォルトで保持します。廃棄は、ほとんどのユーザがNSOによって管理されていない場合でも、ネットワーク上の内容を保持することを好むため、ほとんど使用されません。Reconcile { discard-non-service-config } dry-runを使用すると、CDBのどのデータポイントがサービス設定の一部でないかを調べることができますが、サービスが削除または再展開された場合には削除されます。

作成なし

再配置調整を使用してサービス所有以外のデータと混在させた場合にサービス所有権を修正する代わりに、 nocreateタグを使用して競合を回避することもできます。

これは、XMLサービステンプレートで使用できるタグです。ドキュメントには、「nocreate：マージは、テンプレートにすでに存在する設定項目にのみ影響します。このタグを使用して設定が作成されることはありません。既存のコンフィギュレーション構造を変更するだけです」

このタグの使用には興味深い副作用があります。サービスはノードを作成しないため、ノードの所有権は必要ありません。

これは通常、デバイスが削除を許可していない設定をNSOが削除しようとする状況を防ぐために使用されます。

このタグは子ノードに継承されることに注意してください。つまり、nocreateタグをインターフェイスに追加すると、mergeなどの別のタグでマークしない限り、そのインターフェイス内のすべてのノードにも適用されます

サービステンプレートにnocreateタグを追加します。インターフェイスFE1が存在しない場合は、何も設定されていません。

```
{/device}

  FE1

  {/ipaddress}
```

パッケージを再コンパイルしてリロードし、テストします。

```
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;
             +}
    }
}
```

以前と同じパラメータが定義されていても、インターフェイスまたは基盤となる設定はデバイス設定に作成されません。

マージ

インターフェイス内部の設定にマージタグを追加します。「interface-name」はlist interfaceのキーであるため、タグを追加しないでください。キーは常にリストの動作を継承できる必要があります。パッケージを再コンパイルしてリロードします。

```
         {/device}
```

```
           FE1
```

```
            {/ipaddress}
```

サービスを展開する前に、インターフェイスFE1を手動で設定します。

```
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
```

```
        mydevice0
```

```
          FE1
```

```
          192.0.2.1
```

インターフェイスには隠し参照カウント1があります。インターフェイスはncs_cliを使用して導入されましたが、サービスパッケージにnocreateタグがあります。サービスは所有権を取得しませんでした。デバイス所有である。

プライマリは参照カウント1を持つ：サービスによってのみ所有される

サービスインスタンスが削除されると、ipaddressだけが削除されます。これは、このサービスが完全に所有する唯一の部分であるためです。

翻訳について
シスコは世界中のユーザにそれぞれの言語でサポート コンテンツを提供するために、機械と人による翻訳を組み合わせて、本ドキュメントを翻訳しています。ただし、最高度の機械翻訳であっても、専門家による翻訳のような正確性は確保されません。シスコは、これら翻訳の正確性について法的責任を負いません。原典である英語版（リンクからアクセス可能）もあわせて参照することを推奨します。