LCS を使用した VNF ソフトウェアのアップグレード

ESC は、展開の更新中の VNF ソフトウェア アプリケーションのアップグレードをサポートします。ポリシーデータモデルを使用して、VNF アップグレードをサポートする新しいライフサイクルステージ（条件）が導入されます。VNF アップグレードポリシーは、VM グループごとに異なる場合があります。これらのポリシーは VM のグループに適用され、展開全体ではなく <vm_group> の下で指定できます。

VNF ソフトウェアのアップグレード

ESCは、展開内の初期イメージまたは基本イメージのアップグレードをサポートします。ESC ポリシーフレームワークは、新規および既存の VM のソフトウェアをアップグレードするためのカスタムスクリプトを提供します。ESC ポリシーフレームワークが最新であれば、VM の増分更新がサポートされます。

- 既存の VM のアップグレード：次の ESC ポリシーフレームワークは、ソフトウェアバージョンの更新前にすでに展開されている既存の VM をアップグレードするためのスクリプトをトリガーします。

  ```
  LCS::DEPLOY_UPDATE::POST_VM_SOFTWARE_VERSION_UPDATED
  ```

- 新しい VM のアップグレード：次の ESC ポリシーフレームワークは、導入時、回復時、またはスケールアウト時に新しい VM をアップグレードするためのスクリプトをトリガーします。

  ```
  LCS::DEPLOY::POST_VM_ALIVE
  ```

ボリュームを使用した VNF アップグレードの詳細については、「ボリュームを使用した VNF ソフトウェアのアップグレード」を参照してください。

VNF ソフトウェアバージョンの更新とソフトウェアアップグレードのトリガー

このシナリオでは、カスタムスクリプトを使用してソフトウェアアップグレードをトリガーする手順について説明します。次の例では、CSR VM がアップグレードされます。`csr_dep2.xml` を使用したサービスの更新により、カスタム スクリプト アクション LCS::DEPLOY_UPDATE::POST_VM_SOFTWARE_VERSION_UPDATED がトリガーされます。LCS は最初にその VM のモニタリングを無効にしてから、`csr_upgrade.exp` スクリプトを呼び出します。スクリプトが CSR に接続し、指定された upgrade.bin を CSR のブートフラッシュに scp し、ブートローダに新しい bin ファイルを指定し、CSR VM を再起動します。その後、bootup_time をリセットして、モニタリングを有効にします。bootup_time を使用すると、CSR は ESC によって再展開されることなく再起動を完了できます。

手順

ステップ 1 ESC VM を展開します。

ステップ 2 デイゼロ設定を `/var/tmp/csp-csr-day0-config` として ESC VM にアップロードします。

ステップ 3 カスタム アップグレード スクリプトを ESC VM にアップロードします。たとえば、csr_upgrade.exp スクリプトを `/var/tmp/csr_upgrade.exp` として ESC VM にアップロードします。

ステップ 4 `chmod +x /var/tmp/csr_upgrade.exp` を実行します。

ステップ 5 初期展開データモデル（dep.xml など）を編集して、関連する IP、ユーザ名、パスワード、および CSR のアップグレードバージョンを含めます。

ステップ 6 展開データモデル（dep.xml）のソフトウェアバージョンを編集して、アップグレードされた CSR バージョンを反映させます。

ステップ 7 ESC ユーザのホームディレクトリに CSR アップグレードをアップロードします。

ステップ 8 展開された CSR VM をアップグレードします。`esc_nc_cli edit-config csr_dep2.xml` コマンドを実行します。

ボリュームを使用した VNF ソフトウェアのアップグレード

サービスの初回展開時に、データモデルには、将来のソフトウェアアップグレード用に設定されたポリシーがあります。展開の更新要求を受信すると、展開の更新の一部として VM のアップグレードが開始されます。LCS::DEPLOY_UPDATE::VM_PRE_VOLUME_DETACH は、ESC がボリュームをデタッチする前にトリガーされます。このライフサイクルステージでは、デタッチする前にボリュームをアンマウントするスクリプトがサポートされています。ESC は、古いバージョンのソフトウェアを含む古いボリュームをデタッチし、削除します。ボリューム

が正常にデタッチされると、LCS::DEPLOY_UPDATE::VM_POST_VOLUME_DETACHED がトリガーされます。さらなるクリーンアップのため、この LCS でスクリプトが実行されます。新しいソフトウェアバージョンの新しいボリュームがアタッチされると、LCS::DEPLOY_UPDATE::VM_VOLUME_ATTACHED がトリガーされます。ESC は、ソフトウェアの新しいバージョンを含む新しいボリュームを作成してアタッチします。ボリュームをマウントし、ソフトウェアのインストールをトリガーするスクリプトが実行されます。ボリュームがアタッチされると、ESC が VM のソフトウェアバージョンを更新した後に、LCS::DEPLOY_UPDATE::VM_SOFTWARE_VERSION_UPDATED がトリガーされます。この段階で、ソフトウェアアップグレードの設定を完了するためのスクリプトが実行されます。

VNF ソフトウェアアップグレードのデータモデル：

```
<esc_datamodel xmlns="http://www.cisco.com/esc/esc">
  <tenants>
    <tenant>
      <name>test</name>
      <deployments>
        <deployment>
          <name>dep</name>
          <vm_group>
            <name>Group1</name>
            <volumes>
              <volume nc:operation="delete">
                <name>v1.0</name>
                <volid>0</volid>
              </volume>
              <volume>
                <name>v2.0</name>
                <volid>1</volid>
                <sizeunit>GiB</sizeunit>
                <size>2</size>
                <bus>virtio</bus>
                <type>lvm</type>
                <image>Image-v2</image>
              </volume>
            </volumes>
            <software_version>2.0</software_version>
            <policies>
              <policy>
                <name>SVU1</name>
                <conditions>
                  <condition>
                    <name>LCS::DEPLOY_UPDATE::PRE_VM_VOLUME_DETACH</name>
                  </condition>
                </conditions>
                <actions>
                  <action>
                    <name>LOG</name>
                    <type>pre_defined</type>
                  </action>
                </actions>
              </policy>
              <policy>
                <name>SVU2</name>
                <conditions>
                  <condition>
                    <name>LCS::DEPLOY_UPDATE::POST_VM_VOLUME_ATTACHED</name>
                  </condition>
                </conditions>
                <actions>
```

```
                    <action>
                      <name>LOG</name>
                      <type>pre_defined</type>
                    </action>
                  </actions>
                </policy>
                <policy>
                  <name>SVU3</name>
                  <conditions>
                    <condition>
                      <name>LCS::DEPLOY_UPDATE::POST_VM_SOFTWARE_VERSION_UPDATED</name>
                    </condition>
                  </conditions>
                  <actions>
                    <action>
                      <name>LOG</name>
                      <type>pre_defined</type>
                    </action>
                  </actions>
                </policy>
              </policies>
            </vm_group>
          </deployment>
        </deployments>
      </tenant>
    </tenants>
</esc_datamodel>
```

このデータモデルでは、volid が 0 の既存のボリューム v1.0 が削除されます。volid が 1 の新しいボリューム v2.0 が追加されます。ソフトウェアバージョンである <software_version> 値が 1.0 から 2.0 に変更されます。VNF ソフトウェアアップグレード用に 3 つのポリシーが追加されました。

（注）

- 新しいボリュームを削除して作成する代わりに、ボリュームのプロパティを更新できます。name、vol_id、および image プロパティを保持できます。上記の 3 つのプロパティのいずれかが変更されると、ボリュームが削除され、再度作成されます。
- ボリュームサイズを拡張でき、ブート可能プロパティを変更できます。ボリュームタイプなどのその他のプロパティやイメージプロパティを変更すると、ボリュームが再度作成されます。
- ボリューム ID を更新するには、ボリュームを削除し、別のボリューム ID でボリュームを再度追加する必要があります。
- ESC によって作成されたボリュームは、同じボリューム ID のアウトオブバンドボリュームによって更新することはできません。その逆も同様です。

ボリュームを使用した VNF ソフトウェアアップグレードでサポートされるライフサイクルステージ（LCS）

各ライフサイクルステージには、条件とアクションがあります。条件に基づいて、アクションが実行されます。ポリシー主導型データモデルの詳細については、ポリシー駆動型データモデ

ルを参照してください。VNF ソフトウェアアップグレードには、次の 3 つの条件が設定されています。

条件名	範囲	説明
LCS::DEPLOY_UPDATE::VM_PRE_VOLUME_DETACH	展開	ESC がボリュームをデタッチする直前にトリガーされます
LCS::DEPLOY_UPDATE::POST_VM_VOLUME_DETACHED	展開	ESC がボリュームをデタッチした直後にトリガーされます
LCS::DEPLOY_UPDATE::POST_VM_VOLUME_ATTACHED	展開	ESC が新しいボリュームをアタッチした直後にトリガーされます
LCS:DEPLOY_UPDATE:POST_VM_SOFTWARE_VERSION_UPDATED	展開	ESC が VM のソフトウェアバージョンを更新した直後にトリガーされます

LCS::DEPLOY_UPDATE::PRE_VM_VOLUME_DETACH

この LCS 条件は、ESC がボリュームをデタッチする前にトリガーされます。デタッチする前に、ボリュームをアンマウントするスクリプトが実行されます。

```
<policy>
    <name>SVU1</name>
    <conditions>
        <condition>
            <name>LCS::DEPLOY_UPDATE::PRE_VM_VOLUME_DETACH</name>
        </condition>
    </conditions>
    <actions>
        <action>
            <name>LOG</name>
            <type>pre_defined</type>
        </action>
    </actions>
</policy>
```

LCS::DEPLOY_UPDATE::POST_VM_VOLUME_ATTACHED

この LCS は、ESC が新しいボリュームをアタッチした後にトリガーされます。ボリュームをマウントし、新しいアプリケーションを新しいボリュームにインストールするスクリプトが実行されます。

```
<policy>
    <name>SVU2</name>
    <conditions>
        <condition>
            <name>LCS::DEPLOY_UPDATE::POST_VM_VOLUME_ATTACHED</name>
        </condition>
    </conditions>
    <actions>
        <action>
            <name>LOG</name>
```

```
            <type>pre_defined</type>
         </action>
    </actions>
</policy>
```

LCS::DEPLOY_UPDATE::POST_VM_SOFTWARE_VERSION_UPDATED

この LCS は、ESC が VM のソフトウェアバージョンを更新した後にトリガーされます。ソフトウェアのアップグレードを完了するための最終設定を実行するスクリプトが実行されます。

```
<policy>
   <name>SVU3</name>
   <conditions>
      <condition>
         <name>LCS::DEPLOY_UPDATE::POST_VM_SOFTWARE_VERSION_UPDATED</name>
      </condition>
   </conditions>
   <actions>
      <action>
         <name>LOG</name>
         <type>pre_defined</type>
      </action>
   </actions>
</policy>
```

（注） 上記の3つのポリシーはすべて、LOGアクションをデータモデルサンプルの定義済みアクションとして示しています。スクリプトの実行が必要な場合は、SCRIPT アクションを追加できます。サンプルスクリプトについては、以下の「スクリプトアクション」セクションを参照してください。

スクリプト アクション

上記の例では、すべてのアクションは事前定義されたログです。代わりにカスタムスクリプトを使用できます。

```
<action>
   <name>unmount_volume</name>
   <type>SCRIPT</type>
   <properties>
      <property>
         <name>script_filename</name>
         <value>/opt/cisco/esc/esc-scripts/unmount.sh</value>
      </property>
      <property>
         <name>user_param</name>
         <value>value</value>
      </property>
   </properties>
</action>
```

すべてのプロパティ名と値のペアは、スペースで区切られたパラメータとしてスクリプトに渡されます。上記の例では、unmount.sh 値は次のようにスクリプトによって呼び出されます。

```
/opt/cisco/esc/esc-scripts/unmount.sh user_param value
```

ESC 内部 ID を指定したスクリプトに渡すように、事前に作成されたプロパティ名を設定できます。事前に作成されたプロパティ名は次のとおりです。

```
<property>
  <name>internal_deployment_id</name>
</property>
<property>
  <name>external_deployment_id</name>
</property>
<property>
  <name>deployment_name</name>
</property>
<property>
  <name>internal_tenant_id</name>
</property>
<property>
  <name>external_tenant_id</name>
</property>
```

ESCが生成する、事前に作成されたプロパティ名と値を含むスクリプトの例を次に示します。

```
script_name.sh deployment_name my-deployment-name external_deployment_id
18fbcfd5-8b63-44e0-97ec-68de25902917
external_tenant_id my-tenant-id internal_deployment_id my-tenant-idmy-deployment-name
internal_tenant_id my-tenant-id
```

デフォルトで、ESC ではスクリプトの実行が完了するまでに 15 分かかります。一部のスクリプトは完了までにさらに時間がかかる場合があります。オプションのプロパティを指定して、タイムアウト値を秒単位で延長できます。次の例では、スクリプトのタイムアウトは 3600 秒に設定されています。

```
<property>
    <name>wait_max_timeout</name>
    <value>3600</value>
</property>
```

仮想ネットワーク機能ソフトウェアアップグレードの通知

通知は、VNF ソフトウェアアップグレードの各段階でトリガーされます。

デタッチされたボリューム

```
status SUCCESS
        status_code 200
        status_message Detached 1 volume: [Volume=test-esc-1,volid=1]
        depname dep
        tenant test
        tenant_id 9132cc90b8324a1c95a6c00975af6206
        depid eb4fe3b5-138d-41a3-b6ff-d6fa9035ca6c
        vm_group Group1
        vm_source {
            vmid cd4eeb61-61db-45a6-9da1-793be08c4de6
            hostid 8e96b8830d7bfbb337ce665586210fcca9644cbe238240e207350735
            hostname my-server-5
            software_version 1.0
            interfaces {
                interface {
                    nicid 0
```

```
                    type virtual
                    port_id 26412180-45cf-4f0b-ab45-d05bb7ca7091
                    network 943fda9e-79f8-400c-b442-3506f102721a
                    subnet e313b95c-ca1f-4c81-8d60-c9e721a85d0b
                    ip_address 192.168.0.56
                    mac_address fa:16:3e:18:90:1e
                    netmask 255.255.255.0
                    gateway 192.168.0.1
                }
            }
            volumes {
                volume {
                    display_name test-esc-1__v0_0_0_1
                    external_id 5d008a12-6fb1-492a-b648-4cf7fc8c68b1
                    bus virtio
                    type lvm
                    size 2
                }
            }
        }
        vm_target {
        }
        event {
            type VM_UPDATED
        }
    }
}
```

削除されたボリューム

```
notification {
    eventTime 2016-11-24T00:27:25.457+00:00
    escEvent {
        status SUCCESS
        status_code 200
        status_message Removed 1 volume: [Volume=test-esc-3,volid=1]
        depname dep
        tenant test
        tenant_id 9132cc90b8324a1c95a6c00975af6206
        depid f938ca24-d0c2-42b3-a757-66b0543fe0a6
        vm_group Group1
        vm_source {
            vmid 91379ad1-1cfc-4a10-abaf-068d01ae92b9
            hostid 101f55110748903af4844a2517e854f64843b9ac8d880ad68be8af59
            hostname my-server-4
            software_version 1.0
            interfaces {
                interface {
                    nicid 0
                    type virtual
                    port_id a8201c3e-2c6e-4313-94d0-1b4eee14f08a
                    network 943fda9e-79f8-400c-b442-3506f102721a
                    subnet e313b95c-ca1f-4c81-8d60-c9e721a85d0b
                    ip_address 192.168.0.220
                    mac_address fa:16:3e:eb:bd:77
                    netmask 255.255.255.0
                    gateway 192.168.0.1
                }
            }
        }
        vm_target {
        }
        event {
            type VM_UPDATED
```

```
        }
    }
}
```

アタッチされたボリューム

```
notification {
    eventTime 2016-11-23T19:54:48.105+00:00
        status_message Attached 1 volume: [Volume=test-esc-2,volid=0]
        depname dep
        tenant test
        tenant_id 9132cc90b8324a1c95a6c00975af6206
        depid eb4fe3b5-138d-41a3-b6ff-d6fa9035ca6c
        vm_group Group1
        vm_source {
            vmid cd4eeb61-61db-45a6-9da1-793be08c4de6
            hostid 8e96b8830d7bfbb337ce665586210fcca9644cbe238240e207350735
            hostname my-server-5
            software_version 1.1
            interfaces {
                interface {
                    nicid 0
                    type virtual
                    port_id 26412180-45cf-4f0b-ab45-d05bb7ca7091
                    network 943fda9e-79f8-400c-b442-3506f102721a
                    subnet e313b95c-ca1f-4c81-8d60-c9e721a85d0b
                    ip_address 192.168.0.56
                    mac_address fa:16:3e:18:90:1e
                    netmask 255.255.255.0
                    gateway 192.168.0.1
                }
            }
            volumes {
                volume {
                    display_name test-esc-2__v0_0_0_1
                    external_id bf5c9a01-e9fb-42fa-89ee-73699d6c519c
                    bus virtio
                    type lvm
                    size 2
                }
            }
        }
        vm_target {
        }
        event {
            type VM_UPDATED
        }
    }
}
```

更新されたソフトウェアバージョン

```
notification {
    eventTime 2016-11-23T20:06:56.75+00:00
    escEvent {
        status SUCCESS
        status_code 200
        status_message VM Software Updated. VM name:
[dep_Group1_0_c9edef63-4d9d-43ea-af1b-16527ed2edae], previous version: [1.0], current
version: [1.1]
        depname dep
        tenant test
        tenant_id 9132cc90b8324a1c95a6c00975af6206
```

```
        depid eb4fe3b5-138d-41a3-b6ff-d6fa9035ca6c
        vm_group Group1
        vm_source {
            vmid cd4eeb61-61db-45a6-9da1-793be08c4de6
            hostid 8e96b8830d7bfbb337ce665586210fcca9644cbe238240e207350735
            hostname my-server-5
            software_version 1.1
            interfaces {
                interface {
                    nicid 0
                    type virtual
                    port_id 26412180-45cf-4f0b-ab45-d05bb7ca7091
                    network 943fda9e-79f8-400c-b442-3506f102721a
                    subnet e313b95c-ca1f-4c81-8d60-c9e721a85d0b
                    ip_address 192.168.0.56
                    mac_address fa:16:3e:18:90:1e
                    netmask 255.255.255.0
                    gateway 192.168.0.1
                }
            }
            volumes {
                volume {
                    display_name test-esc-2__v0_0_0_1
                    external_id bf5c9a01-e9fb-42fa-89ee-73699d6c519c
                    bus virtio
                    type lvm
                    size 2
                }
            }
        }
        vm_target {
        }
        event {
            type VM_SOFTWARE_VERSION_UPDATED
        }
    }
}
```

更新されたサービス

```
notification {
    eventTime 2016-11-23T20:06:56.768+00:00
    escEvent {
        status SUCCESS
        status_code 200
        status_message Service group update completed successfully
        depname dep
        tenant test
        tenant_id 9132cc90b8324a1c95a6c00975af6206
        depid eb4fe3b5-138d-41a3-b6ff-d6fa9035ca6c
        vm_source {
        }
        vm_target {
        }
        event {
            type SERVICE_UPDATED
        }
    }
}
```

展開内の VNF のアップグレード

ESC では、次のライフサイクルステージのいずれかで、既存の展開の VNF ソフトウェアをアップグレードできます。

- LCS：PRE SOFTWARE UPGRADE-SCRIPT ACTION
- LCS：POST SOFTWARE UPGRADE-SCRIPT ACTION

NB は、PRE、POST、または BOTH を使用してカスタム アクション スクリプトを実行することを選択できます。

カスタムスクリプトの詳細については、スクリプトアクションのカスタムスクリプトを参照してください。ライフサイクルステージについては、さまざまなステージで定義されているライフサイクルステージ（LCS）ポリシーの条件を参照してください。

LCS_NOTIFY 通知は、ライフサイクルの各ステージでオンまたはオフにできます。software_version の変更では、各 VM の最終通知は VM_SOFTWARE_VERSION_UPDATED となります。ESC は、展開の更新ごとに SERVICE_UPDATED 通知を受信します。

ESC は、既存の展開で次の VNF ソフトウェア アップグレード シナリオをサポートします。

- 展開後の VNF のアップグレード
- 既存の展開での VNF 展開とアプリケーションのアップグレード

既存の展開内で他のリソースを更新する方法の詳細については、既存の展開の更新を参照してください。

展開後の VNF のアップグレード

VNF のアップグレードは、単一または段階的なトランザクションで実行できます。

ESC は、単一のトランザクションで LCS ポリシーを追加し、ソフトウェアバージョンを変更します。

2 段階のトランザクションでは、ESC は最初のトランザクションで LCS ポリシーを追加し、2 番目のトランザクションでソフトウェアバージョンの変更を伴うソフトウェアアップグレードをトリガーします。

通知

- LCS_NOTIFY—LCS::DEPLOY_UPDATE::PRE_VM_SOFTWARE_VERSION_UPDATE
- LCS_NOTIFY—LCS::DEPLOY_UPDATE::POST_VM_SOFTWARE_VERSION_UPDATED
- VM_SOFTWARE_VERSION_UPDATED
- SERVICE_UPDATED

[エラー（Error）]

ESC は、VNF アップグレードプロセスの早期検証を実行します。カスタムスクリプトファイルが存在しない場合は、エラーが発生します。トランザクションは拒否され、通知はNFVOに送信されません。

カスタムスクリプトがタイムアウトすると、エラーが発生します。次の通知がNFVOに送信されます。

- LCS::DEPLOY_UPDATE::PRE_VM_SOFTWARE_VERSION_UPDATE
- LCS::DEPLOY_UPDATE::PRE_VM_SOFTWARE_VERSION_UPDATE
- VM_SOFTWARE_VERSION_UPDATED
- SERVICE_UPDATED

既存の展開での VNF 展開とアプリケーションのアップグレード

VNF の展開およびアプリケーションのアップグレード中に、ESC は次の通知を NFVO に送信します。

- VM_DEPLOYED
- LCS_NOTIFY-LCS::DEPLOY::POST_VM_ALIVE
- VM_ALIVE
- SERVICE_ALIVE

[エラー（Error）]

カスタムスクリプトがタイムアウトすると、エラーが発生します。次の通知がNFVOに送信されます。

- VM_DEPLOYED
- LCS::VM::POST_VM_ALIVE
- VM_DEPLOYED
- SERVICE_ALIVE