OpenStack での仮想ネットワーク機能の展開

OpenStack での仮想ネットワーク機能の展開

ここでは、Elastic Services Controller（ESC）のいくつかの展開シナリオと VNF の展開手順について説明します。次の表に、さまざまな展開シナリオを示します。

シナリオ	説明	リソース	利点
ESC を使用してイメージとフレーバーを作成することにより、単一の VIM に VNF を展開する	展開データモデルは、作成されたイメージとフレーバーを参照して、VNF を展開します。	イメージとフレーバーは、NETCONF/REST API を使用して ESC で作成されます。	• イメージとフレーバーは、複数の VNF 展開で使用できます。 • ESC によって作成されたリソース（イメージ、フレーバー、およびボリューム）を削除できます。

シナリオ	説明	リソース	利点
アウトオブバンドイメージ、フレーバー、ボリューム、およびポートを使用した単一 VIM への VNF の展開	展開データモデルは、OpenStack のアウトオブバンドイメージ、フレーバー、ボリューム、およびポートを参照して、VNF を展開します。	イメージ、フレーバー、ボリューム、およびポートは、ESC を使用して作成されません。	• イメージ、フレーバー、ボリューム、ポートは、複数の VNF 展開で使用できます。 • ESC を使用して作成されていないリソースは削除できません。
アウトオブバンドリソースを使用した複数の VIM への VNF の展開	展開データモデルは、アウトオブバンドイメージ、フレーバー、ネットワーク、および VIM プロジェクトを参照して、VNF を展開します。	イメージ、フレーバー、VIM プロジェクト（ロケータで指定）およびネットワークは、ESC を使用して作成されません。これらは、VIM のアウトオブバンドに存在する必要があります。	展開内の ESC で設定する必要がある（VM を展開するための）VIM を指定できます。

複数の OpenStack VIM に VNF を展開するには、「複数の OpenStack VIM への VNF の展開」を参照してください。

単一の OpenStack VIM での VNF の展開

VNF の展開は、ESC ポータルまたはノースバウンド インターフェイスから発信されるサービス要求として開始されます。サービス要求は XML ペイロードで構成されます。ESC は、次の展開シナリオをサポートします。

- ESC を使用したイメージおよびフレーバの作成による VNF の展開
- アウトオブバンドイメージ、フレーバ、ボリューム、およびポートを使用した VNF の展開

VNF を展開する前に、OpenStack でイメージ、フレーバ、ボリューム、およびポートが使用可能であることを確認するか、これらのリソースを作成する必要があります。イメージ、フレーバ、およびボリュームの作成の詳細については、リソース管理の概要を参照してください。

展開では、展開と同じテナントによってアウトオブバンドポートを作成する必要があります。ポートの設定の詳細については、『*Cisco Elastic Services Controller Administration Guide*』の「Interface Configurations」を参照してください。

複数の VIM に VM を展開するには、「複数の OpenStack VIM への VNF の展開」を参照してください。

展開中、ESCは展開データモデルで展開の詳細を検索します。展開データモデルの詳細については、「Cisco Elastic Services Controller Deployment Attributes」を参照してください。ESC が特定のサービスに対する展開の詳細を見つけることができない場合は、*vm_group* の既存のフレーバとイメージを使用して展開を続行します。ESCがイメージとフレーバの詳細を検出できない場合、展開は失敗します。

重要 ネットワークに使用するサブネットを指定することもできます。展開データモデルでは、サブネットを指定する新しい `subnet` 属性が導入されています。詳細については、「Cisco Elastic Services Controller Deployment Attributes」を参照してください。

（注） SERVICE_UPDATE 設定が失敗すると、VM の最小数と最大数が変化し、スケールインまたはスケールアウトが発生します。OpenStack で発生したエラーのため、ESC は設定内の VM の最小数または最大数をロールバックできません。CDB（ESC DB）が同期していません。この場合、手動ロールバックを実行するには、別の SERVICE_UPDATE 設定を実行する必要があります。

OpenStack での展開では、UUID または名前を使用してイメージとフレーバを参照できます。名前は VIM で一意である必要があります。同じ名前の複数のイメージがある場合、展開は正しいイメージを識別できず、展開は失敗します。

すべての展開およびESCイベント通知にテナントUUIDが表示されます。次に例を示します。

```
<?xml version="1.0" encoding="UTF-8"?>
<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
  <eventTime>2016-01-22T15:14:52.484+00:00</eventTime>
  <escEvent xmlns="http://www.cisco.com/esc/esc">
    <status>SUCCESS</status>
    <status_code>200</status_code>
    <status_message>VIM Driver: VM successfully created,
      VM Name:
[SystemAdminxyz_abc_NwDepMod1_0_5e6b7957-20e7-4df9-9113-e5fc8c047e91]</status_message>
    <depname>test_NwDepModVmGrp1</depname>
    <tenant>admin</tenant>
    <tenant_id>62cd11f560b44bf5815eaad41fc94c80</tenant_id>
 </event>
```

再起動時間パラメータ

再起動時間パラメータが展開要求に導入されます。これにより、展開におけるリカバリの再起動待機時間をよりきめ細かく制御できます。展開では、VMが再起動すると、モニタに再起動時間が設定されます。VM ALIVE イベントの前に再起動時間が経過すると、VM_RECOVERY_COMPLETE や undeploy などの次のアクションが実行されます。

（注） 再起動時間が指定されていない場合は、ブートアップ時間が使用されます。

データモデルの変更は次のとおりです。

```
<?xml version="1.0" encoding="UTF-8"?>
<esc_datamodel xmlns="http://www.cisco.com/esc/esc">
   <tenants>
      <tenant>
         <name>tenant</name>
         <deployments>
            <deployment>
               <name>depz</name>
               <vm_group>
                  <name>g1</name>
                  <image>Automation-Cirros-Image</image>
                  <flavor>Automation-Cirros-Flavor</flavor>
                  <reboot_time>30</reboot_time>
                  <recovery_wait_time>10</recovery_wait_time>
                  <interfaces>
                     <interface>
                        <nicid>0</nicid>
                        <port>pre-assigned_IPV4_1</port>
                        <network>my-network</network>
                     </interface>
                  </interfaces>
                  <kpi_data>
                     <kpi>
                        <event_name>VM_ALIVE</event_name>
                        <metric_value>1</metric_value>
                        <metric_cond>GT</metric_cond>
                        <metric_type>UINT32</metric_type>
                        <metric_collector>
                           <nicid>0</nicid>
                           <type>ICMPPing</type>
                           <poll_frequency>3</poll_frequency>
                           <polling_unit>seconds</polling_unit>
                           <continuous_alarm>false</continuous_alarm>
                        </metric_collector>
                     </kpi>
                  </kpi_data>
                  <rules>
                     <admin_rules>
                        <rule>
                           <event_name>VM_ALIVE</event_name>
                           <action>ALWAYS log</action>
                           <action>TRUE servicebooted.sh</action>
                           <action>FALSE recover autohealing</action>
                        </rule>
                     </admin_rules>
                  </rules>
                  <config_data />
                  <scaling>
                     <min_active>1</min_active>
                     <max_active>2</max_active>
                     <elastic>true</elastic>
                  </scaling>
                  <recovery_policy>
                     <recovery_type>AUTO</recovery_type>
                     <action_on_recovery>REBOOT_ONLY</action_on_recovery>
                     <max_retries>1</max_retries>
                  </recovery_policy>
               </vm_group>
            </deployment>
         </deployments>
      </tenant>
```

```
   </tenants>
</esc_datamodel>
```

通知の例は次のとおりです。

```
20:43:48,133 11-Oct-2016 WARN  ===== SEND NOTIFICATION STARTS =====
20:43:48,133 11-Oct-2016 WARN  Type: VM_RECOVERY_INIT
20:43:48,134 11-Oct-2016 WARN  Status: SUCCESS
20:43:48,134 11-Oct-2016 WARN  Status Code: 200
20:43:48,134 11-Oct-2016 WARN  Status Msg: Recovery event for
VM [dep-12_CSR1_c_0_37827511-be08-4702-b0bd-1918cb995118] triggered.
20:43:48,134 11-Oct-2016 WARN  Tenant: gilan-test-5
20:43:48,134 11-Oct-2016 WARN  Service ID: NULL
20:43:48,134 11-Oct-2016 WARN  Deployment ID: f6ff8164-fe6d-4589-84fa-f39d676e9231
20:43:48,134 11-Oct-2016 WARN  Deployment name: dep-12
20:43:48,134 11-Oct-2016 WARN  VM group name: CSR1_cirros
20:43:48,134 11-Oct-2016 WARN  VM Source:
20:43:48,134 11-Oct-2016 WARN  VM ID: 90d2066c-9a07-485b-8f72-b51026a62922
20:43:48,134 11-Oct-2016 WARN  Host ID:
69c3fba0a5b5ffff211bd05b9da7e2130d98d005a9bef71ace7d09ff
20:43:48,134 11-Oct-2016 WARN  Host Name: my-server
20:43:48,134 11-Oct-2016 WARN  [DEBUG-ONLY] VM IP: 192.168.0.75;
20:43:48,135 11-Oct-2016 WARN  =====  SEND NOTIFICATION ENDS  =====
20:43:56,149 11-Oct-2016 WARN
20:43:56,149 11-Oct-2016 WARN  ===== SEND NOTIFICATION STARTS =====
20:43:56,149 11-Oct-2016 WARN  Type: VM_RECOVERY_REBOOT
20:43:56,149 11-Oct-2016 WARN  Status: SUCCESS
20:43:56,149 11-Oct-2016 WARN  Status Code: 200
20:43:56,150 11-Oct-2016 WARN  Status Msg: VM
[dep-12_CSR1_c_0_37827511-be08-4702-b0bd-1918cb995118] is rebooted.
20:43:56,150 11-Oct-2016 WARN  Tenant: gilan-test-5
20:43:56,150 11-Oct-2016 WARN  Service ID: NULL
20:43:56,150 11-Oct-2016 WARN  Deployment ID: f6ff8164-fe6d-4589-84fa-f39d676e9231
20:43:56,150 11-Oct-2016 WARN  Deployment name: dep-12
20:43:56,150 11-Oct-2016 WARN  VM group name: CSR1_cirros
20:43:56,150 11-Oct-2016 WARN  VM Source:
20:43:56,151 11-Oct-2016 WARN    VM ID: 90d2066c-9a07-485b-8f72-b51026a62922
20:43:56,151 11-Oct-2016 WARN    Host ID:
69c3fba0a5b5ffff211bd05b9da7e2130d98d005a9bef71ace7d09ff
20:43:56,151 11-Oct-2016 WARN    Host Name: my-server
20:43:56,152 11-Oct-2016 WARN    [DEBUG-ONLY] VM IP: 192.168.0.75;
20:43:56,152 11-Oct-2016 WARN  =====  SEND NOTIFICATION ENDS  =====
20:44:26,481 11-Oct-2016 WARN
20:44:26,481 11-Oct-2016 WARN  ===== SEND NOTIFICATION STARTS =====
20:44:26,481 11-Oct-2016 WARN  Type: VM_RECOVERY_COMPLETE
20:44:26,481 11-Oct-2016 WARN  Status: FAILURE
20:44:26,481 11-Oct-2016 WARN  Status Code: 500
20:44:26,481 11-Oct-2016 WARN  Status Msg: Recovery: Recovery completed with errors
```

複数の OpenStack VIM への VNF の展開

ESC を使用して、同じタイプの複数の VIM に VNF を展開できます。ESC は、複数の OpenStack VIM での VNF の展開をサポートします。OpenStack の単一インスタンスに VM を展開するには、OpenStack での仮想ネットワーク機能の展開 （1 ページ）を参照してください。

VNF を複数の VIM に展開するには、次の手順を実行する必要があります。

- VIM コネクタとそのクレデンシャルを設定します。
- ESC 内にテナントを作成する

VIM コネクタは VIM を ESC に登録します。VNF を複数の VIM に展開するには、VIM の各インスタンスに VIM コネクタとそのクレデンシャルを設定する必要があります。VIM コネクタは、インストール時に bootvm.py パラメータを使用するか、VIM コネクタ API を使用して設定できます。デフォルトの VIM コネクタは、単一の VIM 展開に使用されます。マルチ VIM 展開では、VIM コネクタを指定するためにロケータ属性が使用されます。

通常、マルチ VIM 展開をサポートする ESC は以下を備えています。

- ESC がリソースを作成および管理するデフォルトの VIM
- 展開のみがサポートされているデフォルト以外の VIM

詳細については、VIM コネクタの管理を参照してください。

（vim_mapping 属性が false に設定されている）ESC 内のテナントであるデータモデル階層内のルートテナントと、ロケータ属性内に配置されたアウトオブバンド VIM テナントが、複数の VIM に VNF を展開するために使用できる必要があります。ルートテナントが存在しない場合、ESC はマルチ VIM 展開中にテナントを作成できます。複数の ESC テナントを作成できます。ユーザは、複数の VIM に複数のテナントを使用できます。詳細については、テナントの管理を参照してください。

マルチ VIM 展開では、VM グループごとにターゲット VIM を指定できます。各 VM グループを異なる VIM に展開できますが、VM グループ内の VM は同じ VIM に展開されます。

マルチ VIM 展開を有効にするには、データモデルの VM グループにロケータ属性を追加する必要があります。ロケータノードは、次の属性で構成されます。

（注） ロケータ属性が展開に存在する場合、VM はロケータで指定された VIM に展開されます。ロケータ属性が展開に存在しない場合、VM はデフォルトの VIM に展開されます。デフォルトの VIM も存在しない場合、要求は拒否されます。

- vim_id：ターゲット VIM の VIM ID。ESC は vim_id を定義し、vim_connector ID にマッピングします。VIM コネクタは、vim_id で指定された VIM に展開する前に存在している必要があります。
- vim_project：ターゲット VIM で作成されたテナント名。これは、OpenStack に存在するアウトオブバンドテナントまたはプロジェクトです。

（注） ESC は、マルチ VIM 展開でポート、イメージ、フレーバ、ボリュームなどのアウトオブバンドリソース（既存のリソース）のみをサポートします。アウトオブバンドポートは、展開と同じテナントによって作成する必要があります。

ただし、マルチ VIM 展開では、デフォルト以外の VIM でロケータ属性を使用してエフェメラルボリュームのみを作成できます。その他のリソースは、デフォルト以外の VIM では作成できません。

VM のリカバリ、VM のスケールインとスケールアウトは、VM が展開されている同じ VIM 内でサポートされます。異なる VIM で VM を拡張またはリカバリすることはできません。

次の例では、esc-tenant は ESC 内のテナントです。VIM テナントへのマッピングはなく、VM はこの esc-tenant に展開されません。アウトオブバンドで作成される vim_project、project-test-tenant（ロケータ属性内）は、VM が展開されているテナントです。

```
<tenants>
   <tenant>
      <name>esc-tenant</name>
      <deployments>
         <deployment>
            <name>dep-1</name>
            <vm_group>
               <name>group-1</name>
               <locator>
                  <vim_id>vim-1</vim_id>
                  <vim_project>project-test-tenant</vim_project>
               </locator>
            </vm_group>
         </deployment>
      </deployments>
   </tenant>
</tenants>
```

ロケータ属性を使用して、単一の VIM に VNF を導入することもできます。つまり、ロケータ属性を持つデータモデルは、単一の OpenStack VIM で VM を導入するためにも使用できます。ロケータ属性（ESC リリース 2.x データモデル）なしで展開するには、単一の OpenStack VIM での VNF の展開 （2 ページ）を参照してください。

展開データモデルは次のとおりです。

```
<?xml version="1.0" encoding="UTF-8"?>
<esc_datamodel xmlns="http://www.cisco.com/esc/esc"
xmlns:ns0="http://www.cisco.com/esc/esc"
xmlns:ns1="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:ns2="urn:ietf:params:xml:ns:netconf:notification:1.0"
xmlns:ns3="http://www.cisco.com/esc/esc_notifications">
   <tenants>
      <tenant>
         <name>test-esc-tenant1</name>
         <deployments>
            <deployment>
               <name>dep-1</name>
               <vm_group>
                  <name>g1</name>
                  <locator>
                     <vim_id>vim1</vim_id>
```

```
                    <vim_project>project-test</vim_project>
                 </locator>
                 <bootup_time>150</bootup_time>
                 <recovery_wait_time>30</recovery_wait_time>
                 <flavor>Automation-Cirros-Flavor</flavor>
                 <image>Automation-Cirros-Image</image>
                 <interfaces>
                    <interface>
                       <nicid>0</nicid>
                       <network>my-network</network>
                    </interface>
                 </interfaces>
                 <scaling>
                    <min_active>1</min_active>
                    <max_active>1</max_active>
                    <elastic>true</elastic>
                 </scaling>
                 <kpi_data>
                    <kpi>
                       <event_name>VM_ALIVE</event_name>
                       <metric_value>1</metric_value>
                       <metric_cond>GT</metric_cond>
                       <metric_type>UINT32</metric_type>
                       <metric_collector>
                          <type>ICMPPing</type>
                          <nicid>0</nicid>
                          <poll_frequency>3</poll_frequency>
                          <polling_unit>seconds</polling_unit>
                          <continuous_alarm>false</continuous_alarm>
                       </metric_collector>
                    </kpi>
                 </kpi_data>
                 <rules>
                    <admin_rules>
                       <rule>
                          <event_name>VM_ALIVE</event_name>
                          <action>ALWAYS log</action>
                          <action>TRUE servicebooted.sh</action>
                          <action>FALSE recover autohealing</action>
                       </rule>
                    </admin_rules>
                 </rules>
                 <config_data />
              </vm_group>
           </deployment>
        </deployments>
     </tenant>
  </tenants>
</esc_datamodel>
```

アウトオブバンドリソースを使用し、展開の一部としてルートテナントを作成するサンプルのマルチ VIM 展開データモデル。

```
<esc_datamodel>
    <tenants>
        <tenant>
            <!-- This root level tenant is an ESC tenant either previously created or
created here marked by vim_mapping atrribute. -->
            <name>esc-tenant-A</name>
            <vim_mapping>false</vim_mapping>
            <deployments>
                <deployment>
                    <name>dep-1</name>
                    <vm_group>
```

```
                        <name>Grp-1</name>
                        <locator>
                            <vim_id>SiteA</vim_id>
                            <!-- vim_project: OOB project/tenant that should already
exist in the target VIM -->
                            <vim_project>Project-X</vim_project>
                        </locator>
                        <!-- All other details in vm group remain the same. -->
                        <flavor>Flavor-1</flavor>
                        <image>Image-1</image>
            ...
            ...
          </vm_group>
                </deployment>
            </deployments>
        </tenant>
    </tenants>
</esc_datamodel>
```

ESC が要求を受け入れるには、マルチ VIM 展開で指定されたすべての VIM が設定され、CONNECTION_SUCCESSFUL ステータスである必要があります。展開で指定された VIM が到達不能またはその他のステータスである場合、要求は拒否されます。

マルチ VIM 展開の VM にはアフィニティルールとアンチアフィニティルールを適用できます。詳細については、OpenStack のアフィニティおよびアンチアフィニティルールを参照してください。

マルチ VIM 展開は、ライフサイクルステージ（LCS）を使用したリカバリをサポートします。サポートされている LCS の詳細については、リカバリポリシー（ポリシーフレームワークを使用）を参照してください。既存のマルチ VIM 展開を更新できます。ただし、VM グループ内のロケータ属性は更新できません。既存の展開の更新に関する詳細については、既存の展開の更新を参照してください。