



インターフェイスの設定

- [インターフェイス設定 \(1 ページ\)](#)
- [ハードウェア アクセラレーションのサポート \(OpenStack のみ\) \(16 ページ\)](#)

インターフェイス設定

インターフェイス設定を使用すると、ネットワーク、サブネット、IP アドレス、MAC アドレス、VIM インターフェイス名、モデルなど、インターフェイスのさまざまな設定を選択できます。

この項では、Elastic Services Controller (ESC) の基本的インターフェイス設定および高度なインターフェイス設定と、これらを設定する手順について説明します。

基本的なインターフェイスの設定

ESC データモデルでは、インターフェイスは VM に接続されている VNIC を参照します。VM グループの下に1つ以上のインターフェイスを追加できます。interface セクションには、VNIC を設定するための詳細が表示されます。

この項では、Elastic Services Controller (ESC) の基本的なインターフェイス設定について説明します。

基本的なインターフェイス設定

この項では、次のような基本的なインターフェイス設定について説明します。

- ネットワーク
- サブネット
- IP アドレス
- MAC アドレス
- Elastic Services Controller (ESC) の場合は、VIM インターフェイス名など。

インターフェイス名の設定

VIM インターフェイス名を設定するには、展開 XML ファイル内のインターフェイスに属性 `<vim_interface_name>` を指定します。インターフェイス名を生成するときに特定の名前を使用するには、`<vim_interface_name>` を使用します。これらの属性が指定されていない場合、ESC はインターフェイス名を自動生成します。この名前は、`deployment_name`、`group_name`、およびランダムな UUID 文字列の組み合わせになります。例：

```
my-deployment-na_my-gro_0_8053d7gf-hyt33-4676-h9d4-9j4a5599472t.
```



(注) この機能は現在、OpenStack でのみサポートされています。

VM グループに伸縮性があり、`vim_interface_name` が指定されている場合は、2 番目のインターフェイス名以降のインターフェイス名の後に数値インデックスが追加されます（最初の名前は変更されません）。たとえば、指定したインターフェイス名が

```
<vim_interface_name>interface_1</vim_interface_name>
```

と設定されており、スケーリングが 3 に設定されている場合は、3 つの異なるインターフェイス名（`interface_1`、`interface_1_1`、および `interface_1_2`）で 3 つの VM が作成されます。VM グループの VM が 1 つのみの場合は、カスタムインターフェイス名に「`<index>`」は追加されません。単一の展開に複数の VM グループを含めることができます。また、必要に応じて、個々の VM グループで異なる

`vim_interface_name` 値を指定できます。たとえば、展開に 2 つの VM グループがある場合、最初のグループで `vim_interface_name` を指定すると、すべての VM に前述のようにその名前が生成されます。2 番目の VM グループでは `vim_interface_name` を指定しないため、このグループから作成されたすべての VM 名が自動生成されます。同じインターフェイス名は、必要に応じて、同じ VM グループ内の別の `interface` セクション、展開内の別の VM グループ、または異なる展開内で使用できます。

属性 `<vim_interface_name>` または `<port>` を同じインターフェイスに使用した場合、`vim_interface_name` 値は無視され、`port` 属性内の値が使用されます。

```
<esc_datamodel xmlns="https://www.cisco.com/esc/esc"> <tenants><tenant>
<name>Admin</name>
<deployments>
<deployment>
<deployment_name>NwDepModel_nosvc</deployment_name>
<interface>
<nicid>0</nicid>
<vim_interface_name>interface_1</vim_interface_name>
<network>my-network</network>
</interface>
```



(注) インターフェイス名には最大 61 文字を使用できます。特殊文字は使用できず、英数字と「`_`」および「`-`」のみを使用できます。次に、カスタムポート名を使用した出力例を示します。展開時に `vim_interface_name` を設定した場合は、同じ値が出力に表示されます。展開時にこの値を設定しなかった場合は、ESC がポート名を自動生成します。

- 次に、カスタムインターフェイス名を追加した後に `esc_nc_cli` スクリプトを使用して取得した出力の運用データの例を示します。 `vim_interface_name` という新しい要素がインターフェイス要素の下に表示されます。

```
[admin@esc-3-1-xxx]$ esc_nc_cli --user <username> --password <password> get
esc_datamodel/opdata
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="1">
. . .
  <interface>
    <nicid>0</nicid>
    <type>virtual</type>
    <port_id>e4111069-5d00-493b-8ea9-1a2ca134b5c8</port_id>
    <vim_interface_name>interface_1</vim_interface_name>      <!-- NEW IN OUTPUT
-->
    <network>c7fafeca-aa53-4349-9b60-1f4b92605420</network>
    <subnet>255.255.255.0</subnet>
    <ip_address>192.168.2.1</ip_address>
    <mac_address>fa:16:3e:d7:5e:da</mac_address>
    <netmask>255.255.240.0</netmask>
    <gateway>192.168.2.255</gateway>
  </interface>
```

- 次に、REST API を使用して取得した出力の運用データの例を示します。

```
GET http://localhost:8080/ESCManager/v0/deployments/example-deployment-123
| xmllint --format -
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<deployments>
. . .
  <interface>
    <network_uuid>c7fafeca-aa53-4349-9b60-1f4b92605420</network_uuid>
    <gateway>172.16.0.1</gateway>
    <ip_address>172.16.12.251</ip_address>
    <mac_address>fa:16:3e:30:0c:99</mac_address>
    <netmask>255.255.240.0</netmask>
    <nic_id>0</nic_id>
    <port_forwarding/>
    <port_uuid>1773cdbf-fe5f-4af1-adff-3a9c1dd1c47d</port_uuid>
    <vim_interface_name>interface_1</vim_interface_name>      <!-- NEW IN
OUTPUT -->
    <security_groups/>
    <subnet_uuid>7b2ce63b-eb20-4ff8-8d49-e46ee8dde0f5</subnet_uuid>
    <type>virtual</type>
  </interface>
```

上記のすべてのシナリオでは、`vim_interface_name` が `deployment.xml` に指定されていない場合でもこの要素は出力に含められますが、インターフェイス名は内部生成されます。

例：

```
<vim_interface_name>vm-name-deployme_Grp1_1_0f24cd7e-cae7-402e-819a-5c84087103ba</vim_interface_name>
```

MAC アドレスの割り当て

VMware vCenter での ESC の展開では、MAC アドレスプールから MAC アドレスの範囲または MAC アドレスリストを使用して MAC アドレスを割り当て、VM をネットワークに展開できます。

MAC アドレスは次の方法で割り当てることができます。

インターフェイスの使用

```
<interfaces>
  <interface>
    <nicid>1</nicid>
    <network>MANAGEMENT_NETWORK</network>
    <ip_address>172.16.0.11</ip_address>
    <mac_address>fa:16:3e:73:19:a0</mac_address>
  </interface>
</interfaces>
```

スケーリング時に、MAC アドレスリストまたはMAC アドレスの範囲をMAC アドレスプールから割り当てることができます。

```
<scaling>
  <min_active>2</min_active>
  <max_active>2</max_active>
  <elastic>true</elastic>
  <static_ip_address_pool>
    <network>MANAGEMENT_NETWORK</network>
    <ip_address>172.16.0.11</ip_address>
    <ip_address>172.16.0.12</ip_address>
    <ip_address>172.16.0.13</ip_address>
  </static_ip_address_pool>
  <static_mac_address_pool>
    <network>MANAGEMENT_NETWORK</network>
    <mac_address>fa:16:3e:73:19:a0</mac_address>
    <mac_address>fa:16:3e:73:19:a1</mac_address>
    <mac_address>fa:16:3e:73:19:a2</mac_address>
  </static_mac_address_pool>
</scaling>
```

MAC アドレスの範囲を使用してMAC アドレスを割り当てます。

```
<scaling>
  <min_active>2</min_active>
  <max_active>2</max_active>
  <elastic>true</elastic>
  <static_ip_address_pool>
    <network>MANAGEMENT_NETWORK</network>
    <ip_address_range>
      <start>172.16.0.25</start>
      <end>172.16.0.27</end>
    </ip_address_range>
  </static_ip_address_pool>
  <static_mac_address_pool>
    <network>MANAGEMENT_NETWORK</network>
    <mac_address_range>
      <start>fa:16:3e:73:19:b0</start>
      <end>fa:16:3e:73:19:b2</end>
    </mac_address_range>
  </static_mac_address_pool>
</scaling>
```



- (注) 既存の展開内や、サービスアップデート内のVMインスタンスのスケーリング時（最小値および最大値が1よりも大きい場合）は、MAC または IP プールを変更できません。

VMware vCenter では、MAC アドレスの割り当て中に、「Generated」または「Assigned」の指定値が正しい範囲内でないか、または重複していると判断された場合、サーバがその値をオーバーライドすることがあります。このため、ESC が MAC アドレスを割り当てることができない場合、その展開は失敗します。

インターフェイスのサブネットの設定

サブネットはデータモデルを介して渡すことができます。インターフェイス内のサブネットは、展開 XML ファイルの `interface` セクションで指定できます。データモデルにサブネットが指定されていない場合、ESC は OpenStack にインターフェイスを作成するためのサブネットを選択し、OpenStack によって作成されたポートのサブネットを使用します。

```
<interface>
  <nicid>0</nicid>
  <network>my-network</network>
  <subnet>my-subnet</subnet>
</interface>
```

`no_gateway` 属性を使用すると、ESC はゲートウェイを無効にした状態でサブネットを作成できます。次に、`no_gateway` 属性を `true` に設定して、ゲートウェイなしでサブネットを作成する例を示します。

```
<networks>
<network>
<name>mgmt-net</name>
<subnet>
<name>mgmt-net-subnet</name>
<ipversion>ipv4</ipversion>
<dhcp>false</dhcp>
<address>172.16.0.0</address>
<no_gateway>true</no_gateway><!-- DISABLE GATEWAY -->
<gateway>172.16.0.1</gateway>
<netmask>255.255.255.0</netmask>
</subnet>
</network>
</networks>
```

アウトオブバンドポートの設定

また、ESC を使用すると、アウトオブバンドポートを VNF に接続することもできます。これを行うには、サービス要求を開始している間に展開要求ファイルで UUID またはポートの名前を渡します。詳細については、『[Cisco Elastic Services Controller User Guide](#)』の「Out-of-band Volumes」の項を参照してください。



- (注) VNF を展開解除または復元している間は、その VNF に接続されているポートは切り離されるだけで、削除されません。ESC は、VM グループのアウトオブバンドポートを使用している間はスケーリングを許可しません。VM グループには、VM のインスタンスを 1 つだけ設定できます。アウトオブバンドポートが使用されている間は、展開の更新時に VM グループのスケールリング値を更新できません。

```
<esc_datamodel xmlns="https://www.cisco.com/esc/esc">
  <name>tenant</name>
  <deployments>
    <deployment>
      <name>depz</name>
      <vm_group>
        <name>g1</name>
        <image>Automation-Cirros-Image</image>
        <flavor>Automation-Cirros-Flavor</flavor>
        <bootup_time>100</bootup_time>
        <reboot_time>30</reboot_time>
        <recovery_wait_time>10</recovery_wait_time>
        <interfaces>
          <interface>
            <nicid>0</nicid>
            <port>057a1c22-722e-44da-845b-a193e02807f7</port>
            <network>my-network</network>
          </interface>
        </interfaces>
      </vm_group>
    </deployment>
  </deployments>
</esc_datamodel>
```

デュアルスタックのサポート

デュアルスタック ネットワークを使用すると、複数の IP アドレスを割り当てることができます。これらの複数の IP アドレスは、ESC を使用して、VNF 展開内の所定のインターフェイスへの異なるサブネットに割り当てることができます。

ESC では、デュアルスタックの次の機能がサポートされています。

- ネットワークとサブネットのリストの設定
- ネットワークとサブネットおよび IP アドレスのリストの設定
- ネットワークとアドレスのリストの設定（サブネットなし）
- ネットワークとサブネットおよび IP のリストの指定（同じサブネットで IP が異なる）



- (注) 現在、ESC は OpenStack でのみデュアルスタックをサポートしています。ESC は、OpenStack 展開のためのエンドツーエンド IPv6 をサポートしています。

新しいコンテナ要素の名前付きアドレスがインターフェイスに追加されます。このコンテナには、アドレス要素のリストが含まれています。アドレス要素には address_id (キー) が必要で

す。サブネットおよび固定 IP アドレスのフィールドはオプションですが、いずれか1つを指定する必要があります。

コンテナアドレスは次のとおりです。

```
container addresses {
  list address {
    key "address_id";
    leaf address_id {
      description "Id for the address in address list.";
      type uint16;
      mandatory true;
    }
    leaf subnet {
      description "Subnet name or uuid for allocating IP to this port";
      type types:escnetname;
    }
  }
  leaf ip_address {
    description "Static IP address for this specific subnet";
    type types:escipaddr;
    must "../...../...../scaling/max_active = 1 or
count(...../...../scaling/static_ip_address_pool) > 0"
    {
      error-message "Static ip address pools must be configured when static ip addresses are
configured.";
    }
  }
}
```

デュアルスタックでは、KPIのモニタリングがサポートされるようになりました。新しい要素 `address_id` が `metric_collector` 要素に追加されました。これは、KPIのモニタリングに使用される、指定された NICID 内のアドレスをポイントする値を受け入れます。つまり、インターフェイスの下に定義されているアドレスの1つを KPI のモニタリングに使用できます。

```
...
<interface>
  <nicid>1</nicid>
  <network>demo-net</network>
  <addresses>
    <address>
      <address_id>0</address_id>
      <subnet>demo-subnet</subnet>
    </address>
  </addresses>
</interface>

<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_occurrences_true>5</metric_occurrences_true>
    <metric_occurrences_false>5</metric_occurrences_false>
    <metric_collector>
      <type>ICMPPing</type>
      <nicid>1</nicid>
      <address_id>0</address_id>
      <poll_frequency>10</poll_frequency>
      <polling_unit>seconds</polling_unit>
      <continuous_alarm>>false</continuous_alarm>
    </metric_collector>
  </kpi>
```

```
</kpi_data>
```

...



(注) `metric_collector` 要素の下にある `address_id` は、インターフェイスの下にある `address_id` の 1 つと同じである必要があります。

デュアルスタックインターフェイスは、`day-0` 変数の置換で使えるようになりました。つまり、1 つのインターフェイスの下で定義されている複数のアドレスから値を置き換えることができます。`Day 0` 設定は、`config_data` タグの下にあるデータモデルで定義されます。

複数の IP アドレスを持つデュアルスタックの場合、変数は `NICID_<n>_<a>_<PROPERTY>` 形式になります。それぞれの意味は次のとおりです。

- `<n>` は、インターフェイスの NICID です。
- `<a>` は、そのインターフェイス内のアドレスの `address_id` です。

次に、デュアルスタックからの使用可能な `day-0` 置換変数のリストを示します。

| | | |
|---|---|---------------|
| <code>NICID_n_a_IP_ALLOCATION_TYPE</code> | FIXED DHCP を含む文字列 | ipv4 または ipv6 |
| <code>NICID_n_a_IP_ADDRESS</code> | IP アドレス | ipv4 または ipv6 |
| <code>NICID_n_a_GATEWAY</code> | ゲートウェイ アドレス | ipv4 または ipv6 |
| <code>NICID_n_a_CIDR_ADDRESS</code> | CIDR プレフィックスアドレス | ipv4 または ipv6 |
| <code>NICID_n_a_CIDR_PREFIX</code> | CIDR プレフィックス長の整数 | ipv4 または ipv6 |
| <code>NICID_n_a_NETMASK</code> | IPv4 CIDR アドレスとプレフィックスが存在する場合、ESC は自動的にネットマスク変数を計算して入力します。これは、IPv6 アドレスの場合は置換されないため、使用しないでください。 | ipv4 のみ |

1 つの IP アドレスの `day-0` 設定の詳細については、『[Cisco Elastic Services Controller User Guide](#)』の「Day Zero Configuration」の章を参照してください。

次に、`day-0` 設定の `config_data` で定義されているテンプレートファイルを示します。

```
NICID_0_NETWORK_ID=${NICID_0_NETWORK_ID}
NICID_0_MAC_ADDRESS=${NICID_0_MAC_ADDRESS}

NICID_0_0_IP_ALLOCATION_TYPE=${NICID_0_0_IP_ALLOCATION_TYPE}
NICID_0_0_IP_ADDRESS=${NICID_0_0_IP_ADDRESS}
NICID_0_0_GATEWAY=${NICID_0_0_GATEWAY}
NICID_0_0_CIDR_ADDRESS=${NICID_0_0_CIDR_ADDRESS}
NICID_0_0_CIDR_PREFIX=${NICID_0_0_CIDR_PREFIX}
NICID_0_0_NETMASK=${NICID_0_0_NETMASK}
```



```
NICID_0_1_IP_ALLOCATION_TYPE=${NICID_0_1_IP_ALLOCATION_TYPE}
NICID_0_1_IP_ADDRESS=${NICID_0_1_IP_ADDRESS}
NICID_0_1_GATEWAY=${NICID_0_1_GATEWAY}
NICID_0_1_CIDR_ADDRESS=${NICID_0_1_CIDR_ADDRESS}
NICID_0_1_CIDR_PREFIX=${NICID_0_1_CIDR_PREFIX}
```

次に、データモデルを示します。

```
<?xml version="1.0" encoding="ASCII"?>
<esc_datamodel xmlns="https://www.cisco.com/esc/esc">
  <tenants>
    <tenant>
      <name>dep-tenant</name>
      <deployments>
        <deployment>
          <name>cirros-dep</name>
          <vm_group>
            <name>Grp1</name>
            <bootup_time>600</bootup_time>
            <recovery_wait_time>30</recovery_wait_time>
            <flavor>Automation-Cirros-Flavor</flavor>
            <image>Automation-Cirros-Image</image>
          </vm_group>
          <interfaces>
            <interface>
              <!-- No dual stack support on mgmt interface in ESC 4.1 -->
              <nicid>0</nicid>
              <network>my-network</network>
            </interface>
            <interface>
              <nicid>1</nicid>
              <network>ent-network1</network>
              <addresses>
                <address>
                  <!-- IPv4 Dynamic -->
                  <address_id>0</address_id>
                  <subnet>v4-subnet_A</subnet>
                </address>
                <address>
                  <!-- IPv6 Dynamic -->
                  <address_id>1</address_id>
                  <subnet>v6-subnet_B</subnet>
                </address>
              </addresses>
            </interface>
            <interface>
              <nicid>2</nicid>
              <network>ent-network2</network>
              <addresses>
                <address>
                  <!-- IPv4 Static -->
                  <address_id>0</address_id>
                  <subnet>v4-subnet_C</subnet>
                  <ip_address>172.16.87.8</ip_address>
                </address>
                <address>
                  <!-- IPv6 Static -->
                  <address_id>1</address_id>
                  <subnet>v6-subnet_D</subnet>
                  <ip_address>fd07::110</ip_address>
                </address>
              </addresses>
            </interface>
            <interface>
              <nicid>3</nicid>
```

```

<network>ent-network3</network>
<addresses>
  <address>
    <!-- Only ip config - ipv6 but no subnet -->
    <address_id>0</address_id>
    <ip_address>fd07::110</ip_address>
  </address>
  <address>
    <!-- Only ip config - ipv4 but no subnet -->
    <address_id>1</address_id>
    <ip_address>172.16.88.9</ip_address>
  </address>
</addresses>
</interface>
<interface>
  <nicid>4</nicid>
  <network>ent-network4</network>
  <addresses>
    <address>
      <!-- ipv4 same subnet as address_id 6 -->
      <address_id>0</address_id> //
      <subnet>v4-subnet_F</subnet>
      <ip_address>172.16.86.10</ip_address>
    </address>
    <address>
      <!-- ipv4 same subnet as id 5 -->
      <address_id>1</address_id>
      <subnet>v4-subnet_F</subnet>
      <ip_address>172.16.86.11</ip_address>
    </address>
  </addresses>
</interface>
</interfaces>
<kpi_data>

```

...

複数の IP を使用して正常に展開された後、ESC はアドレスのリストを通知または `opdata` として提供します。

次を含む親 `<interface>` 要素の下にある複数の `<address>` 要素のリスト：

- **address_id** : 入力 XML で指定されたアドレス ID
- **サブネット要素** : サブネット名または UUID
- **ip_address 要素** : そのサブネット上のポートに割り当てられている IP
- **プレフィックス** : サブネット CIDR プレフィックス
- **ゲートウェイ** : サブネット ゲートウェイ アドレス
- ESC 静的 IP サポート

通知:

```

<vm_id>1834124d-b70b-41b9-9e53-fb55d7c901f0</vm_id>
<name>jenkins-gr_g1_0_e8bc9a81-4b9a-437a-807a-f1a9bbc2ea3e</name>
<generated_name>custom_vim_name

<host_id>dc380f1721255e2a7ea15932c1a7abc681816642f75276c166b4fe50</host_id>
<hostname>my-server</hostname>

```

```

<interfaces>
  <interface>
    <nicid>0</nicid>
    <type>virtual</type>
    <vim_interface_name>custom_vim_name
    <port_id>4d57d4a5-3150-455a-ad39-c32fffb10b1</port_id>
    <mac_address>fa:16:3e:d2:50:a5</mac_address>
    <network>45638651-2e92-45fb-96ce-9efdd9ea343e</network>
    <address>
      <address_id>0<address_id>
      <subnet>6ac36430-4f58-454b-9dc1-82f7a796e2ff</subnet>
      <ip_address>172.16.0.22</ip_address>
      <prefix>24</prefix>
      <gateway>172.16.0.1</gateway>
    </address>
    <address>
      <address_id>1<address_id>
      <subnet>8dd9f501-19d4-4782-8335-9aa9fbd4dab9</subnet>
      <ip_address>2002:dc7::4</ip_address>
      <prefix>48</prefix>
      <gateway>2002:dc7::1</gateway>
    </address>
    <address>
      <address_id>2<address_id>
      <subnet>a234501-19d4-4782-8335-9aa9fbd4caf6</subnet>
      <ip_address>172.16.87.8</ip_address>
      <prefix>20</prefix>
      <gateway>172.16.87.1</gateway>
    </address>
  </interface>
</interfaces>

```

opdata の例 :

```

<interfaces>
  <interface>
    <nicid>0</nicid>
    <type>virtual</type>
    <vim_interface_name>custom_vim_name
    <port_id>4d57d4a5-3150-455a-ad39-c32fffb10b1</port_id>
    <mac_address>fa:16:3e:d2:50:a5</mac_address>
    <network>45638651-2e92-45fb-96ce-9efdd9ea343e</network>
    <address>
      <address_id>0</address_id>
      <subnet>6ac36430-4f58-454b-9dc1-82f7a796e2ff</subnet>
      <ip_address>172.16.0.22</ip_address>
      <prefix>24</prefix>
      <gateway>172.16.0.1</gateway>
    </address>
    <address>
      <address_id>1</address_id>
      <subnet>8dd9f501-19d4-4782-8335-9aa9fbd4dab9</subnet>
      <ip_address>2002:dc7::4</ip_address>
      <prefix>48</prefix>
      <gateway>2002:dc7::1</gateway>
    </address>
  </interface>
</interfaces>

```

また、day-0の代入値が出力データで置き換えられていることも確認できます。次に、day-0設定に値が入力された出力データの例を示します。

```

NICID_0_NETWORK_ID=45638651-2e92-45fb-96ce-9efdd9ea343e
NICID_0_MAC_ADDRESS=fa:16:3e:d2:50:a5

```

```

NICID_0_0_IP_ALLOCATION_TYPE=DHCP
NICID_0_0_IP_ADDRESS=172.16.0.22
NICID_0_0_GATEWAY=172.16.0.1
NICID_0_0_CIDR_ADDRESS=172.16.0.0
NICID_0_0_CIDR_PREFIX=24
NICID_0_0_NETMASK=255.255.255.0

NICID_0_1_IP_ALLOCATION_TYPE=DHCP
NICID_0_1_IP_ADDRESS=2002:dc7::4
NICID_0_1_GATEWAY=2002:dc7::1
NICID_0_1_CIDR_ADDRESS=2002:dc7::/48
NICID_0_1_CIDR_PREFIX=48

```

静的 IP をサポートするデュアルスタック

ESC は、静的 IP をサポートするデュアルスタックをサポートします。初期設定の一部として、ユーザは設定するサブネットと IP を指定できます。



- (注) ESC は、スケールリングが **false** または **minimum/maximum = 1** の場合にのみ、静的 IP をサポートします。

アウトオブバンドネットワークを使用して VM を作成し、静的 IP を持つサブネットのリスト（ネットワークに複数のサブネットがある）を指定すると、ESC はサブネットと対応する静的 IP の両方を適用します。

次に、2 つのサブネット（**ipv4** と **ipv6**）が 1 つのインターフェイスに追加された例を示します。

```

<?xml version="1.0" encoding="ASCII"?>
<esc_datamodel xmlns="https://www.cisco.com/esc/esc">
  <tenants>
    <tenant>
      <name>dep-tenant</name>
      <deployments>
        <deployment>
          <name>cirros-dep</name>
          <vm_group>
            <name>Grp1</name>
            <bootup_time>600</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>ent-network2</network>
                <addresses>
                  <address>
                    <!-- IPv4 Static -->
                    <address_id>0</address_id>
                    <subnet>v4-subnet_C</subnet>
                    <ip_address>172.16.87.8</ip_address>
                  </address>
                  <address>
                    <!-- IPv6 Static -->
                    <address_id>1</address_id>

```

```
<subnet>v6-subnet_D</subnet>
  <ip_address>fd07::110</ip_address>
</address>
</addresses>
</interface>
</interfaces>
<kpi_data>
```

VNF の展開については、「OpenStack での仮想ネットワーク機能の展開」を参照してください。

高度なインターフェイス設定

この項では、Elastic Services Controller (ESC) 用の複数のインターフェイス設定と、ハードウェアのインターフェイスを設定する手順について説明します。

基本的なインターフェイス設定の詳細については、「基本的なインターフェイス設定」を参照してください。

高度なインターフェイスの設定

ESC での SR-IOV の設定

Single Root I/O Virtualization (SR-IOV) により、さまざまなゲストオペレーティングシステムを実行している複数の VM が、ホストサーバ内の単一の PCIe ネットワークアダプタを共有できるようになります。また、VM がネットワークアダプタとの間で直接データを移動でき、ハイパーバイザをバイパスすることで、ネットワークのスループットが増加しサーバの CPU 負荷が低下します。

OpenStack 用の ESC での SR-IOV の設定

OpenStack に ESC で SR-IOV を設定する前に、正しいパラメータを使用してハードウェアと OpenStack を設定します。

OpenStack に ESC で SR-IOV を有効にするには、インターフェイスの `type` を `direct` として指定します。次のスニペットは、データモデルの例を示しています。

```
<interfaces>
  <interface>
    <nicid>0</nicid>
    <network>my-network</network>
    <type>direct</type>
  </interface>
</interfaces>
...
```

VMware 用の ESC での SR-IOV の設定

VMware に ESC で SR-IOV を設定する前に、次の点を考慮してください。

- 目的の ESXi ホスト上で SR-IOV 物理機能を有効にします。詳細については、VMware のマニュアルを参照してください。
- SR-IOV を有効にする前に、次の重要な点を考慮してください。

- SR-IOV が VMware でサポートされている物理ネットワークアダプタのリストを確認します。VMware のマニュアルを参照してください。
- SR-IOV が設定されている VM でサポートされていない VM 機能のリストを確認します。VMware のマニュアルを参照してください。
- SR-IOV を使用したクラスタ展開（データモデルの「zone」で定義）では、各 ESXi ホストに同じ物理機能があり、SR-IOV の選択ができるようになっていることを確認します。たとえば、VM が物理機能として vmnic7 を使用する場合は、各ホストに vmnic7 があり、各 vmnic7 の SR-IOV ステータスが有効になっていることを確認します。

VMware に対して ESC で SR-IOV を有効にするには、展開データモデルでインターフェイスの <type> を direct とし、拡張子の <name> を sriov_pf_selection として指定します。インターフェイスタイプの direct は、SR-IOV デバイスを示し、拡張子名の sriov_pf_selection は物理機能を示します。次のスニペットは、データモデルの例を示しています。

```
<vm_group>
...
<interface>
  <nicid>2</nicid>
  <network>MgtNetwork</network>
  <type>direct</type>
</interface>
<interface>
  <nicid>3</nicid>
  <network>MgtNetwork</network>
  <type>direct</type>
</interface>
...
<extensions>
  <extension>
    <name>sriov_pf_selection</name>
    <properties>
      <property>
        <name>nicid-2</name>
        <value>vmnic1,vmnic2</value>
      </property>
      <property>
        <name>nicid-3</name>
        <value>vmnic3,vmnic4</value>
      </property>
    </properties>
  </extension>
</extensions>
</vm_group>
```

許可済みアドレスペアの設定

Cisco Elastic Services Controller を使用すると、ネットワークに関連付けられているサブネットに関係なく、指定されたポートを通過するように展開データモデル内にアドレスペアを指定できます。

アドレスペアは、次の方法で設定されます。

- ネットワークのリスト：特定のインターフェイスにネットワークのリストを指定すると、ESCはこれらのネットワークのOpenStackからサブネットの詳細を取得し、対応するポートまたはインターフェイスに追加します。次に、ネットワークのリストとしてアドレスペアを設定する例を示します。

```
<interface>
  <nicid>1</nicid>
  <network>network1</network>
  <allowed_address_pairs>
    <network>
      <name>bb8c5cfb-921c-46ea-a95d-59feda61cac1</name>
    </network>
    <network>
      <name>6ae017d0-50c3-4225-be10-30e4e5c5e8e3</name>
    </network>
  </allowed_address_pairs>
</interface>
</interfaces>
```

- アドレスのリスト：アドレスのリストを指定した場合、ESCはこれらのアドレスを対応するインターフェイスに追加します。次の例では、アドレスのリストとしてアドレスペアを設定する方法について説明します。

```
<interface>
  <nicid>0</nicid>
  <network>esc-net</network>
  <allowed_address_pairs>
    <address>
      <ip_address>10.10.10.10</ip_address>
      <netmask>255.255.255.0</netmask>
    </address>
    <address>
      <ip_address>10.10.20.10</ip_address>
      <netmask>255.255.255.0</netmask>
    </address>
  </allowed_address_pairs>
</interface>
```

セキュリティグループのルールの設定

Cisco Elastic Services Controller (ESC) を使用すると、セキュリティグループのルールをOpenStackで展開されているインスタンスに関連付けることができます。これらのセキュリティグループのルールは、展開データモデルに必要なパラメータを指定することによって設定されます。セキュリティグループのルールの設定に加えて、いずれかのVNFインスタンスで障害が発生した場合、ESCはインスタンスを復旧し、再展開されているVNFのセキュリティグループのルールを適用します。

セキュリティグループのルールを設定するには、次の手順を実行します。

始める前に

- ESCを使用してテナントを作成したことを確認します。
- セキュリティグループが作成されていることを確認します。
- セキュリティグループの名前またはUUIDがあることを確認します。

ステップ 1 ルートユーザとして ESC VM にログインします。

ステップ 2 次のコマンドを実行して特定のセキュリティグループの UUID を確認します。

```
nova --os-tenant-name <NameOfTheTenant> secgroup-list
```

ステップ 3 展開データモデルでは、次の引数を渡します。

```
<interfaces>
  <interface>
    <nicid>0</nicid>
    <network>my-network</network><!-- depends on network name -->
    <security_groups>
      <security_group>0c703474-2692-4e84-94b9-c29e439848b8</security_group>
      <security_group>bbcd62bc62-a0de-4475-b258-740bfd33861b</security_group>
    </security_groups>
  </interface>
  <interface>
    <nicid>1</nicid>
    <network>sample_VmGrpNet</network><!--depends on network name -->
    <security_groups>
      <security_group>sample_test_SQL</security_group>
    </security_groups>
  </interface>
```

ステップ 4 次のコマンドを実行して、セキュリティグループが VM インスタンスに関連付けられているかどうかを確認します。

```
nova --os-tenant-name <NameOfTenant> show <NameOfVMinstance>
```

ハードウェア アクセラレーションのサポート (OpenStack のみ)

flavor データモデルを使用して OpenStack のハードウェア アクセラレーション機能を設定できます。次のハードウェア アクセラレーション機能を設定できます。

- **vCPU ピニング** : vCPU (仮想中央処理装置) またはある範囲の CPU へのバインディングおよびバインディング解除を可能にし、プロセスが任意の CPU ではなく、指定した CPU 上でのみ実行されるようにします。
- **大規模なページおよび不均等なメモリアクセス (NUMA) の OpenStack パフォーマンスの最適化** : 大規模なページや NUMA のシステムパフォーマンス、つまり、高い負荷を受け入れ、高い負荷を処理するようにシステムを変更するようにシステムの能力を向上させることができます。
- **PCIe パススルーインターフェイスに対する OpenStack サポート** : OpenStack 上のインスタンスへの PCI デバイスの割り当てを可能にします。

次に、フレーバデータモデルを使用してハードウェアアクセラレーション機能を設定する方法について説明します。

```
$ cat example.xml
<?xml version='1.0' encoding='ASCII'?>
<esc_datamodel xmlns="http://www.cisco.com/esc/esc">
  <flavors>
    <flavor>
      <name>testfl16</name>
      <vcpus>1</vcpus>
      <memory_mb>2048</memory_mb>
      <root_disk_mb>10240</root_disk_mb>
      <ephemeral_disk_mb>0</ephemeral_disk_mb>
      <swap_disk_mb>0</swap_disk_mb>
      <properties>
        <property>
          <name>pci_passthrough:alias</name>
          <value>niclg:1</value>
        </property>
      </properties>
    </flavor>
  </flavors>
</esc_datamodel>
$ /opt/cisco/esc/esc-confd/esc-cli/esc_nc_cli --user <username> --password <password>
edit-config ./example.xml
```

VMware vSphere NUMA 属性の追加パラメータの作成

ESC は、追加の設定パラメータを追加することによって VMware vSphere の NUMA を拡張します。

この機能拡張によって、day-0 コンフィギュレーションファイルを介してこれらの値を渡すのではなく、設定パラメータを渡すためのプレフィックスとして、VMware vSphere の追加設定または高度な設定が追加されます。

プレフィックス : extConfigParam

例 :

```
<configuration>
  <dst>extConfigParam:mgmt-ipv4-addr</dst>
  <data>$NICID_1_IP_ADDRESS/16</data>
</configuration>
```

追加設定は、データモデルの変更を最小限に抑え、設定変更を VIM レイヤに制限するのに便利です。

VMware vCenter での PCI または PCIe デバイスのパススルーの設定

ESC は VMware vCenter PCI または PCIe デバイスパススルー (VMDirectPath I/O) をサポートします。これにより、I/O メモリ管理ユニットが搭載されたプラットフォーム上の物理 PCI 機能への VM アクセスが可能になります。

はじめる前に

ホスト VM の PCI/PCIe デバイスでパススルーを有効にするには、vSphere 管理者が vCenter でこれらのデバイスをマークする必要があります。



- (注) PCI 設定後にホストをリブートする必要があります。ホストをメンテナンスモードにし、電源をオフにするか、またはすべての VM を他のホストに移行します。

ESC 展開で PCI デバイスパススルー要求を指定するには、値を *passthrough* に設定して <type> 属性を含めます。特定の *vm_group* またはネットワークに対して選択する PCI デバイスを指定するには、*pci_id* を含めます。次に、データモデルを示します。

```
<tenants>
  <tenant>
    <name>admin</name>
    <deployments>
      <deployment>
        <name>test</name>

        <vm_group>
          <name>test-g1</name>
          <image>uLinux</image>
          <bootup_time>300</bootup_time>
          <recovery_wait_time>10</recovery_wait_time>
          <interfaces>
            <interface>
              <nicid>1</nicid>
              <network>MgtNetwork</network>
              <ip_address>192.168.0.102</ip_address>
            </interface>
            <interface>
              <nicid>2</nicid>
              <network>VM Network</network>
              <type>passthru</type>
              <ip_address>172.16.0.0</ip_address>
            </interface>
            <interface>
              <nicid>3</nicid>
              <network>VM Network</network>
              <type>passthru</type>
              <ip_address>192.168.46.117</ip_address>
            </interface>
            <interface>
              <nicid>3</nicid>
              <type>passthru</type>
              <network>MgtNetwork</network>
              <pci_id>0000:07:10.3</pci_id>
            </interface>
          </interfaces>
        </vm_group>
      </deployment>
    </deployments>
  </tenant>
</tenants>
```

展開が正常に完了すると、*passthru* 値が通知の *interface* セクションと運用データ内に設定されます。

PCI または PCIe PassThrough デバイスの自動選択

ESC では、特定の PCI ID を使用せずに、各展開に 1 つ以上の PCI または PCIe パススルーデバイスを接続する必要があります。ESC は最初にホストを選択します。ESC は、次に使用可能な PCI または PCIe パススルー対応デバイスを選択し、展開時に接続します。使用可能な PCI ま

たは PCIe パススルー対応デバイスがない場合、ESC は展開に失敗します。vSphere 管理者は、ターゲット コンピューティング クラスタ内のすべてのコンピューティングホストに、十分な数の PCI または PCIe パススルー対応デバイスがあることを確認する必要があります。



(注)

- PCI または PCIe パススルーは、ESC 配置アルゴリズムでは考慮されません。たとえば、ESC は PCI または PCIe パススルー要求を完了するために使用可能なリソースがあるため、ホストを選択しません。
- ESC は PCI または PCIe パススルーデバイスをランダムに選択します。ESC では、デバイスのタイプまたは仕様を考慮しません。リストから次に使用可能な PCI または PCIe デバイスを選択します。
- ESC が ESC 配置アルゴリズムに基づいて選択したコンピューティングホストに対して VNF が回復される場合に、そのコンピューティングホストに使用可能な PCI または PCIe パススルー対応デバイスがないと、リカバリは失敗します。
- パススルーが機能するには、DRS をオフにする必要があります。

