OpenStack への Cisco Elastic Services Controller のインストール

この章では、OpenStack に Cisco Elastic Services Controller をインストールする手順について説明します。この章は次の項で構成されています。

インストールのシナリオ

以下の項では、ESC に関する一般的な展開シナリオについて簡単に説明します。

Cisco Elastic Services Controller は、要件に応じてさまざまなモードでインストールできます。各種モードはインストール時に設定します。以下の項では、ESC に関する一般的な展開シナリオについて簡単に説明します。

スタンドアロン型 ESC

スタンドアロンシナリオでは、1 つのアクティブな VM が ESC に展開されます。

高可用性を備えた ESC

ESC は、アクティブ/スタンバイおよびアクティブ/アクティブモデルの形式で高可用性（HA）をサポートします。アクティブ/スタンバイモデルでは、ESC 障害を防止し、サービスの中断を最小限に抑えて ESC サービスを提供するために、ネットワークに 2 つの ESC インスタンスが展開されます。アクティブ ESC インスタンスで障害が発生しても、スタンバイインスタンスが自動的に ESC サービスを引き継ぎます。ESC HA は、ESC で発生する次のシングルポイント障害を解決します。

- ネットワーク障害

- 停電
- VM インスタンスのダウン
- スケジュールされたダウンタイム
- ハードウェアに関する問題
- 内部アプリケーションの障害

（注） ESC VM でのソフトウェアのインストールまたはアップグレードはサポートされていません。さらにサポートが必要な場合は、Cisco TAC にお問い合わせください。

ESC HA アクティブ/スタンバイのインストールの詳細については、QCOW イメージを使用した Cisco Elastic Services Controller のインストール （2 ページ）および『Cisco Elastic Services Controller の VMware vCenter へのインストール』を参照してください。

アクティブ/アクティブモデルの詳細については、「ESC HA アクティブ/アクティブの概要」の章を参照してください。

Cisco Elastic Services Controller セットアップの主要コンポーネント

Cisco Elastic Service Controller（ESC）のセットアップは、次のコンポーネントから構成されます。

- **仮想インフラストラクチャ マネージャ**：Elastic Services Controller（ESC）およびその VNF は、仮想インフラストラクチャ マネージャ（VIM）に展開されます。これには、1 つまたは複数の基盤となる物理ノードが定義されています。
- **ESC 仮想マシン**：ESC VM は、サービスの登録と展開に使用されるすべてのサービスとプロセスを搭載した VM です。これには、ESC マネージャと他のすべてのサービスが含まれます。ESC と通信するためのノースバウンドインターフェイスとして Netconf API、REST API、およびポータルが提供されます。ESC VM には ESC VM と対話するための CLI が実装されています。CLI は 2 つあります。1 つは REST API を使用し、もう 1 つは Netconf API を使用します。

QCOW イメージを使用した Cisco Elastic Services Controller のインストール

QCOW イメージを使用して、OpenStack に Cisco Elastic Services Controller（ESC）をインストールできます。ESC は、実行中の VM インスタンスとして OpenStack に展開され、VNF を管理

します。したがって、ESC では、OpenStack に OpenStack 環境パラメータをインストールする必要がありますホストおよびストレージエリア ネットワークの負荷にもよりますが、10 ～ 20 分のインストール時間がかかります。この手順では、OpenStack で ESC 仮想マシン（VM）を作成する方法について説明します。

始める前に

- 前提条件で指定されているシステム要件をすべて満たしている。
- インストールの準備に示されている情報が既知である。
- ESC イメージファイルを、ESC のインストール先のシステムにコピーします。
- このシステムには OpenStack からアクセスできる必要があります。

手順

ステップ 1 ESC をインストールするシステムにログインします。

ステップ 2 bootvm.py と ESC イメージの互換性を確認します。

```
./bootvm.py --version
```

ESC インストーラの引数の詳細については、「**付録 A**：**Cisco Elastic Service Controller のインストール引数**」を参照してください。

ステップ 3 テキストエディタで、PROJECT-openrc.sh ファイルという名前のファイルを作成し、次の認証情報を追加します。次の例は、admin という名前のプロジェクトの情報を示しています。ここで、OpenStack ユーザ名は admin で、ID ホストはコントローラノードにあります。

（注） OpenStack コマンドラインクライアントに必要な環境変数を設定するには、OpenStack rc ファイルまたは openrc.sh ファイルと呼ばれる環境ファイルを作成する必要があります。このプロジェクト固有の環境ファイルには、すべての OpenStack サービスで使用されるクレデンシャルが含まれています。ESC インストールスクリプトでは、OpenStack で認証とインストールを実行するために、これらの OpenStack 環境パラメータが必要です。すべての OpenStack クレデンシャルが独自の引数を使用して渡される場合、bootvm.py スクリプトはこれらのパラメータを必要としません。

```
export OS_NO_CACHE=true
export OS_TENANT_NAME=admin
export OS_USERNAME=admin
export OS_PASSWORD=admin_pass
export OS_AUTH_URL=http://controller node:35357/v2.0
```

インストールに必要なその他の OpenStack パラメータは次のとおりです：--os_auth_url、--os_username、--os_password、-os_tenant_name、--bs_os_user_domain_name、--bs_os_project_domain_name、--bs_os_identity_api_version、--bs_os_auth_url、--bs_os_username、--bs_os_password、--bs_os_tenant_name、--bs_os_user_domain_name。

OpenStack V2 API の場合は、--os_password、--os_auth_url、--os_username、--os_tenant_name を含むアイテムをグローバル環境変数で定義する必要があります。

OpenStack V3 API の場合は、--os_identity_api_version=3 を設定します。OpenStack V3 API に必要なその他のパラメータは、--os_user_domain_name、--os_project_domain_name、--os_project_name、--os_password、--os_auth_url、--os_username、--os_identity_api_version、--os_ca_cert、--requests_ ca_bundle です。

（注） また、引数、--os_tenant_name、--os_username、--os_password、--os_auth_url では、VIM コネクタもデフォルトで設定されています。VIM コネクタの設定をスキップする場合は、パラメータ（--no_vim_credentials）を bootvm.py で渡します。no_vim_credentials パラメータが指定されている場合、bootvm.py 引数（os_tenant_name、os_username、os_password、os_auth_url）は無視されます。インストール後の VIM コネクタの設定、および VIM コネクタの管理の詳細については、『*Cisco Elastic Services Controller User Guide*』の「Managing VIM Connectors」を参照してください。

（注） --os_ca_cert および --requests_ca_bundle 引数は、https 接続にのみ必要です。

ステップ 4 OpenStack コマンドを実行する任意のシェルで、それぞれのプロジェクトの *PROJECT-openrc.sh* ファイルをソースします。この例では、管理者プロジェクトの *admin-openrc.sh* ファイルを送信します。

```
$ source admin-openrc.sh
```

ステップ 5 環境変数を確認します。

```
$ env | grep OS_
```

ステップ 6 glance コマンドを使用して、ESC イメージファイルを OpenStack イメージに登録します。

```
$ glance image-create \
--name <image_name> \
--is-public=<true or false> or --visibility public or private\
--disk-format  <disk_format> \
--container-format <container_format> \
--file <file>\
--progress
```

設定例を次に示します。

```
$  glance image-create \
--name esc-1_0_01_11_2011-01-01 \
--is-public=<true or false> or --visibility public or private \
--disk-format qcow2 \
--container-format bare \
--file esc-1_0_01_11_2011-01-01.qcow2 \
--progress
```

glance image-create コマンドを使用して、新しいイメージを作成します。このコマンドは、次の引数を使用します。

（注） 「is-public」引数は OpenStack Kilo リリースにのみ適用されます。

引数	説明
name	イメージの名前。

引数	説明
is-public	（任意）イメージを公開アクセスできるようにします。
disk-format	イメージのディスク形式。ESC は qcow2 ディスク形式を使用します。
container-format	イメージのコンテナ形式。ESC は、ベアコンテナ形式を使用します。
file	作成時にアップロードされるディスクイメージを含むローカルファイル。
progress	（任意）アップロードの進行状況バーを表示します。

イメージが正常に登録されているかどうかを以下の手順で確認します。

a) OpenStack コントローラダッシュボードの使用：

- クレデンシャルを使用して OpenStack にログインします。
- [管理者（admin）] > [イメージ（image）] の順に移動します。

  イメージがリストに表示されているかどうかを確認します。

b) Nova CLI の使用：

```
$ nova image-show <image_name>
```

ステップ 7 ESC の標準リソース要件は、4vCPU、8G RAM、および 40GB ディスク容量です。ESC インストールスクリプトは、4vCPU、8G RAM、および 80G ディスク領域の定義を持つ事前定義された「m1.large」フレーバを使用します。40GB のディスク容量を使用するには、最小ディスク容量を持つフレーバーを作成します。

```
$ nova flavor-create ESC_FLAVOR_NAME ESC_FLAVOR_ID 8192 40 4
```

ステップ 8 ESC VM を展開するには、次の手順を実行します。

1. 既存のネットワークが OpenStack コントローラに接続されていることを確認します。Nova CLI を使用してネットワーク接続を確認するには、次を実行します。

   ```
   $ nova net-list
   ```

2. 以前に作成したイメージとフレーバを使用して ESC VM を起動するために ESC が接続するネットワークの ID を記録します。bootvm.py コマンドでは、linux 用の管理者アカウント（ssh/コンソールアクセス）を作成するために、少なくとも 1 つの user_pass 引数が必要です。また、ConfD（netconf/cli アクセス）の管理者アカウントを作成するために、少なくとも 1 つの user_confd_pass が必要です。次に、これらの必須ユーザクレデンシャル引数の構文を示します。

   ```
   --user_pass admin:'PASSWORD-OR-HASH'[:OPTIONAL-PUBLIC-KEY-FILE][:OPTIONAL-ROLE]
   --user_confd_pass admin:'PASSWORD-OR-HASH'[:OPTIONAL-PUBLIC-KEY-FILE]
   ```

   ハッシュされたパスワードの生成は任意です。必要に応じて、プレーンパスワードを選択できます。

   Ubuntu OS でハッシュされたパスワードを生成するには、次のコマンドを使用します。

```
mkpasswd --method=SHA-512 --salt XyZ123 <<< <Password>
```

ESC リリース 5.4 以降では、ユーザ名とパスワードを指定せずに esc_nc_cli コマンドを呼び出すと失敗し、次のメッセージが表示されます。

```
admin@esc$ esc_nc_cli --user <username> --password <password> get
esc_datamodel/opdata/status
ERROR Cannot find file /home/admin/.ssh/confd_id_rsa to support public key
authentication with esc-nc-admin
* Check your arguments or installation.
* Usage without --user --password or --privKeyFile arguments requires ssh keys and
configuration.
* Use the following command to generate new SSH Keys and update to authorized_keys:
sudo escadm confd keygen --user admin
* NOTE: Consult with site security policies for use of public key authentication.
```

ユーザ名とパスワードを指定して esc_nc_cli コマンドを実行する必要があります。

```
admin@esc$ esc_nc_cli --user admin --password ADMIN-CONFD-PASSWORD get
esc_datamodel/opdata/status
Operational Data
/opt/cisco/esc/confd/bin/netconf-console --port=830 --host=127.0.0.1 --user admin
--password=****** --get -x "esc_datamodel/opdata/status"
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="1">
  <data>
    <esc_datamodel xmlns="http://www.cisco.com/esc/esc">
      <opdata>
        <status>OPER_UP</status>
      </opdata>
    </esc_datamodel>
  </data>
</rpc-reply>
```

次に、許可された公開キーを使用して ESC をインストールする例を示します。次の例では、シェル予約済み文字との競合を回避するために一重引用符が使用されています。

```
--user_pass admin:'$algorithm$salt$hash-of-salt-password':$HOME/.ssh/esc_rsa.pub
--user_confd_pass admin:'$algorithm$salt$hash-of-salt-password':$HOME/.ssh/esc_rsa.pub
```

公開キーは、次のようなキーペアの一部として生成されます。

```
ssh-keygen -t rsa -b 1024 -C "esc" -N "" -f ~/.ssh/esc_rsa
```

公開キーと識別キーは、/home/username/.ssh/esc_rsa および esc_rsa.pub ファイルに保存されます。ユーザクレデンシャル引数の例については、「付録 A：Cisco Elastic Services Controller インストーラの引数」を参照してください。

3. ESC VM の詳細を確認し、ESC VM の IP アドレスを含む情報を取得するには、次のコマンドを使用します。

```
$ nova show <esc_vm_name>
```

追加のインストールオプション

- **ConfD SSH キーの作成**：ESC リリース 5.4 以降では、次のコマンドを呼び出して、SSH キーを作成し、公開キー認証を設定できます。

  ```
  sudo escadm confd keygen --user USERNAME
  ```

  パスワードなしの認証を必要とするESCVMの各ユーザアカウントで上記のコマンドを実行します。アクティブ/スタンバイ展開では、両方のノードでコマンドを実行します。

  bootvm.py を使用した ConfD SSH キーの有効化：次の bootvm.py コマンドを有効にすると、ESC リリース 5.4 以前の機能が復元されます。

  ```
  --confd_keygen
  ```

  ConfD cli の詳細については、「コマンドライン インターフェイスを使用した ConfD Netconf/CLI 管理者パスワードの変更」を参照してください。

- **OpenStack IPv6 環境での ESC の展開**：IPv6 で ESC インスタンスを展開する前に、必ず ipv6 アドレスをサポートしている openrc を送信してください。IPv6 環境に ESC を展開するには、次の bootvm 引数を使用します。

  ```
  ./bootvm.py <esc_vm_name-ipv6> --poll --user_rest_pass <username>:<password> --image
   <image_name>
  --net <ipv6_network> --ipaddr <ipv6_ip_address> --enable-http-rest --user_pass
  <username>:<password>
  --user_confd_pass <username>:<password> --etc_hosts_file <hosts-file-name> --route
  <default routing configuration>
  ```

- **DHCP モードでの ESC の展開**：--Ipaddr を指定せずに bootvm.py 引数を使用すると、ESC インスタンスが DHCP モードで展開されます。DHCP ネットワークに ESC を展開するには、次の設定を使用します。

  ```
  ./bootvm.py <esc_vm_name> --image <image_name> --net <IPv6 network> <IPv4 network>
  --flavor <flavor_name>
  --user_pass <username>:<password>
  --user_confd_pass <username>:<password>
  ```

（注） デフォルトでは、ESC は IPv4 ネットワークの DHCP のみをサポートします。IPv6 が使用されている場合は、ESC VM にログインし、「dhclient -6 ethX」（ethX は V6 インターフェイス名）を手動で実行して、V6 DHCP を有効にする必要があります。

複数のネットワーク インターフェイスを使用して ESC を展開するときに、1 つ以上のタイプの DHCP を使用する場合は、`bootvm.py--Defroute N` を使用して、デフォルトルートとゲートウェイ IP を割り当てるインターフェイス インデックスを指定します。

デフォルトでは、`N = 0` です。したがって、`bootvm.py` コマンドラインの最初のネットワーク インターフェイスはデフォルトです。

例：

```
--defroute 1 will assign <network2> with
<default_gateway_ip_address>
./bootvm.py <esc_vm_name> --image <image_id> --net <network1>
<network2> --defroute 1 --gateway_ip
<default_gateway_ip_address>
```

- **ESC インストールでブート可能なボリュームを使用する場合**：ESC インスタンスにボリュームを接続し、ボリューム内からインスタンスを起動することができます。詳細については、「ESC のインストールでブート可能ボリュームを使用」の項を参照してください。

- **ESC へのフローティング IP の割り当て**：フローティング IP を ESC インスタンスに関連付ける場合は、次の手順を実行します。

  1. 使用可能なフローティング IP アドレスを確認し、ESC VM に割り当てます。

     ```
     $ nova floating-ip-list
     $ nova floating-ip-associate esc_vm_name <ip_address>
     ```

  2. または、新しいフローティング IP アドレスを作成し、ESC VM に割り当てます。

     ```
     $ nova floating-ip-create <FLOATING_NETWORK - ID>
     $ nova floating-ip-associate esc_vm_name <ip_address>

     or
     neutron floatingip-create FLOATING_NETWORK
     neutron floatingip-associate floating-ip-ID port-ID
     ```

- **スタティック IP を使用した ESC の展開**：スタティック IP を使用する特定のネットワークで ESC を使用するには（たとえば、network1 における 192.168.0.112）、次に示すように、bootvm コマンドラインに **--ipaddr** および **--gateway_ip** を指定します。

（注） スタティック IP アドレスを割り当てる前に、スタティック IP が使用可能であり、他のマシンで使用されていないことを確認してください。

```
./bootvm.py <esc_vm_name> --image <image_id> --net <network>
--ipaddr <ip_address> --gateway_ip <default_gateway_ip_address> --user_pass
<username>:<password>
--user_confd_pass <username>:<password>
```

- **複数のネットワーク インターフェイスを使用した ESC の展開**：ESC に複数のネットワークを使用するには（たとえば、network1 における 192.168.0.112 および network2 における 10.20.0.112）、次のコマンドラインの **--net**および **--ipaddr** 引数にインターフェイスの IP アドレスとネットワーク名の両方を指定します。さらに、これらのネットワークのゲートウェイから、ESC のデフォルトゲートウェイも選択します。**--gateway_ip** 引数を使用して、ESC のデフォルトゲートウェイを指定します。

```
./bootvm.py <esc_vm_name> --image <image_id> --net <network1>
<network2>  --ipaddr <ip_address1> <ip_address2> --gateway_ip
<default_gateway_ip_address>
--user_pass <username>:<password>
--user_confd_pass <username>:<password>
```

（注） **--flavor** が指定されていない場合、bootvm.py は OpenStack でデフォルトのフレーバ「m1. large」を使用します。

フレーバーが指定されていない場合は、OpenStack に m1.large が存在することを確認し、そのフレーバーが ESC の導入要件に適合することを確認します。

- **ログ転送オプションを使用して ESC を展開します。**ESC ログを rsyslog サーバに転送するには、ESC VM の作成時に rsyslog サーバの IP アドレスを指定します。必要に応じて、使用するポートとプロトコルを指定することもできます。

  たとえば、rsyslog サーバの IP アドレスが 172.16.0.0 で、ログを転送するサーバのポートが 514 で、使用されているプロトコルが UDP である場合、ESC のインストールは次のようになります。

```
./bootvm.py <esc_vm_name> --image <image_id> --net network1 --rsyslog_server 172.16.0.0

--rsyslog_server_port 514 --rsyslog_server_protocol udp --user_pass
<username>:<password>
--user_confd_pass <username>:<password>
```

- **ESC GUI を無効にする**：グラフィカル ユーザ インターフェイスを無効にして ESC VM を起動するには、次のコマンドラインに示すように、--esc_ui_startup 引数の値を変更します。

```
./bootvm.py <esc_vm_name> --image <image_id> --net <network> --user_pass
<username>:<password>
--user_confd_pass <username>:<password>
--esc_portal_startup=False
```

- **ESC の REST インターフェイスを有効にする**：REST インターフェイスをサポートするには、**--enable-https-rest** 引数を指定します。REST インターフェイスは、https または http の両方でアクティブにすることができます。

```
./bootvm.py <esc_vm_name> --image <image_id> --net <network> --user_pass
<username>:<password>
--user_confd_pass <username>:<password> --enable-https-rest

OR

./bootvm.py <esc_vm_name> --image <image_id> --net <network> --user_pass
<username>:<password>
--user_confd_pass <username>:<password> --enable-http-rest
```

-
- **ETSI の REST インターフェイスを有効にする**：ETSI REST のインターフェイスをサポートするには、**--enable-http-etsi** を指定して http 経由でインターフェイスをアクティブにするか、または **--enable-https-etsi** を指定して https 経由でインターフェイスをアクティブにします。

```
./bootvm.py <esc_vm_name> --image <image_id> --net <network> --user_etsi_pass
<username>:<password> --enable-https-etsi . . .

OR

./bootvm.py <esc_vm_name> --image <image_id> --net <network> --user_etsi_pass
<username>:<password>
--user_confd_pass <username>:<password> --enable-http-etsi...
```

（注） 実稼働環境では、https REST インターフェイスと ETSI インターフェイスのみを有効にする必要があります。

- **ETSI OAuth2 クライアントを使用した ESC の導入**：インストール時に ETSI OAuth2 クライアントを追加するには、次のように引数を使用します。インストール後に escadm コマンドを使用して OAuth2 クライアントを追加することもできます。

```
./bootvm.py <esc_vm_name> --image <image_id> --net <network> --
user_etsi_pass <username>:<password> --etsi_oauth2_pass <clientId>:<clientSecret>
--enable-https-etsi ...
```

- **グローバルパラメータを使用した ESC の展開**：インストール中に esc_params_file を使用してグローバル コンフィギュレーションを設定するには、次に示すように引数を使用します。これらのグローバル設定は、インストール後に REST API を使用して変更することもできます。

（注） テナントの作成時に、デフォルトのセキュリティグループがテナントに適用されます。デフォルトでは、セキュリティグループ、*openstack.DEFAULT_SECURITY_GROUP_TO_TENANT* の ESC 設定パラメータは true に設定されています。設定パラメータは、インストール時に設定する必要があります。REST API を使用して、ESC VM のパラメータをクエリまたは更新できます。パラメータが true に設定されている場合は、テナントの作成時にデフォルトのセキュリティグループを作成して割り当てることができます。このパラメータが false に設定されている場合、テナントの作成時にデフォルトのセキュリティグループを作成または割り当てることはできません。sc_params_file を使用して設定できるパラメータの詳細については、「付録 A：Cisco Elastic Services Controller のインストーラの引数」を参照してください。

```
./bootvm.py <esc_vm_name> --image <image_id> --net <network>  --flavor <flavor_name>

--user_pass <username>:<password>:<public key file> --user_confd_pass
<username>:<password>
--esc_params_file <esc parameter configuration file>
```

- **ETSI プロパティを使用した ESC の展開**：インストール中に *etsi_params_file* を使用して ETSI プロパティを設定するには、次のように引数を使用します。これらのプロパティは、インストール後に *etsi-product.properties* ファイルでも変更できます。

（注） etsi_params_file で設定できるプロパティの詳細については、『*ETSI NFV MANO* ユーザガイド』の「ETSI 製品のプロパティ」の章を参照してください。

```
./bootvm.py <esc_vm_name> --image <image_id> --net <network> --flavor <flavor_name>

--user_pass <username>:<password>:<public key file> --user_confd_pass
<username>:<password>
--etsi_params_file <etsi properties file>
```

- **ESC の 2 つのインスタンスを展開して ESC HA（アクティブ/スタンバイ）ペアを構築する**：ESC HA アクティブ/スタンバイの展開についての詳細は、「OpenStack への ESC のインストール」および「VMware への ESC のインストール」章の「高可用性の設定」を参照してください。

- **ダイナミック マッピング ファイルの追加**：Cisco ESC リリース 2.1 以前では、データモデルで定義されたアクションおよびメトリックから、モニタリングエージェントで使用可能な有効なアクションおよびメトリックへのマッピングは、*dynamic_mappings.xml* ファイルを使用して有効化されていました。ファイルは ESC VM に保存され、テキストエディタを使用して変更されました。ESC 2.2 以降には、*esc-dynamic-mapping* ディレクトリと *dynamic_mappings.xml* ファイルがありません。既存の dynamic_mapping xml ファイルを ESC VM に追加する場合は、次の手順を実行します。

1. このファイルを、ホームディレクトリなどのESC以外の場所にバックアップします。
2. ESC VM で *esc-dynamic-mapping* ディレクトリを作成します。読み取りアクセス許可が設定されていることを確認します。
3. 次の bootvm 引数を使用して、ESC VM にインストールします。

```
 --file
root:root:/opt/cisco/esc/esc-dynamic-mapping/dynamic_mappings.xml:<path-to-local-copy-of-dynamic-mapping.xml>
```

アクションとメトリックをマッピングするための CRUD 操作は、REST API を介して使用できます。既存のマッピングを更新するには、REST API を使用してそのマッピングを削除して、新しいマッピングを追加します。

- **ESC VM で confd パスワードを変更する**：管理者は、インストール時に bootvm.py を使用して confd パスワードを設定できます。

```
./bootvm.py --user_pass <username>:<password> --user_confd_pass
admin:'PASSWORD-OR-HASH':OPTIONAL-PUBLIC-KEY
```

インストール後にこのパスワードを再設定するには、次のコマンドを実行します。

```
$ /opt/cisco/esc/confd/bin/ssh admin@localhost -p 2024
$ configure
$ set aaa authentication users user admin password <your_password>
$ commit
$ exit
```

（注） 今後のアップグレードを容易にするために、bootvm.py ファイルを使用して ESC をインストールする際に使用されるすべてのコマンドと引数のコピーを保管するようにしてください。

Cisco Elastic Services Controller でのルート証明書の管理

Cisco Elastic Services Controller（ESC）は、SSL 証明書の検証を有効にするメカニズムを提供します。この機能は現在、OpenStack でのみサポートされています。証明書の検証は、ESC の初回起動時にデフォルトで有効になっています。ただし、ESC を使用すると、これらの SSL 証明書を設定することもできます。このセクションでは、OpenStack で Cisco Elastic Service Controller の証明書の検証を有効化/無効化、追加/削除、または一覧表示する方法について説明します。ESC の起動中、または ESC の起動が完了した後でも、ルート証明書を追加できます。

ルート証明書の検証の有効化/無効化

Cisco Elastic Services Controller は、デフォルトで証明書の検証を有効にします。また、有効または無効にするには、esc_params.conf ファイル の Openstack カテゴリで使用可能なパラメータ DISABLE_CERT_VALIDATION を変更するか、REST インターフェイスを使用するか、または escadm ツールを使用することもできます。

ESC アクティブノードで、コマンド **sudo escadm enable-certificate** または **sudo escadm disable-certificate** を使用して、証明書の検証をそれぞれ有効または無効にします。

ルート証明書の追加

ESC の起動中、または ESC の起動が完了した後でも、ルート証明書を追加できます。証明書を追加する前に、OpenStack 環境ファイルを確認します。OpenStack RC ファイルには、OpenStack で認証とインストールを実行するためのパラメータがあります。パラメータを渡すときに --os_auth_url を指定する必要があります。--os_auth_url は、OpenStack が認証に使用するセキュア（https）または非セキュア（http）キーストーン URL を指定します。

- 起動時にスタンドアロン（のみ）の証明書を追加します。つまり、ESC VM のインストール時に次のようにします。

```
./bootvm.py test-vm --image <image_name> --net <network> [--cert_file CERT_FILE]
[--confd_aes_key CONFD_AES_KEY]
 /home/cisco/openstack.crt
--user_pass <username>:<password> --user_confd_pass <username>:<password>
```

（注） 現在、ESC では、証明書の追加時に keepalived サービスが実行されていないため、インストール中に HA アクティブ/スタンバイの証明書を追加することはサポートされていません。

- ESC インスタンスを起動した後、スタンドアロン/HA アクティブ/スタンバイの証明書を追加します。escadm ツールには、次の引数を持つ **truststore add** オプションがあります。--file 引数は、CA 証明書ファイルを参照します。この引数を使用すると、Java keytool でサポートされている任意のファイル形式（X.509 v1、v2、v3 証明書、および PKCS＃7）をインポートできます。--alias 引数は一意であり、この特定の CA 証明書が付与されている名前を参照します。

  1. CA 証明書ファイルを ESC アクティブ VM にコピーまたは転送します。
  2. 証明書を ESC トラストストアに追加します。これを行うには、次のコマンドを実行します。

     ```
     sudo escadm escadm truststore add --alias [ca cert alias] --file [file path]
     ```

     または

     ```
     sudo escadm truststore escadm truststore --alias [ca cert alias] --file [file
     path]
     ```

  3. 証明書が追加されていることを確認します。

     ```
     sudo escadm truststore show
     ```

     または

     ```
     sudo escadm truststore show
     ```

ルート証明書の削除

escadm ツールには、--alias 引数のみを実行する「truststore delete」オプションがあります。--alias 引数は、削除する CA 証明書の名前を参照します。スタンドアロン/HA アクティブ/スタンバイ ESC VM でこの引数を使用します。

手順

ステップ 1 （アクティブ）ESC で escadm を使用して、ESC トラストストアから証明書を削除します。

```
sudo escadm truststore delete --alias [ca cert alias]
```

または

```
sudo escadm truststore truststore delete --alias [ca cert alias]
```

ステップ 2 証明書が削除されていることを確認します。

```
sudo escadm truststore show
```

または

```
sudo escadm truststore show
```

アップグレード中のルート証明書の管理

- **イメージのアップグレード**：アップグレードをするために ESC DB をバックアップしている場合、他のアクションは必要ありません。ESC DB を復元されると、ESC トラストストアが復元されます。アップグレードのために ESC DB をバックアップしていない場合は、各 CA 証明書を ESC トラストストアに再度追加する必要があります。
- **RPM アップグレード**：このアップグレード方式では、ESCトラストストアをそのまま保持します。つまり、ESC トラストストア内のすべての CA 証明書は、アップグレード後も保持されます。

Cisco Elastic Services Controller でのキーストアの管理

キーストアは、アプリケーションが他のクライアントを使用して自身を認証するために使用する証明書とキーのストレージです。

ESC キーストアは、ESCManager、VIMManager、MONA などのすべてのアプリケーションで使用される証明書を 1 つのみ保持します。

ESCADM には、キーストアを管理するための複数のコマンドが用意されています。

ESC を初めて展開すると、自己署名証明書が作成され、デフォルトでキーストアに保存されます。

特記事項：

- アクティブ/アクティブモードでは、デフォルトの証明書の共通名（CN）は `db.service.consul` です。新しい証明書を設定するときには、同じ CN を使用して設定する必要があります。そうしない場合は、ESC 展開時の MONA での証明書の検証を無効にするため、すべてのノードの Heat テンプレートに次の設定を追加する必要があります。

```
mona:
          certificate_validation: false
```

- すべての escadm キーストアコマンドには、ESC で行使される root 権限が必要です。

escadm キーストアコマンド

- `escadm  keystore show`

  `escadm keystore show` コマンドにより、キーストアに現在保存されている証明書に関する情報が表示されます。作成日、エイリアス、証明書のフィンガープリントなどの情報が表示されます。

  次に例を示します。

```
$ sudo escadm keystore show
Keystore type: PKCS12
Keystore provider: SUN
Your keystore contains 1 entry
esc, Apr 13, 2020, PrivateKeyEntry,
Certificate fingerprint (SHA1):
FF:11:66:3E:93:DD:3A:0B:9A:72:40:16:35:34:D2:22:E1:25:07:80
```

- `escadm keystore export [--out <file path>]`

  export コマンドを実行すると、証明書のすべてのコンテンツが表示されます。`out` オプションが指定されている場合、コンテンツは、以前のオプションで指定されたパスのファイルに保存されます。

- `escadm keystore set-- file <file path>`

  set コマンドにより、現在の証明書が、`file` オプションで指定されたパスを持つファイル内に存在する新しい証明書に置き換えられます。

  ファイルがPEM形式であり、証明書と秘密キーの両方が含まれていることを確認します。

  次の例は、新しい自己署名証明書を生成し、キーストアに対して設定する方法を示しています。

  1. 証明書を生成するには、次のコマンドを使用します。

     ```
     openssl req -newkey rsa:2048 -new -nodes -x509 -days 3650 -keyout key.pem -out
     cert.pem
     ```

     前のコマンドで、次の 2 つのファイルが作成されます。

     `key.pem` と `cert.pem`

  2. 次のコマンドを使用して、2 つのファイルを結合します。

     ```
     cat key.pem > server.pem
     ```

```
cat cert.pem >> server.pem
```

3. 新しい証明書をキーストアに対して設定するには、次のコマンドを使用します。

```
$ sudo escadm keystore set --file server.pem
```

```
Service "keystore" successfully updated ESC keystore and will take effect once
ESC services are restarted by running "sudo escadm restart"
```

（注） 設定が完了した後、秘密キーと証明書を含む残りのファイルすべてを削除してください。

この変更を有効にするには、システムの再起動が必要です。スタンドアロンモードまたは H/A モードで再起動するには、次のコマンドを使用します。

```
sudo escadm restart
```

ESC がアクティブ/アクティブモードで実行されている場合は、同じクラスタ内のすべてのノードを再起動する必要があります。ただし、ESC がアクティブ/アクティブ GEO モードで実行されている場合は、すべてのクラスタの GEO サービスを停止して、GEO スイッチオーバーが実行されないようにする必要があります。

GEO アクティブ/アクティブモードの場合に限り、GEO サービスを停止するには、各クラスタ内の任意のノードにログインして、次のコマンドを使用します。

```
$ sudo escadm geo stop --cluster
```

証明書が元々設定されているクラスタ内の任意のノードにログインできます。次のコマンドを使用します。

```
$ sudo escadm stop --cluster
$ sudo escadm start --cluster
```

GEO アクティブ/アクティブモードの場合は、すべてのノードが稼働状態になったら、任意のノードに再度ログインして GEO サービスを開始します。サービスを開始するには、次のコマンドを使用します。

```
$ sudo escadm geo start --cluster
```

ESC のインストールでブート可能ボリュームを使用

OpenStack のボリュームは取り外し可能なブロックストレージデバイスであり、ESC インスタンスに接続できます。ESC インスタンスをボリュームに 保存し、ボリュームから ECS インスタンスを実行することもできます。

（注）
- 一度に 1 つのボリュームから起動できるのは、1 つの ESC インスタンスだけです。
- Cinder では、ブート可能ボリュームと高可用性（アクティブ/スタンバイおよびアクティブ/アクティブ）を組み合わせた ESC インストールはサポートされていません。

ブート可能ボリュームから ESC インスタスを起動するには、次の手順を実行します。

手順

ステップ 1 ESC イメージまたはブート可能ボリュームを使用して、OpenStack にブート可能ボリュームを作成します。ブート可能ボリュームには、30 GB 以上のディスクサイズが必要です。詳細については、OpenStack のマニュアルを参照してください。

ステップ 2 以下に示すように、`bootvm.py` コマンドを使用して ESC VM を展開します。`--image` 引数の代わりに `--boot_volume` 引数を選択します。

```
./bootvm.py <esc_vm_name> --boot_volume <volume_name_or_id> --net <network> --user_pass
 <username>:<password>
--user_confd_pass <username>:<password> --flavor <flavor_name>
```

（注）
- bootvm.py コマンドには、`--image` と `--boot_volume` のどちらか 1 つを指定する必要があります。両方の引数が使用されたた場合、あるいはどちらの引数も使用されていない場合、インストールは失敗します。
- ブート可能ボリュームから ESC インスタンスを起動すると、ボリュームディスクサイズはフレーバディスクサイズを超えて考慮されます。
- ボリュームがアウトオブバンドで作成されたため、ESC インスタンスを削除しても、そのインスタンスに接続されているボリュームは削除されません。