Cisco Elastic Services Controller 5.2 インストールおよびアップグレードガイド

初版：2020 年 5 月 29 日

最終更新：2020 年 7 月 20 日

最終更新：2020 年 9 月 3 日

最終更新：2020 年 9 月 11 日

シスコシステムズ合同会社
〒107-6227 東京都港区赤坂9-7-1 ミッドタウン・タワー
http://www.cisco.com/jp
お問い合わせ先：シスコ コンタクトセンター
0120-092-255 （フリーコール、携帯・PHS含む）
電話受付時間：平日 10:00～12:00、13:00～17:00
http://www.cisco.com/jp/go/contactcenter/

目 次

このマニュアルについて

このガイドでは、Cisco Elastic Services Controller のインストール要件、手順、およびアップグレード手順について説明します。

- 対象読者 （xi ページ）

対象読者

このガイドは、VNF のプロビジョニング、設定、およびモニタリングを担当するネットワーク管理者を対象としています。Cisco Elastic Services Controller（ESC）とその 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。現在、OpenStack、VMware vCenter、VMware vCloud Director、CSP 2100/5000、および Amazon Web Services（AWS）は、サポートされている VIMs です。管理者は、VIM レイヤ、vCenter、OpenStack および AWS のリソース、ならびに使用するコマンドに精通している必要があります。

Cisco ESC は、サービスプロバイダー（SP）および大企業を対象としています。ESC は、効果的かつ最適なリソース使用率を実現することにより、ネットワークの運用コストの削減に役立ちます。大企業向けに、ESC はネットワーク機能のプロビジョニング、設定、およびモニタリングを自動化します。

用語および定義

次の表で、このガイドで使用されている用語を定義します。

表 *1* : 用語および定義

用語	定義
AWS	Amazon Web Services（AWS）はセキュアなクラウドサービス プラットフォームであり、コンピューティング、データベースストレージ、コンテンツ配信、その他の機能を提供します。
ESC	Elastic Services Controller（ESC）は仮想ネットワーク機能マネージャ（VNFM）であり、仮想ネットワーク機能のライフサイクル管理を実行します。

用語	定義
ETSI	欧州電気通信標準化機構（ETSI）は、欧州内の情報通信技術（ICT）の標準開発において貢献してきた独立標準化機関です。
ETSI 展開フレーバ	展開フレーバの定義には、VNF インスタンスに適用するアフィニティ関係、スケーリング、最小/最大 VDU インスタンス、その他のポリシーと制限に関する情報が含まれています。VNF 記述子（VNFD）で定義された展開のフレーバは、インスタンス化 VNF LCM 操作時に InstantiateVNFRequest ペイロードで *flavour_id* 属性を渡すことによって選択する必要があります。
HA	ESC 高可用性（HA）は、ESC のシングルポイント障害を防止し、ESC のダウンタイムを最小限に抑えるためのソリューションです。
KPI	重要業績評価指標（KPI）は、パフォーマンス管理を測定します。KPI は、どのようなパラメータをいつ、どのように測定するかを指定します。KPI には、特定のパラメータのソース、定義、測定、計算に関する情報が組み込まれています。
MSX	Cisco Managed Services Accelerator（MSX）は、企業とサービスプロバイダーの両方の顧客にクラウドベースのネットワーキングサービスを迅速に導入できるようにするサービスの作成と配信のプラットフォームです。
NFV	ネットワーク機能仮想化（NFV）は、仮想ハードウェアの抽象化を使用して実行するネットワーク機能をハードウェアから分離する原則です。
NFVO	NFV オーケストレータ（NFVO）は、ネットワークサービス（NS）のライフサイクルを管理し、NS ライフサイクル、VNF ライフサイクル（VNFM でサポート）、NFVI リソース（VIM でサポート）の管理を調整して、必要なリソースと接続の割り当てを最適化します。
NSO	Cisco Network Services Orchestrator（NSO）は、サービス アクティベーションのためのオーケストレータであり、純粋な物理ネットワーク、ハイブリッドネットワーク（物理および仮想）、および NFV の使用をサポートします。
OpenStack コンピューティングのフレーバ	フレーバで、Nova コンピューティング インスタンスのコンピューティング、メモリ、およびストレージ容量を定義します。フレーバは、サーバに使用可能なハードウェア設定です。起動可能な仮想サーバのサイズを定義します。
サービス	サービスは、1 つまたは複数の VNF で構成されます。
VDU	仮想化展開ユニット（VDU）は、情報モデルで使用できる構成要素であり、VNF のサブセットの展開と運用動作の説明、またはサブセットにコンポーネントとして含まれていない場合は VNF 全体の説明をサポートします。

用語	定義
VIM	仮想インフラストラクチャ マネージャ（VIM）は、データセンターハードウェアの管理レイヤを追加します。このノースバウンド API は、インスタンス化、終了、スケールインとスケールアウトの手順、ならびに障害とパフォーマンスのアラームの物理リソースと仮想リソースを管理するために、他のレイヤによって使用されます。
VM	仮想マシン（VM）は、オペレーティングシステム OS またはソフトウェアにインストールされているアプリケーションであり、専用ハードウェアを模倣します。エンドユーザは、仮想マシン上でも専用ハードウェア上と同じように操作できます。
VNF	仮想ネットワーク機能（VNF）は、ネットワーク機能仮想化（NFV）インフラストラクチャに展開可能なさまざまなソフトウェアとプロセスを備えた 1 つの VM または 1 つのグループの VM で構成されます。
VNFC	仮想ネットワーク機能コンポーネント（VNFC）は、VNF の複合部分であり、VDU と同義で、VM またはコンテナとして実装できます。
VNFM	仮想ネットワーク機能マネージャ（VNFM）は、VNF のライフサイクルを管理します。

関連資料

Cisco ESC のドキュメントセットは、さまざまな API を使用した VNF のインストール、設定、ライフサイクル管理操作、修復、スケーリング、モニタリング、メンテナンスの実行に役立つ次のガイドから構成されています。

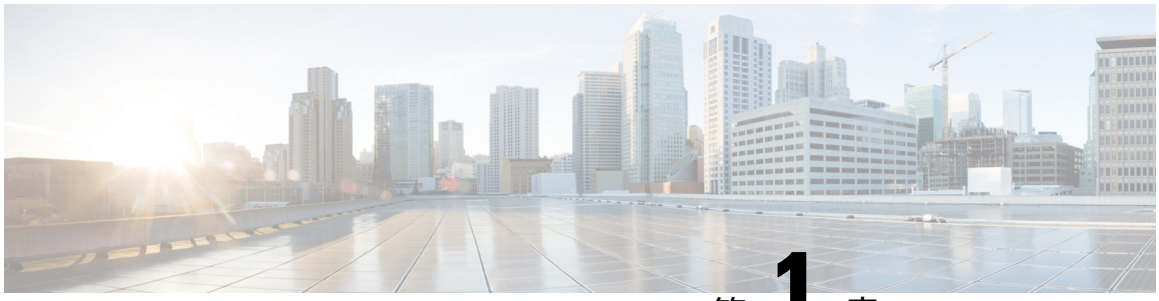
ガイド	このガイドに記載されている情報
Cisco Elastic Services Controller Release Notes	新機能とバグ、既知の問題が記載されています。
Cisco Elastic Services Controller Install and Upgrade Guide	新規インストールとアップグレードのシナリオ、インストール前後のタスク、ESC 高可用性（HA）展開の手順が記載されています。
Cisco Elastic Services Controller User Guide	VNF のライフサイクル管理操作、モニタリング、修復、スケーリングが記載されています。
Cisco Elastic Services Controller ETSI NFV MANO User Guide	ETSI API を使用した VNF のライフサイクル管理操作、モニタリング、修復、スケーリングが記載されています。
Cisco Elastic Services Controller 5.1 Administration Guide	メンテナンス、ESC の正常性のモニタリング、および ESC が生成したシステムログに関する情報が記載されています。

ガイド	このガイドに記載されている情報
Cisco Elastic Services Controller NETCONF API Guide	Cisco Elastic Services Controller NETCONF ノースバウンド API に関する情報とそれらの使用方法が記載されています。
Cisco Elastic Services Controller REST API Guide	Cisco Elastic Services Controller RESTful ノースバウンド API に関する情報とそれらの使用方法が記載されています。
Cisco Elastic Services Controller ETSI REST API Guide	Cisco Elastic Services Controller ETSI API に関する情報と、それらの使用方法が記載されています。
Cisco Elastic Services Controller Deployment Attributes	展開データモデルで使用される展開属性に関する情報が記載されています。
Cisco Elastic Services Controller Open Source	Cisco Elastic Services Controller で使用されているオープンソースソフトウェアのライセンスと通知に関する情報が記載されています。

ドキュメントの入手方法

マニュアルの入手、Cisco Bug Search Tool（BST）の使用、サービス要求の送信、追加情報の収集の詳細については、『What's New in Cisco Product Documentation』を参照してください。このドキュメントは、http://www.cisco.com/c/en/us/td/docs/general/whatsnew/whatsnew.html から入手できます。

『What's New in Cisco Product Documentation』に登録します。ここには、すべての新規および改訂済みの Cisco テクニカル マニュアルが RSS フィードとして掲載されており、コンテンツはリーダー アプリケーションを使用してデスクトップに直接配信されます。RSS フィードは無料のサービスです。



第 1 章

Elastic Services Controller の概要

表 *2 : ESC 5.2* の変更履歴

日付（**Date**）	リビジョン	参照先
2020 年 7 月 20 日	注を追加。	インストールのシナリオ （7 ページ）
2020 年 9 月 3 日	コマンドを更新。	Cisco Elastic Services Controller でのルート証明書の管理 （17 ページ）

Elastic Services Controller の概要

Cisco Elastic Services Controller（ESC）は仮想ネットワーク機能マネージャ（VNFM）であり、仮想ネットワーク機能（VNF）のライフサイクル管理を実行します。ESC は、仮想サービスのプロビジョニングと、正常性および負荷のモニタリングによって、エージェントレスのマルチベンダー VNF 管理を実現します。ESC では、モニタリングのルールを定義するとともに、定義されたルールの結果に基づいてトリガーされるアクションを関連付ける柔軟な対応が可能です。VNFM として、一般的なライフサイクル管理操作に加えて、ESC は、VM で障害が発生した場合の自動 VM リカバリをサポートしており、自動スケールインおよびスケールアウト機能を実行します。ESC は、シスコおよびその他のサードパーティ製アプリケーションと完全に統合されています。

- Cisco Orchestration スイートの一部として、ESC は Cisco Network Services Orchestrator（NSO）を使用してパッケージ化されており、シスコのソリューションである Managed Services Accelerator ソリューション（MSX）内で使用できます。
- ESC は、専用仮想ネットワーク機能マネージャ（SVNFM）として、Cisco Mobility VNF と緊密に統合されます。
- また、ESC は汎用仮想ネットワーク機能マネージャ（GVNFM）としても使用でき、シスコとサードパーティ両方の VNF のライフサイクル管理を提供します。

ESC は、OpenStack、VMware vCenter、KVM または AWS 内の仮想マシンに展開され、仮想インフラストラクチャ マネージャ（VIM）で VNF を管理します。

VNF マネージャとしての Elastic Services Controller は、仮想マネージドサービスと、すべてのサービスプロバイダーの NFV 展開を対象としています（仮想ビデオ、Wi-Fi、認証など）。

ESC は、基本的な VNF と複雑な VNF の両方を管理できます。基本的な VNF には、vFW、vRouter などの単一の VM が含まれます。

複雑な VNF には複数の VM が含まれており、VM 間に依存関係がある単一のエンティティとして組織化されています。

IPv6 のサポート

Elastic Services Controller は、OpenStack で次の IPv6 サポートを提供します。

- VNF 管理
- HA ESC は、IPv4 と IPv6 での VNF を管理します（OpenStack と KVM のみ）。

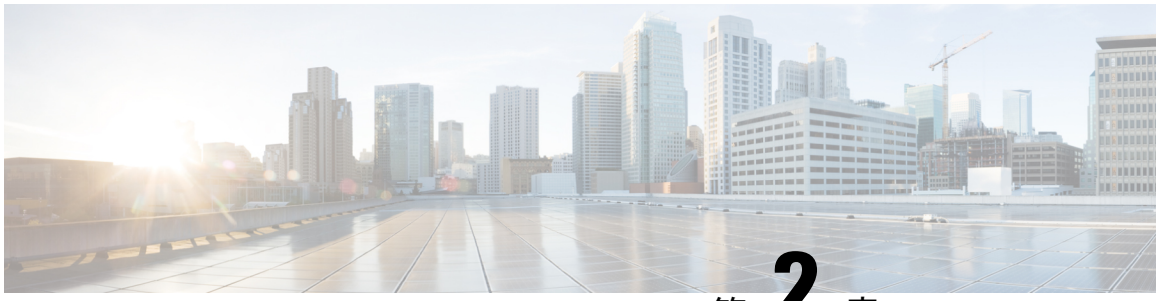
Elastic Services Controller は、ノースバウンド インターフェイス（NFVO から VNFM など）とサウスバウンドインターフェイス（VNFM から VNF など）への IPv6 サポートを提供します。ノースバウンドとサウスバウンド両方の IPv6 を同時にサポートするには、次の前提条件を満たす必要があります。

- OpenStack クラウドコンピューティングが、エンドポイント（IPv6 ベース）を含めて、IPv6 用にセットアップおよび設定されていること。
- OpenStack クラウドコンピューティングに、IPv6 管理で os_api ベースのネットワークを使用するコントローラ、エンドポイント、いくつかのコンピューティングホストが含まれていること。
- ESC のデフォルトのセキュリティグループルールは、IPv6 トラフィックをサポートします。

（注） VM を展開するときに、IPv6 サブネットのアウトオブバンドポートを VM に接続できます。ただし、この VM を削除する場合は、既知の OpenStack の問題により、同じ IPv6 アドレスを別の VM に接続することはできません。

第 部

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



第 2 章

前提条件

このセクションでは、Cisco Elastic Services Controller をインストールするための前提条件について詳しく説明します。

仮想リソースの要件

次の表に、Cisco Elastic Services Controller の仮想リソース要件を示します。

ESC の展開	ESC VM ごとの仮想 CPU	ESC VM ごとの仮想メモリ（GB）	ESC VM ごとの仮想ハードディスク（GB）	サポートされる VM の合計数	VIM
ESC スタンドアロン	4	8	30	1000	サポートされているすべての VIM
ESC アクティブ/スタンバイ HA	4	8	30	2500	サポートされているすべての VIM
ESC アクティブ/アクティブ HA/GEO HA	8	16	60	5000	OpenStack のみ

ソフトウェア要件

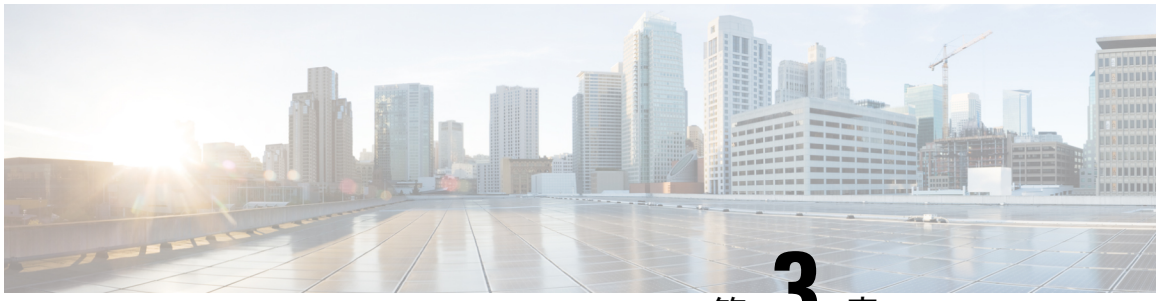
次の表では、OpenStack のソフトウェア要件が記載されています。

要件	説明
サポートされるブラウザ	Google Chrome 44.x 以降
OpenStack バージョン	以下のいずれかに該当するパートナー： • Train • Ocata（V2 および V3） • Queens（Keystone V3）

インストールの準備

インストールを開始する前に、以下のチェックリストを参照して、準備が整っていることを確認します。

要件	ユーザ情報/注記
インストール前の設定	
QCOW イメージの場所	
QCOW イメージ	
インスタンスごとの VM	
IP アドレス	
サブネット マスク	
ホスト名	
ドメイン名	
ゲートウェイ IP アドレス	
admin パスワード	
ESC リリースパッケージ	
ESC.qcow2	ESC インスタンスを起動するためのイメージファイル
bootvm.py	Python 2.7.6 および Python 3.4 と互換性のあるインストールスクリプト



第 3 章

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。プライマリ 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 5.0 以降、アクティブ/パッシブモデルの名称がアクティブ/スタンバイモデルに変更されています。

（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「高可用性アクティブ/スタンバイの構成」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 （5 ページ）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 （6 ページ）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。
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 30GB 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。30GB のディスク領域を使用するには、最小ディスク領域要件を持つフレーバを作成します。

```
$ nova flavor-create ESC_FLAVOR_NAME ESC_FLAVOR_ID 8192 30 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>
```

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

- **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。

- **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-rest . . .

OR

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

（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。

- **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>
```

- **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/confd_cli  -u admin
  $ 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 マスターノードで、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 マスター 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 truststore add --alias [ca cert alias] --file [file path]
```

または

```
sudo escadm truststore truststore add --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 （マスター）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。

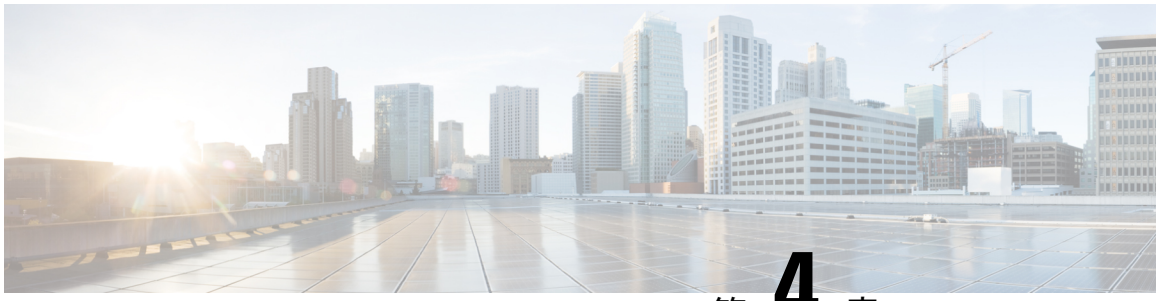
手順

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。



第 4 章

高可用性アクティブ/スタンバイのインストール

この章は、次の項で構成されています。

高可用性アクティブ/スタンバイの概要

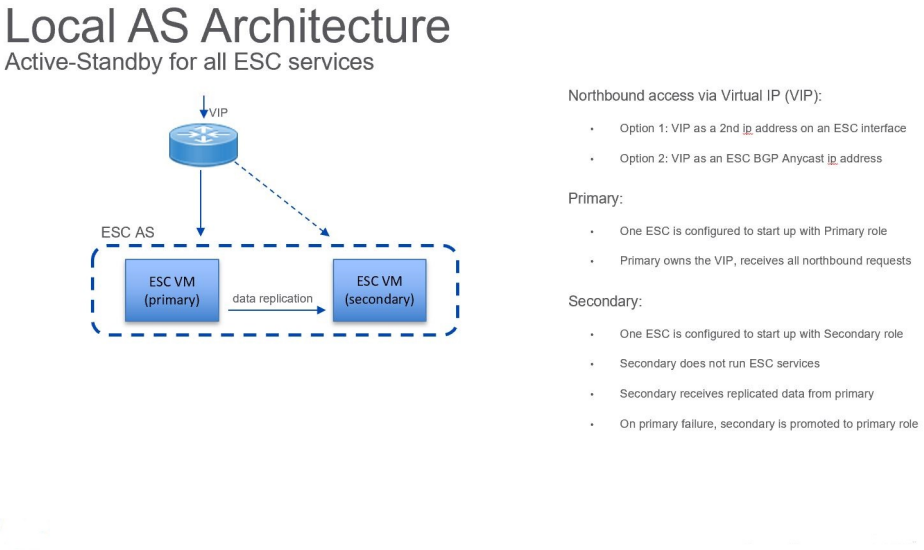
ESC は、アクティブ/スタンバイおよびアクティブ/アクティブモデルの形式で高可用性（HA）をサポートします。アクティブ/スタンバイモデルでは、ESC 障害を防止し、サービスの中断を最小限に抑えて ESC サービスを提供するために、ネットワークに 2 つの ESC インスタンスが展開されます。プライマリ ESC インスタンスで障害が発生しても、スタンバイインスタンスが自動的に ESC サービスを引き継ぎます。ESC HA アクティブ/スタンバイ は、次のシングルポイント障害を解決します。

- 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 5.0 以降、アクティブ/パッシブモデルの名称がアクティブ/スタンバイモデルに変更されています。

ESC アクティブ/スタンバイのアーキテクチャ

図 *1 : Cisco Elastic Services Controller* アクティブ/スタンバイのアーキテクチャ



ハイ アベイラビリティの仕組み

ESC HA アクティブ/スタンバイネットワークは、ESC HA アクティブ/スタンバイペアの単一のインストールとして設定するか、2 つのスタンドアロン ESC ノードとして展開することができます。2 つのノードは、展開後、再設定を経て HA ペアに変換されます。HA 展開は、プライマリとスタンバイの 2 つの ESC インスタンスで構成されます。通常の状況では、プライマリ ESC インスタンスによってサービスが提供されます。対応するスタンバイインスタンスはパッシブ状態になります。スタンバイインスタンスは、プライマリインスタンスと常時通信して、プライマリインスタンスのステータスをモニタします。プライマリ ESC インスタンスに障害が発生すると、スタンバイインスタンスが ESC サービスを自動的に引き継ぎ、最小限の中断で ESC サービスの提供を継続します。

スタンバイインスタンスにもプライマリインスタンスのデータベースの完全なコピーが存在しますが、プライマリインスタンスに障害が発生しない限り、セカンダリインスタンスがアクティブにネットワークを管理することはありません。keepalived サービスは、プライマリとスタンバイ両方のインスタンスのアクティビティステータスをモニタします。プライマリインスタンスに障害が発生すると、スタンバイが自動的に引き継ぎます。プライマリインスタンスの

復元中は、スタンバイインスタンスがプライマリインスタンスを引き継ぎ、サービスを管理します。

障害が発生したインスタンスが復元されたら、必要に応じて手動でスイッチオーバーを開始し、プライマリインスタンスによるネットワーク管理を再開できます。

プライマリとスタンバイ両方の ESC インスタンスは、IPv4 または IPv6 ネットワーク経由でノースバウンド オーケストレーション システムに接続されます。ノースバウンドシステムには、現在のプライマリ ESC 高可用性アクティブ/スタンバイインスタンスにアクセスするための一意の仮想 IP アドレスが割り当てられます。展開された VNF は、別の IPv6 ネットワーク経由で ESC のプライマリとスタンバイ両方のインスタンスに接続されます。

ESC HA アクティブ/スタンバイノードは、keepalived および DRBD（ESC データベースの同期を維持するためのレプリケーションツール）同期ネットワークサービスによって管理されます。keepalived サービスは、プライマリとスタンバイ両方のインスタンスのステータスをモニタしますが、DRBD サービスは、プライマリインスタンス DB をモニタして変更内容をスタンバイインスタンス DB に同期します。これら 2 つのサービスは、同じ VIP ネットワークに配置することも、2 つの異なるネットワークに配置することもできます。ESC インスタンス間の VM ハンドシェイクは、IPv4 または IPv6 ネットワーク上の keepalived を使用して行われます。

ESC 高可用性アクティブ/スタンバイの展開

Cisco Elastic Services Controller（ESC）の高可用性（HA）アクティブ/スタンバイを展開するには、2 つの独立したノード（プライマリおよびスタンバイ）に ESC スタンドアロンインスタンスをインストールします。詳細については、ハイ アベイラビリティの仕組み （24 ページ）を参照してください。プライマリインスタンスとスタンバイインスタンスは、Cinder ボリュームまたはレプリケーションベースのボリューム（DRBD）に接続できます。

ESC HA アクティブ/スタンバイを展開する際、次の展開メカニズムを利用できます。

- 内部ストレージ：ESC HA アクティブ/スタンバイが内部ストレージを使用して構成されている場合、プライマリインスタンスとスタンバイインスタンスは個別のデータベースを備え、常に同期されます。このソリューションでは、ESC HA アクティブ/スタンバイはデータベースレプリケーションを利用して設計され、DRBD はディスクレベルのレプリケーション用ツールとして使用されます。プライマリインスタンスのデータベースは、スタンバイインスタンスのデータベースにも同時にデータを伝送するため、外部ストレージを必要としません。プライマリインスタンスで障害が発生した場合、スタンバイインスタンスには、プライマリインスタンスのロールと、同期されたスタンバイインスタンス固有のデータベースが割り当てられます。

  ESC HA アクティブ/スタンバイは、内部ストレージを利用して展開されます。ESC インスタンスは仮想 IP アドレス（kad_vip 引数）および vrrp インスタンスのインターフェイス（kad_vif 引数）で応答し、プライマリ ESC インスタンスを選択します。信頼性の高いハートビートネットワークを確立するには、プライマリおよびスタンバイ ESC インスタンスを異なる物理ホスト上に配置することが推奨されます。ESC インスタンス間の物理リンクの信頼性（ネットワークインターフェイスの結合など）を考慮することもできます。

- 外部ストレージレプリケーション：このタイプのアーキテクチャでは、ESC HA アクティブ/スタンバイが DRBD を利用して構成されます。プライマリインスタンスとスタンバイインスタンスの両方が、2 つの外部ストレージ （OpenStack Cinder ボリューム）にデータを保存します。各 ESC ノードは Cinder ボリュームによって接続され、ESC データファイルは Cinder ボリュームに保存されます。2 つの ESC ノードのデータは、DRBD で提供されるデータベース レプリケーション メカニズムを利用して同期されます。

HA アクティブ/スタンバイのオプションの違いを次の表に示します。

	内部ストレージベースの ESC HA アクティブ/スタンバイ	外部ストレージレプリケーションベースの ESC HA アクティブ/スタンバイ
データ共有方法	HA アクティブ/スタンバイノード間のデータレプリケーション	2 つの外部ストレージ（Cinder ボリューム）間のデータレプリケーション
インストール方法	インストール後の設定 bootvm のインストール	bootvm のインストール
VIM 対応	OpenStack、VMware、KVM	OpenStack のみ
依存条件	VIM に依存しない	OpenStack Cinder に依存
利点	• 特定の VIM コンポーネントへの依存関係がありません。 • 共有ストレージを必要としないため、市販のハードウェアから HA アクティブ/スタンバイクラスタを柔軟に構築できます。	• データベース レプリケーション メカニズムを利用してデータが同期されます。 • 2 つの Cinder ボリュームが外部ストレージとして使用され、ESC ノードに接続されます。
制限事項	二重障害が発生すると（両方の ESC ノードで問題が発生した場合）、データの一貫性に影響を与える可能性があります。	二重障害が発生すると（両方の ESC ノードで問題が発生した場合）、データの一貫性に影響を与える可能性があります。

内部ストレージを利用した高可用性アクティブ/スタンバイモードの ESC の展開

プライマリおよびスタンバイインスタンスで ESC インスタンスを起動する場合は、次の *bootvm.py* コマンド引数を指定して、内部ストレージに ESC HA アクティブ/スタンバイを展開する必要があります。

- kad_vip

（注） ESC HA アクティブ/スタンバイが展開されると、エンドユーザは *kad_vip* 引数を使用してプライマリ ESC インスタンスにアクセスできます。

- kad_vif
- ha_node_list

これらの引数を指定すると、*bootvm.py* コマンドを使用して、OpenStack で内部ストレージを自動設定できます。*bootvm.py* コマンド引数の使用に関する詳細は、「付録 A：Cisco Elastic Service Controller のインストーラ引数」を参照してください。

ESC HA アクティブ/スタンバイインスタンスを展開するには、両方のノードの bootvm スクリプトで次の引数を指定します。

```
ON HA NODE 1:

$ ./bootvm.py <ESC_HA_Node1>\
--user_pass <username>:<password>\
--user_confd_pass <username>:<password>\
--gateway_ip <default gateway IP address>\
--net <network name1>\
--ipaddr <static ip address>
--image <image_name>\
--avail_zone nova:<openstack zone>\
--ha_node_list=<ESC_HA_NODE1_IP> <ESC_HA_NODE2_IP>\
--db_volume_id <cinder volume id>\
--kad_vip <virtual IP address>\
--kad_vif <VRRP_Interface_Instance>\
--ha_mode drbd


ON HA NODE 2:

$ ./bootvm.py <ESC_HA_Node2>\
--user_pass <username>:<password>\
--user_confd_pass <username>:<password>\
--gateway_ip <default gateway IP address>\
--net <network name1>\
--ipaddr <static ip addresses>\
--image <image_name>\
--avail_zone nova:<openstack zone>\
--ha_node_list=<ESC_HA_NODE1_IP> <ESC_HA_NODE2_IP>\
--db_volume_id <cinder volume id>\
--kad_vip <virtual IP address>\
--kad_vif <VRRP_Interface_Instance>\
--ha_mode drbd
```

または

escadm ツールを使用して、各スタンドアロン型 ESC VM で ESC HA アクティブ/スタンバイのパラメータを再設定することもできます。ESC HA アクティブ/スタンバイを構成するには、「--ha_node_list、--kad_vip、--kad_vif」の 3 つのパラメータが必要です。

（注） 次のコマンドを実行してHAアクティブ/スタンバイを構成する前に、両方のインスタンスがスタンドアロン ESC VM の正常性チェックに合格していることを確認してください。

次に例を示します。

```
$ sudo bash
$ escadm ha set --ha_node_list='<ESC_HA_NODE1_IP> <ESC_HA_NODE2_IP>' --kad_vip <virtual
 IP address> --kad_vif <VRRP_Interface_Instance>
$ sudo escadm reload
$ sudo escadm restart
```

外部ストレージレプリケーションを利用した高可用性アクティブ/スタンバイモードの ESC の展開

外部ストレージレプリケーションを利用したESC HAアクティブ/スタンバイでは、データベースストレージに 2 つの Cinder ボリュームが必要です。

始める前に

- 両方の ESC インスタンスが接続するネットワークおよび IP アドレス
- HAアクティブ/スタンバイのスイッチオーバー用キープアライブインターフェイスおよび仮想 IP

手順

ステップ 1 OpenStack に 2 つの Cinder ボリュームを作成します。Cinder ボリュームサイズは 3 GB に設定する必要があります。

```
$ cinder create --display-name  cindervolume_name_a[SIZE]
$ cinder create --display-name  cindervolume_name_b[SIZE]
```

ステップ 2 作成した Cinder ボリュームのステータスを確認し、展開用の UUID を見つけます。

```
$ cinder list
```

ステップ 3 ESC HAアクティブ/スタンバイインスタンスを展開します。両方のノードで、次の引数を指定した bootvm スクリプトを使用します。

```
ON HA NODE 1:

$ ./bootvm.py <ESC_HA_Node1>\
--user_pass <username>:<password>\
--user_confd_pass <username>:<password>\
--gateway_ip <default gateway IP address>\
--net <network name1>\
--ipaddr <static ip address>\
--image <image_name>\
--avail_zone nova:<openstack zone>\
```

```
--kad_vip <virtual IP address>\
--kad_vif <VRRP_Interface_Instance>\
--ha_node_list=<ESC_HA_NODE1_IP> <ESC_HA_NODE2_IP>\
--db_volume_id <cinder volume id>\
--ha_mode drbd_on_cinder

ON HA NODE 2:

$ ./bootvm.py <ESC_HA_Node2>\
--user_pass <username>:<password>\
--user_confd_pass <username>:<password>\
--gateway_ip <default gateway IP address>\
--net <network name1>\
--ipaddr <static ip address>\
--image <image_name>\
--avail_zone nova:<openstack zone>\
--kad_vip <virtual IP address>\
--kad_vif <VRRP_Interface_Instance>\
--ha_node_list=<ESC_HA_NODE1_IP> <ESC_HA_NODE2_IP>\
--db_volume_id <cinder volume id>\
--ha_mode drbd_on_cinder
```

ステップ 4 両方の VM が再起動されると、ESC VM のキープアライブ状態は、一方の ESC VM はマスターで、もう一方はバックアップになる必要があります。ESC HA アクティブ/スタンバイの状態を確認するには、*$ sudo escadm status --v* コマンドを使用します。

ノースバウンド インターフェイス アクセスの設定

ESC HA アクティブ/スタンバイを設定する場合は、HA アクティブ/スタンバイペアに仮想エニーキャスト IP アドレスを指定することもできます。ノースバウンドインターフェイスおよびサービスポータルは、仮想エニーキャスト IP アドレスを使用して ESC プライマリ HA アクティブ/スタンバイインスタンスにアクセスします。ESC HA アクティブ/スタンバイを展開する場合は、*./bootvm.py* スクリプトで次の引数を使用します。

- --ha_node_list
- --kad_vip
- --kad_vif

これらの引数のさらなる詳細については、「**付録 A**：**Cisco Elastic Services Controller インストーラの引数**」のセクションを参照してください。

ここでは、複数のインターフェイスを使用して ESC HA アクティブ/スタンバイを設定し、仮想エニーキャスト IP アドレスを設定する方法について説明します。

複数のインターフェイスを使用した ESC HA アクティブ/スタンバイの設定

データ同期と VNF モニタリングのために、ネットワーク インターフェイスで DRDB 同期と VRRP ハートビート ブロードキャストを使用して ESC HA アクティブ/スタンバイを設定できます。追加のネットワーク インターフェイスを使用して、ノースバウンドアクセス用の仮想 IP を割り当てることができます。ESC HA アクティブ/スタンバイノードで複数のインターフェ

イスを設定するには、--ha_node_list、--kad_vip、--kad_vif引数を使用して、これらの複数のネットワーク インターフェイス設定を指定します。これらの引数の詳細については、「**付録 A**：**Cisco Elastic Services Controller インストーラの引数**」のセクションを参照してください。

（注） KeepAlivedは、IPv6 VRRPインスタンスで単一のIPv4 VIPアドレスをサポートしていません。

設定手順の例を次に示します。

```
./bootvm.py <esc_ha1> \
--user_pass <username>:<password>
--user_confd_pass <username>:<password>
--image <image_id> \
--net <net-name> \
--gateway_ip <default_gateway_ip_address> \
--ipaddr <ip_address1> <ip_address2> \
--ha_node_list < IP addresses HA nodes1> < IP addresses for HA nodes2> \
--kad_vip <keepaliveD VIP of the HA nodes and the interface for keepaliveD VIP> \  (for
 example: --kad_vip 192.0.2.254:eth2)
--kad_vri <virtual router id of vrrp instance>
--kad_vif <virtual IP of the HA nodes or the interface of the keepalived VRRP> \ (for
example: --kad_vif eth1 )
--ha_mode <HA installation mode> \
--route <routing configuration> \  (for example:192.0.2.254/24:192.168.0.1:eth1 )
--avail_zone nova:<openstack zone> \
```

同様に、3 つのネットワーク インターフェイスを ESC HA アクティブ/スタンバイノードに設定できます。次に、3 つのインターフェイス設定の例と前提条件を示します。

- ネットワーク 1 は、ノースバウンド接続に使用される IPv6 ネットワークです。ESC VIP はこのネットワークに割り当てられ、Orchestrator は ESC VIP を使用して要求を ESC に送信します。
- ネットワーク 2 は、ESC 同期トラフィック（DRDB 同期）と VRRP ハートビートに使用される IPv4 ネットワークです。このネットワークは、OpenStack 接続および VNF モニタリングにも使用されます。
- ネットワーク 3 は、管理に使用される別の IPv4 ネットワークです。SA、rsyslog などでは、このネットワークを使用して ESC を管理できます。

./bootvm.py esc-ha-0 --image ESC-2_2_x_yyy --net network-v6 network --gateway_ip 192.168.0.1 --ipaddr 2001:cc0:2020::fa 192.168.0.239 192.168.5.239 --ha_node_list 192.168.0.239 192.168.0.243 --kad_vip [2001:cc0:2020::fc/48]:eth0 --kad_vif eth1 --ha_mode drbd --route 172.16.0.0:eth1 --avail_zone nova: zone name

./bootvm.py esc-ha-1 --image ESC-2_2_x_yyy --net network-v6 network lab-net-0 --gateway_ip 192.168.0.1 --ipaddr 2001:cc0:2020::fa 192.168.0.239 192.168.5.239 --ha_node_list 192.168.0.239 192.168.0.243 --kad_vip [2001:cc0:2020::fc/48]:eth0 --kad_vif eth1 --ha_mode drbd --route 172.16.0.0:eth1 --avail_zone nova: zone name

ESC HA アクティブ/スタンバイ仮想 IP アドレスの設定

このオプションでは、kad_vip 引数の値は仮想 IP である必要があります。これにより、サービスポータルとノースバウンドがプライマリ ESC にアクセスし、仮想 IP（VIP）を介して ESC HA アクティブ/スタンバイサービスに要求を送信できます。

ノースバウンドと両方の ESC HA アクティブ/スタンバイノードが同じネットワークにある場合は、仮想 IP（VIP）を介して直接接続できます。ノースバウンドが ESC HA アクティブ/スタンバイと同じネットワーク上にない場合は、次の手順を使用して、フローティング IP を ESC HA アクティブ/スタンバイ VIP に割り当てます。

1. ESC の kad_vip の接続先と同じネットワークに VIP アドレス（kad_vip）を使用してポートを作成します。

   ```
   neutron port-create network --name network_vip --fixed-ip
   subnet_id=network-subnet,ip_address=192.168.0.87
   ```

2. ESC HA アクティブ/スタンバイを展開します。「OpenStack で ESC をインストール」の「**高可用性アクティブ/スタンバイの設定**」のセクションを参照してください。

（注） *kad_vip* が上記で作成したポートと同じ IP アドレスであることを確認してください。

3. 上記で作成したポートにフローティング IP を関連付けます。最初の uuid はフローティング IP ID で、2 つ目の uuid はポート ID です。

   ```
   neutron floatingip-associate <floating IP> <port ID>
   ```

   フローティング IP を介して HA アクティブ/スタンバイにアクセスすると、ESC プライマリノードに接続されます。

4. ポータルアクセスの場合、ブラウザからキープアライブネットワークにアクセスできること、および仮想 IP がプライマリノードのポータルにアクセスするための IP アドレスであることを確認してください。

   たとえば、VIP が 192.0.2.254 の場合、https://192.0.2.254:9001/ で ESC HA アクティブ/スタンバイポータルにアクセスします。

BGP を使用した ESC L3 HA アクティブ/スタンバイの設定

ESC HA アクティブ/スタンバイの BGP を設定するには、次の 2 つのオプションがあります。

1. BGP を使用した ESC HA アクティブ/スタンバイ L3 の直接起動
2. 既存 ESC HA アクティブ/スタンバイペアからの POST 設定の使用

ESC HA アクティブ/スタンバイの BGP を設定するには、次のネットワークパラメータが必要です。

- BGP リモート IP
- BGP エニーキャストルーティングのインターフェイスの IP

- ルーティング設定の BGP ローカル AS 番号
- ルーティング設定用の BGP リモート AS 番号
- BGP ルーティングの設定
- --bgp_local_ip
- --bgp_local_router_id

（注） ネイバーを使用して BGP ルータを設定し、再起動する必要があります。ルータがエニーキャスト IP に ping できることを確認します。

BGP ルータで、2 つのネイバーを設定します。次の BGP 設定は、Bird ルータ向けに設計されています。この設定は、ルータ固有です。ルータのタイプごとに、手順は異なります。

次の設定は、bootvm コマンドによって指定されます。

```
protocol bgp E3 from EXABGP {
  neighbor 198.18.42.222 as 65012;
}

protocol bgp E4 from EXABGP {
  neighbor 198.18.61.222 as 65011;
}
```

BGP オプションを使用した ESC VM の起動

```
   ######################################################################
   #   ESC on bgp-001.novalocal is in PRIMARY state.
   ######################################################################

[admin@bgp-001 ~]$ health.sh
============== ESC HA (PRIMARY) with DRBD =================
vimmanager (pgid 4007) is running
monitor (pgid 4135) is running
mona (pgid 4167) is running
drbd (pgid 0) is primary
snmp (pgid 5375) is running
etsi is disabled at startup
pgsql (pgid 4586) is running
keepalived (pgid 3068) is running
portal (pgid 5315) is running
confd (pgid 4417) is running
filesystem (pgid 0) is running
escmanager (pgid 4615) is running
========================================
ESC HEALTH PASSED
[admin@bgp-001 ~]$


   ######################################################################
   #   ESC on bgp-002.novalocal is in BACKUP state.
   ######################################################################

[admin@bgp-002 ~]$ health.sh
============== ESC HA (BACKUP) with DRBD =================
vimmanager is stopped
monitor is stopped
```

```
mona is stopped
drbd (pgid 0) is backup
snmp is stopped
etsi is disabled at startup
pgsql is stopped
keepalived (pgid 3069) is running
portal is stopped
confd is stopped
filesystem is stopped
escmanager is stopped
======================================
ESC HEALTH PASSED
[admin@bgp-002 ~]$
```

BGP POST 設定には次の値を使用します。

```
./bootvm.sh <NETWORK_VM_name> \
--image <ESC_image> \
--ipaddr <static_IP_address1> <IP_address2> <IP_address_3>\
--gateway_ip <gateway IP address of NETWORK> \
--net <net_id1> <net_id2> <net_id3> \
--network_params_file <network_params_file> \
--host_mapping_file <host_mapping_file> \
--avail_zone <openStack_zone> \
--bgp_remote_ip <BGP_remote_IP_address> \
--bgp_local_as <BGP_local_AS_#> \
--bgp_remote_as <BGP_remote_AS_#>\
--bgp_local_router_id <local_BGP_reouter_id> \
--bgp_anycast_ip <BGP_anycast_IP> \
--bgp_md5 <BGP_MD5>
```

それぞれの説明は次のとおりです。

```
--ip_addr: ----> the local IP address of the ESC VM
--net: ----> the network id(s) in OpenStack that ESC will connect to.
--bgp_anycast_ip:  ----> the IP address that NCS will communicate with
--bgp_remote_ip:  ----> this IP address of the external router that ESC will peer with
--bgp_local_as:  ----> local AS for the ESC "router"
--bgp_remote_as:   ----> AS number for the external router ESC will peer with
--bgp_local_router_id: ----> id for the esc "router"
--bgp_md5:    ----> optional - md5 to be used to pair with external router
```

BGP HA アクティブ/スタンバイ ポスト コンフィギュレーションの設定

1. HA のアクティブ/スタンバイインスタンスごとに、ネットワーク インターフェイス ファイルを作成します。

   ```
   # cat /etc/sysconfig/network-scripts/ifcfg-lo:2
   IPV6INIT='no'
   IPADDR='10.0.124.124'                    <------------------ bgp anycast IP
   BROADCAST='10.0.124.255'
   NETWORK='10.0.124.0'
   NETMASK='255.255.255.0'
   DEVICE='lo:2'
   ONBOOT='yes'
   NAME='loopback'
   ```

2. HA のアクティブ/スタンバイインスタンスごとに、次のようにします。

   ```
   Bring lo:2 up
   # ifup lo:2
   ```

ESC HA アクティブ/スタンバイの BGP を設定するには、次に示すように、ESC 仮想マシンに escadm ツールを使用します。

```
$ sudo bash
# escadm bgp set --local_ip LOCAL_IP --anycast_ip ANYCAST_IP --remote_ip REMOTE_IP
--local_as LOCAL_AS --remote_as REMOTE_AS
--local_router_id LOCAL_ROUTER_ID
# escadm reload
# reboot
```

例：

```
[root@bgp-001 admin]# escadm bgp set --local_ip 198.18.42.124 --anycast_ip 10.0.124.124
 --remote_ip 192.168.0.2 --local_as 65124 --remote_as 65000 --local_router_id 198.18.42.124

[root@bgp-002 admin]# escadm bgp set --local_ip 198.18.42.125 --anycast_ip 10.0.124.124
 --remote_ip 192.168.0.2 --local_as 65114 --remote_as 65000 --local_router_id 198.18.42.125
```

BGP ルータの設定

BGP ルータを設定するには、BGP ルータにログインして BGP エニーキャストルーティングを設定します。次のパラメータが必要です。

<Router_AS_#> は上記の *--bgp_remote_as* と同様です

<Esc_ip_address> は、BGP アドバタイズメント用に設定された ESC VM の IP アドレスである必要があります。

<ESC_AS_#> は上記の *--bgp_local_as* と同様です

```
configure

router bgp <Router_AS_#>

neighbor <ESC_IP_address>

remote-as <ESC_AS_#>
  address-family ipv6 unicast
   route-policy anycast-in in
   route-policy anycast-out out

route-policy anycast-in
  pass
end-policy

route-policy anycast-out
  drop
end-policy

commit
```

特記事項

- **ESC HA アクティブ/スタンバイ**
  - HA アクティブ/スタンバイのフェールオーバーには 2 ～ 5 分ほどかかります。スイッチオーバーの間、ESC サービスは使用できなくなります。
  - トランザクション中にスイッチオーバーがトリガーされると、すべての未完了のトランザクションがドロップされます。要求に対する ESC からの応答が受信されない場合、ノースバウンドから要求が再送信される必要があります。

- **外部ストレージ**：
  - プライマリ ESC インスタンスが OpenStack コマンドによって中断された場合は、スイッチオーバーがトリガーされますが、Cinder ボリュームは新しいプライマリ ESC インスタンスに接続されません。これは ESC HA アクティブ/スタンバイの有効な使用例ではありません。
- **内部ストレージ**
  - HA アクティブ/スタンバイソリューションを確立するには、2 つの ESC インスタンスを展開する必要があります。両方の ESC インスタンスが正常に展開され、相互に接続できる場合に、ESC HA アクティブ/スタンバイは動作を開始します。HA アクティブ/スタンバイのパラメータを使用して 1 つの ESC インスタンスのみを展開した場合、ESC インスタンスの状態は「マスターに切り替え中」のままになり、ピアに到達するまでサービスを提供できなくなります。
  - スプリットブレインのシナリオは、可能性は非常に低いとはいえ、ESC HA アクティブ/スタンバイソリューションでも発生する場合があります。
- **ETSI 固有の注意事項**

  ESC は、欧州電気通信標準化機構（ETSI）によって定義された ETSI MANO ノースバウンド API を NFV 管理およびオーケストレーションに対してサポートします。ETSI MANO API は、REST アーキテクチャに基づく別のプログラマチック インターフェイスです。詳細については、『*Cisco Elastic Services Controller User Guide*』の「ETSI MANO Compliant Lifecycle Operations」を参照してください。HA アクティブ/スタンバイモードの ESC で ETSI サービスを有効にする場合は、次の注意事項を考慮してください。
  - *etsi-vnfm.properties* ファイル内の server.address の値は、仮想 IP（VIP）アドレスに設定する必要があります。この IP アドレスは、API コールバックを使用した ETSI サービスへの応答に使用できます。仮想 IP アドレスが指定されていない場合、ETSI サービスの起動に失敗することがあります。
  - ETSI VNFM サービスと escadm スクリプトは、security.user.name プロパティと security.user.password プロパティの値を生成して保持します。手動で変更しないでください。security.user.password がエンコードされます。

高可用性アクティブ/スタンバイのトラブルシューティング

- ネットワーク障害をチェックします。ネットワークに問題が発生している場合は、次の詳細情報をチェックする必要があります。
  - 割り当てられている IP アドレスは正しいもので、OpenStack 設定に基づいている必要があります。

- 各ネットワークインターフェイスのゲートウェイは、ピン可能である必要があります。

- トラブルシューティングの際には、次のログをチェックします。
  - */var/log/esc/escadm.log* の ESC 管理ログ
  - */var/log/esc/escmanager.log* の ESC マネージャのログ
  - */var/log/esc/elector-{pid}.log* の AA エレクタのログ

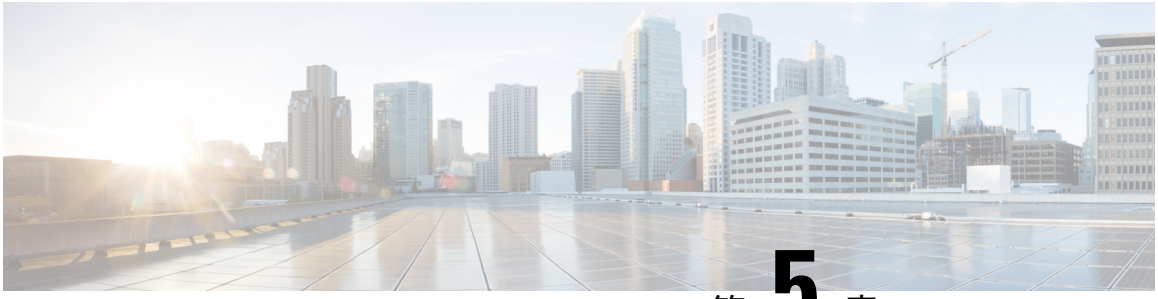
- 内部ストレージソリューションの DRBD（レプリケーションベース ESC HA アクティブ/スタンバイ）を確認します。
  - 次の DRBD 設定ファイルを確認します。

    ```
    /etc/drbd.d/esc.res
    ```

  - DRBD ログへのアクセス
  - `/var/log/messages|grep drbd`

- CLI を使用してログファイルを収集するには、すべての ESC ノードで次のコマンドを使用します。

```
sudo escadm log collect
```



第 5 章

Cisco Elastic Services Controller の高可用性アクティブ/アクティブの概要

この章は、次の項で構成されています。

Cisco Elastic Services Controller のアクティブ/アクティブ HA の概要

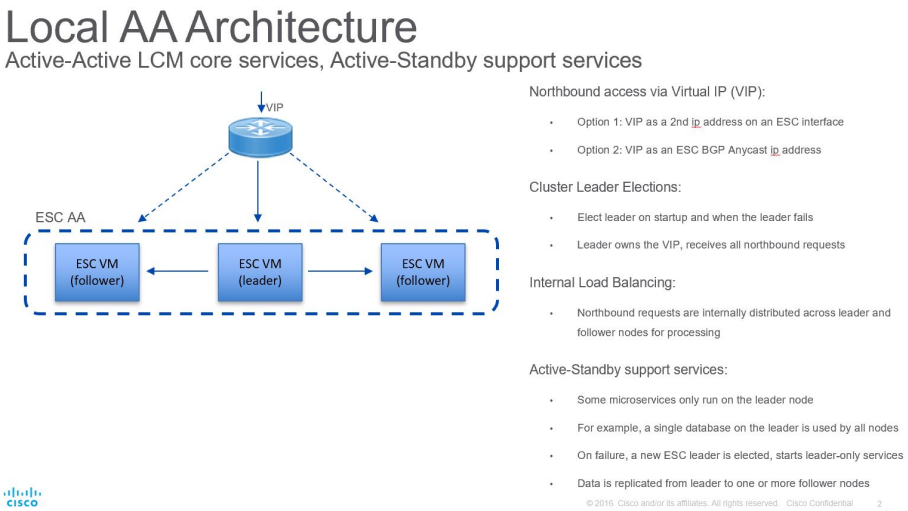
ESC は、アクティブ/アクティブモデルの形式で高可用性（HA）をサポートします。ESC アクティブ/アクティブ HA は、1 つのデータセンター内に 3 つの VM をクラスタとして備えています。。2 つのデータセンターがあります。2 つのデータセンターの内、1 つはアクティブなデータセンターとして機能し、もう 1 つはスタンバイとして機能しますESC アクティブ/アクティブ HA は、Openstack Heat テンプレートを使用して、3 つの VM クラスタをデータセンターに展開します。

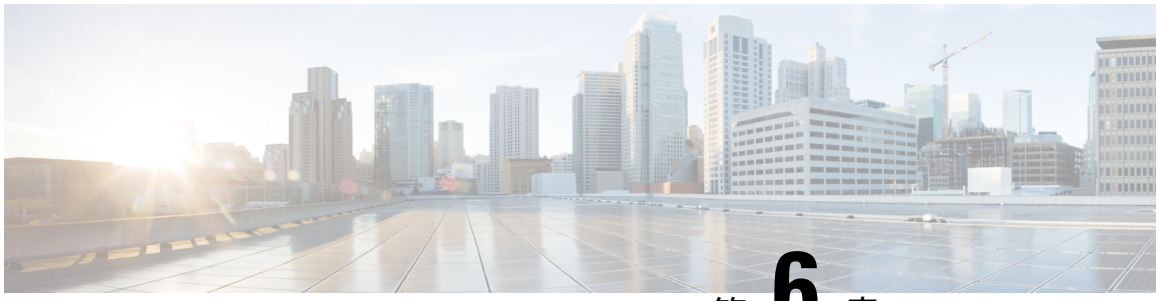
データセンターでは、ESC サービスが各 VM で稼働します。ただし、データセンターでクラスタリーダーとして稼働する ESC は 1 つだけです。DB サービスはクラスリーダーでのみ稼働します。他の 2 つの ESC VM では、ESC サービスがクラスタフォロワーとして稼働します。DB サービスは、ESC サービスリーダーの VM でのみアクティブになります。

DRBD は、ESC VM 間でデータを同期化します。3 つの ESC VM に搭載された ESC サービスは、アクティブな DB サービスに接続されます。リーダーのスイッチオーバーが発生すると、すべての ESC サービスは新たにアクティブになった DB サービスに接続されます。

ESC アクティブ/アクティブアーキテクチャ

図 *2 : Cisco Elastic Services Controller* のアクティブ/アクティブアーキテクチャ





第 6 章

アクティブ/アクティブ高可用性クラスタのインストール

この章は、次の項で構成されています。

アクティブ/アクティブ高可用性クラスタのインストール

アクティブ/アクティブ HA クラスタを設定するには、OpenStack で次のコマンドを実行します。このとき、`openrc my-server` は OpenStack の openrc で、`test` はスタック名にします。

```
source ~/elastic-services-controller/esc-bootvm-scripts/openrc my-server-42
openstack stack create test -t aa.yaml
```

スタックのステータスを確認するには、次のコマンドを使用します。

- `openstack stack list`
- `openstack stack show test`
- `openstack stack event list test`

スタックのステータスが `CREATE_COMPLETE` になると、VM に ssh 接続できます。`openstack stack show test` コマンドを実行すると、3 つの VM の IP アドレスのリストが表示されます。そのリストを使用して VM にアクセスできます。

（注） アクティブ/アクティブ HA クラスタは OpenStack にのみ展開されます。

ユーザ設定のセットアップ

ネットワークやサブネットの設定に基づき、スタティック IP または DHCP を使用して 一部のパラメータを設定し、IPアドレス、フレーバ、イメージ、パスワードなどを割り当てることができます。ESC クラスタが Heat テンプレート（`aa. yaml`）を使用してインスタンス化されている場合は、`aa-param yaml`、Openstack の Heat 環境ファイルでパラメータを設定できます。

ESC クラスタをインスタンス化するには、次の例を使用します。

```
openstack stack create name -t aa.yaml -e aa-params.yaml
```

以下は、環境テンプレートを使用してスタティック IP アドレスを指定し、ポートを設定する例です。

```
sample@my-server-39:~/aa4.5/apr15$ more aa-params.yaml
parameters:
  network_1_name: esc-net
  subnet_name: esc-subnet
  esc_1_ip: 172.23.0.228
  esc_2_ip: 172.23.0.229
  esc_3_ip: 172.23.0.230
```

aa. yaml でユーザ設定可能なパラメータを以下に示します。

```
parameters:
  network_1_name:
    type: string
    description: Name of the image
    default: esc-net
  subnet_name:
    type: string
    description: subnet name
  esc_1_ip:
    type: string
    description: static IP address of esc-1 VM.

  esc_2_ip:
    type: string
    description: static IP address of esc-2 VM.

  esc_3_ip:
    type: string
    description: static IP address of esc-3 VM.


resources:
  esc_1_port:
    type: OS::Neutron::Port
    properties:
      network_id: { get_param: network_1_name }
      fixed_ips: [ { "subnet": { get_param: subnet_name}, "ip_address": { get_param:
esc_1_ip } } ]

  esc_2_port:
    type: OS::Neutron::Port
    properties:
      network_id: { get_param: network_1_name }
      fixed_ips: [ { "subnet": { get_param: subnet_name}, "ip_address": { get_param:
esc_2_ip } } ]

  esc_3_port:
    type: OS::Neutron::Port
```

```
    properties:
      network_id: { get_param: network_1_name }
      fixed_ips: [ { "subnet": { get_param: subnet_name}, "ip_address": { get_param:
esc_3_ip } } ]

...omitting...
```

設定可能なイメージ、フレーバ、およびVM名プレフィックスを環境テンプレートから使用する例を以下に示します。

```
sample@my-server-39:~/aa4.5/apr15$ more aa-params.yaml
parameters:
  nameprefix: abc
  image_name: ESC-5_0_DEV_4
  flavor_name: m1.large
sample@my-server-39:~/aa4.5/apr15$
```

設定可能なイメージ、フレーバ、および vmnameprefix を Heat テンプレートから使用する例を以下に示します。

```
parameters:
  nameprefix:
    type: string
    description: Name prefix of vm
    default: helen
  image_name:
    type: string
    description: Name of the image
    default: ESC-5_0_DEV_4
  flavor_name:
    type: string
    description: Name of the image
    default: m1.large

  esc-1:
    type: OS::Nova::Server
    properties:
      name:
       str_replace:
         template: $nameprefix-esc-1
         params:
           $nameprefix : { get_param: nameprefix }
      image: { get_param: image_name }
      flavor: { get_param: flavor_name }
      ... omitting...
```

アクティブ/アクティブ高可用性クラスタのインストール後の検証

すべての ESC ノードを確認するには、次のコマンドを使用します。ここでは、すべての ESC ノードが各 VM を意味します。

```
sample@my-server-39:~$ openstack --insecure server list | grep abc
| 5ea6fc79-2b2a-4064-9c6a-a83d6b06c225 | abc-test-esc-3
              | ACTIVE  | esc-net=172.23.7.203
                                | ESC-5_0_DEV_13              | m1.large                |
| 10e165d9-5015-4b64-88fe-19e874e6e7c1 | abc-test-esc-1
              | ACTIVE  | esc-net=172.23.7.205
                                | ESC-5_0_DEV_13              | m1.large                |
```

```
| 35f6bad1-865f-4155-8411-d37e2616e079 | abc-test-esc-2
              | ACTIVE  | esc-net=172.23.7.204
                                | ESC-5_0_DEV_13              | m1.large                 |
```

リーダーノードを確認するには、いずれかのノードまたは VM に SSH で接続し、次のコマンドを実行します。

```
[admin@sample-test-esc-1 ~]$ sudo escadm elector dump
{
    "13078@sample-test-esc-3.novalocal:42143": {
        "state": "FOLLOWER",
        "location": "13078@test1-test-esc-3.novalocal:42143",
        "service": "esc_service"
    },
    "13053@sample-test-esc-2.novalocal:50474": {
        "state": "FOLLOWER",
        "location": "13053@sample-test-esc-2.novalocal:50474",
        "service": "esc_service"
    },
    "13187@sample-test-esc-1.novalocal:59514": {
        "state": "LEADER",
        "location": "13187@sample-test-esc-1.novalocal:59514",
        "service": "esc_service"
    }
}
```

アクティブ/アクティブ高可用性クラスタへのデフォルト VIM コネクタの追加

次の 2 つの方法で、3 ESC VM クラスタにデフォルトの VIM コネクタを追加できます。

1. 3 ESC VM クラスタが起動したら、Netconf インターフェイスで、次のコマンドを使用してデフォルトの VIM コネクタを追加します。`vim.xml` は、デフォルトの VIM コネクタの導入用ファイルです。

   ```
   [admin@name-esc-1 ~]$ esc_nc_cli --host db.service.consul --user admin --password
   <admin_password> edit-config vim.xml
   ```

2. デフォルトの VIM コネクタを設定するには、Heat テンプレート day0 ファイル内にデフォルト VIM コネクタの設定を追加する必要があります。aa-day0.yaml ファイルの cloud-config にある `write_files` セクションに次のブロックを追加します。3 ESC VM クラスタが起動すると、デフォルトの VIM コネクタが独自に作成されます。

   次に、Heat テンプレート day0 ファイルでデフォルトの VIM コネクタを設定する例を示します。

   ```
   - path: /opt/cisco/esc/esc-config/esc_params.conf
     content: |
       openstack.os_auth_url=http://10.85.103.38:35357/v3
       openstack.os_project_name=admin
       openstack.os_tenant_name=admin
       openstack.os_user_domain_name=default
       openstack.os_project_domain_name=default
       openstack.os_identity_api_version=3
       openstack.os_image_api_version=2
   ```

```
openstack.os_username=admin
openstack.os_password=password1
```

アクティブ/アクティブクラスタでの BGP の追加

BGP プロセスを開始するには、エニーキャスト IP を lo デバイスに追加します。これは、`sys-cfg. yaml` で設定できます。

次に例を示します。

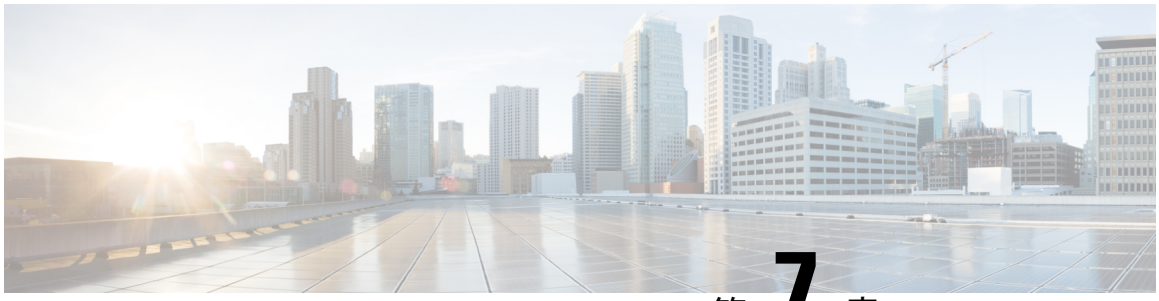
```
#cloud-config
write_files:
 - path: /etc/cloud/cloud.cfg.d/sys-cfg.yaml
   content: |
     network:
       version: 1
       config:
       - type: physical
         name: lo
         subnets:
         - type: static
           address: 172.23.188.188/23
```

Consul に関連付けるアドバタイズ IP を指定する必要があります。`esc-config.yaml` で、次のように追加します。

```
consul:
  advertise_addr: 172.23.1.149
```

BGP セクションを追加する例を以下に示します。

```
bgp:
  depend_on: elector:leader
  anycast_ip: 172.23.188.188/23
  local_as: '65001'
  local_ip: 192.168.1.11
  local_router_id: 192.168.1.11
  remote_as: '65000'
  remote_ip: 192.168.1.12
```



第 7 章

ESC アクティブ/アクティブ高可用性でのクラスタの管理

この章の内容は、次のとおりです。

ESCアクティブ/アクティブ高可用性でのクラスタの管理

ESC アクティブ/アクティブ HA でクラスタを管理するには、任意の ESC ノードで escadm コマンドを呼び出して、アクティブ/アクティブクラスタ内のすべてのノードで実行します。

クラスタレベルの呼び出しでサポートされているコマンドは次のとおりです。

- `escadm start`
- `escadm stop`
- `escadm geo start/stop`
- `escadm vim show`

前のコマンドをアクティブ/アクティブローカルクラスタ内のすべてのノードで実行するには、オプション `--cluster` を追加します。

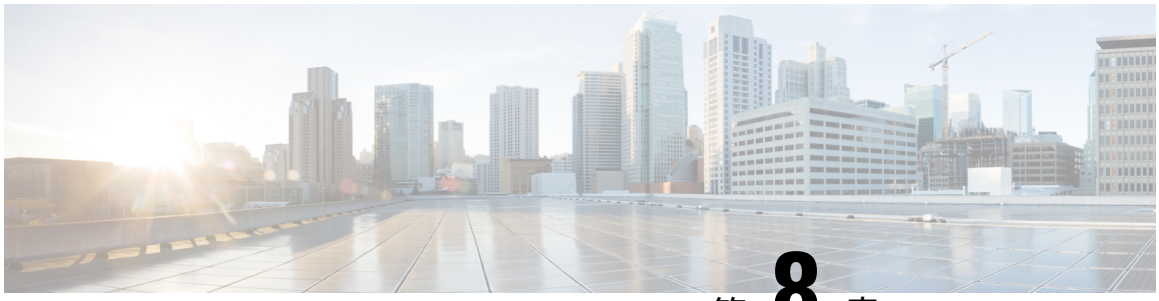
次に例を示します。

```
sudo escadm geo start --v -cluster
```

各ノードのコマンドと exit コードの出力は、実行結果とローカルノードの出力を区別するために、そのノードの IP アドレスとともに表示されます。

`escadm geo start --cluster` の例：

```
[root@name-geo-2-1 admin]# escadm geo start --cluster
192.168.1.13 # remote host
exit status : 0
Starting geo service: [OK]
192.168.1.12 # remote host
exit status : 0
Starting geo service: [OK]
Starting geo service: [OK] # output of the local node
```



第 8 章

アクティブ/アクティブ高可用性での GEO の設定

アクティブ/アクティブ高可用性での GEO の設定

ESC アクティブ/アクティブ HA は、1 つのデータセンターに 3 つの VM をクラスタとして備えています。2 番目のデータセンターは、GEO HA で構成されています。

GEO で事前定義されている 6 つのロールは次のとおりです。

1. `init`：geo サービスの初期ロールを意味します。
2. `pre_primary`
3. `primary`
4. `pre_secondary`
5. `secondary`
6. `unknown`：consul に到達できない場合に使用されます。

GEO は、ロールを別のロールに変更できます。移行は `esc-config. yaml` で定義されます。各移行は、次の 3 つの部分に分かれています。

- `from`：現在のロール
- `goto`：移行先のロール
- `condition`：GEO がロールを変更する条件

移行条件

A/A HA GEO の起動時、プライマリデータセンターの状態は、`init`、`pre_primary`、`primary` の順に移行する必要があります。一方、セカンダリデータセンターの場合は、init、pre_secondary、secondary の順に状態が移行する必要があります。プライマリデータセンターとセカンダリデータセンターの両方で、すべての ESC VM の正常性チェックに合格した場合、ESC A/A HA GEO は稼働中です。使用する準備が整いました。

条件関数

サポートされているすべての条件関数を次に示します。

1. `return`：何も実行せずに引数を返します。
2. `and`：すべての引数が true の場合に true を返します。
3. `or`：引数のいずれかが true の場合に true を返します。
4. `len`：引数の長さを返します。
5. `equals`：すべての引数が等しい場合に true を返します。
6. `true`：python の真理値に対して args をテストできる場合に true を返します。
7. `false`：「true」の逆を意味します。

プライマリデータセンターでの GEO 設定のサンプルを次に示します。

```
on_init: consul start
 on_primary: start
 on_secondary: stop
 on_stop: consul stop
 startup: manual
 transitions:
 - condition:
     return:
       and:
       - equals:
         - len: service1
         - 3
       - equals:
         - len: service2
         - 3
     rise: 3
     service1:
       dc: dc1
       name: consul_agent
       passing: true
       type: service
     service2:
       dc: dc2
       name: geo
       passing: true
       type: service
   from: init
   goto: primary
 - condition:
     fall: 2
     return:
       equals:
```

```
        - len: service
        - 3
      service:
        dc: dc1
        name: consul_agent
    from: primary
    goto: secondary
```

セカンダリデータセンターでの GEO 設定のサンプルを次に示します。

```
 on_init: consul start
  on_primary: start
  on_secondary: stop
  on_stop: consul stop
  startup: manual
  transitions:
  - condition:
      return:
        and:
        - equals:
          - len: service1
          - 3
        - equals:
          - len: service2
          - 3
      rise: 3
      service1:
        dc: dc1
        name: consul_agent
        passing: true
        type: service
      service2:
        dc: dc2
        name: geo
        passing: true
        type: service
    from: init
    goto: secondary
  - condition:
      fall: 2
      return:
        equals:
        - len: service
        - 3
      service:
        dc: dc1
        name: consul_agent
    from: secondary
    goto: primary
```

GEO サービスの確認

アクティブ/アクティブ GEO HA を開始するには、次のコマンドを実行します。

```
escadm geo start
```

GEO ステータスを確認するには、次のコマンドを使用します。

```
[root@test-geo3-ha-1 esc-scripts]# escadm geo status
geo (pgid 3745) is primary
```

現在のデータセンターの GEO サービスを確認するには、次のコマンドを使用します。

```
[root@test-geo3-ha-1 esc-scripts]# escadm geo dump
{
    "37410@test-geo3-ha-2.novalocal:44793": {
        "role": "primary",
        "location": "37410@test-geo3-ha-2.novalocal:44793",
        "service": "geo"
    },
    "43391@test-geo3-ha-3.novalocal:52459": {
        "role": "primary",
        "location": "43391@test-geo3-ha-3.novalocal:52459",
        "service": "geo"
    },
    "37898@test-geo3-ha-1.novalocal:38841": {
        "role": "primary",
        "location": "37898@test-geo3-ha-1.novalocal:38841",
        "service": "geo"
    }
}
```

データセンター内のすべての GEO サービスを確認するには、次のコマンドを使用します。

```
[root@test-geo4-ha-1 admin]# escadm geo dump --all
{
    "3745@test-geo4-ha-1.novalocal:36760": {
        "role": "primary",
        "location": "3745@test-geo4-ha-1.novalocal:36760",
        "service": "geo"
    },
    "3742@test-geo4-ha-6.novalocal:42362": {
        "role": "secondary",
        "location": "3742@test-geo4-ha-6.novalocal:42362",
        "service": "geo"
    },
    "3738@test-geo4-ha-3.novalocal:51936": {
        "role": "primary",
        "location": "3738@test-geo4-ha-3.novalocal:51936",
        "service": "geo"
    },
    "3713@test-geo4-ha-4.novalocal:37604": {
        "role": "secondary",
        "location": "3713@test-geo4-ha-4.novalocal:37604",
        "service": "geo"
    },
    "3710@test-geo4-ha-2.novalocal:44450": {
        "role": "primary",
        "location": "3710@test-geo4-ha-2.novalocal:44450",
        "service": "geo"
    },
    "3714@test-geo4-ha-5.novalocal:34875": {
        "role": "secondary",
        "location": "3714@test-geo4-ha-5.novalocal:34875",
        "service": "geo"
    }
}
```

アクティブ/アクティブ GEO HA の障害インジェクションの制限

ESC アクティブ/アクティブ GEO HA では、一方向の GEO HA フェールオーバー機能がメンテナンスウィンドウで強化されており、GEO HA を正常な状態に戻すことができます。

GEO フェールオーバーが発生し、ESC VM が異常な状態になった場合は、次の手順に従い、手動による介入によって ESC A/A GEO HA を正常な状態に戻します。

手順

ステップ 1 障害が発生し、GEO スイッチによってデータセンター 2（DC2）に切り替わる原因となったデータセンター 1（DC1）の問題を解決します。

ステップ 2 DC1 と DC2 の少なくとも 2 つのノードで Consul が 稼働していることを確認します。

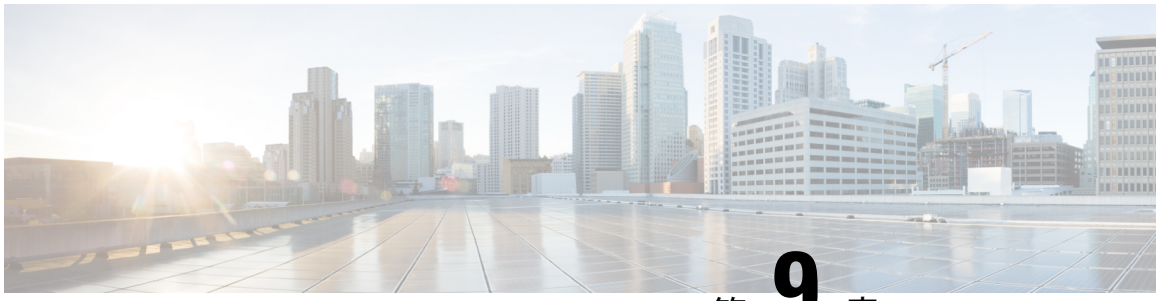
ステップ 3 Consul を稼働中のノードが DC1 に 2 つ以上ある場合、DC2 で `sudo escadm geo replicate -all` コマンドを実行します。

ステップ 4 6 つの ESC VM すべてで `sudo escadm stop` コマンドを実行します。

ステップ 5 6 つの ESC VM すべてで `sudo escadm geo restart` コマンドを実行します。

次のタスク

（注） GEO HA が DC2 にフェールオーバーした後、ESC は DC1 の ESC VM で行われた操作をサポートしません。



第 9 章

ESCアクティブ/スタンバイおよびアクティブ/アクティブHAデータレプリケーションのDRBD暗号化

この章は、次の項で構成されています。

ESC HA データレプリケーションの DRBD 暗号化

ESC は DRBD を使用して、HA クラスタ環境のさまざまなノード間でデータレプリケーションを実行します。DRBD は、クラスタノード上の既存のローカルブロックデバイスを介して論理ブロックデバイスを階層化します。

プライマリノードに書き込まれたデータは、下位層のブロックデバイスに転送され、同時にセカンダリノードに伝送されます。現在、ESC は DRBD デバイスを `/opt/cisco/esc/esc_database` に直接マウントします。

例：

```
# df
Filesystem            1K-blocks    Used Available Use% Mounted on
devtmpfs                2961760       0   2961760   0% /dev
tmpfs                   2972164       4   2972160   1% /dev/shm
tmpfs                   2972164    8748   2963416   1% /run
…
tmpfs                    594436       0    594436   0% /run/user/1004
/dev/mapper/esc_crypt   3028620   57212   2797848   3% /opt/cisco/esc/esc_database
```

ブロックデバイスの暗号化では、ブロックデバイスからの書き込み/読み取り時にデータが透過的に暗号化または復号化されます。基盤となるブロックデバイスは、暗号化されたデータのみを認識します。

`dm-crypt/LUKS` レイヤによってセキュリティが強化されて、ファイルシステムと DRBD デバイス間で DRBD パーティション内のデータが暗号化されます。LUKS（Linux Unified Key Setup）は、ブロックデバイスの暗号化向けの仕様です。

DRBD 暗号化を使用した ESC HA

次の bootvm コマンドは、DRBD が暗号化された状態で ESC HA を起動します。

bootvm.py を使用して DRBD LUKS の暗号化を選択します。ESC VM インスタンスに渡された場合、ESC day-0 user-data/esc-config. yaml と同等の結果になるのに 4 つのバリエーションがあります。

```
bootvm.py --fs_encryption_type luks --fs_luks_key_prompt
bootvm.py --fs_encryption_type luks --fs_luks_key 'LuksKeyValue'
=> injects the luks key into default file location /opt/cisco/esc/esc-config/luks_key

bootvm.py --fs_encryption_type luks --file
root:0400:/opt/cisco/esc/esc-config/luks_key:path-to-local-luks-key-file
=> injects a local file containing the luks key
```

次のコマンドは、ESC VM ファイルシステム上の別のパスにある luks キーファイルを管理するための高度な使用法を示しています。

```
bootvm.py --fs_encryption_type luks --fs_luks_key_file path-on-esc-vm-luks-key-file
--fs_luks_key_prompt
bootvm.py --fs_encryption_type luks --fs_luks_key_file path-on-esc-vm-luks-key-file
--fs_luks_key 'LuksKeyValue'
=> injects the luks key into a different file location

bootvm.py --fs_encryption_type luks --fs_luks_key_file path-on-esc-vm-luks-key-file
--file root:0400:path-on-esc-vm-luks-key-file:path-to-local-luks-key-file
=> injects the luks key as read from a local file into a different file location
```

Heatテンプレートを使用してESCアクティブ/アクティブ構成を展開するように、カスタムユーザデータを使用して ESC をインストールする場合は、次のコマンドを使用します。

luks キーは、day-0 ファイルや属性として esc-config. yaml / filesystem の下に指定します。

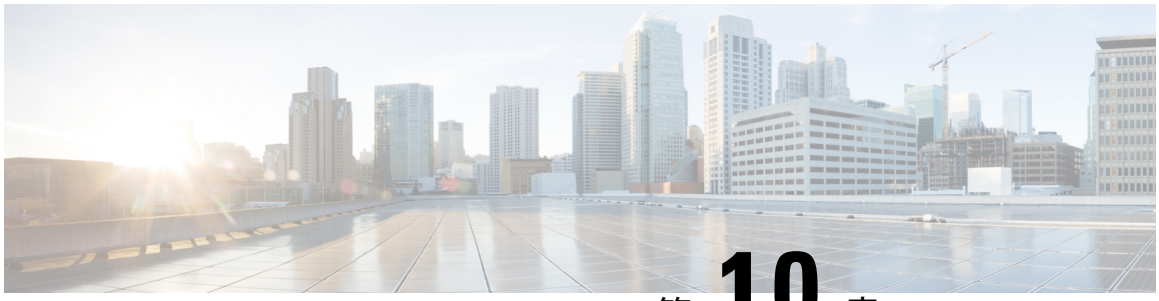
luks キーを base64 としてエンコードします。

```
base64 <<<'LuksKeyValue'
THVrc0tleVZhbHVlCg==
```

次に、以前の luks キーをユーザデータ/クラウド コンフィギュレーション ファイルに挿入します。

```
write_files:
 - path: /opt/cisco/esc/esc-config/luks_key
   owner: root:root
   permissions: '400'
   encoding: b64
   content: THVrc0tleVZhbHVlCg==

 - path: /opt/cisco/esc/esc-config/esc-config.yaml
   owner: root:esc-user
   permissions: '0640'
   content: |
     resources:
       filesystem:
         depend_on: drbd:master
         encryption_type: luks
         luks_key_file: /opt/cisco/esc/esc-config/luks_key
```



第 10 章

ESCアクティブ/アクティブ高可用性のアップグレード

この章は、次の項で構成されています。

ESC アクティブ/アクティブ高可用性のアップグレード

Cisco Elastic Service Controller のアクティブ/アクティブ HA は、ローカルアクティブ/アクティブからアクティブ/アクティブへの簡易アップグレードをサポートしています。

ローカルアクティブ/アクティブからアクティブ/アクティブへの簡易アップグレード

手順

ステップ 1 データベースをバックアップします。詳細については、データベースのバックアップ （55 ページ）のセクションを参照してください。

ステップ 2 古い VM を削除します。詳細については、「古い VM の削除 （56 ページ） 古い VM の削除」の項を参照してください。

ステップ 3 新しい ESC アクティブ/アクティブ VM をインストールします。詳細については、新しい ESC アクティブ/アクティブ VM のインストール （56 ページ）のセクションを参照してください。

ステップ 4 ESC データベースを復元します。詳細については、ESC データベースの復元 （57 ページ）のセクションを参照してください。

データベースのバックアップ

アップグレードの前に、次の手順に従ってデータベースのバックアップを取得します。

手順

ステップ 1 次のコマンドを実行して、ESC リーダ VM をメンテナンスモードにします。

```
escadm op_mode set --mode=maintenance
```

ステップ 2 すべてのESCVMがトランザクション処理を停止するまで待機します。確認するには、次のコマンドを実行します。

```
escadm ip_trans
```

ステップ 3 次のコマンドを実行して、ESC リーダーにデータベースのバックアップを作成します。

```
 escadm backup --file dbback.tar, scp <dbback.tar>
```

ステップ 4 次のコマンドを実行して、すべての ESC VM からログを収集します。

```
escadm log collect
scp
```

古い VM の削除

手順

ステップ 1 次のコマンドを実行して、すべての ESC フォロワー VM と ESC リーダー VM をシャットダウンします。

```
nova stop
```

ステップ 2 次のコマンドを実行して、古いESCアクティブ/アクティブVMをOpenStackから削除します。

```
openstack stack delete {stack name}
```

新しい ESC アクティブ/アクティブ VM のインストール

データベースのバックアップおよび古い ESC アクティブ/アクティブ VM のシャットダウンが完了したら、新規/アップグレードされた（新しいESCパッケージに基づく）アクティブ/アクティブ ESC VM をインストールする必要があります。

手順

ステップ 1 OpenStack では、次のコマンドを実行して新しいイメージを登録します。

```
glance image-create
```

ステップ 2 次のコマンドを実行して、新しい ESC アクティブ/アクティブ VM をインストールします。

```
openstack stack create {stack name} --template {location of the template file}
```

ステップ 3 次のコマンドを実行して、すべての ESC VM の正常性を確認し、フォロワー VM で escadm サービスを停止します。

```
sudo escadm stop for all followers VMs
```

ステップ 4 すべてのフォロワー VM で escadm サービスが停止したら、次のコマンドを実行して、リーダー VM の escadm サービスを停止します。

```
sudo escadm stop
```

ESC データベースの復元

次の手順を使用して、新しい ESC インスタンスで ESC データベースを復元します。

手順

ステップ 1 次のコマンドを実行して、バックアップファイルを新しいリーダーにコピーします。

```
scp
```

ステップ 2 次のコマンドを実行して、ESC リーダーのデータベースを復元します。

```
 sudo escadm restore --file <dbback.tar>
```

復元後、復元プロセスはリーダー VM で escadm サービスを開始します。ただし、すべてのフォロワー VM で escadm サービスは停止されたままです。

ステップ 3 ESC リーダー VM が中断されずにすべてのサービスを実行していることを確認します。

ステップ 4 次のコマンドを実行して、ESC リーダー VM を動作モードにします。

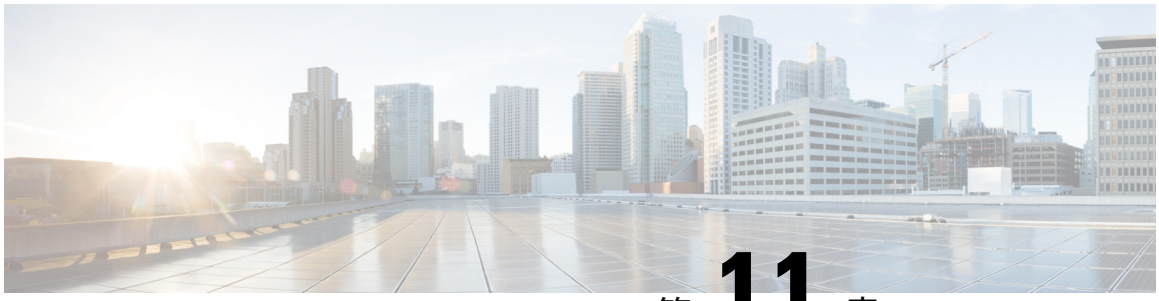
```
sudo escadm op_mode set --mode=operation
```

ステップ 5 次のコマンドを実行して、フォロワー VM で ESC サービスを開始します。

```
sudo escadm start
```

第 II 部

Cisco Elastic Services Controller の VMware vCenter へのインストール



第 11 章

前提条件

このセクションでは、Cisco Elastic Services Controller をインストールするための前提条件について詳しく説明します。

仮想リソースとハイパーバイザの要件

次の表に、VMware vCenter または vSphere に Cisco Elastic Services Controller をインストールするための前提条件を示します。

お使いのハードウェア プラットフォームが VMware でサポートされていることを確認するには、『VMware Compatibility Guide』を参照してください。

要件	説明
システム要件	
仮想 CPU	4 VCPU
メモリ	8 GB RAM
ディスク容量	30 GB
ハイパーバイザ要件	
VMWare vCenter	ESC は、VMware および vCenter バージョン 6.5 をサポートします。

vCenter のリソース

VCenter に作成/インストールするリソース：

- **データセンター**：少なくとも 1 つのデータセンター。詳細については、以下の**重要な注意事項**を参照してください。
- **ホスト**：ターゲットのパフォーマンス目標に基づくホスト設定です。単一の vDS の下の各ホストには、少なくとも 2 つの物理ネットワークインターフェイスカード（NIC）が接続されている必要があります（1 つはデフォルトで vCenter 管理インターフェイス用、もう 1 つは VDS のアップリンクポートグループへの割り当てに使用されます）。この設定は、ホスト間のデータアクセスに必要です。
- **コンピューティングクラスタ**：複数のホストをまとめてグループ化するためにクラスタを作成できます。
- **データストア**：ユーザが DRS を利用する場合は、共有データストアが必要です。
- **分散型スイッチ**：すべての VNF サポートネットワークを含む、少なくとも 1 つの分散スイッチです。

特記事項

VMware に ESC をインストールする際、次の特記事項に留意してください。

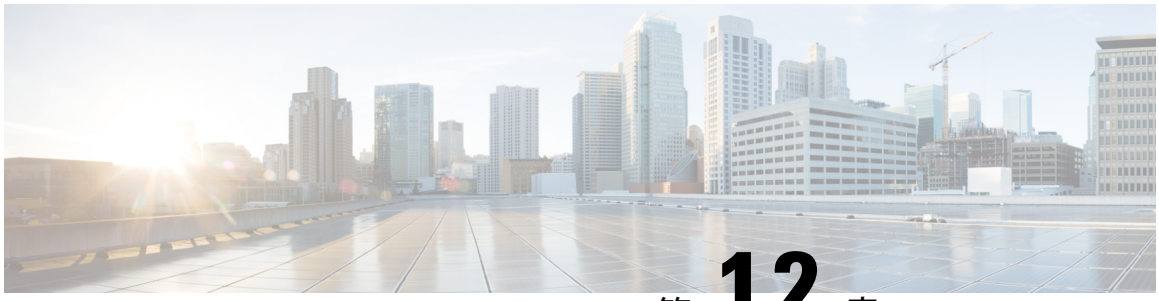
- 1 つの ESC インスタンスでは、以下のみがサポートされます。
  - 複数のデータセンターを対象にした展開、ネットワーク、イメージ、サブネットの作成
  - One vSphere Distributed Switch（VDS）
- DPM、HA アクティブ/スタンバイ、および vMotion を無効にする必要があります。
- DRS が有効になっている場合は、「手動モード」になっている必要があります。
- 耐障害性はサポートされていません。
- データストアクラスタはサポートされていません。クラスタ内またはデータセンター内のフラットなデータストア構造のみがサポートされています。
- ESC はデフォルトのリソースプールのみをサポートします。リソースプールの追加と作成はサポートされていません。
- ESC を使用して作成されたイメージ（テンプレート）は、**/esc-ovas** フォルダ内に保存されます。
- day-0、スマートライセンス、およびその他のサポート対象ファイルは ISO ファイルにパックされており、VM と同じフォルダにアップロードされ、CD-ROM として VM にマウントされます。
- ESC/VIM は、ISO ファイルの生成で渡されるファイル名やファイルコンテンツに対して応答しません。ファイル名とファイルコンテンツは各テンプレートの要件に従って指定する必要があります。たとえば ASAv の場合、day-0 設定は「day0-config」という名前にする

必要があり、スマートライセンストークンは「idtoken」という名前にする必要があります。

- 「ネットワーキングの設定操作がロールバックされ、ホストが vCenter サーバから切断されています」という内容のエラーメッセージが表示された場合は、vCenter の制限が原因です。ロールバックのタイムアウトを延長するには、「トラブルシューティングガイド」(91 ページ) を参照してください。
- Cisco CSR 1000V のすべてのバージョンでは、次の VM 機能と操作はサポートされていません。これらの操作が使用または実行されると、ドロップされたパケット、ドロップされた接続、およびその他のエラー統計情報が検出される可能性があります。

  1. DRS
  2. 中断（Suspend）
  3. スナップショット
  4. 復帰（Resume）

- 展開は共有ストレージなしで処理できますが、ESC はコンピューティングリソースの最適化を保証しません。共有ストレージは、可能な限り多くのホストに関連付ける必要があります。これにより、DRS はリソースのバランスをとることが可能になります。
- VMware のリカバリの一環として再展開が行われるたびに、VM のインターフェイスには異なる MAC アドレスが割り当てられます。
- データモデルで定義されているすべての VM グループは、「ゾーンホスト」配置ポリシーに準拠する必要があります。つまり、展開はホスト、クラスタのいずれかを対象にする必要があります。
- VM に PCI/PCIe パススルーデバイスが接続されていない場合、VM がコンピューティングホスト（ESC 配置アルゴリズムに基づき選択）に回復される際に、リカバリに失敗することがあります。PCI/PCIe パススルーが有効になっているデバイスがないためです。
- PCI/PCIe パススルーが機能するには、DRS がオフになっている必要があります。
- PCI/PCIe パススルーデバイスが接続されている VM で PowerOn エラーが発生した場合は、ここで説明されているソリューションを使用して複製元の VM またはイメージ（テンプレート）を更新します。



第 12 章

Cisco Elastic Services Controller の VMware vCenter へのインストール

この章では、VMware vCenter 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。この章は次のセクションで構成されています。

Cisco Elastic Services Controller の VMware vCenter へのインストール

Cisco Elastic Services Controller は、VMware ESXi ハイパーバイザにインストールして、VMware の vSphere クライアントを使用してアクセスまたは管理できます。VMware 環境に Cisco Elastic Services Controller をインストールするには、オープン仮想アプライアンス（OVA）パッケージを使用します。

VMware vSphere クライアントは、ESXi に直接接続するか、または vCenter サーバへの接続を介して、vSphere に接続できます。VCenter を介して接続すると、ESXi に直接接続した場合には提供されない多くの機能が提供されます。VCenter サーバが使用可能で、ESXi に関連付けられている場合は、VCenter を介した接続を推奨します。

Cisco Elastic Services Controller のインストールに向けた準備

Cisco Elastic Services Controller をインストールしてネットワーク接続を設定するには、いくつかの質問に答える必要があります。質問の中には、仮想マシンがインストールされているネットワーキング環境に関するものと、インストールされている特定の仮想マシンに固有の値に関するものがあります。

インストールを開始する前に、以下のチェックリストを参照して、準備が整っていることを確認します。

要件	ユーザ情報/注記
OVA イメージの場所	
OVA イメージ	
vSphere Web クライアント	
ホスト名	
IP アドレス	
サブネット マスク	
ネットワーク	
vCenter IP	
vCenter ポート（vCenter Port）	
vCenter ログイン情報	
データセンター名（Datacenter Name）	
データストアのホスト（Datastore Host）	
コンピュータクラスタ名	

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

Cisco Elastic Services Controller をインストールするには、最初に正しいインストールファイルをダウンロードする必要があります。

VSphere を使用して、ESXi のインストールまたは vCenter サーバに直接接続し、OVA の展開先である ESXi のインストールを選択します。

この手順では、VMware に Elastic Services Controller OVA イメージを展開する方法について説明します。

始める前に

- キーボードを英語（米国）に設定します。
- Elastic Services Controller OVA イメージが VMware vSphere Client から使用できることを確認します。
- 「第 6 章：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ことを確認します。
- 「Cisco Elastic Services Controller のインストールに向けた準備」に記載されている情報を収集します。

手順

ステップ 1 VMware vSphere Client を使用して vCenter Server にログインします。

ステップ 2 **[vCenterHome]** > **[ホストおよびクラスタ（Hosts and Clusters）]** の順に選択します。ESC を展開するホストを右クリックし、**[OVFテンプレートを展開（Deploy OVF Template）]** を選択します。

ステップ 3 ウィザードで、次のテーブルで説明されている情報を入力します。

画面	アクション
ソースの選択（Select source）	Elastic Services Controller OVA を選択します。
詳細の確認（Review details）	OVF テンプレートの詳細を確認します。
名前とフォルダの選択（Select name and folder）	名前を入力し、VMのフォルダを選択します。
設定の選択（Select configuration）	次のいずれかの展開設定を選択します。 • 大規模な 1 つのネットワーク • 大規模な 2 つのネットワーク • 大規模な 3 つのネットワーク
リソースの選択（Select resource）	ESC テンプレートを実行するためのホストまたはクラスタを選択します。
ストレージの選択（Select Storage）	VM のファイルとプロビジョニングタイプを保存する場所を選択します。ストレージは、ローカルか、NFS や SAN などの共有リモートにできます。 **シンプロビジョニングの形式**または**シックプロビジョニングの形式**のどちらかを選択し、VM 仮想ディスクを保存できます。
ネットワークの選択（Select networks）	[設定の選択（Select configuration）] オプションで選択した展開のネットワーク設定に基づいて、vCenterに事前設定されたネットワークを ESC ネットワーク インターフェイスに割り当てることができます。
テンプレートのカスタマイズ（Customize template）	
ブートストラップのプロパティ	
ユーザ名（Username）	リモートログインの管理者ユーザ名。

画面	アクション
パスワード（Password）	管理者パスワード。
ホスト名（Host name）	VM ホスト名
ネットワーク IP（Network IP）	VM IP アドレス
ネットワークゲートウェイ（Network Gateway）	ゲートウェイ IP アドレス。
Https Rest の有効化（Enable Https Rest）	外部 REST インターフェイスをポート 8443 で HTTPS を介して有効にします。
ポータルの起動の有効化（Enable Portal startup）	ポータルの起動をポート 9001（https の場合）で有効にします。
vCenter サーバの VIM 設定（VIM Settings of vCenter Server）	
vCenter IP	VNF 展開用の vCenter サーバの IP アドレス。
vCenter ポート（vCenter Port）	VCenter サーバのポート。
vCenter ユーザ名（vCenter Username）	VCenter サーバにアクセスするためのユーザ名。
vCenter パスワード（vCenter Password）	VCenter サーバにアクセスするためのパスワード。
データセンター名（Datacenter Name）	VNF 展開のターゲット vCenter 内のデータセンター名（マルチ VDC がサポートされた後のデフォルト VDC）
データストア名（Datastore Name）	すべてのイメージ（テンプレート）の宛先データストアは、ESC を使用して作成されます。
データストアのホスト（Datastore Host）	ESC を介して作成されるすべてのイメージ（テンプレート）の宛先コンピューティングホスト。
終了準備の完了（Ready to Complete）	展開設定を確認します。 **注意** 不一致があると、VM の起動時に問題が発生する可能性があります。IP アドレス、サブネットマスク、およびゲートウェイ情報が正しいかを慎重に確認します。
公開キー（3Public Key）	管理者が承認したリモートログイン用の公開キー。

画面	アクション
ConfD ユーザ名（ConfD Username）	Netconf および ConfD CLI の管理者ユーザ名。
ConfD パスワード（ConfD Password）	Netconf および ConfD CLI の管理者パスワード。
ConfD 公開キー（ConfD Public Key）	Netconf および ConfD CLI の管理者が承認した公開キー。

ステップ 4 [展開後に電源をオン（power on after deployment）] チェックボックスをオンにし、展開後に VM の電源をオンにします。

ステップ 5 [終了（Finish）] をクリックします。
進行状況インジケータには、Elastic Services Controller が展開されるまでタスクの進行状況を表示されます。

ステップ 6 Elastic Services Controller が正常に展開された後、[終了（Close）] をクリックします。

ステップ 7 Elastic Services Controller VM の電源をオンにします。

OVA ツールを使用した Elastic Services Controller のインストール

OVA イメージを使用して Elastic Services Controller をインストールする方法に加えて、VMware OVF ツール（コマンドラインクライアント）を使用して、VMware vCenter または vSphere に Elastic Services Controller をインストールすることもできます。

コマンドラインから Elastic Services Controller（ESC）をインストールするには、次の手順を実行します。

手順

ステップ 1 プローブモードを使用して、OVA パッケージのプロパティを確認します。プローブモードでは、ソースのコンテンツを調査することができます。

プローブモードを起動するには、ソースのみ、ターゲットなしで **ovftool** コマンドを使用します。

```
>ovftool <source locator>
```

次の例は、ESC OVA をプローブした結果を示しています。

```
NETWORK_OVA=(Path to the OVA Package)

NETWORK_HOSTNAME="$(User Name)"
NETWORK_GATEWAY="192.0.2.1"
NETWORK_NET1_IP="192.0.2.0.xx/24" #
NETWORK_NET2_IP="192.51.100.xx/24"
ADMIN_USERNAME="(admin name)"
ADMIN_PASSWORD="(password)"
HTTPS_REST="True"
```

```
VMWARE_VCENTER_PORT='80'
VMWARE_VCENTER_IP='192.0.2.0.xx'
VMWARE_DATASTORE_HOST='192.0.2.0.xx'
VMWARE_DATACENTER_NAME='DC-NETWORK-1'
VMWARE_DATASTORE_NAME='cluster-datastore1'
VMWARE_COMPUTE_CLUSTER_NAME='DC-CLUSTER-1'
VMWARE_VCENTER_USERNAME='root'
VMWARE_VCENTER_PASSWORD='password'
VMWARE_VCENTER_FOLDER="$USER"

# All valid deployment options:
#          4CPU-8GB  (default)
#          4CPU-8GB-2Net
#          4CPU-8GB-3Net
DEPLOYMENT_OPTION="4CPU-8GB-2Net"
```

ステップ 2 ESC OVA を展開する前に、OVA パッケージのプロパティを設定します。次の OVA パッケージのプロパティが ESC OVA で更新されていることを確認します。NETWORK_OVA、NETWORK_HOSTNAME、VMWARE_VCENTER_FOLDER、NETWORK_NET1_IP、NETWORK_NET2_IP、VMWARE_VCENTER_FOLDER

OVA 記述子には、OVA パッケージの設定プロパティが含まれています。一度に設定できるプロパティは 1 つだけですが、コマンドごとにオプションの複数のインスタンスを指定できます。複数のプロパティマッピングの場合は、オプションを繰り返し、空白で区切ります（例：*--prop:p1=v1 --prop:p2=v2 --prop:p3=v3*）。

```
>.ovftool/ovftool\
--powerOn \
--acceptAllEulas \
--noSSLVerify \
--datastore=$VMWARE_DATASTORE_NAME \
--diskMode=thin \
--name=$NETWORK_HOSTNAME \
--deploymentOption=$DEPLOYMENT_OPTION \
--vmFolder=$VMWARE_VCENTER_FOLDER \
--prop:admin_username=$ADMIN_USERNAME --prop:admin_password=$ADMIN_PASSWORD \
--prop:admin_username=admin \
--prop:admin_password='Strong4Security!' \
--prop:confd_admin_username=admin \
--prop:confd_admin_password='Strong4Security!' \
--prop:network_hostname=$NETWORK_HOSTNAME \
--prop:vmware_vcenter_port=$VMWARE_VCENTER_PORT \
--prop:vmware_vcenter_ip=$VMWARE_VCENTER_IP \
--prop:vmware_datastore_host=$VMWARE_DATASTORE_HOST \
--prop:vmware_datacenter_name=$VMWARE_DATACENTER_NAME \
--prop:vmware_vcenter_username=$VMWARE_VCENTER_USERNAME \
--prop:vmware_datastore_name=$VMWARE_DATASTORE_NAME \
--prop:vmware_compute_cluster_name=$VMWARE_COMPUTE_CLUSTER_NAME \
--prop:vmware_vcenter_password=$VMWARE_VCENTER_PASSWORD \
--prop:net1_ip=$NETWORK_NET1_IP \
--prop:net2_ip=$NETWORK_NET2_IP \
--prop:gateway=$NETWORK_GATEWAY \
--prop:https_rest=$HTTPS_REST \
--net:"Network1=VM Network" --net:"Network2=MgtNetwork" --net:"Network3=VNFNetwork" \
    $NETWORK_OVA
vi://$VMWARE_VCENTER_USERNAME:$VMWARE_VCENTER_PASSWORD@$VMWARE_VCENTER_IP/$VMWARE_DATACENTER_NAME/
host/$VMWARE_COMPUTE_CLUSTER_NAME
```

次に、プロパティを使用してユーザのログイン情報を渡す高度な例のいくつかを示します。

パスワードハッシュを用いた高度な使用例：

```
--prop:admin_username=admin \
--prop:admin_password='$6$wnOi$UDOQmkKm2tQtr2jDVNhoo4wS42ffYYmzxMXLDugfzTbTXmMQDw146VzpxQvMumeaa125.agYHZUqQ8L.scM2v0'
 \
--prop:confd_admin_username=admin \
--prop:confd_admin_password='$6$wnOi$UDOQmkKm2tQtr2jDVNhoo4wS42ffYYmzxMXLDugfzTbTXmMQDw146VzpxQvMumeaa125.agYHZUqQ8L.scM2v0'
 \
```

パスワードハッシュと承認済みの公開キーを用いた高度な使用例：

```
--prop:admin_username=admin \
--prop:admin_password='$6$wnOi$UDOQmkKm2tQtr2jDVNhoo4wS42ffYYmzxMXLDugfzTbTXmMQDw146VzpxQvMumeaa125.agYHZUqQ8L.scM2v0'
 \
--prop:admin_public_key='ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAIEAu+nkTtu2pShVbTYL+mmKxtmzM5dNXFy8IeX/lH5fXsODH1EAySlzHGFXq36RT5vIG/
+c2uV8fRsWaY7xXDrdGICxfkPuEj2UQH2MQx2yFjMFcaSAT56hsqE= admin@net' \
--prop:confd_admin_username=admin \
--prop:confd_admin_password='$6$wnOi$UDOQmkKm2tQtr2jDVNhoo4wS42ffYYmzxMXLDugfzTbTXmMQDw146VzpxQvMumeaa125.agYHZUqQ8L.scM2v0'
 \
--prop:confd_admin_public_key='ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAIEAu+nkTtu2pShVbTYL+mmKxtmzM5dNXFy8IeX+xXu6TT2sTsxkKtVy8uOAeBp1qKzkp+
c2uV8fRsWaY7xXDrdGICxfkPuEj2UQH2MQx2yFjMFcaSAT56hsqE= admin@net' \
```

（注） 上の例の変数（IP アドレス、root パスワード、VM 名など）は、ご使用のシステムの値に置き換える必要があります。

ステップ 3 VMware OVF ツールで OVA を展開する場合、次のコマンドシンタックスを使用します。

```
>ovftool <source locator> <target locator>
```

<*source locator*> は OVA パッケージのパス、<*target locator*> は、仮想マシン、OVA パッケージまたは VI のパスターゲットです。VI の場所とは、vSphere、VMware Server、ESXi といった VMware 製品上の場所を指します。VMware OVF ツールの詳細については、VMware OVF ツールのユーザマニュアルを参照してください。

ESC VM が VMware に展開され、自動的に電源がオンになります。

Cisco Elastic Services Controller 仮想マシンの電源投入

Cisco Elastic Services Controller の仮想マシン（VM）の電源をオンにするには、次の手順を実行します。

（注） [電源オン（Power On）] をクリックする前に、要件に基づいてメモリと CPU を設定する必要があります。VM を起動すると、シャットダウンするまでメモリや CPU の設定を変更できません。

手順

ステップ 1 VM を展開した後、vSphere で仮想マシン名を選択して右クリックし、[コンソールを開く（Open Console）] を選択します。

ステップ 2 [電源オン（Power On）] ボタン（  ）をクリックします。新しく展開されたマシンの初回起動時に、ルート（システム）パスワードを入力するように求められます。これは、Cisco Elastic Services Controller ポータルのパスワードとは異なります。設定によって初期化の方法が異なる場合があります。

（注） これは、Cisco Elastic Services Controller ポータルを搭載した基盤となる Linux オペレーティングシステムのルートパスワードを指します。このパスワードを 2 回入力するように求められます。今後、さまざまな場面で、基盤となる Linux オペレーティングシステムへのルートアクセスが必要になります。そのため、このパスワードを覚えておいてください。

[エンドユーザライセンス契約（End User License Agreement）] ウィンドウが初回起動時に表示されます。ライセンス契約のすべてに目を通し、ライセンス条項を理解して同意した場合にのみ、y（Yes）と入力します。

次のステップ：Cisco Elastic Services Controller 仮想マシン

Cisco Elastic Services Controller ポータルへのログイン

ESC ポータルにログインするには、次を参照してください。ESC ポータルへのログイン（138 ページ）

自動的に電源をオンにするための仮想マシンの設定

ESXi ハイパーバイザレイヤに電力が復旧されたときに、ESC VM の電源を自動的にオンにするように ESXi ハイパーバイザを設定できます。

（注） VM の電源を手動でオンにする必要があります。

手順

ステップ 1 VSphere クライアントで、接続先の ESXi マシンを選択します。特定の VM を選択するのではなく、VM が存在する ESXi ハイパーバイザを選択します。

ステップ 2 [設定（Configuration）] タブを選択します。

ステップ 3 [ソフトウェア（Software）] エリアの下にある [仮想マシンの起動/シャットダウン（Virtual Machine Startup/Shutdown）] リンクをクリックします。ウィンドウ内のリストに VM が表示されます。

ステップ 4 ページの右上隅にある [プロパティ...（Properties...）] リンクをクリックします。表示されない場合は、表示されるまでウィンドウのサイズを変更します。

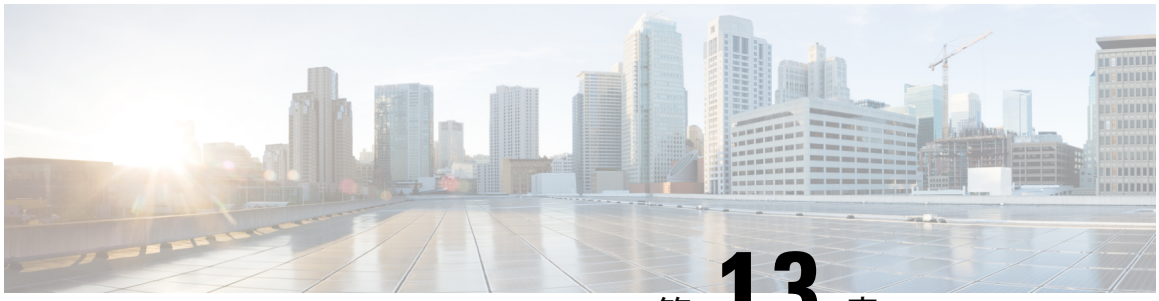
[仮想マシンの起動/シャットダウン（Virtual Machine Startup/Shutdown）] ページが表示されます。

ステップ 5 [システムによる仮想マシンの自動起動と自動停止を許可（Allow Virtual machines to start and stop automatically with the system）] チェックボックスをオンにします。

ステップ 6 ESC を稼働している仮想マシンを選択し、右側にある [上へ移動（Move up）] ボタンを使用して、[自動起動（Automatic Startup）] というラベル名のグループに移動します。

ステップ 7 [OK] をクリックします。

これにより、ESXi ハイパーバイザに電力が復旧されるたびに、ESC VM の電源が自動的にオンになります。



第 13 章

高可用性のインストール

この章は、次の項で構成されています。

高可用性アクティブ/スタンバイの概要

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。プライマリ 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 アクティブ/スタンバイ は、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 5.0 以降、アクティブ/パッシブモデルの名称がアクティブ/スタンバイモデルに変更されています。

高可用性アクティブ/スタンバイの仕組み

高可用性アクティブ/スタンバイの展開は、プライマリとスタンバイの 2 つの ESC インスタンスで構成されます。通常の状況下では、プライマリ ESC インスタンスによってサービスが提供されます。対応するスタンバイインスタンスはパッシブ状態になります。スタンバイインスタンスは、プライマリインスタンスと常時通信して、プライマリインスタンスのステータスをモニタします。プライマリ ESC インスタンスに障害が発生すると、スタンバイインスタンスが ESC サービスを自動的に引き継ぎ、最小限の中断で ESC サービスの提供を継続します。

スタンバイインスタンスにもプライマリインスタンスのデータベースの完全なコピーが存在しますが、プライマリインスタンスに障害が発生しない限り、セカンダリインスタンスがアクティブにネットワークを管理することはありません。プライマリインスタンスに障害が発生したときに、スタンバイインスタンスが自動的に引き継ぎます。プライマリインスタンスの復元中、スタンバイインスタンスがプライマリインスタンスを引き継ぎ、サービスを管理します。

障害が発生したインスタンスが復元されると、元のプライマリインスタンスを使用してネットワーク管理を再開するためのフェールバック操作を開始できます。

ESC インスタンスは、キープアライブサービスを使用して管理されます。ESC インスタンス間の VM ハンドシェイクは、IPv4 ネットワーク上で キープアライブサービスを介して行われます。

ユーザデータを使用した ESC 高可用性アクティブ/スタンバイの展開（HA アクティブ/スタンバイペア）

始める前に

- Cisco Elastic Services Controller（ESC）高可用性（HA）アクティブ/スタンバイでは、キープアライブを維持し、プライマリノードとスタンバイノード間でデータベースを複製するためのネットワークが必要です。両方の ESC VM には、同じネットワークに接続する少なくとも 1 つのネットワークインターフェイスが必要であり、ネットワークを介して相互に通信できる必要があります。
- 2 つの ESC VM が異なるホストとデータストアに配置されることを確認し、シングルポイント障害を防止できるようにします。

ESC HA アクティブ/スタンバイを、次のいずれかの方法で VMware vCenter または vSphere に展開できます。

- ESC HA アクティブ/スタンバイを高可用性アクティブ/スタンバイペアとしてユーザデータを使用して展開する（ESC 4.2 でサポート）
- ESC HA アクティブ/スタンバイを、2 つのスタンドアロンインスタンスとして展開し、POST 設定を使用してそれらを高可用性ペアとして設定します。詳細については、「ESC 高可用性アクティブ/スタンバイの展開（スタンドアロンインスタンス）」のセクションを参照してください。

ESC HA アクティブ/スタンバイを、高可用性アクティブ/スタンバイペアとしてユーザデータを使用して VMware vCenter または vSphere に展開するには、ユーザデータファイルを HA アクティブ/スタンバイインスタンスごとに定義し、次に、各インスタンスのユーザデータを ovftool を介して指定します。ユーザデータのエンコードは、ovftool スクリプトの一連のコマンドを介して行われ、その結果は、ovftool の「–prop:user-data = 」プロパティの変数として設定されます。

（注） 「admin user/password」および「confd user/password」プロパティは、必須の OVF プロパティです。これらのプロパティは、ユーザデータファイルでは定義できません。

- ESC HA アクティブ/スタンバイの 2 つの VM を定義します。

ユーザデータ 1

```
#cloud-config
ssh_pwauth: True
write_files:
 - path: /etc/cloud/cloud.cfg.d/sys-cfg.yaml
   content: |
     network:
       version: 1
       config:
       - type: nameserver
         address:
         - 161.44.124.122
       - type: physical
         name: eth0
         subnets:
         - type: static
           address: 172.16.0.0
           netmask: 255.255.255.0
           routes:
           - gateway: 172.16.0.0
             network: 0.0.0.0
             netmask: 0.0.0.0
 - path: /opt/cisco/esc/esc-config/esc-config.yaml
   content: |
      resources:
        confd:
          option: start-phase0
        drbd:
          nodes:
          - 172.16.0.0
          - 172.16.1.0
          run_forever: true
        esc_service:
          depend_on: filesystem
```

```
          type: group
        escmanager:
          depend_on:
          - pgsql
          - mona
          - vimmanager
        etsi:
          depend_on: pgsql
          startup: false
        filesystem:
          depend_on: drbd:master
        keepalived:
          vip: 172.16.2.0
        portal:
          depend_on: escmanager
          startup: false
        snmp:
          startup: false
runcmd:
 - [ cloud-init-per, once, escadm_ovf_merge, sh, -c, "/usr/bin/escadm ovf merge"]
 - [ cloud-init-per, once, escservicestart, sh, -c, "chkconfig esc_service on &&
service esc_service start"]
```

ユーザデータ 2

```
#cloud-config
ssh_pwauth: True
write_files:
 - path: /etc/cloud/cloud.cfg.d/sys-cfg.yaml
   content: |
     network:
       version: 1
       config:
       - type: nameserver
         address:
         - 161.44.124.122
       - type: physical
         name: eth0
         subnets:
         - type: static
           address: 172.16.1.0
           netmask: 255.255.255.0
           routes:
           - gateway: 172.16.0.0
             network: 0.0.0.0
             netmask: 0.0.0.0
 - path: /opt/cisco/esc/esc-config/esc-config.yaml
   content: |
      resources:
        confd:
          option: start-phase0
        drbd:
          nodes:
          - 172.16.0.0
          - 172.16.1.0
          run_forever: true
        esc_service:
          depend_on: filesystem
          type: group
        escmanager:
          depend_on:
          - pgsql
          - mona
          - vimmanager
        etsi:
```

```
          depend_on: pgsql
          startup: false
        filesystem:
          depend_on: drbd:master
        keepalived:
          vip: 172.16.2.0
        portal:
          depend_on: escmanager
          startup: false
        snmp:
          startup: false
runcmd:
 - [ cloud-init-per, once, escadm_ovf_merge, sh, -c, "/usr/bin/escadm ovf merge"]
 - [ cloud-init-per, once, escservicestart, sh, -c, "chkconfig esc_service on &&
service esc_service start"]
```

- 各 VM インスタンスについて、OVFtool を 2 回呼び出す必要があります。各インスタンスは、ハッシュ化されたユーザデータを指す「--prop:user-data」プロパティを提供する必要があります。
- ここでは、172.16.0.0 および 172.16.1.0（フローティング）IP をインスタンスに、172.16.2.0 を KAD_VIP として使用する HA アクティブ/スタンバイインスタンスのペアをブートする例を示しています。

```
user_data_1=`cat ./user-data-1`
user_data_2=`cat ./user-data-2`
dec_user_data_1=`echo "$user_data_1" | base64 | tr -d '[:space:]'`
dec_user_data_2=`echo "$user_data_2" | base64 | tr -d '[:space:]'`
# vcenter-16 is the developer lab for vmware5
ESC_OVA=/scratch/BUILD-${ESC_IMAGE}/BUILD-${ESC_IMAGE}/ESC-${ESC_IMAGE}.ova
# All valid deployment options:
#          2CPU-4GB
#          4CPU-8GB  (default)
#          4CPU-8GB-2Net
#          4CPU-8GB-3Net
DEPLOYMENT_OPTION="4CPU-8GB-2Net"
deploy_vmware_vm1() {
/usr/bin/ovftool \
--powerOn \
--acceptAllEulas \
--noSSLVerify \
--datastore=$VM_WARE_DATASTORE_NAME \
--diskMode=thin \
--name=$INSTANCE_NAME"-0" \
--deploymentOption=$DEPLOYMENT_OPTION \
--vmFolder=$FOLDER \
--prop:admin_username=$ESC_VM_USERNAME --prop:admin_password=$ESC_VM_PASSWORD \
--prop:esc_hostname=$INSTANCE_NAME"-0" \
--prop:rest_username=$REST_USERNAME \
--prop:rest_password=$REST_PASSWORD \
--prop:portal_username=$PORTAL_USERNAME \
--prop:portal_password=$PORTAL_PASSWORD \
--prop:confd_admin_username=$CONFD_USERNAME \
--prop:confd_admin_password=$CONFD_PASSWORD \
--prop:vmware_vcenter_port=$VMWARE_VCENTER_PORT \
--prop:vmware_vcenter_ip=$VM_WARE_VCENTER_IP \
--prop:vmware_datastore_host=$VM_WARE_DATASTORE_HOST \
--prop:vmware_datacenter_name=$VM_WARE_DATACENTER_NAME \
--prop:vmware_vcenter_username=$VM_WARE_VCENTER_USERNAME \
--prop:vmware_datastore_name=$VM_WARE_DATASTORE_NAME \
--prop:vmware_vcenter_password=$VM_WARE_VCENTER_PASSWORD \
--prop:net1_ip=$NET1_IP1 \
--prop:net2_ip=$NET2_IP1 \
```

```
--prop:gateway=$ESC_GATEWAY \
--prop:https_rest=$HTTPS_REST \
--prop:user-data=$dec_user_data_1 \
--net:"Network1=VM Network" --net:"Network2=MgtNetwork" --net:"Network3=VNFNetwork"
 \
    $ESC_OVA
vi://$VM_WARE_VCENTER_USERNAME:$VM_WARE_VCENTER_PASSWORD@$VM_WARE_VCENTER_IP/
$VM_WARE_DATACENTER_NAME/host/$VM_WARE_DATASTORE_CLUSTER
}
deploy_vmware_vm2() {
/usr/bin/ovftool \
--powerOn \
--acceptAllEulas \
--noSSLVerify \
--datastore=$VM_WARE_DATASTORE_NAME \
--diskMode=thin \
--name=$INSTANCE_NAME"-1" \
--deploymentOption=$DEPLOYMENT_OPTION \
--vmFolder=$FOLDER \
--prop:admin_username=$ESC_VM_USERNAME --prop:admin_password=$ESC_VM_PASSWORD \
--prop:esc_hostname=$INSTANCE_NAME"-1" \
--prop:rest_username=$REST_USERNAME \
--prop:rest_password=$REST_PASSWORD \
--prop:portal_username=$PORTAL_USERNAME \
--prop:portal_password=$PORTAL_PASSWORD \
--prop:confd_admin_username=$CONFD_USERNAME \
--prop:confd_admin_password=$CONFD_PASSWORD \
--prop:vmware_vcenter_port=$VMWARE_VCENTER_PORT \
--prop:vmware_vcenter_ip=$VM_WARE_VCENTER_IP \
--prop:vmware_datastore_host=$VM_WARE_DATASTORE_HOST \
--prop:vmware_datacenter_name=$VM_WARE_DATACENTER_NAME \
--prop:vmware_vcenter_username=$VM_WARE_VCENTER_USERNAME \
--prop:vmware_datastore_name=$VM_WARE_DATASTORE_NAME \
--prop:vmware_vcenter_password=$VM_WARE_VCENTER_PASSWORD \
--prop:net1_ip=$NET1_IP2 \
--prop:net2_ip=$NET2_IP2 \
--prop:gateway=$ESC_GATEWAY \
--prop:https_rest=$HTTPS_REST \
--prop:user-data=$dec_user_data_2 \
--net:"Network1=VM Network" --net:"Network2=MgtNetwork" --net:"Network3=VNFNetwork"
 \
    $ESC_OVA
vi://$VM_WARE_VCENTER_USERNAME:$VM_WARE_VCENTER_PASSWORD@$VM_WARE_VCENTER_IP/
$VM_WARE_DATACENTER_NAME/host/$VM_WARE_DATASTORE_CLUSTER
}
deploy_vmware_vm1
deploy_vmware_vm2
```

- VM が正常に展開された後に、ESC HA アクティブ/スタンバイのステータスを確認できます。1 つの VM インスタンスが MASTER として起動され、他の VM インスタンスが BACKUP であることがわかります。

ESC高可用性アクティブ/スタンバイの展開（スタンドアロンインスタンス）

VMware vCenter または vSphere で ESC HA アクティブ/スタンバイを展開するには、2 つの別個のスタンドアロンノードを最初にインストールする必要があります。スタンドアロン ESC イ

ンスタンスがインストールされた後、次を使用して、これらのノードがプライマリとスタンバイになるように再設定します。

- kad_vip
- kad_vif
- ha_node_list

（注）
- ESC VM ごとに、*escadm* ツールを実行して ESC HA アクティブ/スタンバイパラメータを設定した後、escadm サービスをリロードして再起動する必要があります。
- ESC HA アクティブ/スタンバイを展開する際、kad_vip 引数を使用すると、エンドユーザがプライマリ ESC インスタンスにアクセスできるようになります。

手順

ステップ 1 ESC スタンドアロンインスタンスにログインします。

ステップ 2 管理者ユーザとして、プライマリインスタンスとスタンバイインスタンスの両方で*escadm* ツールを実行し、対応する引数を指定します。

- **kad_vip**：keepalived VIP（仮想 IP）の IP アドレスと keepalived VIP のインターフェイスを指定します（ESC-HA アクティブ/スタンバイ）。
- **kad_vif**：keepalived 仮想 IP と keepalived VRRP のインターフェイスを指定します（ESC-HA アクティブ/スタンバイ）。VIP インターフェイスが引数 *kad_vip* を使用してすでに指定されている場合は、引数 kad_vip を使用して keepalived VRRP のインターフェイスのみを指定することもできます。
- **ha_node_list**：DRDB 同期のため、プライマリ/スタンバイクラスタに含まれる HA アクティブ/スタンバイノードの IP アドレスのリストを指定します。この引数は、レプリケーションベースの HA アクティブ/スタンバイソリューションのみに使用されます。複数のネットワークインターフェイスを持つ ESC インスタンスの場合、IP アドレスは、引数 *--kad_vif* で指定されたネットワーク内にある必要があります。

```
$ sudo escadm ha set --kad_vip= <ESC_HA_VIP> --kad_vif= <ESC_KEEPALIVE_IF>
--ha_node_list= <ESC_NODE_1_IP> <ESC_NODE_2_IP>
$ sudo escadm reload
$ sudo escadm restart
```

ステップ 3 再起動後、1 つの ESC VM はプライマリ状態になり、もう 1 つはスタンバイ状態になる必要があります。

ステップ 4 VIP が外部から到達可能になるように、両方の VM で許可されたアドレスペアに VIP を追加します。

ステップ 5 各 ESC インスタンスのステータスを確認します。

```
# sudo escadm status
```

次の表に、ステータスを確認するための他のコマンドをいくつか示します。

ステータス	CLI コマンド
ESC HA アクティブ/スタンバイのロール	`cat /opt/cisco/esc/keepalived_state`
ESC の正常性	`sudo escadm health`
ESC サービスのステータス	詳細情報（VIM マネージャ、SNMP、ポータル、ESC マネージャ、keepalived のステータスなど）を表示するには、「–v」を追加します。 `sudo escadm status --v` 詳細なステータスを確認するには、/var/log/esc/escadm.log をチェックします。

ESC HA アクティブ/スタンバイに関する特記

- HA アクティブ/スタンバイフェールオーバーには、動作可能な管理対象 VNF の数に基づいて約 2 ー 5 分かかります。ESC サービスは、スイッチオーバー時間中は使用できません。
- トランザクション中にスイッチオーバーがトリガーされると、すべての未完了のトランザクションがドロップされます。要求は、ESC からの応答を受信しない場合、ノースバウンドインターフェイスによって再送信される必要があります。

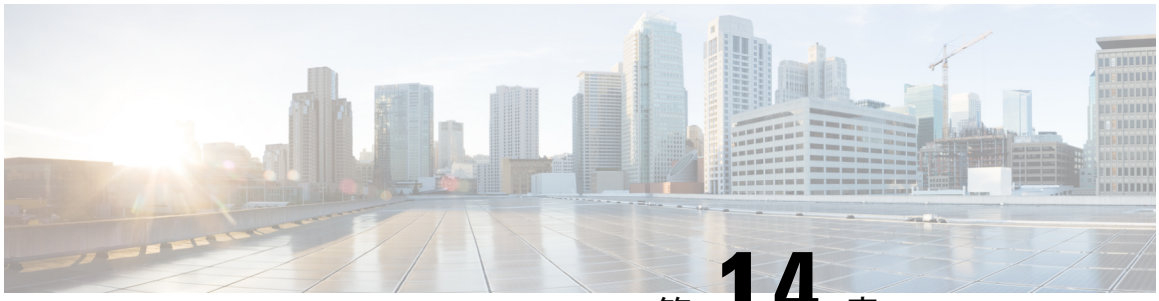
高可用性アクティブ/スタンバイのトラブルシューティング

- ネットワーク障害をチェックします。ネットワークに問題が発生している場合は、次の詳細情報をチェックする必要があります。
  - 割り当てられている IP アドレスは正しいもので、OpenStack 設定に基づいている必要があります。
  - 各ネットワークインターフェイスのゲートウェイで ping が応答する必要があります。
- トラブルシューティングの際には、次のログをチェックします。
  - ESC マネージャログ：*/var/log/esc/escmanager.log*
  - キープアライブログ：*/var/log/messages*　（grep keepalived を実行）

- ESC サービスステータスログ：*/Var/log/esc/escadm.log*

第 III 部

Cisco Elastic Services Controller のカーネルベース仮想マシン（KVM）へのインストール



第 14 章

Cisco Elastic Services Controller のカーネルベース仮想マシン（KVM）へのインストール

この章では、カーネルベース仮想マシンに Cisco Elastic Services Controller をインストールする方法について説明します。この章は次のセクションで構成されています。

Cisco Elastic Services Controller のカーネルベース仮想マシンへのインストール

Cisco Elastic Services Controlle は、カーネルベースの仮想マシンにインストールできます。Cisco Elastic Services Controller をカーネルベースの仮想マシンにインストールするには、libvirt を使用します。

カーネルベース仮想マシンに Cisco Elastic Services Controller をインストールするための準備

カーネルベースの仮想マシンで Cisco Elastic Services Controller を実行する予定の場合は、次のように設定されていることを確認してください。

	注記
Python 2.7 または 3.x	Linux にデフォルトでインストールされています
python-setuptools	Linux にデフォルトでインストールされています

	注記

pip	**RHEL** の場合：
	`# easy_install pip`
	pip を使用したインストールではソースファイルがコンパイルされるため、RHEL では gcc と python の開発パッケージも必要です。RHEL にこれらのパッケージをインストールするには、次の手順に従います。
	`# yum install gcc python-devel`
	Ubuntu の場合は、デフォルトでインストールされています。pip を使用したインストールではソースファイルがコンパイルされるため、Ubuntu では gcc と python の開発パッケージも必要です。Ubuntu にこれらのパッケージをインストールするには、次の手順に従います。
	`# apt-get install python-dev`

OpenStack クライアント	`# pip install python-keystoneclient`
	`# pip install python-cinderclient`
	`# pip install python-novaclient`
	`# pip install python-neutronclient`

genisoimage	**RHEL** の場合：
	`# yum install  genisoimage`
	Ubuntu の場合：
	`# apt-get install genisoimage`

libvirt および virtinst	**RHEL 6.X** の場合：
	`# yum install libvirt-python python-virtinst`
	RHEL 7.X の場合：
	`# yum install libvirt-python virt-install`
	Ubuntu の場合：
	`# apt-get install libvirt-dev`
	`# pip install libvirt-python`

（注） libvirt はデフォルトのネットワークを自動的に作成します。

Elastic Services Controller のカーネルベース仮想マシンへのインストール

カーネルベース仮想マシンにスタンドアロンの Elastic Services Controller（ESC）をインストールするには、次の手順を実行します。

手順

ステップ 1 OpenStack ログイン情報を含む openerc ファイルから変数をロードします。

```
cat ./openrc.sh
export OS_TENANT_NAME='<OS tenant username>'
export OS_USERNAME='<OS username>'
export OS_PASSWORD='<OS password>'
export OS_AUTH_URL='http://<Openstack Host>:5000/v2.0/'

source ./openrc.sh
```

ステップ 2 ESC qcow2 イメージと bootvm.py をカーネルベース VM にコピーします。

ステップ 3 次のコマンドのいずれかを使用して、libvirt のインストール時に作成されたデフォルトネットワーク上のカーネルベース VM で ESC を起動します。

```
./bootvm.py --user_pass <username>:<password> --user_confd_pass <username>:<password>
--libvirt --image <image_name> esc-vm --net <default network>
```

ステップ 4 次のコマンドを使用して、静的 IP を持つデフォルトネットワーク上のカーネルベース VM で ESC を起動します。

```
./bootvm.py --user_pass <username>:<password> --user_confd_pass <username>:<password>
--libvirt --image <image_name> esc-vm --net <network> --ipaddr <ip_address>
```

ステップ 5 ネットワークで使用されている IP アドレスのリストを取得します。HA アクティブ/スタンバイの bootvm.py コマンドと kad_vip の両方に対して、リストに含まれていない IP アドレスを使用します。ネットワークの最初の 3 オクテット（つまり 192.168.122）を特定し、次のコマンドで渡します。

```
arp -an | grep 192.168.122
```

ステップ 6 高可用性のカーネルベース VM に ESC をインストールするには、両方の HA ノードに対して次のコマンドを 2 回使用します。

（注） 2 番目の bootvm.py コマンドでは、他の HA インスタンス名を使用します。

```
./bootvm.py --user_pass <username>:<password> --user_confd_pass <username>:<password>
--libvirt --image <image_name> --ha_mode drbd --gateway_ip <default_gateway_ip_address>
 --ipaddr <ip_address>
 --ha_node_list <ha peer ip addresses separated by comma> --kad_vip <vip address> esc-ha-1
 --net <network>
```

次の手順：Cisco Elastic Services Controller カーネルベース仮想マシン

Cisco Elastic Services Controller ポータルへのログイン

ESC ポータルにログインするには、次を参照してください。ESC ポータルへのログイン （138 ページ）

カーネルベース仮想マシン（KVM）の ESC インストールの確認

カーネルベースの仮想マシンに ESC を展開した後、次の手順を使用して展開を確認します。

手順

ステップ 1 ESC VM が起動したことを確認するには、次のコマンドを使用します。

```
$ virsh list
```

ステップ 2 ESC VM の IP アドレスを取得するには、次のコマンドを使用します。

```
$ arp -an | grep <ip_address>
```

ステップ 3 SSH を使用して ESC に接続し、プロセスが実行されていることを確認します。

```
$ ssh USERNAME@ESC_IP
```

トラブルシューティングのヒント

ネットワークの状態のため、または ESC の起動に失敗したために SSH アクセスを使用できない場合は、コンソール（ESC VM イメージで有効になっている場合）または VNC アクセスを使用して ESC に接続できます。VNC 経由で ESC VM にアクセスするには、次の手順を実行します。

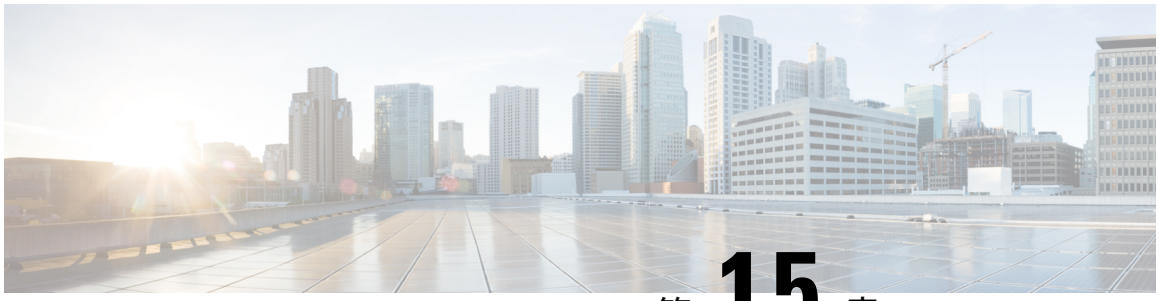
1. VNC ポートを特定します。

   ```
   virsh dumpxml 10 | fgrep vnc
   ```

2. リモート VNC クライアントからの接続を可能にするため、ローカル VNC ポートへの SSH トンネルを作成します。

第 IV 部

Cisco Elastic Services Controller の Amazon Web Services（AWS）へのインストール



第 15 章

Cisco Elastic Services Controller の Amazon Web Services へのインストール

この章では、AWS 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。この章は次のセクションで構成されています。

前提条件

次に、AWS で ESC インスタンスのインストールを開始する前に完了する必要がある前提条件を示します。

（注） ESC AMI イメージが AWS アカウントと共有されている場合は、これらの前提条件を無視して、ESC のインストールに AMI イメージを直接使用することができます。

手順

ステップ 1 AWS CLI を設定します。pip を使用して AWS CLI をインストールできます。詳細については、AWS ドキュメントを参照してください。

ステップ 2 アカウント情報に基づいて AWS CLI のクレデンシャルを設定します。

ステップ 3 Amazon S3 バケットを作成します。ESC イメージをアップロードするには、バケットにこれを使用します。

（注） VM のインポートを許可する vmimport というロールを持っている必要があります。また、このロールに IAM ポリシーをアタッチする必要があります。詳細については、AWS での S3 バケットの作成に関するマニュアルを参照してください。

ステップ 4 ESC ova ファイルから vmdk ファイルを抽出します。

```
$ tar xvf ESC-<latest image file>.ova ESC-<latest image file>-disk1.vmdk
```

AWS での Elastic Services Controller インスタンスのインストール

「前提条件」の項に記載されたタスクを完了すると、次の手順に沿って AWS で ESC インスタンスを展開および起動できます。

手順

ステップ 1 ESC イメージをアップロードして登録します。

a) S3 バケットに vmdk イメージをアップロードします。

```
aws s3 cp <esc-vmdk-file> s3://<S3 bucket name>/
```

b) イメージを登録します。

```
aws ec2 import-image --description "<esc-vmdk-file>" --disk-containers
file://containers.json
```

ステップ 2 ユーザデータを作成します。

a) ESC VM のユーザを作成します。ユーザが存在しない VM にはアクセスできません。Sudo アクセスおよび ssh キーを使用して「admin」ユーザを設定することをお勧めします。

b) write_files コマンドを使用して、ユーザデータに esc-config.yaml を作成します。

インスタンスのタイプに応じて、各インスタンスに最大 15 のインターフェイスを実装できます。

（注） 2 つのインターフェイスを使用する場合は、あらかじめ 2 つのネットワークインターフェイスを作成してください。2 つのインターフェイスが異なるサブネット上にある場合、同じ可用性ゾーンに属している必要があります。AWS コンソールからインスタンスを起動するときに、[インスタンスの詳細設定（Configure Instance Details）] タブにインターフェイスの詳細を追加します。

c) esc_service を有効にして開始します。

完全なユーザデータの例を以下に示します。

```
#cloud-config
# It is recommended to disable password authentication for ssh when ESC runs in public
cloud such as AWS.
ssh_pwauth: False
users:
 - name: admin
   # Put admin in 'esc-user' group, otherwise some scripts of ESC might fail when running
 as admin.
   groups: esc-user
```

```
   gecos: User created by cloud-init
   # This is an example of the hashed password for 'admin'.
   passwd:
$6$rounds=656000$pswsUsR7Iz9NIfA4$7E1sEGV8rhDieNDhc824lYwL3cQ8Rsgp9Nds.OZBe9rG/DE56YWk0kDZoB.DsjATrj9pcBnAe.rSOpWll2r0N/

   # The public key for admin user. Replace it with your public key to login.
   ssh-authorized-keys:
     - ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCqGLe4EVVI/rQy4e4jZUEnc5PvYItc39x5fz9rRggZzpwYzKXSj+UnWQMgvkIai+
Mv5vIPiEYISNZx9PmUKayZaLr/2GiLPmnPNEgyzvjD5v77vV3Ag7eHfLXKLmu7ausYqFKEEbNgSTGC1PWhoz2geY4zND9hS3eMhNvxNSLpbo3ftzamQoqtWSx2aRc81M/
piy6NcBzJ3JeH4rOk9bQ+QxRAYm3bOlq/qRfuoxmrsgd68xAIXeDWyGumEIhXN9MDEcQMIWO54fiPQgkqKbZWztH2EBnE9/B6rZCRBUUvdoQhQt2L/
hbCZN1k+oqQ53rlG/BjT09CGfYbgoHq2v
   # false allows you to sudo with the password.
   lock-passwd: false
   homedir: /home/admin
   # sudo settings
   sudo: ALL=(ALL) ALL
write_files:
 - path: /etc/cloud/cloud.cfg.d/sys-cfg.yaml
   content: |
     network:
       version: 1
       config:
       # You must define the name server when you use the static IP address.
       - type: nameserver
         address:
         - 172.31.0.2
       # Define physical network interface
       - type: physical
         name: eth0
         subnets:
         # Define the static IP address
         - type: static
           address: 172.31.5.66
           netmask: 255.255.240.0
           # Define the routes
           routes:
           - gateway: 172.31.0.1
             # 0.0.0.0 means the default gateway
             network: 0.0.0.0
             netmask: 0.0.0.0
 # ESC service config file
 - path: /opt/cisco/esc/esc-config/esc-cfg.yaml
   content: |
     confd:
       # AAA users for ConfD
       init_aaa_users:
       # Public key for ConfD user 'admin'
       - key:
c3NoLXJzYSBBQUFBQjNOemFDMXljMkVBQUFBREFRQUJBQUFCQVFDeFkwMzByaEMzSXlWekF2bStISVlmMmpkdm
RUZndTTEpCRjVPTjZoUEgvK2FBTkkzb0NCSmJndjhPdjRtVXUvYmlCYmsyS240QW52Ni9ROE1YWGducnZST241MlJuODN2ejRCWTAw
T1h2SzZrT2YrUnZkSDFtNjhscVlrWU9uZVErNEtOak5tQXRwV0huT0xCZE1mZ2pzTmF1SlF1QVJUMEtDS2VBS3k4aUVqSUZpZDhWZ3
NiSlA0aDNpTzdjcTkza0E1ZGFQb0xiNWRKRVp3ZW15WS9ENGp6ZnJUeDVKWFFuMy80SDdaQVZPaWcyNzBGUnlGVkZhNFl1VXNYcDk1d3
QveHdpc0RUREVCYTYydjkxQzdXamtaNy9rYkRlRW9VSU9OZExqdEdvbU84c2JRUUJoZHBVTTZlNXJkeUl2VzQ3YTZYOFA5N2lBR3JrQ09
qMWVHNkYgeGlhb3hpbnlAWElBT1hJTlktTS1SRVhXCg==
         # Note: 'admin' is the only user supported and you cannot change the name here.

         name: admin
         # Hashed password for admin user.
         passwd:
$6$rounds=656000$d4hZHtniblo4/b0m$fD3./1H3jcPlWAENviFlu70i5wKnH9DIasDwIkL.p70UFZlfalzD9O7utLlNdKXwudnNhxIOrvYagkBfq6AWh.

     # No specific settings for esc service. Leave it empty.
     esc_service: {}
```

```
runcmd:
 - [ cloud-init-per, once, escservicestart, sh, -c, "chkconfig esc_service on && service
 esc_service start"]
```

ユーザデータに 2 つのインターフェイスを定義する例を以下に示します。

```
 - path: /etc/cloud/cloud.cfg.d/sys-cfg.yaml
   content: |
     network:
       version: 1
       config:

       - type: physical
         name: eth0

         subnets:

         - type: static
           address: 172.31.5.66
           netmask: 255.255.240.0
           # Define the routes
           routes:
           - gateway: 172.31.0.1
             # 0.0.0.0 means the default gateway
             network: 0.0.0.0
             netmask: 0.0.0.0

       - type: physical
         name: eth1

         subnets:

         - type: static
           address: 172.31.51.220
           netmask: 255.255.240.0
```

ステップ 3　AWS で ESC VM を起動します。

次のいずれかの方法で、ESC VM を起動します。

- **ポータルから起動**：

  1. EC2 管理コンソール、IMAGES/AMI に移動します。インポートしたイメージを選択し、[起動（Launch）] をクリックします。
  2. インスタンスタイプを選択します。インスタンスタイプとして t2.xlarge を選択します。
  3. インスタンスの詳細を設定します。ユーザデータ、ストレージ、タグ名などの詳細を追加します。2 つのインターフェイスを使用する際は、ネットワークインターフェイスの作成と設定をここで行います。
  4. セキュリティグループを設定します。ssh のみを有効にします。
  5. [作成（Launch）] をクリックします。

- **コマンドラインから起動**：イメージ、サブネット、セキュリティグループを選択し、次のコマンドを使用して ESC VM をインスタンス化します。

```
aws ec2 run-instances --subnet-id <subnet id> --image-id <image id>
```

```
--security-group-ids <security group id> --count 1
--instance-type <instance> --key-name <key name>  --user-data <user data file location>
 --associate-public-ip-address
```

（注） ESC は AWS での HA アクティブ/スタンバイのインストールをサポートしていません。

次のタスク

ESC VM を起動した後、`$ sudo escadm status` コマンドを使用して ESC サービスのステータスを確認します。

第 V 部

Cisco Elastic Services Controller の Cisco Cloud Services Platform 2100 へのインストール

第 16 章

Cisco Elastic Services Controller の Cisco Cloud Services Platform 2100 へのインストール

この章では、CSP 2100 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。この章は次のセクションで構成されています。

前提条件

CSP 2100 で ESC インスタンスのインストールを開始するための前提条件は、次のとおりです。

- 仮想 CPU：4 基（最小）
- メモリ：8 GB
- ディスクサイズ：30 GB

Elastic Services Controller インスタンスの CSP 2100 へのインストール

前提条件のセクションに記載されたタスクを完了すると、次の手順を使用して、CSP 2100 で ESC インスタンスを展開および起動できます。次に、CSP 2100 で使用可能な別の展開方法を 3 つ示します。

- シングルインターフェイスとデュアルインターフェイスを使用した ESC
- ESC HA アクティブ/スタンバイのインストール

CSP 2100 のサンプルファイルで使用される変数のリストについては、CSP 2100 のサンプルファイルで使用される変数リスト （111 ページ） を参照してください。

シングルインターフェイスとデュアルインターフェイスを使用した ESC

CSP に ESC をインストールするには、day0 設定ファイルとして次の形式でユーザデータを作成する必要があります。

day0 ファイルを設定ドライブおよびユーザデータとして記述するシングルインターフェイスの例を次に示します。

```
#cloud-config
users:
 - name: admin          # The user's login name
   gecos: admin         # The user name's real name
   groups: esc-user     # add admin to group esc-user
   passwd: $6$saltsalt$9PDBehueUG4XTLEj6BFZA5MDGh/XeQ6QPbf9HYLU3RifHj1
                        # The hash -- not the password itself -- of the password you
want
                        #           to use for this user. You can generate a safe hash
via:
                        #               mkpasswd --method=SHA-512 --rounds=4096
   lock-passwd: false   # Defaults to true. Lock the password to disable password login
                        # Set to false if you want to password login
   homedir: /home/admin # Optional. Set to the local path you want to use. Defaults to
/home/<username>
   sudo: ALL=(ALL) ALL  # Defaults to none. Set to the sudo string you want to use

ssh_pwauth: True        # Defaults to False. Set to True if you want to enable password
 authentication for sshd.
write_files:
# ESC Configuration
- path: /opt/cisco/esc/esc-config/esc-config.yaml
  content: |
    resources:
      confd:
        init_aaa_users:
        - key: c3NoLXJzYSBBQUFBQjNOemFDMXljMkVBQUF
          passwd: $6$rounds=4096$adWFd7LUn2PEUPWtWPl5tCD7pO9bae672T1
        option: start-phase0
      escmanager:
        open_ports:
        - '8080'
        - '8443'
        url:
        - http://0.0.0.0:8080/ESCManager
        - https://0.0.0.0:8443/ESCManager
      esc_service:
        type: group
# Params
- path: /opt/cisco/esc/esc-config/esc_params.conf
  content: |
    default.active_vim=CSP
    default.enable_cascade_deletion=true
# Networking
- path: /etc/sysconfig/network-scripts/ifcfg-eth0
  content: |
    DEVICE="eth0"
    BOOTPROTO="none"
    ONBOOT="yes"
    TYPE="Ethernet"
    USERCTL="yes"
    IPADDR="VAR_NETWORK0_IPADDR"
```

```
    NETMASK="VAR_NETWORK0_NETMASK"
    GATEWAY="VAR_NETWORK0_GATEWAY"
    DEFROUTE="yes"
    NM_CONTROLLED="no"
    IPV6INIT="no"
    IPV4_FAILURE_FATAL="yes"
bootcmd:
- [ cloud-init-per, once, disable_ipv6_eth0, sh, -c, "echo net.ipv6.conf.eth0.disable_ipv6
 = 1 >> /etc/sysctl.conf"]
- [ cloud-init-per, once, update_host_name, sh, -c, "echo VAR_LOCAL_HOSTNAME >>
/etc/hostname && hostnamectl set-hostname VAR_LOCAL_HOSTNAME"]
- [ cloud-init-per, once, update_hosts, sh, -c, "echo 127.0.0.1 VAR_LOCAL_HOSTNAME >>
/etc/hosts"]
- [ cloud-init-per, once, add_name_server, sh, -c, "echo nameserver VAR_NAMESERVER_IP
>> /etc/resolv.conf"]
- [ cloud-init-per, once, add_ntp_server, sh, -c, "echo server VAR_NTP_SERVER iburst >>
 /etc/ntp.conf"]
- [ cloud-init-per, once, enable_ecdsa-sha2-nistp521, sh, -c, "/usr/bin/ssh-keygen -f
/etc/ssh/ssh_host_ecdsa_521_key -t ecdsa -b 521 -N ''"]
- [ cloud-init-per, once, enable_ecdsa-sha2-nistp384, sh, -c, "/usr/bin/ssh-keygen -f
/etc/ssh/ssh_host_ecdsa_384_key -t ecdsa -b 384 -N ''"]
- [ cloud-init-per, once, enable_ssh_rsa, sh, -c, "sed -i '/ssh_host_rsa_key/s/^#//g'
/etc/ssh/sshd_config"]
runcmd:
- [ cloud-init-per, once, apply_network_config, sh, -c, "systemctl restart network"]
- [ cloud-init-per, once, stop_chronyd, sh, -c, "systemctl stop chronyd;systemctl disable
 chronyd"]
- [ cloud-init-per, once, start_ntp, sh, -c, "systemctl enable ntpd;systemctl start
ntpd"]
- [ cloud-init-per, once, set_timezone, sh, -c, "timedatectl set-timezone VAR_TIMEZONE"]
- [ cloud-init-per, once, confd_keygen_root, sh, -c, "/usr/bin/escadm confd keygen --user
 root"]
- [ cloud-init-per, once, confd_keygen_admin, sh, -c, "/usr/bin/escadm confd keygen
--user admin"]
- [ cloud-init-per, once, esc_service_start, sh, -c, "chkconfig esc_service on && service
 esc_service start"] # You must include this line
```

day0 ファイルを設定ドライブおよびユーザデータとして記述するデュアルインターフェイスの例を次に示します。

ESC では、静的 IPv4 を使用してイーサネットベースの物理ネットワークデバイスを設定できます。

```
#cloud-config
users:
 - name: admin          # The user's login name
   gecos: admin         # The user name's real name
   groups: esc-user     # add admin to group esc-user
   passwd: $6$saltsalt$9PDBehueUG4XTLEj6BFZA5MDGh/XeQ6QPbf9HYLU3RifHj1
                        # The hash -- not the password itself -- of the password you
want
                        #           to use for this user. You can generate a safe hash
via:
                        #               mkpasswd --method=SHA-512 --rounds=4096
   lock-passwd: false   # Defaults to true. Lock the password to disable password login
                        # Set to false if you want to password login
   homedir: /home/admin # Optional. Set to the local path you want to use. Defaults to
/home/<username>
   sudo: ALL=(ALL) ALL  # Defaults to none. Set to the sudo string you want to use

ssh_pwauth: True        # Defaults to False. Set to True if you want to enable password
 authentication for sshd.
write_files:
# ESC Configuration
```

```
- path: /opt/cisco/esc/esc-config/esc-config.yaml
  content: |
    resources:
      confd:
        init_aaa_users:
        - key: c3NoLXJzYSBBQUFBQjNOemFDMXljMkVBQUF
          passwd: $6$rounds=4096$adWFd7LUn2PEUPWtWPl5tCD7pO9bae672T1
        option: start-phase0
      escmanager:
        open_ports:
        - '8080'
        - '8443'
        url:
        - http://0.0.0.0:8080/ESCManager
        - https://0.0.0.0:8443/ESCManager
      esc_service:
        type: group
# Params
- path: /opt/cisco/esc/esc-config/esc_params.conf
  content: |
    default.active_vim=CSP
    default.enable_cascade_deletion=true
# Networking
- path: /etc/sysconfig/network-scripts/ifcfg-eth0
  content: |
    DEVICE="eth0"
    BOOTPROTO="none"
    ONBOOT="yes"
    TYPE="Ethernet"
    USERCTL="yes"
    IPADDR="VAR_NETWORK0_IPADDR"
    NETMASK="VAR_NETWORK0_NETMASK"
    GATEWAY="VAR_NETWORK0_GATEWAY"
    DEFROUTE="yes"
    NM_CONTROLLED="no"
    IPV6INIT="no"
    IPV4_FAILURE_FATAL="yes"
- path: /etc/sysconfig/network-scripts/ifcfg-eth1
  content: |
    DEVICE="eth1"
    BOOTPROTO="none"
    ONBOOT="yes"
    TYPE="Ethernet"
    USERCTL="yes"
    IPADDR="VAR_NETWORK1_IPADDR"
    NETMASK="VAR_NETWORK1_NETMASK"
    GATEWAY="VAR_NETWORK1_GATEWAY"
    DEFROUTE="no"
    NM_CONTROLLED="no"
    IPV6INIT="no"
    IPV4_FAILURE_FATAL="yes"
bootcmd:
- [ cloud-init-per, once, disable_ipv6_eth0, sh, -c, "echo net.ipv6.conf.eth0.disable_ipv6
 = 1 >> /etc/sysctl.conf"]
- [ cloud-init-per, once, update_host_name, sh, -c, "echo VAR_LOCAL_HOSTNAME >>
/etc/hostname && hostnamectl set-hostname VAR_LOCAL_HOSTNAME"]
- [ cloud-init-per, once, update_hosts, sh, -c, "echo 127.0.0.1 VAR_LOCAL_HOSTNAME >>
/etc/hosts"]
- [ cloud-init-per, once, add_name_server, sh, -c, "echo nameserver VAR_NAMESERVER_IP
>> /etc/resolv.conf"]
- [ cloud-init-per, once, add_ntp_server, sh, -c, "echo server VAR_NTP_SERVER iburst >>
 /etc/ntp.conf"]
- [ cloud-init-per, once, enable_ecdsa-sha2-nistp521, sh, -c, "/usr/bin/ssh-keygen -f
/etc/ssh/ssh_host_ecdsa_521_key -t ecdsa -b 521 -N ''"]
```

```
- [ cloud-init-per, once, enable_ecdsa-sha2-nistp384, sh, -c, "/usr/bin/ssh-keygen -f
/etc/ssh/ssh_host_ecdsa_384_key -t ecdsa -b 384 -N ''"]
- [ cloud-init-per, once, enable_ssh_rsa, sh, -c, "sed -i '/ssh_host_rsa_key/s/^#//g'
/etc/ssh/sshd_config"]
runcmd:
- [ cloud-init-per, once, apply_network_config, sh, -c, "systemctl restart network"]
- [ cloud-init-per, once, stop_chronyd, sh, -c, "systemctl stop chronyd;systemctl disable
 chronyd"]
- [ cloud-init-per, once, start_ntp, sh, -c, "systemctl enable ntpd;systemctl start
ntpd"]
- [ cloud-init-per, once, set_timezone, sh, -c, "timedatectl set-timezone VAR_TIMEZONE"]
- [ cloud-init-per, once, confd_keygen_root, sh, -c, "/usr/bin/escadm confd keygen --user
 root"]
- [ cloud-init-per, once, confd_keygen_admin, sh, -c, "/usr/bin/escadm confd keygen
--user admin"]
- [ cloud-init-per, once, esc_service_start, sh, -c, "chkconfig esc_service on && service
 esc_service start"] # You must include this line
```

day0 ファイルで使用する ESC パスワードの作成

Cloud-Init day0 ファイルを使用して ESC インスタンスを展開する場合、パスワードは、プレーンテキストではなくハッシュとして渡す必要があります。

ハッシュ化されたパスワードを作成するには、mkpasswd ツールを使用します。次に、mkpasswd ツールを使用して、ハッシュ化されたパスワードを作成する例を示します。

```
~$ mkpasswd --method=SHA-512 --rounds=4096
Password:
$6$rounds=4096$Yo11pRsFO$iT5SGMJ6z8WEnmj8TKMdInblgWeb/UChmrsQs3aspx8j.yUuuhxKk2XScOkerWWxpqD5F0sLfC5kzT5t2xGkL1
```

手順

ステップ 1 CSP へのユーザデータファイルのアップロード

ESC を展開するには、ユーザデータファイルをまず CSP ノードにアップロードする必要があります。

（注） イメージと day0 ファイルをアップロードするパスは、次のとおりです。/osp/repository

```
scp user-data-esc admin@<CSP_IP_ADDRESS>:/osp/repository
```

ステップ 2 ESC VM の展開

ESC VM をホストする CSP ノードに送信されるように設定を編集する必要があります。

次に、シングルインターフェイス用の展開データモデルを示します。デュアルインターフェイスの場合は、2 つのインターフェイスがあります。<name>ESC-SA-2-IF</name>

```
<?xml version="1.0"?>
<services xmlns="http://www.cisco.com/ns/test/service">
  <service>
    <name>VAR_SERVICE_NAME</name>
    <memory>8192</memory> <!-- minimum 8G -->
    <numcpu>4</numcpu> <!-- minimum 4 -->
    <disk_size>30.0</disk_size> <!-- minimum 30G -->
    <disk-resize>true</disk-resize>
    <iso_name>ESC-5_0_0_xxx</iso_name> <!-- the name of the ESC image already on the CSP
 -->
    <power>on</power>
```

```
    <ip>172.20.117.40</ip>
    <!-- add the ip for display in the CSP web/console interfaces -->
    <vnc_password>password1</vnc_password>
    <!-- to secure the VNC console session -->
    <vnics>
      <!-- This interface aligns with eth0 in the user-data file -->
      <vnic>
        <nic>0</nic>
        <vlan>1</vlan>
        <tagged>false</tagged>
        <type>access</type>
        <passthrough_mode>none</passthrough_mode>
        <model>virtio</model>
        <network_name>VAR_NETWORK0_NAME</network_name>
      </vnic>
      <!-- This interface aligns with eth1 in the user-data file -->
      <!-- If not using 2 interfaces, this vnic block can be removed -->
      <vnic>
        <nic>1</nic>
        <vlan>1</vlan>
        <tagged>false</tagged>
        <type>access</type>
        <passthrough_mode>none</passthrough_mode>
        <model>virtio</model>
        <network_name>VAR_NETWORK1_NAME</network_name>
      </vnic>
    </vnics>
    <disk_type>ide</disk_type>
    <day0_filename>user-data-esc</day0_filename> <!-- this name MUST match the name of
the file that was copied to the CSP -->
    <day0-dest-filename>user-data</day0-dest-filename>  <!-- mandatory value -->
    <day0-volume-id>cidata</day0-volume-id>  <!-- mandatory value -->
  </service>
</services>
```

ステップ 3 設定の送信

（ConfD 付属の）Netconf コンソールを使用して、CSP ノードで ESC を展開します。

```
$ netconf-console --port=2022 --host=<CSP_IP_ADDRESS> --user=CSP_ADMIN_USERNAME
--password=CSP_ADMIN_PASSWORD --edit-config=deployESCHA1.xml
```

HA の場合は、2 番目の ESC の設定を使用してコマンドを繰り返します。

ステップ 4 VIM コネクタの設定

ESC が起動したら、VIM コネクタを設定します。

CSP に ESC をインストールする場合、デフォルトでは VIM コネクタは追加されません。VNF を管理するには、VIM コネクタを作成する必要があります。

ステップ 5 VIM コネクタの追加

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。

ESC HA アクティブ/スタンバイのインストール

CSP に ESC をインストールするには、day0 設定ファイルとして次の形式でユーザデータを作成する必要があります。HA の場合は、VM ごとに 1 つのファイルを作成する必要があります。

day0 ファイルで使用する ESC パスワードの作成については、「**day0 ファイルで使用する ESC パスワードの作成**」のセクションを参照してください。

次に、設定ドライブおよびユーザデータとして day0 ファイルを記述するノード 1 に ESC HA アクティブ/スタンバイをインストールする例を示します。

```
user-data sample - HA Node 1
#cloud-config
users:
 - name: admin          # The user's login name
   gecos: admin         # The user name's real name
   groups: esc-user     # add admin to group esc-user
   passwd: $6$saltsalt$9PDBehueUG4XTLEj6BFZA5MDGh/XeQ6QPbf9HYLU3RifHj1
                        # The hash -- not the password itself -- of the password you
want
                        #           to use for this user. You can generate a safe hash
via:
                        #               mkpasswd --method=SHA-512 --rounds=4096
   lock-passwd: false   # Defaults to true. Lock the password to disable password login
                        # Set to false if you want to password login
   homedir: /home/admin # Optional. Set to the local path you want to use. Defaults to
/home/<username>
   sudo: ALL=(ALL) ALL  # Defaults to none. Set to the sudo string you want to use

ssh_pwauth: True        # Defaults to False. Set to True if you want to enable password
 authentication for sshd.

write_files:
# ESC Configuration
- path: /opt/cisco/esc/esc-config/esc-cfg.yaml
  content: |
    ha:
      vri: VAR_NETWORK0_KADVRI
      mode: drbd
      vip: VAR_NETWORK0_KADVIP
      vif: eth0
      nodes:
      - ipaddr: VAR_NETWORK0_IPADDR
      - ipaddr: VAR_NETWORK0_IPADDR2
    confd:
      init_aaa_users:
      - name: admin
        passwd: $6$rounds=4096$adWFd7LUn2PEUPWtWPl5tCD7pO9bae672T1
      escmanager:
        open_ports:
        - '8080'
        - '8443'
        url:
        - http://0.0.0.0:8080/ESCManager
        - https://0.0.0.0:8443/ESCManager
    esc_service: {}
# Params
- path: /opt/cisco/esc/esc-config/esc_params.conf
  content: |
    default.active_vim=CSP
    default.enable_cascade_deletion=true
# Networking
```

```
- path: /etc/sysconfig/network-scripts/ifcfg-eth0
  content: |
    DEVICE="eth0"
    BOOTPROTO="none"
    ONBOOT="yes"
    TYPE="Ethernet"
    USERCTL="yes"
    IPADDR="VAR_NETWORK0_IPADDR"
    NETMASK="VAR_NETWORK0_NETMASK"
    GATEWAY="VAR_NETWORK0_GATEWAY"
    DEFROUTE="yes"
    IPV6INIT="no"
    IPV4_FAILURE_FATAL="yes"
bootcmd:
- [ cloud-init-per, once, disable_ipv6_eth0, sh, -c, "echo net.ipv6.conf.eth0.disable_ipv6
 = 1 >> /etc/sysctl.conf"]
- [ cloud-init-per, once, update_host_name, sh, -c, "echo VAR_LOCAL_HOSTNAME >>
/etc/hostname && hostnamectl set-hostname VAR_LOCAL_HOSTNAME"]
- [ cloud-init-per, once, update_hosts, sh, -c, "echo 127.0.0.1 VAR_LOCAL_HOSTNAME >>
/etc/hosts"]
- [ cloud-init-per, once, add_name_server, sh, -c, "echo nameserver VAR_NAMESERVER_IP
>> /etc/resolv.conf"]
- [ cloud-init-per, once, add_ntp_server, sh, -c, "echo server VAR_NTP_SERVER iburst >>
 /etc/ntp.conf"]
- [ cloud-init-per, once, enable_ecdsa-sha2-nistp521, sh, -c, "/usr/bin/ssh-keygen -f
/etc/ssh/ssh_host_ecdsa_521_key -t ecdsa -b 521 -N ''"]
- [ cloud-init-per, once, enable_ecdsa-sha2-nistp384, sh, -c, "/usr/bin/ssh-keygen -f
/etc/ssh/ssh_host_ecdsa_384_key -t ecdsa -b 384 -N ''"]
- [ cloud-init-per, once, enable_ssh_rsa, sh, -c, "sed -i '/ssh_host_rsa_key/s/^#//g'
/etc/ssh/sshd_config"]
runcmd:
- [ cloud-init-per, once, apply_network_config, sh, -c, "systemctl restart network"]
- [ cloud-init-per, once, stop_chronyd, sh, -c, "systemctl stop chronyd;systemctl disable
 chronyd"]
- [ cloud-init-per, once, start_ntp, sh, -c, "systemctl enable ntpd;systemctl start
ntpd"]
- [ cloud-init-per, once, set_timezone, sh, -c, "timedatectl set-timezone VAR_TIMEZONE"]
- [ cloud-init-per, once, confd_keygen_root, sh, -c, "/usr/bin/escadm confd keygen --user
 root"]
- [ cloud-init-per, once, confd_keygen_admin, sh, -c, "/usr/bin/escadm confd keygen
--user admin"]
- [ cloud-init-per, once, esc_service_start, sh, -c, "chkconfig esc_service on && service
 esc_service start"] # You must include this line
```

手順

ステップ 1 CSP へのユーザデータファイルのアップロード

ESC を展開するには、ユーザデータファイルをまず CSP ノードにアップロードする必要があります。

（注） イメージと day0 ファイルをアップロードするパスは、次のとおりです。/osp/repository

```
scp user-data-esc-ha-1 CSP_ADMIN_USERNAME@<CSP_IP_ADDRESS>:/osp/repository

scp user-data-esc-ha-2 CSP_ADMIN_USERNAME@<CSP_IP_ADDRESS>:/osp/repository
```

ステップ 2 ESC VM の展開

ESC VM をホストする CSP ノードに送信されるように設定を編集する必要があります。

次に、ノード 1 での ESC HA アクティブ/スタンバイの展開データモデルを示します。

```
deployESC-HA-1.xml
<?xml version="1.0"?>
<services xmlns="http://www.cisco.com/ns/test/service">
  <service>
    <name>VAR_SERVICE_NAME</name>
    <memory>8192</memory> <!-- minimum 8G -->
    <numcpu>4</numcpu> <!-- minimum 4 -->
    <disk_size>30.0</disk_size> <!-- minimum 30G -->
    <disk-resize>true</disk-resize>
    <iso_name>ESC-5_0_0_xxx</iso_name> <!-- the name of the ESC image already on the CSP
 -->
    <power>on</power>
    <ip>172.20.117.40</ip>
    <!-- add the ip for display in the CSP web/console interfaces -->
    <vnc_password>password1</vnc_password>
    <!-- to secure the VNC console session -->
    <vnics>
      <!-- This interface aligns with eth0 in the user-data file -->
      <vnic>
        <nic>0</nic>
        <vlan>1</vlan>
        <tagged>false</tagged>
        <type>access</type>
        <passthrough_mode>none</passthrough_mode>
        <model>virtio</model>
        <network_name>VAR_NETWORK0_NAME</network_name>
      </vnic>
      <!-- This interface aligns with eth1 in the user-data file -->
      <!-- If not using 2 interfaces, this vnic block can be removed -->
      <vnic>
        <nic>1</nic>
        <vlan>1</vlan>
        <tagged>false</tagged>
        <type>access</type>
        <passthrough_mode>none</passthrough_mode>
        <model>virtio</model>
        <network_name>VAR_NETWORK1_NAME</network_name>
      </vnic>
    </vnics>
    <disk_type>ide</disk_type>
    <day0_filename>user-data-esc</day0_filename> <!-- this name MUST match the name of
the file that was copied to the CSP -->
    <day0-dest-filename>user-data</day0-dest-filename>  <!-- mandatory value -->
    <day0-volume-id>cidata</day0-volume-id>  <!-- mandatory value -->
  </service>
</services>
```

次に、ノード 2 での ESC HA アクティブ/スタンバイの展開データモデルを示します。

```
deployESC-HA-2.xml
deployESC-HA-1.xml
<?xml version="1.0"?>
<services xmlns="http://www.cisco.com/ns/test/service">
  <service>
    <name>VAR_SERVICE_NAME</name>
    <memory>8192</memory> <!-- minimum 8G -->
    <numcpu>4</numcpu> <!-- minimum 4 -->
    <disk_size>30.0</disk_size> <!-- minimum 30G -->
    <disk-resize>true</disk-resize>
    <iso_name>ESC-5_0_0_xxx</iso_name> <!-- the name of the ESC image already on the CSP
 -->
    <power>on</power>
    <ip>172.20.117.40</ip>
```

```
    <!-- add the ip for display in the CSP web/console interfaces -->
    <vnc_password>password1</vnc_password>
    <!-- to secure the VNC console session -->
    <vnics>
      <!-- This interface aligns with eth0 in the user-data file -->
      <vnic>
        <nic>0</nic>
        <vlan>1</vlan>
        <tagged>false</tagged>
        <type>access</type>
        <passthrough_mode>none</passthrough_mode>
        <model>virtio</model>
        <network_name>VAR_NETWORK0_NAME</network_name>
      </vnic>
      <!-- This interface aligns with eth1 in the user-data file -->
      <!-- If not using 2 interfaces, this vnic block can be removed -->
      <vnic>
        <nic>1</nic>
        <vlan>1</vlan>
        <tagged>false</tagged>
        <type>access</type>
        <passthrough_mode>none</passthrough_mode>
        <model>virtio</model>
        <network_name>VAR_NETWORK1_NAME</network_name>
      </vnic>
    </vnics>
    <disk_type>ide</disk_type>
    <day0_filename>user-data-esc</day0_filename> <!-- this name MUST match the name of
the file that was copied to the CSP -->
    <day0-dest-filename>user-data</day0-dest-filename>  <!-- mandatory value -->
    <day0-volume-id>cidata</day0-volume-id>  <!-- mandatory value -->
  </service>
</services>
```

ステップ 3 設定の送信

（ConfD 付属の）Netconf コンソールを使用して、CSP ノードで ESC を展開します。

```
$ netconf-console --port=2022 --host=<CSP_IP_ADDRESS> --user=<CSP_ADMIN_USERNAME>
--password=<CSP_ADMIN_PASSWORD> --edit-config=deployESC-HA-1.xml

$ netconf-console --port=2022 --host=<CSP_IP_ADDRESS> --user=<CSP_ADMIN_USERNAME>
--password=<CSP_ADMIN_PASSWORD> --edit-config=deployESC-HA-2.xml
```

ステップ 4 VIM コネクタの設定

ESC が起動したら、VIM コネクタを設定します。

CSP に ESC をインストールする場合、デフォルトでは VIM コネクタは追加されません。VNF を管理するには、VIM コネクタを作成する必要があります。

ステップ 5 VIM コネクタの追加

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。

CSP 2100 のサンプルファイルで使用される変数リスト

ユーザデータファイルを作成する場合や ESC を設定する場合、サンプルファイルで使用されている次の変数リストの値を準備する必要があります。

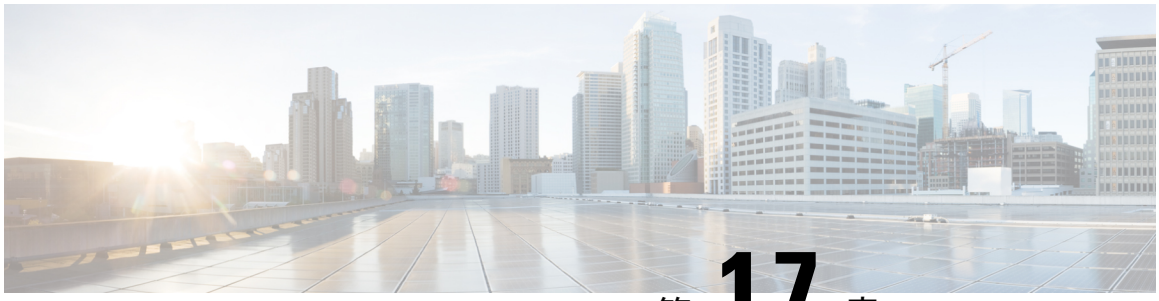
表 *3 :* 変数リスト

変数名	目的
VAR_TIMEZONE	ESC クロックで使用されるタイムゾーン
VAR_SERVICE_NAME	CSP の ESC サービス名
VAR_NTP_SERVER	NTP サーバの IP アドレス
VAR_NETWORK1_NETMASK	eth1 インターフェイスのネットマスク（デュアルインターフェイス ESC）
VAR_NETWORK1_NAME	ESC の eth1 インターフェイスが存在する CSP 上のネットワーク名（デュアルインターフェイス ESC）
VAR_NETWORK1_IPADDR	eth1 インターフェイスの IP アドレス（デュアルインターフェイス ESC）
VAR_NETWORK1_GATEWAY	eth1 インターフェイスのゲートウェイ（デュアルインターフェイス ESC）
VAR_NETWORK0_NETMASK	eth0 インターフェイスのネットマスク
VAR_NETWORK0_NAME	ESC の eth0 インターフェイスが存在する CSP 上のネットワーク名
VAR_NETWORK0_KADVRI	HA に使用される VRRP ID。HA ペアのサブネット内で一意である必要があり、両方の ESC で使用されているものと同じ値である必要があります。 範囲は 1 ～ 254 です。
VAR_NETWORK0_KADVIP	現在のマスター ESC に接続する HA ペアの VIP
VAR_NETWORK0_IPADDR2	他の ESC の eth0 インターフェイスに割り当てられた IP アドレス
VAR_NETWORK0_IPADDR	ESC の IP アドレス（eth0 インターフェイス）
VAR_NETWORK0_GATEWAY	eth0 インターフェイスのゲートウェイ

変数名	目的
VAR_NAMESERVER_IP	DNS サーバの IP アドレス
VAR_LOCAL_HOSTNAME	ESC のホスト名
CSP_IP_ADDRESS	使用する CSP 2100 の IP アドレス

第 VI 部

インストール後のタスク



第 17 章

インストール後のタスク

この章は、次の項で構成されています。

（注） cloud-init day-0 コンフィギュレーション ファイルを変更する場合は、[編集しない（do not edit）] メッセージを無視することをお勧めします。

ESC パスワードの変更

初回ログイン時には、デフォルトのパスワードを強制的に変更する必要があります。ポータルでは、この手順をバイパスすることはできず、デフォルトのパスワードを変更するまでこのページに戻ります。パスワードを初めて変更した後、このセクションで説明されている手順を使用してパスワードを変更できます。また、ユーザが複数のブラウザまたはタブを持っている場合、または同じユーザが 2 台以上のコンピュータからログインしている場合、ユーザの 1 人がパスワードを変更すると、全員がログオフされ、新しいパスワードを再入力するように求められます。ユーザセッションの有効期限は 1 時間であるため、ユーザがポータルで 1 時間アクティブでない場合、ポータルはセッションを期限切れにし、ユーザは再ログインする必要があります。パスワードを忘れた場合は、パスワードを更新したり、ランダムに生成したりすることもできます。

ここでは、パスワードを変更する方法について説明します。

REST の例：

```
sudo escadm rest set --username {USERNAME} --password {PASSWORD}
```

ETSI の例：

```
sudo escadm etsi set --rest_user {USERNAME:PASSWORD}
```

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

ESC をインストールした後、Confd 管理者パスワードを変更するには、次の手順を実行します。

手順

ステップ 1 ESC VM にログインします。

```
$ ssh USERNAME@ESC_IP
```

ステップ 2 管理者ユーザに切り替えます。

```
[admin@esc-ha-0 esc]$ sudo bash
[sudo] password for admin:
```

ステップ 3 ConfD CLI をロードします。

```
$ /opt/cisco/esc/confd/bin/confd_cli  -u admin
```

ステップ 4 新しい管理者パスワードを設定します。

```
$ configure
$ set aaa authentication users user admin password <new password>
```

ステップ 5 変更内容を保存します。

```
$ commit
```

ESC における ConfD の読み取り専用ユーザグループの作成

ESC の ConfD は、readonly という名前の新しいグループを導入することで強化されています。読み取り専用グループのメンバーの場合は、情報を取得するだけで、権限を変更することはできません。

Bootvm のロール名として「readonly」を使用できます。次の例は、ConfD で 2 人のユーザを作成する方法を示しています。1 つは管理者専用で、もう 1 つは読み取り専用です。

```
# bootvm.py name-500-105-100 --user_confd_pass admin:admin --user_confd_pass
readonly:readonly::readonly --user_pass admin:admin --image ESC-5_0_0_105 --net esc-net
```

HA A/A では、aa-day0.yaml のグループ名として「readonly」を使用できます。次が例になります。

```
confd:
  init_aaa_users:
  - group: readonly
    name: admin
```

```
    passwd:
$6$rounds=4096$Ps1JIjKihRTF$fo8XPBxwEHJWWfNiXDnO269rlhAxAhWBcPBfGnZxy1gM3QMxcN8jJ6guWt9Bu.ZkWdPt3hr0OghO73Wr3iDHb0
```

ESC vm が展開された後に、confd の読み取り専用ユーザを作成することもできます。次の手順では、「test」という名前の confd 読み取り専用ユーザと「test」というパスワードを作成します。

```
[root@name-500-155 admin]#  /opt/cisco/esc/confd/bin/confd_cli --user admin
admin connected from 127.0.0.1 using console on name-500-155
admin@name-500-155> configure
Entering configuration mode private
[ok][2019-12-06 18:17:39]
[edit]
admin@name-500-155% set aaa authentication users user test uid 9000 gid 9000 password
$0$test homedir /var/confd/homes/test ssh_keydir /var/confd/homes/test/.ssh
[ok][2019-12-06 18:19:15]
[edit]
admin@name-500-155% set nacm groups group readonly user-name test
[ok][2019-12-06 18:19:41]
[edit]
admin@name-500-155% commit
Commit complete.
[ok][2019-12-06 18:19:47]
[edit]
admin@name-500-155%
```

読み取り専用ユーザとして、リモートで ConfD にアクセスすることもできます。

```
name@my-server-39:~$ ssh -p 2024 readonly@172.29.0.57
readonly@172.29.0.57's password:
readonly connected from 10.85.103.46 using ssh on name-500-156
readonly@name-500-156> configure
Entering configuration mode private
[ok][2019-12-13 16:15:33]
[edit]
readonly@name-500-156% show esc_datamodel
tenants {
    tenant admin {
        description      "Built-in Admin Tenant";
        managed_resource false;
        vim_mapping      true;
    }
}
[ok][2019-12-13 16:15:38]
[edit]
```

読み取り専用の ConfD グループに分類され、変更権限を必要とする場合、ConfD の ESC からアクセス拒否エラーが送信されます。次に、アクセス拒否エラーメッセージの例を示します。

```
$ esc_nc_cli --user readonly --password readonly edit-config dep.xml
Configure
/opt/cisco/esc/confd/bin/netconf-console --port=830 --host=127.0.0.1 --user=readonly
--password=****** --edit-config=/tmp/d.xml
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="1">
  <rpc-error>
    <error-type>application</error-type>
    <error-tag>access-denied</error-tag>
    <error-severity>error</error-severity>
  </rpc-error>
</rpc-reply>
```

ESC が PAM/IDM を使用するように設定されている場合は、次のようにします。IDM サーバのグループは、ConfD のグループに直接マッピングされます。したがって、読み取り専用ユーザは、IDM グループ「readonly」にマッピングする必要があります。

次に例を示します。

```
$ ipa group-find  --all readonly
---------------
1 group matched
---------------
  dn: cn=readonly,cn=groups,cn=accounts,dc=linuxsysadmins,dc=local
  Group name: readonly
  GID: 5003
  Member users: readonly
  ipantsecurityidentifier: S-1-5-21-2222126199-2113948134-574478857-1003
  ipauniqueid: 858b8cda-0d34-11ea-bca8-525400b29c19
  objectclass: top, groupofnames, nestedgroup, ipausergroup, ipaobject, posixgroup,
ipantgroupattrs
----------------------------
Number of entries returned 1
----------------------------
```

Linux アカウントのパスワードの変更

手順

ステップ 1 ESC VM にログインします。

```
$ ssh USERNAME@ESC_IP
```

ステップ 2 ランダムなパスワードを更新または生成するには、次のコマンドを使用します。

```
/usr/bin/pwqcheck
/usr/bin/pwqgen
```

ESC ポータルパスワードの変更

ユーザは、デフォルトの管理者パスワードを更新またはリセットできます。

手順

ステップ 1 ESC VM にログインします。

ステップ 2 ルートユーザに切り替えます。

ステップ 3 デフォルトの管理者パスワードを更新するか、またはランダムにパスワードを生成するには、次のいずれかの方法を使用します。

- Escadm ユーティリティを使用：

  デフォルトの管理者パスワード（admin/********）を更新する場合：

```
[root@anyname-v44-52 admin]# escadm portal set --username admin --password ********
Successfully updated password for username admin
```

ランダムなパスワードを生成する場合：

```
[root@anyname-v44-52 admin]# escadm portal set --username admin
Would you like to use the generated password: "Accent5omit&Wide"?[y|n]y
Successfully updated password for username admin
```

--must_change 変数は、次のログイン時にパスワードを変更するようユーザに要求します。

--must_change 変数は、REST ユーザには適用されません。

```
[root@anyname-v44-52 admin]# escadm portal set --username admin --must_change
Would you like to use the generated password: "Rainy4Dozen&Behave"?[y|n]y
Successfully reset password for username admin. User must change the password at the
 next login.
```

- 特定のパスワードにリセット：

```
[root@anyname-v44-52 admin]# escadm portal set --username admin --password P@55w0rd!
 --must_change
Successfully reset password for username admin. User must change the password at the
 next login.
```

- bootvm コマンドラインを使用：

```
--user_portal_pass admin:<new password>
```

- ESC ポータルを使用：

1. ユーザ名とパスワードを使用して ESC ポータルにログインします。
2. ナビゲーションメニューの [アカウントの設定（Accounts Settings）] を選択します。
3. [古いパスワード（Old Password）] フィールドに古いパスワードを入力し、[新しいパスワード（New Password）] および [パスワードの確認（Confirm Password）] フィールドに新しいパスワードを入力します。
4. [パスワードを更新（Update Password）] をクリックします。

Cisco Elastic Services Controller での着脱可能な認証モジュール（PAM）サポートの設定

ESC サービスを設定して、ESC のユーザ認証にプラグ可能な認証モジュール（PAM）を使用することができます。PAM をサポートする Cisco Elastic Services を使用すると、ESC で LDAP 認証を有効にすることもできます。PAM が設定されていない場合、ESC は ESC サービスごとにデフォルトの認証方式を引き続き使用します。次の表に、各 ESC サービスに対して PAM 認証を有効にするコマンドを示します。

表 *4 : ESC* サービス用の *PAM* の設定

ESC サービス/コンポーネント	コマンド
ESCManager（REST インターフェイス）	`sudo escadm escmanager set --auth PAM:[:<pam_service_name>]`
ESC Monitor（ヘルス API）	`sudo escadm monitor set --auth PAM:[:<pam_service_name>]`
Confd	`sudo escadm confd set --auth PAM:[:<pam_service_name>]`
ポータル	`sudo escadm portal set --auth PAM[:<pam_service_name>]`
ETSI	`sudo escadm etsi set --pam_service <pam_service_name>`

（注）
- ESC VM 内で実行される SSHD サービスは、すでに PAM 認証をデフォルトで使用しています。
- いずれかのコンポーネントが PAM サービスを指定せずに PAM 認証を設定した場合、ESC はデフォルトで PAM サービス「system-auth」になります。

ESC サービス/コンポーネントへの PAM ユーザの追加

次の ESC サービスグループに PAM ユーザを追加できます。

- rest-user
- confd-user
- portal-user
- etsi-user

次の手順を実行して、PAM ユーザを ESC サービス/コンポーネントに追加します。

手順

ステップ 1 ESC VM にログインします。

ステップ 2 ルートユーザに切り替えます。

ステップ 3 次のコマンドを使用して、PAM ユーザを追加します。

```
passwd pamuser
            Changing password for user pamuser.
```

```
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

ステップ 4 次のコマンドを使用して、PAM ユーザを ESC サービス/コンポーネントグループに追加します。

```
usermod -a -G <ESC Service Group> pamuser
```

（注） PAMユーザは、Confdサービスの管理者または読み取り専用グループに追加する必要があります。

Cisco Elastic Services Controller を ID 管理クライアントとして 設定

前提条件

- ID 管理クライアント（IDM）サーバが起動して稼働中であることを確認します。
- ESC で DNS サーバが稼働状態にあることを確認します。DNS サーバが稼働中、ESC インスタンスはホスト名を使用して IDM サーバと対話します。

次の例は、ESC（esc-client-500.linuxsysadmins.local）が IDM サーバ（idmns.linuxsysadmins.local）に到達する様子を示しています。

```
[root@esc-client-500 admin]# ping idmns
PING idmns.linuxsysadmins.local (192.168.222.176) 56(84) bytes of data.
64 bytes from idmns.linuxsysadmins.local (192.168.221.176): icmp_seq=1 ttl=64 time=0.492
 ms
64 bytes from idmns.linuxsysadmins.local (192.168.221.176): icmp_seq=2 ttl=64 time=0.457
 ms
64 bytes from idmns.linuxsysadmins.local (192.168.221.176): icmp_seq=3 ttl=64 time=0.645
 ms
```

IDM は sssd を使用して設定できます。IDM サーバと連携するために ESC サービスの設定を開始するには、ESC の PAM 設定ファイルで、`/etc/pam.d/syste-auth`、`system-auth-esc-sssd` を指定します。

```
# cd /etc/pam.d
# ln -sf system-auth-esc-sssd  system-auth
# ls -al /etc/pam.d/system-auth
lrwxrwxrwx. 1 root root 20 Nov 13 00:39 /etc/pam.d/system-auth -> system-auth-esc-sssd
```

各 ESC サービスに対して IDM 認証を有効にするためのコマンド一覧を次の表に示します。

表 *5 : ESC* サービスに対する *IDM* の設定

ESC サービス/コンポーネントコマンド	コマンド
ESCManager	`# escadm escmanager set --auth PAM:system-auth-esc-sssd`

ESC サービス/コンポーネントコマンド	コマンド
ETSI	`# escadm etsi set --pam_service system-auth-esc-sssd`
ConfD	`# escadm confd set --auth PAM:system-auth-esc-sssd`

ID ポリシーおよび監査クライアントとしての Cisco Elastic Services Controller の設定

ESC をアイデンティティポリシーおよび監査クライアント（IPA）クライアントとして設定するには、次のコマンドを実行します。

```
ipa-client-install
```

次に、IPA クライアントとして ESC を設定する例を示します。

```
[root@esc-client-500 admin]# ipa-client-install --domain linuxsysadmins.local --server
idmns.linuxsysadmins.local --realm LINUXSYSADMINS.LOCAL
WARNING: ntpd time&date synchronization service will not be configured as
conflicting service (chronyd) is enabled
Use --force-ntpd option to disable it and force configuration of ntpd

Autodiscovery of servers for failover cannot work with this configuration.
If you proceed with the installation, services will be configured to always access the
discovered server for all operations and will not fail over to other servers in case of
 failure.
Proceed with fixed values and no DNS discovery? [no]: yes
Client hostname: esc-client-500.linuxsysadmins.local
Realm: LINUXSYSADMINS.LOCAL
DNS Domain: linuxsysadmins.local
IPA Server: idmns.linuxsysadmins.local
BaseDN: dc=linuxsysadmins,dc=local

Continue to configure the system with these values? [no]: yes
Skipping synchronizing time with NTP server.
User authorized to enroll computers: admin
Password for admin@LINUXSYSADMINS.LOCAL:
Successfully retrieved CA cert
    Subject:     CN=Certificate Authority,O=LINUXSYSADMINS.LOCAL
    Issuer:      CN=Certificate Authority,O=LINUXSYSADMINS.LOCAL
    Valid From:  2019-11-12 23:23:32
    Valid Until: 2039-11-12 23:23:32

Enrolled in IPA realm LINUXSYSADMINS.LOCAL
Created /etc/ipa/default.conf
Configured sudoers in /etc/nsswitch.conf
Configured /etc/sssd/sssd.conf
Configured /etc/krb5.conf for IPA realm LINUXSYSADMINS.LOCAL
trying https://idmns.linuxsysadmins.local/ipa/json
[try 1]: Forwarding 'schema' to json server 'https://idmns.linuxsysadmins.local/ipa/json'
trying https://idmns.linuxsysadmins.local/ipa/session/json
[try 1]: Forwarding 'ping' to json server
'https://idmns.linuxsysadmins.local/ipa/session/json'
[try 1]: Forwarding 'ca_is_enabled' to json server
'https://idmns.linuxsysadmins.local/ipa/session/json'
Systemwide CA database updated.
Adding SSH public key from /etc/ssh/ssh_host_ecdsa_521_key.pub
```

```
Adding SSH public key from /etc/ssh/ssh_host_ecdsa_384_key.pub
Adding SSH public key from /etc/ssh/ssh_host_rsa_key.pub
Adding SSH public key from /etc/ssh/ssh_host_ed25519_key.pub
Adding SSH public key from /etc/ssh/ssh_host_ecdsa_key.pub
[try 1]: Forwarding 'host_mod' to json server
'https://idmns.linuxsysadmins.local/ipa/session/json'
Could not update DNS SSHFP records.
SSSD enabled
Configured /etc/openldap/ldap.conf
Configured /etc/ssh/ssh_config
Configured /etc/ssh/sshd_config
Configuring linuxsysadmins.local as NIS domain.
Client configuration complete.
The ipa-client-install command was successful
```

REST 要求の認証

ESC REST および ETSI REST API は、HTTP 基本アクセス認証を使用します。この場合、ESC クライアントは、REST 要求を行うときにユーザ名とパスワードを提供する必要があります。ユーザ名とパスワードは、送信中に Base64 でエンコードされますが、暗号化もハッシュ化もされません。HTTPS は基本認証と組み合わせて使用され、暗号化を提供します。

ここでは、ESC REST および ETSI REST 認証について、インターフェイスのデフォルトパスワードを変更する方法、および ESC クライアントから許可された要求を送信する方法について説明します。

REST 認証

デフォルトでは、REST 認証は有効に設定されています。REST 認証を無効にするには、引数 **--disable-rest-auth** を bootvm に渡すことができます。シスコでは、実稼働環境でこれを使用することは推奨していません。

ESC は、ポート 8443 経由の https 通信もサポートしています。ESC は、クライアントが https 通信を開始するために信頼する必要がある自己署名証明書を生成します。デフォルトでは、REST は HTTP として有効になっており、localhost に制限されています。

ESCは、追加の bootvm.py 引数（**--enable-https-rest** または **--enable-http rest**）を使用して HTTPS または HTTP 上の REST への外部アクセスを有効にしてインストールできます。

必要に応じて、有効になっている外部 REST API のみを使用することをお勧めします。有効にした場合、**bootvm.py --enable-https-rest --user_rest_pass USERNAME:PASSWORD** を使用することを推奨します。

（注） REST API への http および https インターフェイスを有効にするには、**--enable-https-rest** または **--enable-http-etsi-rest** を渡すか、もしくは bootvm.py スクリプトへの引数の両方を渡すようにしてください。REST 認証が無効になっていない場合は、**--user_rest_pass** または **--enable-https-rest** を使用しているときに、**--user_rest_pass** を渡す必要があります。ESC VM が起動した後に https または http を有効にするには、以下に指定された escadm コマンドを使用します。

```
sudo escadm escmanager set --url
http://127.0.0.1:8080/ESCManager,https://0.0.0.0:8443/ESCManager
```

ESC が HA アクティブ/スタンバイモードの場合は、ピアインスタンスの設定を変更する必要があります。

ETSI REST 認証の有効化

ETSI REST http または https インターフェイスが有効になっている場合は、ETSI API へのすべての要求に認証データが含まれている必要があります。**--enable-http-etsi-rest** または **--enable-https-etsi** 引数をそれぞれ使用して、http および https インターフェイスを ESC bootvm.py インストールスクリプトに対して有効にすることができます。

両方のインターフェイスを同時に有効にすることはできますが、実稼働環境では https インターフェイスのみを有効にする必要があります。

（注） ESC VM が起動した後に http または https を有効にするには、次に指定された escadm コマンドを使用します。

```
sudo escadm etsi enable_http_rest
```

または

```
sudo escadm etsi enable_https_rest
```

その後、ETSI サービスを再起動します。

REST インターフェイスパスワードの変更

REST インターフェイスには、デフォルトのユーザ名/パスワード（admin/<default_password>）が 1 つしかありません。パスワードは、起動後に ESC VM CLI から escadm tool を使用して更新できます。REST API を使用してパスワードを更新することもできます。

手順

ステップ 1 ESC VM にログインします。

ステップ 2 既存のパスワードを新しいものに置き換えるには、次のいずれかのオプションを使用します。

- ESC VM CLI から escadm ツールを使用すると、ランダムなパスワードを生成できます。

```
[root@test-v44-52 admin]# escadm rest set --help
usage: escadm rest set [-h] [-v] --username USERNAME [--password PASSWORD]

optional arguments:
  -h, --help           show this help message and exit
  -v, --v, --verbose   show verbose output
  --username USERNAME
  --password PASSWORD  new password or use randomly generated password if no
                       password provided
```

- REST API の使用

  ```
  http://[ESCVM_IP]:8080/ESCManager/v0/authentication/setpassword?userName=admin&password=yourPassword
  ```

  または

  ```
  https://[ESCVM_IP]:8443/ESCManager/v0/authentication/setpassword?userName=admin&password=yourPassword
  ```

ETSI REST インターフェイスのパスワードの変更

ETSI RESTインターフェイスには、デフォルトのユーザ名/パスワード（admin/<default_password>）が 1 つしかありません。パスワードは、起動後に ESC VM CLI から escadm tool を使用して更新できます。

手順

ステップ 1 ESC VM にログインします。

ステップ 2 デフォルトのETSI RESTユーザ名とパスワードを設定するには、次のコマンドを使用します。

```
sudo escadm etsi set --rest_user username:password
```

または

```
[admin@xyz-esc-4-4-0-59-keep ~]$ escadm etsi set --help
usage: escadm etsi set [-h] [-v] [--startup {0,1,true,false,manual,auto}]
[--rest_user REST_USER] [--pam_service PAM_SERVICE]

optional arguments:
-h, --help show this help message and exit
-v, --v, --verbose show verbose output
--startup {0,1,true,false,manual,auto}
set to false|0|manual to disable etsi at startup.
--rest_user REST_USER
Set the user for rest. Format username:password
--pam_service PAM_SERVICE
Specify a PAM service to use for authentication. This
will override the rest user. To revert to the using
the rest user for authentication, supply an empty
string.
```

承認済み REST 要求の送信

許可された要求を送信するには、ESC クライアントが次のヘッダーを使用して要求を送信する必要があります。

```
Authorization: Basic YWRtaW46Y2lzY28xMjM=
```

ここで、*YWRtaW46Y2lzY28xMjM=* は、デフォルトのユーザ名/パスワードの Base64 でエンコードされた文字列です。

ほとんどのライブラリと Web クライアントは、ユーザ名/パスワードを提供するためのインターフェイスを備えており、アプリケーションはユーザ名/パスワードをエンコードし、HTTP 基本認証ヘッダーを追加します。

デフォルトのクレデンシャルを使用する例：

HTTP の場合：

http://[ESCVM_IP]:8080/ESCManager/v0/tenants/

HTTPS の場合：

https://[ESCVM_IP]:8443/ESCManager/v0/tenants/

承認済みの ETSI REST 要求の送信

許可された要求を送信するには、ESC クライアントが次のヘッダーを使用して要求を送信する必要があります。

```
Authorization: Basic YWRtaW46Y2lzY28xMjM=
```

ここで、*YWRtaW46Y2lzY28xMjM=* は、デフォルトのユーザ名/パスワードの Base64 でエンコードされた文字列です。

ほとんどのライブラリと Web クライアントは、ユーザ名/パスワードを提供するためのインターフェイスを備えており、アプリケーションはユーザ名/パスワードをエンコードし、HTTP 基本認証ヘッダーを追加します。

デフォルトのクレデンシャルを使用する例：

HTTP の場合：

http://[ESCVM_IP]: 8250/vnflcm/v1/vnf_lcm_op_occs

HTTPS の場合：

http://[ESCVM_IP]: 8251/vnflcm/v1/vnf_lcm_op_occs

OpenStack ログイン情報の設定

VIM クレデンシャルを渡さずに ESC が展開された場合、ESC VIM および VIM ユーザ API（REST または Netconf API）を介して VIM クレデンシャルを設定できます。

（注） ESC は、次の条件を満たしている場合にのみノースバウンド設定要求を受け入れます。

- ESC には、API（REST/Netconf）を介して設定された VIM または VIM ユーザが含まれています。
- ESC には VIM または VIM ユーザが設定されており、ESC は VIM に到達できます。
- ESC には VIM または VIM ユーザが設定されており、ESC はユーザを認証できます。

Netconf API を使用した設定

- **Netconf を使用した VIM クレデンシャルの提供**：

```
<esc_system_config xmlns="http://www.cisco.com/esc/esc">
  <vim_connectors>
    <!--represents a vim-->
    <vim_connector>
      <!--unique id for each vim-->
      <id>my-server-30</id>
      <!--vim type [OPENSTACK|VMWARE_VSPHERE|LIBVIRT|AWS|CSP]-->
      <type>OPENSTACK</type>
      <properties>
        <property>
          <name>os_auth_url</name>
          <value>http://<os_ip:port>/v3</value>
        </property>
        <!-- The project name for openstack authentication and authorization -->
        <property>
          <name>os_project_name</name>
          <value>vimProject</value>
        </property>
        <!-- The project domain name is needed for openstack v3 identity api -->
        <property>
          <name>os_project_domain_name</name>
          <value>default</value>
        </property>
      </properties>
      <users>
        <user>
          <id>admin</id>
          <credentials>
            <properties>
              <property>
                <name>os_password</name>
                <value>********</value>
              </property>
              <!-- The user domain name is needed for openstack v3 identity api -->

              <property>
                <name>os_user_domain_name</name>
                <value>default</value>
              </property>
            </properties>
          </credentials>
        </user>
```

```
      </users>
    </vim_connector>
  </vim_connectors>
</esc_system_config>
```

（注）

- ESC 3.0 以降では、複数の VIM コネクタがサポートされていますが、1 つの ESC 内では 1 つのタイプの VIM のみがサポートされています。たとえば、すべての VIM コネクタが OpenStack 専用である必要があります。1 つの ESC VIM には 2 つの VIM コネクタを設定できません。1 つは OpenStack、1 つは VMware を指します。
- 1 つの VIM がデフォルトの VIM として選択されます。これは、すべての pre 3.0 設定要求とデータモデルをサポートしています。
- 展開はデフォルトの VIM ではない VIM で行うことができます。デフォルト以外の VIM への展開では、すべてのアウトオブバンドリソース（一時ボリュームを除く）を持つ必要があります。イメージ、フレーバ、ネットワークなどのその他の設定は、デフォルトの VIM ではない VIM で実行できます。
- デフォルトの VIM コネクタは自動プロビジョニングされ、次のシナリオで設定する必要はありません。
  - ESC 起動中に VIM クレデンシャルが渡された場合。
  - 2.3.x から 3.0 にアップグレードする場合。
- Openstack create VIM コネクタのデータモデルの変更は、移行によるアップグレード中に処理されます。「os_tenant_name」および「os_project_domain_name」プロパティは、VIM コネクタのプロパティに移動され、「os_ternant_name」は「os_project_name」に変更されます。
- デフォルトの VIM コネクタでは、正常に認証されると、それらのプロパティを更新できなくなります。
- VIM ユーザは、いつでも削除、再作成、またはそのプロパティを更新できます。

- **Netconf を使用した VIM コネクタの更新**：

```
<esc_system_config xmlns="http://www.cisco.com/esc/esc">
  <vim_connectors>
    <vim_connector nc:operation="replace">
      <id>example_vim</id>
      <type>OPENSTACK</type>
```

```
      <properties>
        <property>
          <name>os_auth_url</name>
          <value>{auth_url}</value>
        </property>
        <property>
          <name>os_project_name</name>
          <value>vimProject</value>
        </property>
        <!-- The project domain name is only needed for openstack v3 identity api
-->
        <property>
          <name>os_project_domain_name</name>
          <value>default</value>
        </property>
        <property>
          <name>os_identity_api_version</name>
          <value>3</value>
        </property>
      </properties>
    </vim_connector>
  </vim_connectors>
</esc_system_config>
```

- Netconf を使用した VIM ユーザの更新：

```
<esc_system_config xmlns="http://www.cisco.com/esc/esc">
  <vim_connectors>
    <vim_connector>
      <id>example_vim</id>
        <users>
          <user nc:operation="replace">
            <id>my_user</id>
            <credentials>
             <properties>
              <property>
                <name>os_password</name>
                <value>********</value>
              </property>
              <!-- The user domain name is only needed for openstack v3 identity api
 -->
              <property>
                <name>os_user_domain_name</name>
                <value>default</value>
              </property>
             </properties>
            </credentials>
          </user>
        </users>
    </vim_connector>
  </vim_connectors>
</esc_system_config>
```

- Netconf を使用した VIM コネクタの削除：

```
<esc_system_config xmlns="http://www.cisco.com/esc/esc">  <vim_connectors>
    <vim_connector nc:operation="delete">
      <id>example_vim</id>
    </vim_connector>
  </vim_connectors>
</esc_system_config>
```

- コマンドを使用した VIM コネクタの削除：

```
$/esc_nc_cli delete-vim-connector <vim connector id>
```

- コマンドを使用した VIM ユーザの削除：

```
$/esc_nc_cli delete-vim-user <vim connector id> <vim user id>
```

REST API を使用して設定

- REST を使用した VIM の追加：

```
POST /ESCManager/v0/vims/
HEADER: content-type, callback

<?xml version="1.0"?>
<vim_connector xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <id>example_vim</id>
  <type>OPENSTACK</type>
  <properties>
    <property>
      <name>os_auth_url</name>
      <value>{auth_url}</value>
    </property>
    <property>
      <name>os_project_name</name>
      <value>vimProject</value>
    </property>
    <!-- The project domain name is only needed for openstack v3 identity api -->
    <property>
      <name>os_project_domain_name</name>
      <value>default</value>
    </property>
    <property>
      <name>os_identity_api_version</name>
      <value>3</value>
    </property>
  </properties>
</vim_connector>
```

- REST を使用した VIM ユーザの追加：

```
POST /ESCManager/v0/vims/{vim_id}/vim_users
HEADER: content-type, callback

<?xml version="1.0"?>
<user xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <id>my_user</id>
  <credentials>
    <properties>
      <property>
        <name>os_password</name>
        <value>********</value>
      </property>
      <!-- The user domain name is only needed for openstack v3 identity api -->
      <property>
        <name>os_user_domain_name</name>
        <value>default</value>
      </property>
    </properties>
  </credentials>
</user>
```

- REST を使用した VIM の更新：

```
PUT /ESCManager/v0/vims/{vim_id}
HEADER: content-type, callback

<?xml version="1.0"?>
<vim_connector xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <!--unique id for each vim-->
  <id>example_vim</id>
  <type>OPENSTACK</type>
  <properties>
    <property>
      <name>os_auth_url</name>
      <value>{auth_url}</value>
    </property>
    <property>
      <name>os_project_name</name>
      <value>vimProject</value>
    </property>
    <!-- The project domain name is only needed for openstack v3 identity api -->
    <property>
      <name>os_project_domain_name</name>
      <value>default</value>
    </property>
    <property>
      <name>os_identity_api_version</name>
      <value>3</value>
    </property>
 </properties>
</vim_connector>
```

- REST を使用して VIM ユーザの更新：

```
PUT /ESCManager/v0/vims/{vim_id}/vim_users/{vim_user_id}
HEADER: content-type, callback

<?xml version="1.0"?>
<user xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <id>my_user</id>
  <credentials>
    <properties>
      <property>
        <name>os_password</name>
        <value>********</value>
      </property>
      <!-- The user domain name is only needed for openstack v3 identity api -->
      <property>
        <name>os_user_domain_name</name>
        <value>default</value>
      </property>
    </properties>
  </credentials>
</user>
```

- REST を使用した VIM の削除：

```
DELETE /ESCManager/v0/vims/{vim_id}
```

- REST を使用した VIM ユーザの削除：

```
DELETE /ESCManager/v0/vims/{vim_id}/vim_users/{user_id}
```

- 各 VIM または VIM ユーザの設定が完了した後の通知例：

```
<?xml version="1.0" encoding="UTF-8"?>
<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
  <eventTime>2016-10-06T16:24:05.856+00:00</eventTime>
  <escEvent xmlns="http://www.cisco.com/esc/esc">
    <status>SUCCESS</status>
    <status_code>200</status_code>
    <status_message>Created vim connector successfully</status_message>
    <vim_connector_id>my-server-30</vim_connector_id>
    <event>
      <type>CREATE_VIM_CONNECTOR</type>
    </event>
  </escEvent>
</notification>
```

特記事項：

- ESC 3.0 では、Openstack VIM の複数の VIM コネクタを追加できます。各 VIM コネクタでは、1 つの VIM ユーザのみを持つことができます。
- VIM ユーザ名とパスワードはいつでも更新できます。VIM エンドポイントは、ESC を使用してリソースを作成した後は更新できません。
- VIM が接続され、VIM ユーザが認証されると、VIM を削除または更新することができなくなります。また、VIM ユーザのみを削除または更新できます。
- VIM プロパティまたは VIM ユーザログイン情報プロパティの名前は大文字と小文字が区別されません。たとえば、OS_AUTH_URL と os_auth_url は ESC にとっては同じです。

ESC 仮想マシンの再設定

ここでは、次のトピックについて取り上げます。

- rsyslog の再設定
- NTP の再設定
- DNS の再設定
- ホストの再設定
- タイムゾーンの再設定

rsyslog の再設定

rsyslog パラメータはオプションです。ESC VM を起動した後にカスタマイズが必要になった場合は、ESC VM（/etc/rsyslog.d/）内のファイルを編集できます。

手順

ステップ 1 **rsyslog ファイルの編集**：

- ブートアップ時のログ転送の設定を指定しなかった場合は、/etc/rsyslog.d/ に /etc/rsyslog.d/log-forwarding.conf のようなファイルを作成できます。
- インストール時にログ転送を指定した場合は、ファイルを編集するだけで済みます。ファイルは /etc/rsyslog.d/20-cloud-config.conf である可能性があります。複数の rsyslog サーバにログを転送するには、このファイルで次の行を編集します。

```
*.* @[server_ip]:port
```

（注）
- サーバの IP アドレスを指定する前に「@@」を入力してください（rsyslog サーバへのログの転送に使用されるプロトコルが TCP である場合）。
- サーバの IP アドレスを指定する前に「@」を入力してください（rsyslog サーバへのログの転送に使用されるプロトコルが UDP である場合）。
- server_ip には、rsyslog サーバの IPv4 アドレスと IPv6 アドレスのいずれかを使用できます。
- IPv6 サーバアドレスが指定されている場合は、server_ip を囲む「[ ]」を「:port#」から分離する必要があります。

rsyslog の設定の詳細については、Red Hat のマニュアルを参照してください。

ステップ 2 **ESC ログファイルの設定**：rsyslog サーバにどの ESC ログファイルを転送するかを設定します。

a) /etc/rsyslog.d/ に移動して、**log-esc.conf** などの設定ファイルを作成または変更します。サンプルとして log-esc.conf のコピーを作成します。

b) rsyslog サーバに転送するすべてのファイルに対して、次のブロックを指定します。

```
$InputFileName /var/log/esc/escmanager.log
$InputFileTag esc-manager:
$InputFileStateFile stat-esc-manager
$InputFileSeverity info
$InputRunFileMonitor
```

次に例を示します。

```
$InputFileName /var/log/esc/file1.log
$InputFileTag file1:
$InputFileStateFile stat-file1
$InputFileSeverity info
$InputRunFileMonitor

$InputFileName /var/log/esc/file2.log
$InputFileTag file2:
$InputFileStateFile stat-file2
$InputFileSeverity info
$InputRunFileMonitor
```

ステップ 3 rsyslog サービスを再起動します。

```
# service rsyslog restart
```

ステップ 4 転送されたログを受信するようにサーバ側を設定します。

a) 指定されたサーバで、/etc/rsyslog.conf に移動し、TCP または UDP に基づいてクライアントからのログをリッスンするかどうかに応じて、以下に示す行をコメント解除します。

```
#$ModLoad imudp
#$UDPServerRun 514
```

b) ファイルを終了します。最後の手順として、このコマンドを実行します。

```
sudo service rsyslog restart
```

サーバは、TCP/UDP を使用してポート 514 でログをリッスンするようになりました。

NTP の再設定

手順

ステップ 1 vi などのテキストエディタで NTP 設定ファイル /etc/ntp.conf を開きます。ファイルがまだ存在しない場合は、新しいファイルを作成します。

```
# vi /etc/ntp.conf
```

ステップ 2 パブリック NTP サーバのリストを追加または編集します。インストール時に NTP サーバを指定しない場合、ファイルに次のデフォルト行が含められますが、必要に応じて自由に変更または拡張できます。

```
server 0.rhel.pool.ntp.org iburst
server 1.rhel.pool.ntp.org iburst
server 2.rhel.pool.ntp.org iburst
server 3.rhel.pool.ntp.org iburst
server <your_ntp_server_ip> iburst
```

各行の最後にある iburst ディレクティブは、初期同期を高速化します。

ステップ 3 サーバのリストアップが完了したら、同じファイルで適切な権限を設定し、localhost のみに無制限のアクセス権を付与します。該当する行が設定ファイルに含まれていることを確認してください。

```
restrict default kod nomodify notrap nopeer noquery
restrict -6 default kod nomodify notrap nopeer noquery
restrict 127.0.0.1
restrict -6 ::1
```

ステップ 4 すべての変更を保存し、エディタを終了して、NTP デーモンを再起動します。

```
# service ntpd restart
```

ステップ 5 起動時に ntpd が開始されていることを確認してください。

```
# chkconfig ntpd on
```

DNS の再設定

手順

ステップ 1 /etc/resolv.conf ファイルには、DNS クライアント（リゾルバ）の設定が含まれています。通常、次のように表示されます。

```
search domain.com
nameserver 8.8.4.4
```

その結果、`/etc/resolv.conf` に次のような記述が含められます。

```
Created by cloud-init on instance boot automatically, do not edit.
;
#Generated by esc-cloud
domain cisco.com
search cisco.com
nameserver 8.8.4.4
```

（注） ファイルを変更する場合は、`do not edit` メッセージを無視することをお勧めします。

ステップ 2 「nameserver」項目の IP アドレスを変更するか、または新しいネームサーバレコードを追加できます。

```
search domain.com
nameserver <your_first_dns_ip>
nameserver <your_second_dns_ip>
```

ステップ 3 ネットワークサービスを再起動します。

```
service network restart
```

ホストの再設定

/etc/hosts ファイルを使用して、ホストを追加、編集、または削除できます。このファイルには、IP アドレスと対応するホスト名が含まれています。IP アドレスが DNS にリストされていないコンピュータがネットワークに含まれている場合は、それらのコンピュータを /etc/hosts ファイルに追加することをお勧めします。

手順

ステップ 1 DNS にリストされていない IP アドレスを/etc/hosts ファイルに追加します。

ステップ 2 ネットワークを再起動して、変更内容を有効にします。

```
service network restart
```

タイムゾーンの再設定

ESC VM の場合、/etc の「localtime」ファイルは、タイムゾーンに関する情報が含まれているファイルのリンクまたはコピーです。/usr/share/zoneinfo からゾーン情報ファイルにアクセスします。タイムゾーンを変更するには、/usr/share/zoneinfo のゾーン情報ファイルで、現在の国や都市、または同じタイムゾーンにある都市を検索し、/etc ファイル内の localtime にリンクします。

```
$ ln \-sf /usr/share/zoneinfo/America/Los_Angeles /etc/localtime
```

ESC 設定と他のインストール後操作の確認

ここでは、escadm ツールを使用したインストール後の各種チェックおよび操作について説明します。

既存の ESC 設定の確認

`escadm dump` コマンドを使用すると、現在の ESC 設定を yaml 形式で表示できます。出力結果には、 ESC のさまざまなサービスが表示されます。

```
$ sudo escadm dump

resources:
  confd:
    init_aaa_users:
    - name: admin
      passwd:
    option: start-phase0
  esc_service:
    group:
    - confd
    - mona
    - vimmanager
    - pgsql
    - escmanager
    - portal
    - monitor
    - snmp
    type: group
  escmanager: {}
  mona: {}
  monitor: {}
  pgsql: {}
  portal: {}
  snmp:
    run_forever: true
  vimmanager: {}
```

VIM 設定の確認

`escadm vim show` コマンドを使用すると、VI Mの設定が正しく入力されているかを確認できます。

```
$ sudo escadm vim show

[
    {
        "status": "CONNECTION_SUCCESSFUL",
        "status_message": "Successfully connected to VIM",
        "type": "OPENSTACK",
        "id": "default_openstack_vim",
        "properties": {
            "property": [
                {
                    "name": "os_auth_url",
                    "value": "http://10.85.103.143:35357/v3"
                }
            ]
        }
    }
]
```

ESC サービスのスタートアップ問題に関するトラブルシューティング

問題：インストール時に `sudo escadm status` を使用して ESC サービスのステータスを確認する際に発生する問題

原因：サービスの中には、開始に時間がかかるものや、開始時に問題が発生するものがあります。

解決策：

1. 次のいずれかの方法で、問題を特定します。
   - ログ /var/log/esc/escadm.log の確認

     ```
     $ cat /var/log/esc/escadm.log
     2017-06-01 20:35:02,925: escadm.py(2565): INFO: promote drbd to master...
     2017-06-01 20:35:02,934: escadm.py(2605): INFO: Waiting for at least one drbd to
      be UptoDate...
     2017-06-01 20:35:02,942: escadm.py(2616): INFO: Waiting for peer drbd node to be
      demoted...
     2017-06-01 20:35:14,008: escadm.py(2423): INFO: mount: /dev/drbd1
     /opt/cisco/esc/esc_database
     2017-06-01 20:35:14,017: escadm.py(1755): INFO: Starting filesystem service: [OK]
     2017-06-01 20:35:15,039: escadm.py(1755): INFO: Starting vimmanager service: [OK]
     2017-06-01 20:35:16,116: escadm.py(1755): INFO: Starting monitor service: [OK]
     2017-06-01 20:35:17,163: escadm.py(1755): INFO: Starting mona service: [OK]
     2017-06-01 20:35:18,440: escadm.py(1755): INFO: Starting snmp service: [OK]
     2017-06-01 20:35:21,397: escadm.py(1770): INFO: Starting confd service:[FAILED]
     2017-06-01 20:35:28,304: escadm.py(1755): INFO: Starting pgsql service: [OK]
     2017-06-01 20:35:29,331: escadm.py(1755): INFO: Starting escmanager service: [OK]
     2017-06-01 20:35:30,354: escadm.py(1755): INFO: Starting portal service: [OK]
     2017-06-01 20:35:31,523: escadm.py(1755): INFO: Starting esc_service service:
     [OK]
     ```

   - ESC サービスの詳細出力を表示するには、「-v」を escadm ステータスに追加します。

```
$ sudo escadm status --v
0 ESC status=0 ESC HA Master Healthy
pgsql (pgid 61397) is running
vimmanager (pgid 61138) is running
monitor (pgid 61162) is running
mona (pgid 61190) is running
drbd is master
snmp (pgid 61541) is running
filesystem (pgid 0) is running
<<service>> is dead
keepalived (pgid 60838) is running
portal (pgid 61524) is running
confd (pgid 61263) is running
escmanager (pgid 61491) is running
```

2. 問題のあることが特定されたサービスのステータスを確認し、これらのサービスを手動で開始します。

```
$ sudo escadm <<service>> status// If the status is stopped or dead, manually start
 the services using the next command.

$ sudo escadm <<service>> start --v
```

ESC ポータルへのログイン

（注）

- ESC ポータルはデフォルトで有効になっています。インストール時に ESC ポータルが無効になっていないことを確認する必要があります。ESC ポータルの有効化または無効化の詳細については、「QCOW イメージを使用した Cisco Elastic Services Controller のインストール （9 ページ）」を参照してください。
- ESC ポータルへの初回ログイン時に、デフォルトパスワードの変更を求められます。

ESC ポータルにログインするには、次の手順を実行します。

始める前に

- ESC のインスタンスを登録します。ESC のインスタンスの登録における詳細については、次を参照してください。QCOW イメージを使用した Cisco Elastic Services Controller のインストール （9 ページ）
- ユーザ名とパスワードを取得していることを確認します。

手順

ステップ 1 Web ブラウザを使用して、ESC とポート 8443 の IP アドレスを入力します。

例：

たとえば、ESC の IP アドレスが 192.0.2.254 の場合は、次のように入力します。

https://192.0.2.254: 8443 [https 経由でログイン]

セキュリティ アラート メッセージが表示されます。

ステップ 2 [Yes] をクリックしてセキュリティ証明書を受け入れます。ログイン ページが表示されます。

ステップ 3 ユーザ名とパスワードを入力して、[ログイン（Login）] をクリックします。

初回ログイン時には、ログインページが再表示され、パスワードの変更を求められます。

ステップ 4 [古いパスワード（Old Password）] フィールドに古いパスワードを入力し、[新しいパスワード（New Password）] および [パスワードの確認（Confirm Password）] フィールドに新しいパスワードを入力します。

ステップ 5 [パスワードの更新（Update Password）] をクリックするか、Enter を押します。

（注）
- UI が応答しなくなった場合は、ESC シェルプロンプトから **sudo escadm portal restart** を実行して UI を再起動します。
- ESC ポータルは 1 人のユーザのみをサポートします。
- 現在、事前インストールされた自己署名証明書はHTTPSをサポートしています。ESCポータルの処理を進める前に、ユーザは自己署名証明書を確認する必要があります。
- HTTPS 通信モードでは、OpenStack によって返される URL プロトコルタイプが HTTPS ではない場合、VNF コンソールへのアクセスが無効になることがあります。セキュリティ上の理由から、HTTPS で実行している間は、安全性の低い通信は拒否されます。

第 VII 部

Cisco Elastic Services Controller のアップグレード

第 18 章

メンテナンスモードでの ESC

この章は、次の章で説明されています。

ESC をメンテナンスモードにする

ESC データベースをバックアップおよび復元するには、ESC をメンテナンスモードにする必要があります。この操作を行うには、次のセクションで指定されているように、escadm ツールを使用します。

始める前に

（注） ESC リリース4.4 以降、HA アクティブ/スタンバイの切り替え前または DB の復元前に ESC がメンテナンスモードだった場合は、HA アクティブ/スタンバイの切り替え後または DB の復元後も ESC はメンテナンスモードのままになります。

メンテナンスモードの間、

- ノースバウンド要求は ESC によってブロックされ、ESC はメンテナンスモード通知で応答します。
- REST 要求のみが、ESC は一時的に使用不可になっているという応答を受信します。ConfD 要求は、メンテナンスモード拒否メッセージを取得するか、または（テナントがすでに存在する場合のテナント要求の作成など）すべての冪等要求に対する OK メッセージを取得します。
- モニタリングアクションが一時停止されます。
- すべての進行中の要求とトランザクションは、進行を継続します。

escadm ツールの使用

ESC は、escadm ツールを使用してメンテナンスモードにすることができます。

手順

ステップ 1 VM シェルから ESC をメンテナンスモードにします。

```
sudo escadm op_mode set --mode=maintenance
```

```
Set mode to MAINTENANCE
Operation Mode = MAINTENANCE
```

ステップ 2 操作モードを照会する場合にはいつでも、

```
sudo escadm op_mode show
```

例：

```
Operation Mode = OPERATION
```

ステップ 3 実行中のトランザクションがないときに、メンテナンスモードにします。escadm ツールで ipt_check フラグを使用して、ESC に進行中のトランザクションがない場合にのみ、ESC をメンテナンスモードにすることができます。ESC に進行中のトランザクションがあり、ESC をメンテナンスモードにしない場合は、フラグを true に設定します。

```
sudo escadm op_mode set --mode=maintenace --ipt_check=true
```

[ipt_check] オプションが true に設定されている場合、escadm ツールは実行中の操作があるかどうかをチェックします。実行中の操作がある場合、escadm ツールは ESC をメンテナンスモードにしません。

ESC を操作モードにする

escadm ツールを使用して、ESC を操作モードにします。

```
sudo escadm op_mode set --mode=operation
```

応答は次のとおりです。

```
Set mode to OPERATION
Operation Mode = OPERATION
```

次のコマンドを使用して、ESC の動作モードをいつでも確認できます。

```
sudo escadm op_mode show
```

ESC スタンドアロンインスタンスからのデータベースのバックアップ

- 次の前提条件を考慮する必要があります。

- データベースを保存し、バックアップログを作成するには、3 番目のマシンが必要です。
- ESC はデータベーススキーマのダウングレードをサポートしていません。データベースを以前の ESC バージョンに復元すると、予期しない問題が発生する可能性があります。

- バックアッププロセスを開始する前に、外部ストレージスペースを確認します（OpenStack コントローラ内、または ESC によってアクセス可能なシステム）。バックアップ/復元は汎用形式で表現でき、escadm ツール（scp://<username>:<password>@<backup_ip>:<filename>）で使用されます。この形式では、3 番目のマシンのログイン情報、IP アドレス、およびファイルストレージパスが必要です。また、localhost IP をバックアップ IP として使用し、ESC VM 内にデータベースをバックアップしてから、そのファイルを外部ストレージにコピーすることもできます。

スタンドアロン型 ESC または HA アクティブ/スタンバイ（マスターノード）から ESC データベースをバックアップするには、次の手順を実行します。

手順

ステップ 1 ESC VM にログインして、メンテナンスモードに設定し、次を実行します。

```
$ sudo escadm op_mode set --mode=maintenance
```

ステップ 2 ESC がメンテナンスモードになっていることを確認するには、次のコマンドを実行します。

```
$ sudo escadm op_mode show
```

ステップ 3 データベースをバックアップします。ルートユーザとして次のコマンドを実行します。

```
# sudo escadm backup --file /tmp/db_file_name.tar.bz2
scp://<username>:<password>@<backup_vm_ip>:<filename>
```

ステップ 4 ESC を操作モードに戻すには、次のコマンドを実行します。

```
$ sudo escadm op_mode set --mode=operation
$ sudo escadm op_mode show
```

ステップ 5 古いESC VMからすべてのログを収集し、バックアップします。ルートユーザとして以下のコマンドを実行します。

```
# sudo escadm log collect
```

タイムスタンプログファイルは、/var/tmp/esc_log <timestamp>.tar.bz2 に生成されます。

（注） ダイナミック マッピング ファイルが ESC サービスによって使用されている場合は、ダイナミック マッピング ファイルを ESC ログと同じタイミングでバックアップする必要があります。ダイナミック マッピング ファイルのデフォルトパスは */opt/cisco/esc/esc-dynamic-mapping/dynamic_mappings.xml* です。

ステップ 6 データベースのバックアップが正常に完了したら、Horizon/Kilo または Nova コマンドを使用して、古い ESC VM をシャットダウンします。VMware vSphere を基盤とした ESC VM インスタンスの場合は、VMware クライアントダッシュボードを使用してプライマリインスタンスをシャットダウンします。OpenStack で VM をシャットダウンする例を以下に示します。

```
$ nova stop OLD_ESC_ID
```

ステップ 7 古い VM から古いポートを切り離し、古い ESC ノードの名前を変更します。OpenStack で VM を分離し、名前を変更する例を以下に示します。

```
nova interface-detach ESC_NAME port-id-of-ESC_NAME
nova rename ESC_NAME ESC_NAME.old
```

VMWare で、古い VM に別の IP アドレスを割り当ててから、古い VM の名前を変更します。

ESCHAアクティブ/スタンバイインスタンスのデータベースのバックアップ

- 次の前提条件を考慮する必要があります。
  - データベースを保存し、バックアップログを作成するには、3 番目のマシンが必要です。
  - ESC はデータベーススキーマのダウングレードをサポートしていません。データベースを以前の ESC バージョンに復元すると、予期しない問題が発生する可能性があります。
- バックアッププロセスを開始する前に、使用可能な外部ストレージスペースがあることを確認します（OpenStack コントローラ内、または ESC によってアクセス可能なシステム）。バックアップ/復元は汎用形式で表現でき、escadm ツール（scp://<username>:<password>@<backup_ip>:<filename>）で使用されます。この形式では、3 番目のマシンのログイン情報、IP アドレス、およびファイルストレージパスが必要です。また、localhost IP をバックアップ IP として使用し、ESC VM 内にデータベースをバックアップしてから、そのファイルを外部ストレージにコピーすることもできます。

スタンドアロン型 ESC または HA アクティブ/スタンバイ（マスターノード）から ESC データベースをバックアップするには、次の手順を実行します。

手順

ステップ 1 スタンバイ ESC ノードで次の手順を実行します。

a) SSH を使用してスタンバイ ESC インスタンスに接続します。

```
$ ssh <username>@<backup_vm_ip>
```

b) ESC インスタンスがスタンバイになっていることを確認し、スタンバイ ESC HA アクティブ/スタンバイインスタンス名をメモします。

```
$ sudo escadm status --v
```

「BACKUP」という出力値が表示される場合、そのノードはスタンバイ ESC ノードです。

c) アクセス権を管理者ユーザに変更します。

```
sudo bash
```

d) スタンバイ ESC VM からすべてのログを収集し、バックアップします。

```
$ sudo escadm log collect
```

タイムスタンプログファイルは、/var/tmp/esc_log <timestamp>.tar.bz2 に生成されます。

e) OpenStack Kilo/Horizon で Nova コマンドまたは VMware クライアントを使用して、スタンバイ ESC インスタンスをシャットダウンします。OpenStack で VM をシャットダウンする例を以下に示します。

```
$ nova stop OLD_ESC_STANDBY_ID
```

ステップ 2 マスター ESC ノードで次の手順を実行します。

a) SSH を使用してプライマリ ESC インスタンスに接続します。

```
$ ssh <username>@<master_vm_ip>
```

b) アクセス権を管理者ユーザに変更します。

```
$ sudo bash
```

c) ESC インスタンスがマスターであることを確認し、マスター ESC HA アクティブ/スタンバイインスタンス名をメモします。

```
$ sudo escadm status --v
```

「MASTER」という出力値が表示される場合、そのノードはプライマリ ESC ノードです。

d) ESC HA アクティブ/スタンバイのマスターノードからデータベースファイルをバックアップします。

```
$ sudo escadm backup --file /tmp/db_file_name.tar.bz2
scp://<username>:<password>@<backup_vm_ip>:<filename>
```

e) マスター ESC VM からログを収集し、バックアップします。

```
$ sudo escadm log collect
```

タイムスタンプログファイルは、/var/tmp/esc_log <timestamp>.tar.bz2 に生成されます。

（注） ダイナミック マッピング ファイルが ESC サービスによって使用されている場合は、ダイナミックマッピングファイルを ESC ログと同じタイミングでバックアップする必要があります。ダイナミック マッピング ファイルのデフォルトパスは */opt/cisco/esc/esc-dynamic-mapping/dynamic_mappings.xml* です。

ステップ 3 OpenStack Kilo/Horizon で Nova コマンドを使用して、プライマリ ESC インスタンスをシャットダウンします。VMware vSphere を基盤とした ESC VM インスタンスの場合は、VMware クライアントダッシュボードを使用してプライマリインスタンスをシャットダウンします。OpenStack で VM をシャットダウンする例を以下に示します。

```
$ nova stop OLD_ESC_MASTER
```

ESC HA アクティブ/スタンバイインスタンスが正常にシャットダウンされたかどうかを確認するには、**nova list** コマンドまたは **nova show OLD_ESC_STANDBY** コマンドを使用します。

ステップ 4 （OpenStack の場合のみ）古い ESC VM からポートを切り離し、古い VM の名前を変更します。

アップグレードされた ESC VM を古いインスタンスと同じ IP アドレスや同じインスタンス名で稼働する必要がある場合は、各インスタンスからポートを切り離し、古い ESC VM をシャットダウンしてから、古い ESC インスタンス名を変更します。

古い VMware プライマリインスタンスを使用する場合は、別の IP アドレスを割り当て、VM 名を変更します。古い VMware プライマリインスタンスを使用しない場合は、古い VM を削除し、アップグレードした新規 VMware プライマリインスタンスに同じ IP アドレスを使用できます。古い VM を削除した後、古いインスタンス名と IP アドレスを引き続き使用できます。

ポートを切り離して、古い VM 名を変更するための OpenStack コマンドを以下に示します。

```
nova interface-list ESC_NAME
nova interface-detach ESC_NAME port-id-of-ESC_NAME
nova rename ESC_NAME ESC_NAME.old
```

ESC データベースの復元

始める前に

データベースを復元するには、次の手順を実行します。

- スタンドアロン型 ESC インスタンスで、ESC サービスを停止します。`# sudo escadm stop` を実行します。
- HA アクティブ/スタンバイタイプのインスタンスでは、最初にバックアップの escadm を停止し、その後にマスターの ESC HA アクティブ/スタンバイインスタンスを停止します。`# sudo escadm stop` を実行します。
- すべてのサービスを停止する必要があります。ステータスを確認するには、`# sudo escadm status --v` を実行します。

手順

ステップ 1 データベースを復元します。ESC VM から次のコマンドを実行します。

```
$ scp <username>@<server_ip>:/path/db.tar.bz2     /tmp/
$ sudo escadm restore -file /tmp/db.tar.bz2
```

ステップ 2 URL に ESC パスワードを入力します。または、上記のコマンドを実行した後に手動で ESC パスワード入力します。

ステップ 3 ESC サービスを再起動して、次のコマンドを実行すると、データベースの復元が完了します。

```
$ sudo escadm restart
```

（注） ESC メンテナンスモードでは、ノースバウンド要求と VNF のモニタリングがブロックされます。ただし、ESC がメンテナンスモードに入る前に、ノースバウンド要求を受けて実行中のトランザクションがあった場合、バックアップおよび復元プロセスによって、それらのトランザクションに制限が生じる可能性があります。

- バックアップと復元によってトランザクションが中断された場合、ESC は、展開、ネットワークの構築、およびサブネット作成要求に対してエラーを報告します。ノースバウンドはこれらのエラーメッセージを処理しますが、場合によっては（OpenStack から UUID を取得する前に ESC が中断されるなど）、ネットワークやサブネットの漏えいが発生することがあります。
- ESC は、サービスチェーンのアップグレードに対してエラーを報告し、サービスを再作成するためのサービスチェーンの展開解除と展開（ダウングレードとアップグレードではなく）を要求します。

第 19 章

Cisco Elastic Services Controller のアップグレード

Cisco Elastic Service Controller は、次の 2 種類のアップグレードをサポートしています。

- **Backup and Restore アップグレード**：このアップグレードプロセスでは、ESC キープアライブデーモンの停止（ESC HA の場合）、データベースのバックアップ、ESC インスタンスの停止と名前変更（または削除）、ESCインスタンスの再インストール、データベースの復元が行われます。ESC 5.2 アップグレードでサポートされる ESC バージョンについては、次の表を参照してください。
- **インサービスアップグレード**：ESC は、最小限のダウンタイムで、アクティブ/スタンバイ高可用性ノードのインサービスアップグレードをサポートします。

ESCインスタンスは、スタンドアロンインスタンスとして、または高可用性ペアとしてアップグレードできます。アップグレード手順は、スタンドアロンの場合と高可用性ペアの場合で異なります。

この章では、ESC スタンドアロンインスタンスと ESC 高可用性インスタンスをアップグレードする方法について、それぞれの手順を示します。ESCインスタンスのアップグレードを決定する前に、次の手順を確認する必要があります。インストールのシナリオ （7 ページ） を参照し、インストールシナリオの詳細を確認してください。

- ESC は、先行する 2 つのマイナーリリースからの直接アップグレードのみをサポートします。たとえば ESC 2.3 は、ESC 2.1 と ESC 2.2 からの直接アップグレードをサポートします。直接アップグレードでサポートされているバージョンよりも古いリリースの場合は、段階的なアップグレードを実行する必要があります。

  （注） ESC メジャーリリースの例：ESC 3.1

  ESC マイナーリリースの例：ESC 3.1.1

  ESC パッチリリースの例：ESC 3.1.0.116

- RPM パッケージ（この章では RPM アップグレードと呼ばれる）を使用した ESC のアップグレードは、同じリリース番号の ESC パッチリリース間の ESC アップグレードのみに

適用されます。たとえば、ESC 3.1.0.116 から ESC 3.1.0.150 へのアップグレードのような場合です。

- マイナーリリース間（ESC 2.3.1 から ESC 2.3.2 へのアップグレードなど）またはメジャーリリース間（ESC 3.0 から ESC 4.0 へのアップグレードなど）で ESC をアップグレードする場合は、qcow2 イメージを使用した Backup and Restore アップグレードプロセスでアップグレードできます。

- ESC をアップグレードするには、ESC のインストールプロセスに精通している必要があります。
  - OpenStack については、OpenStack のインストール手順を参照してください。第 4 章「OpenStack への Cisco Elastic Services Controller のインストール」を参照してください。
  - VMware については、VMware のインストール手順を参照してください。第 7 章「Cisco Elastic Services Controller の VMware vCenter へのインストール」を参照してください。
  - ESC HA については、ESC HA のインストール手順を参照してください。第 5 章「OpenStack での高可用性の設定」と第 8 章「VMware での高可用性の設定」を参照してください。

表 *6 : ESC 5.2* へのアップグレードでサポートされる *ESC* バージョン

仮想インフラストラクチャ マネージャ	**Backup and Restore** アップグレードでサポートされるバージョン	インサービスアップグレードでサポートされるバージョン
OpenStack	5.1、5.0	5.1、5.0
VMware	5.1、5.0	5.1、5.0
CSP	5.1、5.0	5.1、5.0

特記事項

- 新しい ESC バージョンにアップグレードした後も、ESC サービスは、以前のリリースで展開されたすべての VNF のライフサイクルを管理します。既存の VNF に新しい機能（新しいデータモデルを使用）を適用するには、既存の VNF を展開解除して再展開する必要があります。
- アップグレードは、アクティブ/アクティブ HA のみでサポートされます。

（注） ETSI 展開では、ESC のアップグレードで VIP を変更しないでください。

ETSI の REST スキーマを変更した場合、http から https への変更中、ESC は、既存の展開に対する ESC コアからのリカバリ通知の送信を停止します。

スタンドアロン ESC インスタンスのアップグレード

スタンドアロン ESC インスタンスをアップグレードするには、次のタスクを実行します。

1. ESC データベースをバックアップします。詳細については、「ESC スタンドアロンインスタンスからのデータベースのバックアップ」を参照してください。

（注） 展開で使用されるカスタムスクリプトをバックアップし、データベース復元の前にそれらを復元するには、バックアップにもスクリプトを再インストールする必要があります。

2. ESC インスタンスを再展開します。詳細については、以下の「**アップグレードを目的とした ESC の展開**」のセクションを参照してください。
3. 新しい ESC インスタンスで ESC データベースを復元します。詳細については、以下の「**ESC データベースの復元**」の項を参照してください。

アップグレードを目的とした ESC の展開

古い ESC VM をバックアップしてシャットダウンした後、新規またはアップグレード済み（新しい ESC パッケージに基づく）の ESC VM をインストールする必要があります。ESC インストールのすべてのパラメータは、古い ESC VM 展開と同じである必要があります。

- OpenStack の場合、新しいイメージ名で Glance コマンドを使用して新しい ESC qcow2 イメージを登録した後、新しい bootvm.py スクリプトと新しいイメージ名を使用して ESC VM をインストールする必要があります。

（注） OpenStack で、古い ESC VM がフローティング IP で割り当てられていた場合は、インストール後に新しい ESC VM を同じフローティング IP に関連付ける必要があります。

- VMWare の場合、新しい ESC OVA ファイルを使用して ESC VM をインストールする必要があります。他のすべての設定とプロパティ値は、古い VM と同じである必要があります。

ESC データベースの復元

次の手順を使用して、新しい ESC インスタンスで ESC データベースを復元します。

手順

ステップ 1 SSH を使用して新しい ESC インスタンスに接続します。

```
$ ssh USERNAME@NEW_ESC_IP
```

ステップ 2 ESC サービスを停止します。

```
$ sudo escadm stop
```

ステップ 3 ESC サービスのステータスをチェックして、すべてのサービスが停止していることを確認します。

```
$ sudo escadm status
```

ステップ 4 データベースファイルを復元します。

```
$ scp://<username>:<password>@<backup_ip>:<filename>
$ sudo escadm restore --file /path/where/file/scp-ed/to/db.tar.bz2
```

ステップ 5 ESC サービスを開始します。

```
$ sudo escadm start
```

ESC サービスが開始されると、スタンドアロン ESC のアップグレードが完了します。新しい ESC VM で `$ sudo escadm status` を実行することにより、新しい ESC サービスの正常性を確認できます。

ステップ 6 Openstack で、データベースを正常に復元した後、古い ESC インスタンスを削除します。

```
$ nova delete OLD_ESC_ID
```

特記事項：

新しい ESC バージョンにアップグレードした後も、ESC サービスは、古いバージョンによって展開されたすべての VNF のライフサイクル管理を継続します。ただし、古いバージョンの ESC によって展開された VNF への新しい機能（新しいデータモデルを使用）の適用は保証されません。新しい ESC バージョンの新しい機能を既存の VNF に適用する場合は、既存の VNF を展開解除して再展開する必要があります。

ESC HA アクティブ/スタンバイインスタンスのアップグレード

ESC HA アクティブ/スタンバイノードをアップグレードするには、次のタスクを実行します。

1. 古い ESC HA アクティブ/スタンバイプライマリインスタンスからデータベースをバックアップします。詳細については、「ESC HA アクティブ/スタンバイインスタンスのデータベースのバックアップ」を参照してください。

（注） 展開で使用されているカスタムスクリプトをバックアップし、データベースを復元する前にそれらを復元するには、バックアップにもスクリプトをコピーする必要があります。

2. 新しい ESC バージョンに基づいて、新しい ESC HA アクティブ/スタンバイノードを展開します。詳細については、次の「**アップグレードを目的とした ESC の展開**」のセクションを参照してください。
3. プライマリ ESC インスタンス（スタンバイ ESC インスタンスはプライマリ ESC インスタンスと同期）のデータベースを復元します詳細については、次の「**新しいマスターおよびスタンバイインスタンスでの ESC データベースの復元**」のセクションを参照してください。

アップグレードを目的とした ESC HA アクティブ/スタンバイノードの展開

2 つの古い ESC VM をバックアップしてシャットダウンした後、新しい ESC パッケージに基づいて新しい ESC VM をインストールします。

- OpenStack の場合、新しいイメージ名と Glance コマンドを使用して新しい ESC qcow2 イメージを登録した後、新しい bootvm.py スクリプトと新しいイメージ名を使用して ESC VM をインストールする必要があります。他のすべての bootvm.py 引数は、古い VM のセットアップに使用したものと同じである必要があります。
- VMWare の場合、HA のアクティブ/スタンバイペアを VMware で起動するには 2 つの手順があります。1）2 つのスタンドアロンインスタンスを設定 2）HA アクティブ/スタンバイ情報を使用して各インスタンスを再設定他のすべての設定とプロパティ値は、古い VM と同じにする必要があります。
- VIP がノースバウンドアクセスに使用されている場合は、古い HA アクティブ/スタンバイペアを再設定するために使用したものと同じ VIP を新しい展開で保持してください。

新しいマスターおよびスタンバイの ESC インスタンスでの ESC データベースの復元

手順

スタンバイ ESC インスタンスをシャットダウンします。

ステップ 1 SSH を使用してスタンバイ ESC インスタンスに接続します。

```
$ ssh USERNAME@ESC_STANDBY_IP
```

ステップ 2 ESC インスタンスがスタンバイになっていることを確認し、スタンバイ ESC HA アクティブ/スタンバイインスタンス名をメモします。

```
$ sudo escadm status
```

「BACKUP」という出力値が表示される場合、そのノードはスタンバイ ESC ノードです。

（注） ダイナミック マッピング ファイル（dynamic_mapping.xml）が ESC サービスで使用されている場合、ダイナミック マッピング ファイルをバックアップ ESC VM に復元する必要があります。スタンバイ ESC ノードの電源をオフにする前に、バックアップ ダイナミック マッピング ファイル（dynamic_mapping .xml）をパス */opt/cisco/esc/esc-dynamic-mapping/* にコピーする必要があります。

ステップ 3 スタンバイ ESC インスタンスは、OpenStack Kg/Horizon を介して、Nova コマンドを使用してシャットダウンします。VMware vSphere を基盤とした ESC VM インスタンスの場合は、VMware クライアントダッシュボードを使用してプライマリインスタンスをシャットダウンします。OpenStack でのスタンバイ ESC インスタンスのシャットダウンの例を次に示します。

```
$ nova stop NEW_ESC_STANDBY_ID
```

新しいマスター ESC インスタンスのデータベースを復元します。

ステップ 4 SSH を使用してプライマリ ESC インスタンスに接続します。

```
$ ssh USERNAME@ESC_MASTER_IP
```

ステップ 5 ESC インスタンスがプライマリであることを確認します。

```
$ sudo escadm status
```

出力値に「MASTER」が表示されている場合、ノードはマスター ESC ノードです。

ステップ 6 マスターノードの ESC サービスを停止し、ステータスを確認して、サービスが停止していることを確認します。

```
$ sudo escadm stop
$ sudo escadm status
```

ステップ 7 データベースファイルを復元します。

```
$ sudo escadm restore --file /tmp/db.tar.bz2
$ scp://<username>:<password>@<backup_ip>:<filename>
```

（注） ダイナミック マッピング ファイル（dynamic_mapping.xml）が ESC サービスで使用されている場合、ダイナミック マッピング ファイルを ESC VM に復元する必要があります。ESC ノードを起動する前に、バックアップ ダイナミック マッピング ファイル（dynamic_mapping.xml）をパス */opt/cisco/esc-dynamic-mapping/* にコピーする必要があります。

ステップ 8 VM を再起動して、完全な ESC サービスを再起動します。

```
$ sudo escadm restart
```

ステップ 9 `$ Sudo escadm status` を使用して、ESC サービスのステータスを確認します。

ステップ 10 スタンバイ ESC ノードを起動します。

OpenStack Nova/Horizon または VMware クライアントを使用して、スタンバイ ESC ノードの電源をオンにします。スタンバイノードを起動した後、ESC HA アクティブ/スタンバイアップグレードのプロセスを完了する必要があります。

ステップ 11 古い HA アクティブ/スタンバイインスタンスを OpenStack Nova/Horizon または VMware クライアントから削除します。OpenStack で VM を削除する例を次に示します。

```
$ nova delete OLD_ESC_MASTER_RENAMED OLD_ESC_STANDBY_RENAMED
```

VNF モニタリングルールのアップグレード

VNF モニタリングルールをアップグレードするには、*dynamic_mappings.xml* ファイルをバックアップしてから、アップグレードされた ESC VM にファイルを復元する必要があります。詳細については、「バックアップと復元の手順」を参照してください。HA アクティブ/スタンバイインスタンスのアップグレードについては、「ESC HA アクティブ/スタンバイインスタンスのアップグレード」を参照してください。スタンドアロンインスタンスのアップグレードについては、「スタンドアロン ESC インスタンスのアップグレード」を参照してください。

OpenStack での ESC HA アクティブ/スタンバイノードのインサービスアップグレード

ESC RPM パッケージを使用した OpenStack でのインサービスアップグレード

手順

ステップ 1 ESC データベースとログファイルをバックアップします。

a) プライマリノードから ESC データベースのバックアップを実行します。データベースのバックアップの詳細については、「ESC HA アクティブ/スタンバイインスタンスのデータベースのバックアップ」を参照してください。

b) プライマリ VM とセカンダリ VM の両方からすべてのログを収集してバックアップします。ログをバックアップするには、次のコマンドを実行します。

```
# sudo escadm log collect
```

（注） タイムスタンプファイルは、/var/tmp/esc_log-<timestamp>.tar.bz2 に生成されます。

c) ESC VM からデータベースのバックアップファイルとログファイル（/tmp/esc_log-.tar.bz2 に生成されています）* をコピーします。

ステップ 2 ESC HA アクティブ/スタンバイセカンダリ VM にログインし、escadm サービスを停止します。

```
$ sudo escadm stop
```

ステップ 3 ESC VM が [停止（STOP）] の状態になっていることを確認します。ESC が [停止（STOP）] の状態に切り替わるまでにいくらか時間がかかる場合があります。ESC のステータスが [停止（STOP）] の状態になると、ESC は HA アクティブ/スタンバイクラスタの一部ではなり、HA アクティブ/スタンバイ機能が一時的に失われることに注意してください。

```
$ sudo escadm status

Expected output:
ESC status=0 ESC HA is stopped
```

ステップ 4 ESC VM にアップグレード用の RPM ファイルをコピーし、rpm コマンドを実行してアップグレードします。

```
$ sudo rpm -Uvh /home/admin/cisco-esc-3.1.0-145.x86_64.rpm
```

ステップ 5 escadm サービスを開始します。

```
$ sudo escadm start
```

ステップ 6 ESC HA アクティブ/スタンバイプライマリ VM にログインし、プライマリ VM で手順 2 ～ 6 を繰り返します。プライマリ ESC VM で escadm サービスを停止すると、フェールオーバーがトリガーされ、アップグレードされたセカンダリ VM がプライマリロールを引き継ぐことに注意してください。

ステップ 7 各インスタンスの ESC バージョンをチェックして、バージョンが正しくアップグレードされていることを確認し、ESC サービスが新しいプライマリ VM で正しく実行されていることを確かめます。

```
# esc_version
# health.sh (in Primary VM)
```

ESC qcow2 イメージを使用した OpenStack でのインサービスアップグレード

手順

ステップ 1 ESC データベースとログファイルをバックアップします。

a) プライマリノードから ESC データベースのバックアップを実行します。データベースのバックアップの詳細については、「ESC HA アクティブ/スタンバイインスタンスのデータベースのバックアップ」を参照してください。

b) プライマリ VM とセカンダリ VM の両方からすべてのログを収集してバックアップします。ログをバックアップするには、次のコマンドを実行します。

```
# sudo escadm log collect
```

（注） タイムスタンプファイルは、/var/tmp/esc_log-<timestamp>.tar.bz2 に生成されます。

c) ESC VM からデータベースのバックアップファイルとログファイル（/tmp/esc_log-.tar.bz2 に生成されています）* をコピーします。

ステップ 2 新しいバージョンの ESC イメージを使用してセカンダリインスタンスを再展開し、データが同期されるまで待機します。

a) Horizon Web UI または nova CLI を使用してセカンダリインスタンスを削除します。OpenStack コントローラで、nova クライアント経由で次のコマンドを実行します。

```
nova delete <secondary_vm_name>
```

b) 新しい ESC イメージを OpenStack Glance に登録して、再展開の使用状況を確認します。

```
glance image-create --name <image_name> --disk-format qcow2 --container-format bare
 --file <esc_qcow2_file>
```

c) 新しいイメージバージョンに基づいて、セカンダリ ESC VM インスタンスを再展開します。新しい ESC パッケージ（bootvm.py および新しく登録したイメージ）を使用して、新規のセカンダリインスタンスを再インストールします。他のすべてのインストールパラメータは、以前の ESC VM 展開と同じである必要があります。たとえば、ホスト名、IP アドレス、gateway_ip、ha_node_list、kad_vip、kad_vif には同じ値を使用する必要があります。アップグレードされたバージョンの新規 ESC インスタンスが稼働したら、そのインスタンスがセカンダリ状態になります。

d) 新規インスタンスにログインして次のコマンドを実行し、新しい ECS ノードの同期状態を確認します。

```
$ drbdadm status
```

drbdadm ステータスが出力されるまで待機し、両方のノードの出力が次のように「UpToDate」になっているかを確認します。これは、新規 ESC インスタンスで、プライマリインスタンスからのデータ同期が完了していることを示します。

バックアップ/セカンダリの例

```
esc role:Secondary
disk:UpToDate
101.1.0.119:7789 role:Primary
peer-disk:UpToDate
```

プライマリ/マスター ESC の例

```
esc role:Primary
disk:UpToDate
101.1.0.120:7789 role:Secondary
peer-disk:UpToDatee
```

ステップ 3 セカンダリインスタンスで keepalived サービスを停止し、プライマリインスタンスの電源をオフにします。次に、セカンダリ keepalived サービスを開始します。

a) keepalived サービスを停止するには、次のコマンドを使用します。

```
escadm keepalived stop
```

b) プライマリインスタンスにログインして、ESC プライマリノードをメンテナンスモードに設定します。

```
$ sudo escadm op_mode set --mode=maintenance
```

次のステップに進む前に、実行中のトランザクションがないことを確認してください。実行中のトランザクションがないことを確認するには、次のコマンドを実行します。

```
For ESC 2.3:
$ sudo escadm ip_trans
```

ESC 2.3 より前のバージョンの場合は、escmanager ログ（/var/log/esc/escmanager.log）に新しいトランザクションがないことを確認します。

c) アップグレードされたセカンダリインスタンスにログインし、ESC サービスをシャットダウンします。

```
$ sudo escadm stop
```

d) OpenStack Nova クライアント/Horizon を使用してプライマリインスタンスの電源をオフにし、オフになっていることを確認します。OpenStack コントローラで次を実行します。

```
$ nova stop <primary_vm_name>
$ nova list | grep <primary_vm_name>
```

e) 以前にアップグレードされたセカンダリインスタンス（停止状態）にログインし、ESC サービスを再起動します。セカンダリ ESC インスタンスはプライマリインスタンスのロールを引き継ぎ（スイッチオーバーがトリガーされます）、新しいバージョンでサービスの提供を開始します。

```
$ sudo escadm start
```

ステップ 4 新しいプライマリインスタンスの ESC バージョンをチェックし、バージョンが正しくアップグレードされていることを確認します。

```
$ sudo escadm status (check ha status)

Expected output:
0 ESC status=0 ESC Master Healthy

$ esc_version (check esc version)
version : 3.x.x
release : xxx
```

ステップ 5 新しい ESC イメージを使用して、旧プライマリインスタンスを再展開します。

古いプライマリリインスタンスを削除し、新しい ESC パッケージ（bootvm.py および新しく登録したイメージ）を使用して再展開します。

a) 新たに展開したインスタンスにログインして、HA ステータスを確認します。新しいインスタンスはセカンダリ状態になっている必要があります。

```
$ sudo escadm status --v
```

b) 次のコマンドを実行して、新しい ESC セカンダリノードの同期状態を確認します。

```
$ drbdadm status
```

drbdadm ステータスの出力に「UpToDate」と表示されるまで待機します。

c) 新しい ESC セカンダリノードの場合、正常性チェックに合格し、ESC バージョンが正しくアップグレードされていることを確認します。

```
$ sudo escadm status (check ha status)
Expected output:
0 ESC status=0 ESC Master Healthy
$ esc_version (check esc version)version : 2.x.x
release : xxx
$ health.sh
Expected output:
ESC HEALTH PASSED
```

ステップ 6 最初にアップグレードしたプライマリインスタンスに戻り、正常性とキープアライブの状態をチェックします。

```
$ drbdadm status

Expected output:
1:esc/0  Connected Primary/Secondary UpToDate/UpToDate /opt/cisco/esc/esc_database ext4
 2.9G 52M 2.7G 2%

$ sudo escadm status (check ha status)
Expected output:
0 ESC status=0 ESC Master Healthy

$ esc_version (check esc version) Expected output:
version : 2.x.x
release : xxx

$ health.sh (check esc health)
Expected output:
ESC HEALTH PASSED
```

（注） クイックロールバック：アップグレードに失敗した場合は、アップグレードされたインスタンスをシャットダウンし、古いプライマリインスタンスを起動して、クイックロールバックを実行します。

インサービスアップグレードのロールバック手順

1. ESC VM からデータベースとログのバックアップファイルを任意の場所にコピーします。
2. 残りの ESC インスタンスを削除したら、古いバージョンの qcow2 イメージを使用して ESC HA アクティブ/スタンバイ VM を再展開します。
3. データベースを復元します。「HA アクティブ/スタンバイデータベース復元のための Backup and Restore を使用した ESC HA アクティブ/スタンバイインスタンスのアップグレード」の手順に従います。
4. データベースの復元後、ESC サービスを古いバージョンに戻す必要があります。

カーネルベース仮想マシン（KVM）の ESC H アクティブ/スタンバイノードのインサービスアップグレード

ESC RPM パッケージを使用した KVM でのインサービスアップグレード

この手順を使用して、カーネルベースの仮想マシンでサービスの中断を最小限に抑えながら、ESC 高可用性ノードをアップグレードします。

手順

ステップ 1 ESC データベースとログファイルをバックアップします。

a) プライマリノードから ESC データベースのバックアップを実行します。データベースのバックアップの詳細については、「ESC HA アクティブ/スタンバイインスタンスのデータベースのバックアップ」を参照してください。

b) プライマリ VM とセカンダリ VM の両方からすべてのログを収集してバックアップします。ログをバックアップするには、次のコマンドを実行します。

```
$ sudo escadm log collect
```

（注） タイムスタンプログファイルは、/var/tmp/esc_log-<timestamp>.tar.bz2 に生成されます。

c) ESC VM からデータベースのバックアップファイルとログファイル（/tmp/esc_log-.tar.bz2 に生成）* をコピーします。

ステップ 2 ESC HA アクティブ/スタンバイセカンダリ VM にログインし、ESC サービスを停止します。

```
$ sudo escadm stop
```

ステップ 3 セカンダリ ESC VM が停止状態になっていることを確認します。

```
$ sudo escadm status --v
```

ESC ステータスが 0 の場合、HA は停止しています。

ステップ 4 セカンダリ VM で rpm コマンドを実行して、アップグレードします。

```
$ sudo rpm -Uvh /home/admin/cisco-esc-<latest rpm filename>.rpm
```

ステップ 5 プライマリインスタンスにログインして、ESCプライマリノードをメンテナンスモードに設定します。

```
$ sudo escadm op_mode set --mode=maintenance
```

実行中のトランザクションや新しいトランザクションがアップグレード時にないことを確認してください。次のコマンドを使用して、実行中のトランザクションを確認します。

```
$ sudo escadm ip_trans
```

ESC 2.3 よりも古いビルドの場合は、escmanager ログから、（/var/log/esc/escmanager.log）のトランザクションを確認する必要があります。

ステップ 6 ESCプライマリノードの電源をオフにして、完全にシャットダウンされていることを確認します。KVM ESC コントローラで、次のコマンドを実行します。

```
$ virsh destroy <primary_vm_name>

$ virsh list --all
```

ステップ 7 アップグレードされた ESC インスタンス（以前のセカンダリインスタンス）にログインし、ESC サービスを開始します。アップグレードされた VM はプライマリロールを引き継ぎ、ESC サービスを提供します。

```
$ sudo escadm restart
$ sudo escadm monitor start
```

ステップ 8 新しいプライマリインスタンスのESCバージョンを確認し、正しいバージョンにアップグレードされているかを確認します。VM がプライマリ状態になったら、ESC サービスが新しいプライマリ VM で適切に稼働していることを確認します。

```
$ sudo escadm status
Expected output:
0 ESC status=0 ESC Master Healthy

$ esc_version

$ health.sh
Expected output:
ESC HEALTH PASSED
```

ステップ 9 古いプライマリインスタンスの電源をオンにします。KVM ESC コントローラで、次のコマンドを実行します。

```
$ virsh start <primary_vm_name>
```

ステップ 10 古い ESC バージョンを搭載している VM にログインして、ステップ 2、3、4、および 7 を VM で繰り返します。

ESC OVA イメージを使用した KVM でのインサービスアップグレード

手順

ステップ 1 ESC データベースとログファイルをバックアップします。

a) プライマリノードから ESC データベースのバックアップを実行します。データベースのバックアップの詳細については、「ESC HA アクティブ/スタンバイインスタンスのデータベースのバックアップ」を参照してください。

b) プライマリ VM とセカンダリ VM の両方からすべてのログを収集してバックアップします。ログをバックアップするには、次のコマンドを実行します。

```
$ sudo escadm log collect
```

（注） タイムスタンプログファイルは、/var/tmp/esc_log-<timestamp>.tar.bz2 に生成されます。

c) ESC VM からデータベースのバックアップファイルとログファイル（/tmp/esc_log-.tar.bz2 に生成）* をコピーします。

ステップ 2 セカンダリ ESC インスタンスを再展開します。セカンダリインスタンスに新しい ESC イメージを登録します。

a) lib vert Virsh コマンドを使用して、セカンダリインスタンスを削除します。KVM ホストで、次のコマンドを実行します。

```
$ Virsh destroy the <secondary_vm_name>
$ Virsh undefine --remove-all-storage <secondary_vm_name>
```

b) 再展開の使用のために、新しい ESC イメージを Kvm ホストにコピーします。

```
sshpass -p "host Password' scp  /scratch/BUILD-2_x_x_x/BUILD-2_x_x_x/ESC-2_x_x_x.qcow2
 root@HOSTIP:
```

c) 新しいイメージバージョンに基づいて、セカンダリ ESC VM インスタンスを再展開します。新しい ESC パッケージ（bootvm.py および新しく登録したイメージ）を使用して、新規のセカンダリインスタンスを再インストールします。他のすべてのインストールパラメータは、以前の ESC VM 展開と同じである必要があります。たとえば、ホスト名、IP アドレス、gateway_ip、ha_node_list、kad_vip、kad_vif には同じ値を使用する必要があります。アップグレードされたバージョンの新規 ESC インスタンスが稼働したら、そのインスタンスがセカンダリ状態になります。

d) 新規インスタンスにログインして次のコマンドを実行し、新しい ECS ノードの同期状態を確認します。

```
$ drbdadm status
```

drbdadm ステータスが出力されるまで待機し、両方のノードが次の出力のように「UpToDate」になっていることを確認します。これは、新規 ESC インスタンスで、プライマリインスタンスからのデータ同期が完了していることを示します。

```
esc/0 Connected Secondary/Primary UpToDate/UpToDate
```

ステップ 3 セカンダリインスタンスで keepalived サービスを停止し、プライマリインスタンスの電源をオフにします。次に、セカンダリ keepalived サービスを開始します。

a) プライマリインスタンスにログインして、ESC プライマリノードをメンテナンスモードに設定します。

```
$ sudo escadm op_mode set --mode=maintenance
```

次のステップに進む前に、実行中のトランザクションがないことを確認してください。実行中のトランザクションがないことを確認するには、次のコマンドを実行します。

```
$ sudo escadm ip_trans
```

escmanager ログ（/var/log/esc/escmanager.log）に新しいトランザクションがないことを確認します。

b) アップグレードされたセカンダリインスタンスにログインして、キープアライブサービスをシャットダウンします。

```
$ sudo escadm stop
```

c) プライマリインスタンスの電源をオフにして、完全にオフになっていることを確認します。KVM ESC コントローラで、次を実行します。

```
$ virsh destroy <primary_vm_name>
$ virsh list --all
```

d) 以前にアップグレードされたセカンダリインスタンス（停止状態）にログインし、ESC サービスを起動します。セカンダリ ESC インスタンスはプライマリインスタンスのロールを引き継ぎ（スイッチオーバーがトリガーされます）、新しいバージョンでサービスの提供を開始します。

```
$ sudo escadm restart
```

ステップ 4 新しいプライマリインスタンスの ESC バージョンをチェックし、バージョンが正しくアップグレードされていることを確認します。

```
$ sudo escadm status (check ha status)

Expected output:
0 ESC status=0 ESC Master Healthy

$ esc_version (check esc version)
version : 4.1.x
release : xxx

$ health.sh (check esc health)

Expected output:
ESC HEALTH PASSED
```

ステップ 5 新しい ESC イメージを使用して、旧プライマリインスタンスを再展開します。

古いプライマリリインスタンスを削除し、新しい ESC パッケージ（bootvm.py および新しく登録したイメージ）を使用して再展開します。他のすべてのインストールパラメータは、古い ESC VM の展開と同じにする必要があります。たとえば、ホスト名、IP アドレス、gateway_ip、ha_node_list、kad_vip、kad_vif は同じ値にする必要があります。

a) 新たに展開したインスタンスにログインして、HA ステータスを確認します。新しいインスタンスはセカンダリ状態になっている必要があります。

```
$ sudo escadm status
```

b) 次のコマンドを実行して、新しい ESC セカンダリノードの同期状態を確認します。

```
$ drbdadm status
```

drbdadm ステータスの出力に「UpToDate」と表示されるまで待機します。

c) 新しい ESC セカンダリノードの場合、正常性チェックに合格し、ESC バージョンが正しくアップグレードされていることを確認します。

```
$ sudo escadm status (check ha status)
Expected output:
0 ESC status=0 ESC Master Healthy
$ esc_version (check esc version)version : 4.1.x
release : xxx
$ health.sh
Expected output:
ESC HEALTH PASSED
```

ステップ 6 最初にアップグレードしたプライマリインスタンスに戻り、正常性とキープアライブの状態をチェックします。

```
$ drbdadm status
Expected output:
1:esc/0  Connected Primary/Secondary UpToDate/UpToDate /opt/cisco/esc/esc_database ext4
 2.9G 52M 2.7G 2%

$ sudo escadm status (check ha status)
Expected output:
0 ESC status=0 ESC Master Healthy

$ esc_version (check esc version) Expected output:
version : 2.x.x
release : xxx

$ health.sh (check esc health)
Expected output:
ESC HEALTH PASSED
```

（注） クイックロールバック：アップグレードに失敗した場合は、アップグレードされたインスタンスをシャットダウンし、古いプライマリインスタンスを起動して、クイックロールバックを実行します。

インサービスアップグレードのロールバック手順

1. 残りの ESC インスタンスを削除したら、古いバージョンの qcow2 イメージを使用して ESC HA アクティブ/スタンバイ VM を再展開します。
2. データベースを復元します。「HA アクティブ/スタンバイデータベース復元のための Backup and Restore を使用した ESC HA アクティブ/スタンバイインスタンスのアップグレード」の手順に従います。
3. データベースの復元後、ESC サービスを古いバージョンに戻す必要があります。

VMware での ESC HA アクティブ/スタンバイノードのインサービスアップグレード

ESC RPM パッケージを使用した VMware でのインサービスアップグレード

サービスの中断を最小限に抑えながら、ESC 高可用性ノードを一度に 1 つアップグレードするには、次の手順を使用します。このプロセスでは、ESC HA アクティブ/スタンバイのレプリケーションおよびフェールオーバー機能が活用されるため、手動でデータベースを復元することなく、アップグレードされた新規ノードに ESC サービスをスムーズに移行できます。

手順

ステップ 1 ESC データベースとログファイルをバックアップします。

a) プライマリノードから ESC データベースのバックアップを実行します。データベースのバックアップの詳細については、「ESC HA アクティブ/スタンバイインスタンスのデータベースのバックアップ」を参照してください。

b) プライマリ VM とセカンダリ VM の両方からすべてのログを収集してバックアップします。ログをバックアップするには、次のコマンドを実行します。

```
# sudo escadm log collect
```

c) ESC VM からデータベースのバックアップファイルとログファイル（/tmp/esc_log-.tar.bz2 に生成されます）* をコピーします

ステップ 2 ESC HA アクティブ/スタンバイセカンダリ VM にログインして、キープアライブサービスを停止します。

```
$ sudo escadm stop
```

ステップ 3 セカンダリ ESC VM が停止状態になっていることを確認します。

```
$ sudo escadm status --v
```

ESC ステータスが 0 の場合、HA は停止しています。

ステップ 4 セカンダリ VM で rpm コマンドを実行して、アップグレードします。

```
$ sudo rpm -Uvh /home/admin/cisco-esc-2.2.9-50.rpm
```

ステップ 5 プライマリインスタンスにログインして、ESCプライマリノードをメンテナンスモードに設定します。

```
$ sudo escadm op_mode set --mode=maintenance
```

実行中のトランザクションや新しいトランザクションがアップグレード時にないことを確認してください。ESC 2.3 から次のコマンドを実行すると、実行中のトランザクションを確認できます。

```
$ sudo escadm ip_trans
```

ESC 2.3 より前のビルドの場合は、escmanager ログを確認し、このログファイルに新しいトランザクションが記録されていないかを確認する必要があります。ログファイルは（/var/log/esc/escmanager.log）にあります。

ステップ 6 ESC プライマリノードの電源をオフにします。VMware vSphare クライアントで、[ホーム（Home）] > [インベントリ（Inventory）] > [VM とテンプレート（VMs and Templates）] を選択します。左側のパネルからプライマリインスタンス名を右クリックして、[電源（Power）] > [電源オフ（Power Off）] を選択します。

ステップ 7 アップグレードされたESCインスタンス（以前のセカンダリインスタンス）にログインして、キープアライブサービスを開始します。アップグレードされたVMはプライマリロールを引き継ぎ、ESC サービスを提供します。

```
$ sudo escamd restart
```

ステップ 8 新しいプライマリインスタンスのESCバージョンを確認し、正しいバージョンにアップグレードされているかを確認します。VM がプライマリ状態になったら、ESC サービスが新しいプライマリ VM で適切に稼働していることを確認します。

```
$ sudo escadm status
Expected output:
0 ESC status=0 ESC Master Healthy

$ esc_version

$ health.sh
Expected output:
ESC HEALTH PASSED
```

ステップ 9 古いプライマリインスタンスの電源をオンにします。VMware vSphare クライアントで、[ホーム（Home）] > [インベントリ（Inventory）] > [VM とテンプレート（VMs and Templates）] を選択します。左側のパネルからプライマリインスタンス名を右クリックして、[電源（Power）] > [電源オン（Power On）] を選択します。

ステップ 10 古い ESC バージョンを搭載している VM にログインして、ステップ 2、3、4、および 7 を VM で繰り返します。

ESC qcow2 イメージを使用した VMware でのインサービスアップグレード

手順

ステップ 1 ESC データベースとログファイルをバックアップします。

a) プライマリノードから ESC データベースのバックアップを実行します。データベースのバックアップの詳細については、「ESC HA アクティブ/スタンバイインスタンスのデータベースのバックアップ」を参照してください。

b) プライマリ VM とセカンダリ VM の両方からすべてのログを収集してバックアップします。ログをバックアップするには、次のコマンドを実行します。

```
# sudo escadm log collect
```

（注） タイムスタンプログファイルは、/var/tmp/esc_log-<timestamp>.tar.bz2 に生成されます。

c) ESC VM からデータベースのバックアップファイルとログファイル（/tmp/esc_log-.tar.bz2 に生成）* をコピーします。

ステップ 2 セカンダリ ESC インスタンスを再展開します。セカンダリインスタンスに新しい ESC イメージを登録し、データが同期されるまで待機します。

a) セカンダリインスタンスを削除します。セカンダリ ESC インスタンスを削除するには、まず vSphere クライアントからインスタンスの「電源をオフ」にし、次に [ディスクから削除（Delete from Disk）] オプションを使用する必要があります。VMware vSphare クライアントで、[ホーム（Home）] > [インベントリ（Inventory）] > [VM とテンプレート（VMs and Templates）] を選択します。左側のパネルからプライマリインスタンス名を右クリックして、[電源（Power）]>[電源オフ（Power Off）] を選択します。次に、セカンダリインスタンスを削除するには、[ホーム（Home）] > [インベントリ（Inventory）] > [VM とテンプレート（VMs and Templates）] を選択します。左側のパネルからインスタンス名を右クリックして、[ディスクから削除（Delete from Disk）] を選択します。

b) 新しいイメージバージョンに基づいて、セカンダリ ESC VM インスタンスを再展開します。新しい ESC パッケージ（bootvm.py および新しく登録したイメージ）を使用して、新規のセカンダリインスタンスを再インストールします。アップグレードされたバージョンの新規 ESC インスタンスが稼働したら、そのインスタンスがセカンダリ状態になります。

c) 新規インスタンスにログインして次のコマンドを実行し、新しい ECS ノードの同期状態を確認します。

```
$ drbdadm status
```

drbdadm ステータスが出力されるまで待機し、両方のノードの出力が次のように「UpToDate」になっているかを確認します。これは、新規 ESC インスタンスで、プライマリインスタンスからのデータ同期が完了していることを示します。

```
esc/0 Connected Secondary/Primary UpToDate/UpToDate
```

ステップ 3 セカンダリインスタンスで keepalived サービスを停止し、プライマリインスタンスの電源をオフにします。次に、セカンダリ keepalived サービスを開始します。

a) プライマリインスタンスにログインして、ESCプライマリノードをメンテナンスモードに設定します。

```
$ sudo escadm op_mode set --mode=maintenance
```

次のステップに進む前に、実行中のトランザクションがないことを確認してください。実行中のトランザクションがないことを確認するには、次のコマンドを実行します。

```
For ESC 2.3:
$ sudo escadm ip_trans
```

ESC 2.3 より前のバージョンの場合は、escmanager ログ（/var/log/esc/escmanager.log）に新しいトランザクションがないことを確認します。

b) アップグレードされたセカンダリインスタンスにログインして、キープアライブサービスをシャットダウンします。

```
$ sudo escadm stop
```

c) プライマリインスタンスの電源をオフにして、プライマリインスタンスの電源がオフになっていることを確認します。VMware vSphare クライアントで、[ホーム（Home）] > [インベントリ（Inventory）] > [VM とテンプレート（VMs and Templates）] を選択します。左側のパネルからプライマリインスタンス名を右クリックして、[電源（Power）] > [電源オフ（Power Off）] を選択します。

d) 以前にアップグレードされたセカンダリインスタンス（停止状態）にログインし、キープアライブサービスを開始します。セカンダリESCインスタンスはプライマリインスタンスのロールを引き継ぎ（スイッチオーバーがトリガーされます）、新しいバージョンでサービスの提供を開始します。

```
$ sudo escadm start
```

ステップ 4 新しいプライマリインスタンスの ESC バージョンをチェックし、バージョンが正しくアップグレードされていることを確認します。

```
$ sudo escadm status --v(check ha status)

Expected output:
0 ESC status=0 ESC Master Healthy

$ esc_version (check esc version)
version : 3.x.x
release : xxx

$ health.sh (check esc health)

Expected output:
ESC HEALTH PASSED
```

ステップ 5 新しい ESC イメージを使用して、旧プライマリインスタンスを再展開します。

古いプライマリリインスタンスを削除し、新しいESCパッケージ（bootvm.pyおよび新しく登録したイメージ）を使用して再展開します。他のすべてのインストールパラメータは、古いESCVMの展開と同じにする必要があります。たとえば、ホスト名、IPアドレス、gateway_ip、

ha_node_list、kad_vip、kad_vif は同じ値にする必要があります。削除する場合は、[ホーム（Home）] > [インベントリ（Inventory）] > [VM とテンプレート（VMs and Templates）] を選択します。左側のパネルからインスタンス名を右クリックして、[ディスクから削除（Delete from Disk）] を選択します。

a) 新たに展開したインスタンスにログインして、HA ステータスを確認します。新しいインスタンスはセカンダリ状態になっている必要があります。

```
$ sudo escadm status
```

b) 次のコマンドを実行して、新しい ESC セカンダリノードの同期状態を確認します。

```
$ drbdadm status
```

drbdadm ステータスの出力に「UpToDate」と表示されるまで待機します。

c) 新しいESCセカンダリノードの場合、正常性チェックに合格し、ESCバージョンが正しくアップグレードされていることを確認します。

```
$ sudo escadm status (check ha status)
Expected output:
0 ESC status=0 ESC Master Healthy
$ esc_version (check esc version)version : 3.x.x
release : xxx
$ health.sh
Expected output:
ESC HEALTH PASSED
```

ステップ 6 最初にアップグレードしたプライマリインスタンスに戻り、正常性とキープアライブの状態をチェックします。

```
$ drbdadm status
Expected output:
1:esc/0  Connected Primary/Secondary UpToDate/UpToDate /opt/cisco/esc/esc_database ext4
 2.9G 52M 2.7G 2%

$ sudo escadm status (check ha status)
Expected output:
0 ESC status=0 ESC Master Healthy

$ esc_version (check esc version) Expected output:
version : 3.x.x
release : xxx

$ health.sh (check esc health)
Expected output:
ESC HEALTH PASSED
```

（注） クイックロールバック：アップグレードに失敗した場合は、アップグレードされたインスタンスをシャットダウンし、古いプライマリインスタンスを起動して、クイックロールバックを実行します。

インサービスアップグレードのロールバック手順

1. ESC VM からデータベースとログのバックアップファイルを任意の場所にコピーします。
2. 残りの ESC インスタンスを削除したら、古いバージョンの qcow2 イメージを使用して ESC HA アクティブ/スタンバイ VM を再展開します。
3. データベースを復元します。「HA アクティブ/スタンバイデータベース復元のための Backup and Restore を使用した ESC HA アクティブ/スタンバイインスタンスのアップグレード」の手順に従います。
4. データベースの復元後、ESC サービスを古いバージョンに戻す必要があります。

CSP での ESC HA アクティブ/スタンバイノードのインサービスアップグレード

CSP で ESC HA アクティブ/スタンバイノードのインサービスアップグレードを実行するには、次の手順を実行します。

始める前に

次のコマンドを使用して、アップグレードの前に ESC HA ノードが適切に稼働していることを確認します。

```
# escadm status
```

1 つのノードがマスター状態で、もう 1 のノードがバックアップ状態である必要があります。次のコマンドを使用して、マスターノードを確認します。

```
# health.sh
```

正常性チェックに合格すると、次が表示されます。

```
ESC HEALTH PASSED
```

手順

ステップ 1 スタンバイインスタンスのシャットダウン

VM の電源をオフにする前に、バックアップ ESC VM をアップグレードします。アップグレードするには、次の手順に従います。

1. 次のコマンドを使用して、バックアップ ESC VM からログを収集し、別のマシンにコピーします。

```
# collect_esc_log.sh
# scp /tmp/LOG_PACKAGE_NAME <username>@<backup_vm_ip>:<filepath>
```

2. CSP を使用して、スタンバイ ESC の電源をオフにします。

ステップ 2 新しい ESC パッケージを使用して、スタンバイノードを再展開します。

スタンバイ ESC VM の電源をオフにした後、アップグレードするために、新しい ESC パッケージを使用して新しい ESC VM をインストールします。以前の ESC VM とは異なる ESC パッケージを使用する場合を除き、ESC インストールで使用するその他すべてのパラメータは、以前の ESC VM 展開と同じにする必要があります。

ESC ノードがバックアップ状態になっていることを確認します。次のコマンドを使用して、新しい ESC スタンバイノードの同期状態を確認します。

```
# drbdadm status
```

次の出力が表示されるまで待機します。これは、新しい ESC VM がマスターノードからのデータ同期を完了しており、最新の状態であることを示しています。

```
[admin@esc-xyx-upgradetestha1-4-5-0-105 ~]$ drbdadm status
esc role:Secondary
disk:UpToDate
172.20.117.55:7789 role:Primary
peer-disk:UpToDate
```

ダイナミック マッピング ファイル（dynamic_mapping xm）が ESC サービスによって使用されている場合は、ESC VM に復元する必要があります。バックアップファイルを `/opt/cisco/esc-dynamic-mapping/` パスにコピーします。

ステップ 3 マスターノードを停止し、スイッチオーバーをトリガーします。

CSP を使用してマスターインスタンスの電源をオフにします。その後、HA スイッチオーバーが自動的にトリガーされ、スタンバイインスタンスが新しいバージョンの ESC サービスを引き継ぎます。新しい ESC インスタンスがマスターになった後、新しい ESC マスターノードが正常性チェックに合格したかを確認します。

ステップ 4 古いマスターを置き換えるための新しい ESC ノードの展開

新しい ESC パッケージを使用して以前のマスターインスタンスをアップグレードするため、新しい ESC VM をインストールします。古いバージョンの ESC VM の電源をオフにする前に、次のコマンドを使用して、ESC VM からログを収集して別のマシンにコピーします。

```
# collect_esc_log.sh
# scp /tmp/LOG_PACKAGE_NAME <username>@<backup_vm_ip>:<filepath>
```

（注） ダイナミック マッピング ファイルが ESC サービスによって使用されている場合は、ダイナミック マッピング ファイルを ESC ログと同じタイミングでバックアップする必要があります。ダイナミック マッピング ファイルのデフォルトパスは `/opt/cisco/esc/esc-dynamic-mapping/dynamic_mappings.xml` です。

CSP を介して古いバージョンの ESC VM の電源をオフにします。

ESC ノードがバックアップ状態になっていることを確認します。次のコマンドを使用して、新しい ESC スタンバイノードの同期状態を確認します。

```
# drbdadm status
```

次の出力が表示されるまで待機します。これは、新しいESC VMがマスターノードからのデータ同期を完了しており、最新の状態であることを示しています。

```
[admin@esc-xyz-upgradetestha1-4-5-0-105 ~]$ drbdadm status
esc role:Secondary
disk:UpToDate
172.20.117.55:7789 role:Primary
peer-disk:UpToDate
```

ダイナミック マッピング ファイル（dynamic_mapping xm）が ESC サービスによって使用されている場合は、ESC VM に復元する必要があります。バックアップファイルを `/opt/cisco/esc-dynamic-mapping/` パスにコピーします。

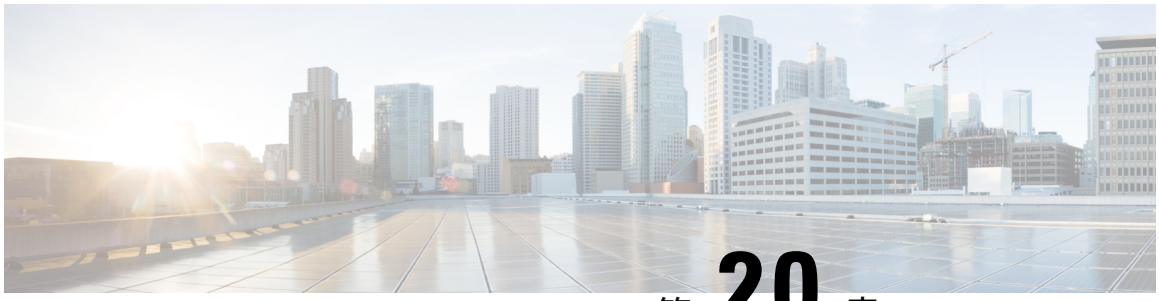
（注） ダイナミック マッピング ファイルが ESC サービスによって使用されている場合は、ダイナミック マッピング ファイルをESC VMに復元する必要があります。バックアップダイナミックマッピングファイルを、ダイナミックマッピングファイルのデフォルトパス `/opt/cisco/esc/esc-dynamic-mapping/dynamic_mappings.xml` にコピーします。

インサービスアップグレードが正常に完了すると、古い ESC インスタンスを削除できます。

（注） 新しい ESC バージョンにアップグレードした後も、ESC サービスは、古いバージョンによって展開されたすべてのVNFのライフサイクル管理を継続します。新しいESCバージョンの新機能を既存の VNF に適用する場合は、それらの VNF の展開を解除し、新たに展開を実行します。

第 VIII 部

Cisco Elastic Services Controller のインストールに関するトラブルシューティング



第 20 章

ESCに関する問題のトラブルシューティング

この章は、次の項で構成されています。

ESC ログメッセージの表示

ログメッセージは、VNF ライフサイクル全体にわたって ESC イベント用に作成されます。これらには、外部メッセージ、ESC から他の外部システムへのメッセージ、エラーメッセージ、警告、イベント、障害などがあります。ログファイルは、`/var/log/esc/escmanager_tagged.log` にあります。

次に、ログメッセージの形式を示します。

```
date=<time-date>] [loglevel=<loglevel>] [tid=<transactionid>] [cl=<classifications>]
[tags=<tags>] [msg=<message>
```

次に、ログの例を示します。

```
date=15:43:58,46022-Nov-2016]
[loglevel=ERROR ] [tid=0793b5c9-8255-47f3-81e6-fbb59f6571f7] [cl=OS ]
[tags=wf:create_vm,eventType:VM_DEPLOY_EVENT,tenant:CSCvd94541,depName:test-dep,vmGrpName:test-VNF,
vmName:test-dep_test_0_dc3f406c-05ca-43b3-af21-0841e3b029a0]
[tags=wf:create_vm,eventType:VM_DEPLOY_EVENT,tenant:test,depName:test-dep,vmGrpName:test-VNF,
vmName:test-dep_test_0_dc3f406c-05ca-43b3-af21-0841e3b029a0] [msg=sleepingfor5seconds
to allow vm to become ACTIVE instance id:
162344f7-78f9-4e45-9f23-34cf87377fa7
name:test-dep_test_0_dc3f406c-05ca-43b3-af21-0841e3b029a0
```

要求を受信すると、一意のトランザクション ID を自動生成する RequestDetails オブジェクトが作成されます。この値は、すべてのスレッドで転送されます。分類とタグは任意です。これらは、読みやすくするためにログメッセージに追加されたプレフィックスであり、デバッグに役立ちます。分類とタグを使用すると、ログメッセージを簡単に解析し、ログ分析ツールでフィルタリングすることができます。

次に、サポートされている分類を示します。

NBI	「com.cisco.esc.rest」「com.cisco.esc.filter」（ノース バウンド インターフェイス：証明書）
SBI	「com.cisco.esc.rest」：ソースはコールバックハンドラまたは「EventsResource」（サウスバウンドインターフェイス、ESC と VIM 間）
SM	「com.cisco.esc.statemachines」は StateMachine を意味します。この分類は、StateMachine カテゴリのログを示します。
MONITORING	「com.cisco.esc.monitoring」「com.cisco.esc.paadaptor」（MONA 関連ログ）
DYNAMIC_MAPPING	「com.cisco.esc.dynamicmapping」「com.cisco.esc.db.dynamicmapping」（MONA 関連ログ）
CONFD	「com.cisco.esc.confd」
CONFD_NOTIFICATION	「com.cisco.esc.confd.notif」「com.cisco.esc.confd.ConfdNBIAdapter」
OS	「com.cisco.esc.vim.openstack」
LIBVIRT	「com.cisco.esc.vim.vagrant
VIM	「com. esc. vim」
REST_EVENT	「ESCManager_Event」「com.cisco.esc.util.RestUtils」。ログ内の REST 通知を示します。
WD	「com.cisco.esc.watchdog」
DM	「com.cisco.esc.datamodel」「com.cisco.esc.jaxb.parameters」（データモデルとリソースオブジェクト）
DB	「com.cisco.esc.db」（データベース関連ログ）
GW	「com.cisco.esc.gateway」
LC	「com.cisco.esc.ESCManager」（スタートアップ関連ログ）
SEC	「com.cisco.esc.jaas」
MOCONFIG	「com.cisco.esc.moconfig」（MOCONFIG オブジェクト関連ログ。これは ESC 開発者用の内部ログです）
POLICY	「com.cisco.esc.policy」（サービス/VM ポリシー関連ログ）
TP	「com.cisco.esc.threadpool」

ESC	「com.cisco.esc」上記にないその他のパッケージ

次に、サポートされているタグを示します。

- **ワークフロー [wf:]**：RequestDetails オブジェクトのアクションとリソースを使用して生成されます。例：「wf: create_network」
- **イベントタイプ [eventType:]**：現在のアクションをトリガーしたイベント。例：「eventType:VM_DEPLOY_EVENT」
- **リソースベース**：これらの値は、イベントで使用されるパラメータのタイプに基づいて生成されます。階層 （テナント、VM グループなど）がログに追加されます。

テナント	[tenant:<tenant name>]
ネットワーク	[tenant:<tenant id>, network:<network name>] （注） テナントは、該当する場合にのみ表示されます。
サブネット	[tenant:<tenant name or id>, network:<network name or id>, subnet:<subnet name>] （注） テナントは、該当する場合にのみ表示されます。
ユーザ	[tenant:<tenant name>, user:<user name or id>] （注） テナントは、該当する場合にのみ表示されます。
イメージ	[image:<image name>]
フレーバ	[flavor:<flavor name>]
配置	[tenant:<tenant name or id>, depName:<deployment name>]
DeploymentDetails	[tenant:<tenant name or id>, depName:<deployment name>, vmGroup:<vm group name>, vmName:<vm name>]
スイッチ	[tenant:<tenant name or id>, switch:<switch name>]
音量	[volume:<volume name>]
サービス	[svcName:<Service Registration name>]

さらに、分析とログの管理を促進するため、ESC ログを rsyslog サーバに転送することもできます。

ConfD API を使用したログのフィルタリング

ConfD API に導入されたログフィルタを使用して、ESC でログ（展開ログやエラーログなど）を照会および取得できます。テナント、展開名、およびVM名の新しいフィルタが導入されました。これにより、ConfD API のログフィルタを使用して、最新のエラーログの ESC ログをさらに照会することができます。ESC と OS 間の通信に関連する ESC ログを取得することもできます（分類タグを「OS」に設定します）。

次に、ConfD API ログを取得するためのログ形式を示します。

```
date=<time-date>] [loglevel=<loglevel>] [tid=<transactionid>] [cl=<classifications>]
[tags=<tags>] [msg=<message>
```

次に、サンプルログの例を示します。

```
date=15:43:58,46022-Nov-2016] [loglevel=ERROR ] [tid=0793b5c9-8255-47f3-81e6-fbb59f6571f7]
 [cl=OS ]
[tags=wf:create_vm,eventType:VM_DEPLOY_EVENT,tenant:test,depName:test-dep,vmGrpName:test-VNF,
vmName:test-dep_test_0_dc3f406c-05ca-43b3-af21-0841e3b029a0]
[msg=sleepingfor5seconds to allow vm to become ACTIVE instance id:
162344f7-78f9-4e45-9f23-34cf87377fa7
name:test-dep_test_0_dc3f406c-05ca-43b3-af21-0841e3b029a0
```

ログレベル、分類、およびタグのパラメータは、ログを取得するために相互に依存します。次の組み合わせを使用してログを正常に取得できます。

- log_level=ERROR, classifications=OS, tags=(depName:test-dep)
- log_level=ERROR, classifications=OS, tags=(tenant: test)

ログフィルタは、次の条件がすべて満たされたときに値を返します。

- ログ レベル
- 分類（指定されている場合）
- タグ（指定されている場合）

（注） 複数の分類がリストされている場合は、1 つ以上の分類に一致する必要があります。同じことが、タグにも適用されます。

たとえば、次のログフィルタ条件では、前述のログサンプルを返しません。

```
log_level=ERROR, classifications=VIM, tags=(depName:test-dep)
```

ログレベルとタグが一致していても、分類の VIM が一致していないので値は返されません。

次に、データモデルを示します。

```
rpc filterLog {
    description "Query and filter escmanager logs using given parameters";
    tailf:actionpoint escrpc;
    input {
      leaf log_level {
        mandatory false;
       description "One of DEBUG / INFO / WARNING / ERROR / TRACE / FATAL. Results will
 include all logs at and
                     above the level specified";
        type types:log_level_types;
        default ERROR;
      }
      leaf log_count {
        mandatory false;
        description "Number of logs to return";
        type uint32;
        default 10;
      }
```

```
      container classifications {
        leaf-list classification {
          description "Classification values to be used for the log filtering.  For
example: 'OS', 'SM'.
                      Logs containing any of the provided classification values will be
 returned.";
          type types:log_classification_types;
        }
      }
      container tags {
        list tag {
          key "name";
          leaf name {
            mandatory true;
          description "Tag name to be used for the log filtering.  For example: 'tenant',
 'depName'.
                       Logs containing any of the provided tag name plus the tag values
 will be returned.";
            type types:log_tag_types;
          }
          leaf value {
            mandatory true;
           description "Tag value pairs to be used for the log filtering.  For example:
 'adminTenant', 'CSRDeployment'";
            type string;
          }
        }
      }
    }
    output {
      container filterLogResults {
        leaf log_level {
          description "Log level used to filter for the logs.";
          type types:log_level_types;
        }
        list logs {
          container classifications {
            leaf-list classification {
              description "Classifications used to filter for the logs.";
              type types:log_classification_types;
            }
          }
          container tags {
            list tag {
              key "name";
              leaf name {
                mandatory true;
                description "Tag name used to filter for the logs.";
                type types:log_tag_types;
              }
              leaf value {
                mandatory true;
                description "Tag value used to filter for the logs.";
                type string;
              }
            }
          }
          leaf log_date_time {
            description "Timestamp of the log.";
            type string;
          }
          leaf log_message {
            description "The log message.";
            type string;
```

```
        }
      }
    }
  }
}
```

NETCONF コンソールまたは esc_nc_cli を使用して、ConfD API ログを照会できます。

- NETCONF コンソールを使用して、次のクエリを実行します。

```
/opt/cisco/esc/confd/bin/netconf-console --port=830 --host=127.0.0.1 --user=admin
--privKeyFile=/home/admin/.ssh/confd_id_dsa --privKeyType=dsa --rpc=log.xml
```

- esc_nc_cli を使用して、次のクエリを実行します。

```
./esc_nc_cli filter-log log.xml
```

次に、log.xml の例を示します。

```
<filterLog xmlns="https://www.cisco.com/esc/esc">
  <log_level>INFO</log_level>
  <log_count>1</log_count>
  <classifications>
    <classification>OS</classification>
    <classification>SM</classification>
  </classifications>
  <tags>
    <tag>
      <name>depName</name>
      <value>CSR_ap1</value>
    </tag>
    <tag>
      <name>tenant</name>
      <value>admin</value>
    </tag>
  </tags>
</filterLog>
```

応答は次のとおりです。

```
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="1">
  <filterLogResults xmlns="https://www.cisco.com/esc/esc">
    <log_level>INFO</log_level>
    <logs>
      <classifications>
        <classification>OS</classification>
        <classification>SM</classification>
      </classifications>
      <tags>
        <tag>
          <name>depName</name>
          <value>CSR_ap1</value>
        </tag>
        <tag>
          <name>tenant</name>
          <value>admin</value>
        </tag>
      </tags>
      <log_date_time>13:06:07,575 31-Oct-2016</log_date_time>
      <log_message> No pending work flow to start.</log_message>
    </logs>
  </filterLogResults>
</rpc-reply>
```

（注） ロギング API の応答は XML 形式です。ログメッセージに XML 文字が含まれている場合はその文字がエスケープされるため、XML 準拠は解除されません。

一般的なインストールエラー

ここでは、一般的なインストールの問題とそのトラブルシューティング方法について説明します。

問題/エラー	考えられる理由	ユーザのアクション

インストール時に `sudo escadm status` を使用して ESC サービスのステータスを確認する際に発生する問題	サービスの中には、開始に時間がかかるものや、開始時に問題が発生するものがあります。	**1.** 次のいずれかの方法で、問題を特定します。
		• ログ /var/log/esc/escadm.log の確認
		`$ cat /var/log/esc/escadm.log`
		• 「-V」を escadm ステータスに追加して、ESC サービスの詳細出力を表示し、サービスが稼働中であることを確認します。
		2. 問題のあることが特定されたサービスのステータスを確認し、これらのサービスを手動で開始します。
		`$ sudo escadm <<service>> status//`
		`If the status is stopped or dead, manually start the services`
		`using the next command.`
		`$ sudo escadm <<service>> start --v`

ESC のインストール中に発生する認証に関するエラー	OpenStack のログイン情報に関する引数がありません。	OpenStack RC ファイルを取得し、OpenStack クライアントが正常に動作していることを確認します。
ESC HA 関連（アクティブ/スタンバイ）の問題		
ネットワークの問題		次の状態がないかどうかを確認します。 • 使用される両方の ESC ノードの静的 IP アドレスが、OpenStack の設定に基づいている。 • 各ネットワークインターフェイスのゲートウェイがアクセス可能である。

問題/エラー	考えられる理由	ユーザのアクション
ESC マスターノードが「マスターに切り替え中」の状態のままである。	これは、次の問題が原因である可能性があります。 • ESC HA アクティブ/スタンバイノードが、初回インストール中にピアに到達できない。 • データベースの問題（データベースの移行、データベースファイルの破損など）が原因で、ESC サービス（tomcat）が適切に起動できない。 • CDB ファイルが破損しているため、confd を開始できない。 • ファイルシステムに問題があるため、PostgreSQL を開始または初期化できない。 • ESC ノード間の接続が低速である。	次のことを確認します。 • ESC マスターノードとスタンバイノード間の接続。初回インストールでは、ESC マスターサービスがスタンバイノードに到達できない場合は起動しません。両方の ESC ノードが正常に展開され、相互に到達可能であることを確認する必要があります。 • ESC では /var/log/esc/esc_haagent.log（ESC 2.x）または /var/log/esc/escadm.log（ESC 3.x）にログが記録されており、問題のあるサービスを特定できます。 • esc_service および postgresql の問題については、/var/log/esc/escmanager.log にログが記録されます。

問題/エラー	考えられる理由	ユーザのアクション
ESC HA アクティブ/スタンバイの MTU 問題		ESC VM の場合は、ネットワークインターフェイスの MTU を 1500 から 1450 に減らします。MTU 値を減らすには、次の手順を実行します。 **1.** 変更するインターフェイスを特定します。/etc/sysconfig/network-scripts/ifcfg-ethX からインターフェイスにアクセスします。 X は変更するインターフェイス番号を表します。 **2.** VIM などのテキストエディタを使用して、インターフェイスの MTU 項目を追加または編集します（例：set MTU = 1450） **3.** インターフェイスを再起動します。 `network service restart`

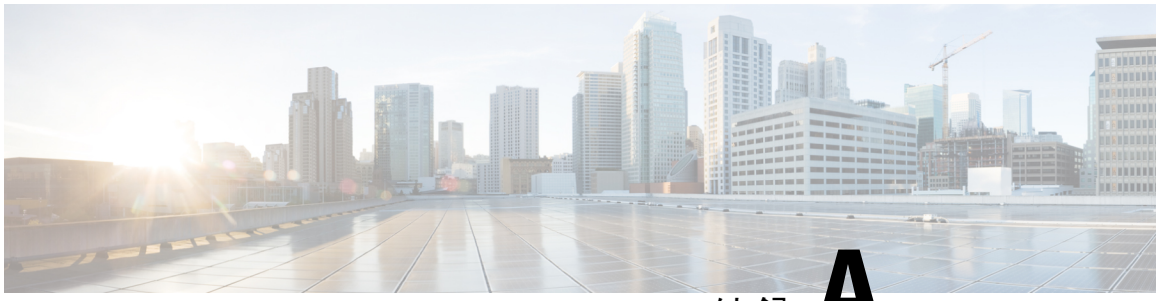
その他の問題		ESC HA アクティブ/スタンバイをトラブルシューティングする際に役立つログがいくつかあります。
		• ESC マネージャのログは /var/log/esc/escmanager.log に格納されています。
		• ESC サービスのスタートアップ/停止に関する ESC HA アクティブ/スタンバイログは、/var/log/esc/esc_haagent.log（ESC 2.X の場合）および /var/log/esc/escadm.log（ESC 3.X の場合）に格納されています。
		• キープアライブ構成の場合：
		  • /etc/keepalived/keepalived.conf でコンフィギュレーション ファイルを確認します。
		  • キープアライブのログは、grep keepalived または vrrp を実行すると、/var/log/messages に格納されます。
		• DRBD 構成の場合：
		  • DRBD 構成は、/etc/drbd.d/esc.res のコンフィギュレーション ファイルで確認できます。
		  • DRBD のログは、grep drbd を実行すると /var/log/messages に格納されます。

問題/エラー	考えられる理由	ユーザのアクション
ESC サービスのスタートアップに関する問題		

インストール時に sudo escadm status --v を使用すると、サービスステータスによって ETSI サービスが停止していることが示されます。	ホストに複数の IP アドレスがある場合、ETSI サービスは、コールバック URL の生成時に使用する IP アドレスを特定できません。
	この原因を検証するには、ログファイル /var/log/esc/etsi-vnfm/etsi-vnfm.log に次のような例外が記録されていないかを確認します。
	コンフィギュレーションファイルで、server.host プロパティを設定します。
		1. server.host プロパティは、次のように設定します。
		`sudo escadm etsi set -server_host <IP>`
		2. ETSI サービスを開始します。
		`sudo escadm etsi start`

ESC のフェールオーバーシナリオ

- お客様が多数の VNF を展開する場合
- お客様が DB をバックアップする場合
- VNF 内の 1 つ以上の VM が再展開によって回復（新しい VM ID）
- ESC エラー：永続的な障害が発生
- お客様の DB 復元：1 つ以上の VM ID で不一致がある場合
- 同じ VNF は失敗しますが、ESC が回復しようとして失敗するのは、VIM 上で VM ID によって検出されないためです。また、ポートが使用中のために再展開に失敗します。
- これを回避するには、VM をアウトオブバンドで削除して復旧します。 VM が削除されると、ESC で検出されなくなり、ポートが再展開で使用可能になります。



付録 A

Cisco Elastic Service Controller のインストール引数

ESC インスタンスを起動するには、次の *bootvm.py* スクリプト引数を指定する必要があります。

引数	説明
esc_hostname	ESC VM インスタンスのホスト名を指定します。
--image	ESC インスタンスを起動するために OpenStack Glance で使用されるイメージ ID を指定します。
--boot_volume	ESC インスタンスを起動するブート可能な外部ボリュームのボリューム名または ID を指定します。
--ignore-ssl-errors	「ignoreSslErrors」を「yes」に設定します。信頼できるルート証明書がインストールされていない場合に、開発環境またはテスト環境への展開に役立ちます。
--managers	SNMP トラップが配信される場所のカンマ区切りリストです。次の形式で指定する必要があります。 `udp:ipv4/port" or "udp:[ipv6]/port`
--net	ESC が接続する OpenStack 内のネットワーク ID または名前を指定します。
--ipaddr	（任意）ネットワークで ESC に割り当てられる IP アドレスを指定します。 （注） この IP アドレスは、--net 引数の net_id に対応している必要があります。
--gateway_ip	（任意）ESC のデフォルトゲートウェイの IP アドレスを指定します。

Cisco Elastic Service Controller のインストール引数

引数	説明
--os_auth_url	（任意）os_auth_url によって認証に使用される OpenStack Keystone の URL を指定します。
--os_username	（任意）os_username によって認証に使用される OpenStack Keystone のユーザ名を指定します。
--os_password	（任意）os_password によって認証に使用される OpenStack Keystone のパスワードを指定します。
--os_tenant_name	（任意）os_tenant_name によって ESC 展開に使用される OpenStack テナント名を指定します。
--bs_os_auth_url	（任意）bs_os_auth_url によって認証に使用される OpenStack Keystone の URL を指定します。
--bs_os_username	（任意）bs_os_username によって認証に使用される OpenStack Keystone のユーザ名を指定します。
--bs_os_password	（任意）bs_os_password によって認証に使用される OpenStack Keystone のパスワードを指定します。
--bs_os_tenant_name	（任意）bs_os_tenant_name によって ESC 展開に使用される OpenStack テナント名を指定します。
--flavor	（任意）ESC VMを起動するための OpenStack フレーバーの IDを指定します。
--security_rules_file	（任意）ESC VM のセキュリティルール（IP、ポートセキュリティ）を定義するファイルを指定します。
--etc_hosts_file	（任意）ESC VM の hosts ファイル（/etc/hosts）にエントリを追加するためのファイルを指定します。
--avail_zone	（任意）ESC 展開に使用される OpenStack ゾーンを指定します。
--esc_params_file	（任意）ESC展開用のデフォルトパラメータファイルを指定します。
--db_volume_id	（任意）ESC HAアクティブ/スタンバイ（ESC-HA アクティブ/スタンバイ）のデータベースストレージにマウントするための Cinder ボリューム ID を指定します。

引数	説明
--ha_node_list	（任意）プライマリ/スタンバイクラスタに含まれる HA アクティブ/スタンバイノードの IP アドレスのリストを指定します。複数のネットワーク インターフェイスを持つESC ノードの場合、これらの IP は、データ同期に使用されるネットワーク内のアドレスである必要があります。 （注） この引数は、レプリケーションベースの HA アクティブ/スタンバイソリューションのみに使用されます。
--kad_vip	（任意）keepalived VIP（仮想IP）の IP アドレスと keepalived VIP のインターフェイスを指定します（ESC-HA アクティブ/スタンバイ）。 VIP のインターフェイスを指定する形式の例として、--kad_vip 192.0.2.1:eth2 や --kad_vip [2001:cc0:2020::fc]:eth2 があります。
--kad_vif	（任意）keepalived 仮想 IP と keepalived VRRP のインターフェイスを指定します（ESC-HA アクティブ/スタンバイ）。VIP インターフェイスが引数 *kad_vip* を使用してすでに指定されている場合は、引数 --kad_vif を使用して keepalived VRRP のインターフェイスのみを指定することもできます。
--kad_vri	VRRP インスタンスの仮想ルータ ID を指定します。kad_vri に指定できる値は 0 ～ 254 です。同じ HA アクティブ/スタンバイ内の ESC VM は、同じ kad_vri 番号を使用する必要があります。L3 HA アクティブ/スタンバイに kad_vip が使用されない場合は、kad_vir を使用する必要があります。それ以外の場合は、kad_vri 引数を省略できます。
--route	ESC VM のルーティング設定を指定します。
--ntp_server	（任意）NTP サーバのアドレスを指定します。
--rsyslog_server	（任意）ESC がログを送信する rsyslog サーバの IP アドレスを指定します。
--rsyslog_server_port	（任意）ESC がログを送信する rsyslog サーバのポートを指定します。
--rsyslog_server_protocol	（任意）サーバにログを転送するために ESC によって使用されるプロトコルを指定します。

引数	説明
--secure	（任意）セキュリティの設定を有効にします。次の値を指定できます。 • **A**：root が完全にロックアウトされます。コンソールからでも root としてログインすることはできません。 • **B**：SELinux が強制モードで実行されます。 • **C**：IPv4/IPv6 テーブルが開始されます。 • **D**：SSH パスワード認証が無効になります。ESC VM に SSH でログインするには、秘密キーが必要です。 • **E**：confd のホストキーが再作成されます。
--host_mapping_file	（任意）VNF 展開用のホストマッピングファイルを指定します。
--version	（任意）bootvm.py のバージョンを出力して終了します。
--rng_virtio	RNG Virtio デバイスを使用した Libvirt/KVM での ESC VM のインストールと展開を有効にします。デフォルト値は次のとおりです。device=/dev/random rate_period=1000 rate_bytes=1024

引数	説明
--user_pass	

引数	説明
	--user_confd_pass とともに、3.0 以降に必須の引数です。 この引数により、ESC VM にアクセスするユーザが追加されます。管理者権限を持たないユーザ（非管理者/非 root ユーザ）を指定するには、この引数を使用します。user_name: password の形式を使用します。Linux（SSH/コンソールアクセス）用の管理者アカウントを作成するには、bootvm.py コマンドに少なくとも 1 つの user_pass 引数を含めることが必要です。必須のユーザクレデンシャル引数の構文は、次のとおりです。 `--user_pass admin:'PASSWORD-OR-HASH'[:OPTIONAL-PUBLIC-KEY-FILE][:OPTIONAL-ROLE]` このユーザは次の操作のみを実行できます。 • SSH を使用して ESC にログインする。 • Netconf CLI（esc_nc_cli、netconf-console など）にアクセスして操作する。 • /var/logs/esc から ESC 関連ログを読み取る。 • ローカルホスト経由で REST インターフェイスにアクセスする。 このユーザは次の操作を実行できません。 • ESC DB にアクセスし、ESC システムを再設定する。 • システムレベルのログにアクセスする。 • rsyslog、keepalived、DRDB などのシステムレベルのコンポーネントを設定する。 • 暗号化キーと、REST インターフェイスまたは ESC ログからの値にアクセスする。 次に、管理者アカウント用の --user_pass と、より強力なクリアテキストパスワードの例を示します。シェルの予約済み文字との競合を回避するには、引用符を使用します。 -user_pass admin:'Strong4Security!'. ESC をインストールする別の方法として、両方の管理者アカウントにパスワードハッシュを使用する例を示します。シェルの予約済み文字との競合を回避するには、引用符を使用します。 --user_pass admin:'$algorithm$salt$hash-of-salt-password' ESC 2.1 以降では、この属性の公開キーが受け入れられます。たとえば、次の例では、ユーザ「admin」のパスワードとして「admin321」が生成され、キーファイルとして */tmp/abc.pub* を使用して公開キーが挿入されます。

引数	説明
	`--user_pass admin:admin321:/tmp/abc.pub`
--user_confd_pass	confd ユーザを変更するために使用されます。ConfD（netconf/cli アクセス）用の管理者アカウントを作成するには、bootvm.py コマンドに少なくとも 1 つの user_confd_pass を含めることが必要です。必須のユーザクレデンシャル引数の構文は、次のとおりです。 `--user_confd_pass admin:'PASSWORD-OR-HASH'[:OPTIONAL-PUBLIC-KEY-FILE]` 次に、管理者アカウント用の --user_confd_pass と、より強力なクリアテキストパスワードの例を示します。シェルの予約済み文字との競合を回避するには、引用符を使用します。 --user_confd_ pass:'Strong4Security!'. ESC をインストールする別の方法として、両方の管理者アカウントにパスワードハッシュを使用する例を示します。シェルの予約済み文字との競合を回避するには、引用符を使用します。 --user_confd_ pass:'$algorithm$salt$hash-of-salt-password' ESC 2.1 以降では、この属性の公開キーが受け入れられます。たとえば、次の例では、ユーザ「admin」のパスワードとして「admin321」が生成され、キーファイルとして */tmp/abc.pub* を使用して公開キーが挿入されます。--user_confd_ pass:admin321:/tmp/abc.pub
--esc_portal_startup	（任意）ESC ポータルを開始します。
--log	（任意）ログファイルを指定します。デフォルトでは、stdout にログが記録されます。
--esc_monitor_check_ips	（任意）esc_monitor によってモニタする必要がある IP アドレスを指定します（HA アクティブ/スタンバイフェールオーバーの場合）。
--enable-https-rest	（任意）作成された ESC VM のセキュアな REST インターフェイスを有効にします。
--enable-http-rest	（任意）作成された ESC VM の非セキュアな REST インターフェイスを有効にします。
--disable-rest-auth	（任意）REST API 認証を無効にします。 （注） 実稼働環境で REST 認証を無効にすることはできません。

引数	説明
--enable-snmp-agent	（任意）SNMP サービスの自動起動を有効にします。デフォルト値は [いいえ（False）] です。
--ha_mode	HA アクティブ/スタンバイインストールのための ESC HA アクティブ/スタンバイモードを指定します。HA アクティブ/スタンバイで使用可能な次のオプションのいずれかを指定します。**no_ha**：HA なし、**cinder**：共有 Cinder ボリューム、**drbd**：組み込み DRBD、**drbd_on_cinder**：Cinder ボリュームによる DRBD
--enable-https-etsi	（任意）作成された ESC VM 用のセキュアな ETSI REST インターフェイスを有効にします。
--enable-http-etsi	（任意）作成された ESC VM 用の非セキュアな ETSI REST インターフェイスを有効にします。実稼働環境でこのインターフェイスを有効にすることは推奨されません。
--encrypt_key	暗号用のキーを指定します。
--proxy	特定のポートでプロキシを使用します。
--noproxy	プロキシを使用しないホストを一覧表示します。
--kad_unicast_src_ip	ユニキャストの送信元 IP アドレスを指定します。これは ESC VM がユニキャスト（L3）VRRP 通信に使用するインターフェイスの IP アドレスです。 例：--kad_unicast_src_ip 10.0.0.1
--kad_unicast_peer	ユニキャストのピア IP アドレスを指定します。これは ESC ピア VM がユニキャスト（L3）VRRP 通信に使用するインターフェイスの IP アドレスです。 例：--kad_unicast_peer 10.0.0.1

引数	説明
--placement_hint	この引数と、サーバグループ、samehost、differenthost フィルタを使用して、ESC HA アクティブ/スタンバイ仮想マシンの配置を指定します。 例： • --placement_hint different_host=2b299428-e7a7-4528-8566-9a4970183c6a（ID は VM UUID である必要があります） • --placement_hint same_host=2b299428-e7a7-4528-8566-9a4970183c6a（ID は VM UUID である必要があります） • --pacement_hint group=4c7758ab-e9cb-4cf0-8f02-344ec666365b（ID はサーバグループ UUID である必要があります）
--format {json}	この引数を使用して、出力内の成功および障害メッセージをキャプチャします。 例：$ ./bootvm.py --image ESC-2_3_0_8 --net network --format json --test-0 { "status" : "Success" , "vm_uuid" : "UUID" }
--user_rest_pass	REST API にアクセスするユーザを追加します。形式は username: password です。このオプションは繰り返し指定できます。
--user_portal_pass	ポータルユーザを追加します。形式は username: password です。このオプションは繰り返し指定できます。
--user_etsi_pass	ETSI REST API にアクセスするユーザを追加します。形式は username:password です。このオプションは繰り返し指定できます。

引数	説明
--no_vim_credentials	VIM ログイン情報を渡さずに ESC を展開するには、この引数を使用します。この引数を使用すると、次のパラメータはインストール中に渡されません。 • --os_auth_url • --os_username • --os_password • --os_tenant_name 展開が完了した後、ユーザは、ESC の VIM/VIM ユーザ API（REST/Netconf）を使用して、これらの VIM ログイン情報を設定できます。REST API および Netconf を使用した設定の詳細については、「インストール後のタスク」の章にある「*ESC* インストール後の *VIM* ログイン情報の設定」を参照してください。
--etsi_startup	この引数は、ESC 4.4 以降では廃止されており、今後のリリースでは使用できません。--etsi_startup を使用すると、使用できる代わりの適切な引数を含むエラーメッセージが表示されます。「--enable-etsi-http」と「--enable-etsi-https」を参照してください。

Cisco Elastic Service Controller インストーラファイルの参照

ファイル	説明
security_rules_file	このファイルには次のものが含まれています。 • テナントのセキュリティグループを作成するためのセキュリティルール。 • テナントへのトラフィックを許可する設定。
etc_hosts_file	このファイルには、/etc/hosts ファイルに追加する 1 つ以上のエントリが含まれています。
esc_params_file	このファイルには、ESC のさまざまなパラメータを設定するための情報が含まれています。esc_params_file で設定できるパラメータの詳細については、下の表で説明します。
host_mapping_file	このファイルには、ホストに基づいてネットワークをマッピングするための情報が含まれています。

ESC 設定パラメータ

このファイルを使用して、インストール時にさまざまな ESC パラメータを設定できます。設定可能なパラメータを表に示します。

このファイルを使用した設定の例を次に示します。

```
openstack.endpoint=adminURL
affinity.filter=ServerGroupAffinity
```

表 *7: ESC* 設定パラメータ

esc_param.conf	タイプ	デフォルト値	説明
default.vm_recovery_retries_max	数値	3	許容されるリカバリの試行回数（VMあたり）。

openstack.endpoint	文字列	publicURL	ESC の keystone エンドポイント値を設定するパラメータ。オプション：adminURL、publicURL
			CLIまたはRESTサービスを使用して、デフォルト値を変更できます。
			CLI を使用：
			`$ sudo escadm escmanager config set --key openstack.endpoint --value publicURL`
			`{ "category": "OPENSTACK", "type": "STRING", "value": "publicURL", "key": "ENDPOINT" }`
			REST を使用：
			`$ curl -X PUT http://172.16.0.1:8080/ESCManager/v0/config/openstack/endpoint/publicURL`

log_level	文字列	INFO	ロギングのレベル。オプション：INFO、Trace、DEBUG
affinity.filter	文字列	SameHostFilter	PolicyEngine を構築し、VM ポリシーテーブルを初期化するために使用される定数文字列。 オプション：SameHostFilter、ServerGroupAffinity
anti_affinity.filter	文字列	DifferentHostFilter	PolicyEngine を構築し、VM ポリシーテーブルを初期化するために使用される定数文字列。 オプション：DifferentHostFilter

Cisco Elastic Service Controller のインストール引数

（注） ESC の場合、ESC ポリシーエンジンに対してデフォルトで SameHostFilter と DifferentHostFilter が使用されますが、OpenStack の場合、デフォルトではこれらのフィルタが設定されないことがあります。その場合、OpenStack の nova サービスの **/etc/nova/nova.conf** ファイルにある次のスケジューラオプションに、SameHostFilter と DifferentHostFilter を追加する必要があります。

```
scheduler_default_filters = RetryFilter, AvailabilityZoneFilter,RamFilter, ComputeFilter,
 ComputeCapabilitiesFilter,
ImagePropertiesFilter, ServerGroupAntiAffinityFilter, ServerGroupAffinityFilter,
DifferentHostFilter, SameHostFilter
```

OpenStack 用の ServerGroupAntiAffinityFilter

ESC は、OpenStack 用の ServerGroupAntiAffinityFilter の使用に対応できます。

REST

```
PUThttp://localhost:8080/ESCManager/v0/config/anti_affinity/filter/ServerGroupAntiAffinity

PUThttp://localhost:8080/ESCManager/v0/config/affinity/filter/ServerGroupAffinity
```

CLI

```
sudo escadm escmanager config set --key ANTI_AFFINITY.FILTER --value
ServerGroupAntiAffinity
sudo escadm escmanager config set --key AFFINITY.FILTER --value ServerGroupAffinity
```

重要なポイント

OpenStack の ServerGroupAntiAffinityFilter は、inter-dep anti-affinit、scaling の使用や、servergroup とデフォルトフィルタ（samehost/differenthost）の混合使用をサポートしていません。ServerGroupAntiAffinity フィルタを使用している場合、VM グループ内での配置は許可されません。VM ベースの配置ポリシーには **<placement_group >** のみを使用できます。 vm_group ごとに 1 つの VM を使用できます。2 つの異なる placement_group に単一の VM グループを追加することはできません。

ESC サービス、ポート、およびセキュリティグループの概要

表 *8 :* 外部サービス（***External Services***）

	サービス	カンファレンスの公開/非公開（**Visibility**）	任意かどうか	インターフェイス	プロトコル	ポート
1	sshd	外部（オーケストレーション）	いいえ	0.0.0.0	TCP	22

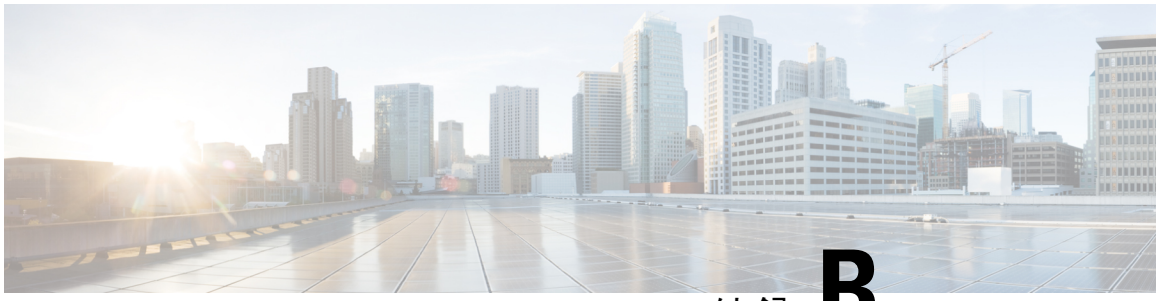
Cisco Elastic Service Controller のインストール引数

	サービス	カンファレンスの公開/非公開（Visibility）	任意かどうか	インターフェイス	プロトコル	ポート
2	ESC Web UI/ポータル（HTTPS）	外部（オーケストレーション）	はい（代わりに REST および/または Netconf を使用可）	0.0.0.0	TCP	443（以前は 9001）
3	ESC Netconf API	外部（オーケストレーション）	はい（代わりに REST および/またはポータルを使用可）	0.0.0.0	TCP	830
4	ESC SNMP	外部（オーケストレーション）	はい（カスタムユーザデータ/esc-config.yaml のみで設定可能）	0.0.0.0	TCP	2001
5	ESC DRBD（HA アクティブ/スタンバイレプリケーション）	外部（オーケストレーション）	いいえ。HA アクティブ/スタンバイの設定に必要。	0.0.0.0	TCP	7789
6	ESC REST API（HTTPS）	外部（オーケストレーション）	はい（代わりにポータルおよび/または Netconf を使用可）	0.0.0.0	TCP	8443
7	ESC Keepalived	外部（オーケストレーション）	いいえ。HA アクティブ/スタンバイの設定に必要。	0.0.0.0	マルチキャスト VRRP	該当なし
8	ETSI-VNFM（HTTP）	外部	はい（etsi-production.properties を使用して設定可）	0.0.0.0	TCP	8250

Cisco Elastic Service Controller のインストール引数

	サービス	カンファレンスの公開/非公開（Visibility）	任意かどうか	インターフェイス	プロトコル	ポート
9	ETSI-VNFM（HTTPS）	外部	はい（etsi-production.properties を使用して設定可）	0.0.0.0	TCP	8251
10	ESC ヘルス API	外部（オーケストレーション）	いいえ	0.0.0.0	TCP	8060
11	D-MONA REST API	外部	いいえ	0.0.0.0	TCP	8443
12	Consul サービス [1]	外部	いいえ	0.0.0.0	TCP	8300、8301、8302
13	ConfD	外部 [2]	A/A セットの場合はいいえ	ESC ノード IP に限定 [3]	TCP	4565
14	PostgreSQL	外部 [4]	A/A セットの場合はいいえ	ESC ノード IP に限定 [5]	TCP	7878
15	ESCManager RMI レジストリ [6]	外部	A/A セットの場合はいいえ	ESC ノード IP に限定	TCP	8679
16	ESCManager RMI サービス [7]	外部	A/A セットの場合はいいえ	ESC ノード IP に限定	TCP	8680

1 A/A ESC セットのみに必要です。それ以外の場合、ポートはリッスンされません。
2 ESC 5.0 以降のみに導入
3 ESC A/A セット（3 VM）
4 ESC 5.0 以降のみに導入
5 ESC A/A セット（3 VM）
6 A/A ESC セットのみに必要です。それ以外の場合、ポートはリッスンされません。
7 A/A ESC セットのみに必要です。それ以外の場合、ポートはリッスンされません。



付録 B

CSP 2100 のサンプルファイルで使用される変数リスト

ユーザデータファイルを作成する場合や ESC を設定する場合、サンプルファイルで使用されている次の変数リストの値を準備する必要があります。

表 *9*: 変数リスト

変数名	目的
VAR_TIMEZONE	ESC クロックで使用されるタイムゾーン
VAR_SERVICE_NAME	CSP の ESC サービス名
VAR_NTP_SERVER	NTP サーバの IP アドレス
VAR_NETWORK1_NETMASK	eth1 インターフェイスのネットマスク（デュアルインターフェイス ESC）
VAR_NETWORK1_NAME	ESC の eth1 インターフェイスが存在する CSP 上のネットワーク名（デュアルインターフェイス ESC）
VAR_NETWORK1_IPADDR	eth1 インターフェイスの IP アドレス（デュアルインターフェイス ESC）
VAR_NETWORK1_GATEWAY	eth1 インターフェイスのゲートウェイ（デュアルインターフェイス ESC）
VAR_NETWORK0_NETMASK	eth0 インターフェイスのネットマスク
VAR_NETWORK0_NAME	ESC の eth0 インターフェイスが存在する CSP 上のネットワーク名

CSP 2100 のサンプルファイルで使用される変数リスト

変数名	目的
VAR_NETWORK0_KADVRI	HA に使用される VRRP ID。HA ペアのサブネット内で一意である必要があり、両方のESCで使用されているものと同じ値である必要があります。 範囲は 1 ～ 254 です。
VAR_NETWORK0_KADVIP	現在のマスター ESC に接続する HA ペアの VIP
VAR_NETWORK0_IPADDR2	他の ESC の eth0 インターフェイスに割り当てられた IP アドレス
VAR_NETWORK0_IPADDR	ESC の IP アドレス（eth0 インターフェイス）
VAR_NETWORK0_GATEWAY	eth0 インターフェイスのゲートウェイ
VAR_NAMESERVER_IP	DNS サーバの IP アドレス
VAR_LOCAL_HOSTNAME	ESC のホスト名
CSP_IP_ADDRESS	使用する CSP 2100 の IP アドレス