Cisco ACI Multi-Site Orchestrator のインストール

この章の内容は、次のとおりです。

- Cisco ACI Multi-Site Orchestrator の展開に関するガイドライン （1 ページ）
- Python を使用した Cisco ACI Multi-Site Orchestrator の展開 （2 ページ）
- ESX で VCenter を使用せずに Cisco ACI Multi-Site Orchestrator を直接展開する （6 ページ）
- OVA を使用した Cisco ACI Multi-Site Orchestrator の展開 （9 ページ）

Cisco ACI Multi-Site Orchestrator の展開に関するガイドライン

Cisco ACI Multi-Site Orchestrator はいくつかの方法で展開できます。たとえば、vCenter で OVA を使用する方法、vCenter を使用せずに ESX サーバに直接導入する方法、Python スクリプトを使用する方法などがあります。Cisco ACI Multi-Site Orchestrator リリース 2.0(1) 以降を展開する場合は Python の使用をお勧めします。手動による多くの手順が自動化されるほか、後で Cisco ACI Multi-Site Orchestrator のソフトウェア アップグレードを行う場合にリモートでの実行がサポートされます。

Docker サブネットに関する考慮事項

Multi-Site Orchestrator のアプリケーション サービスは Docker コンテナ内で実行されます。内部の 10.0.0.0/24 ネットワークが Docker のスウォーム アプリケーション サービスにより使用され、Multi-Site Orchestrator のインストール時は変更することができません。ファブリックにある他のサービスがこのネットワークに存在しないようにしてください。

VMware vSphere の要件

次の表は、Cisco ACI Multi-Site Orchestrator を使用する場合の VMware vSphere の要件をまとめたものです。

（注） VM ごとに、vCPU、メモリ、およびディスク容量に関する以下の要件を確保し、それらが共有リソース プールの一部とならないようにしてください。

表 *1 : VMware vSphere* の要件

Cisco ACI Multi-Site Orchestrator のバージョン	VMware vSphere の要件
リリース 2.0(1)	• ESXi 6.0 以降 • 6 個の vCPU（8 個の vCPU を推奨） • 24 GB の RAM • 64 GB のディスク

Python を使用した Cisco ACI Multi-Site Orchestrator の展開

以降の各セクションでは、Python を使用して Cisco ACI Multi-Site Orchestrator を準備し展開する方法について説明します。

Cisco ACI Multi-Site Orchestrator を展開するための Python 環境のセットアップ

このセクションでは、Python を使用して Cisco ACI Multi-Site Orchestrator を展開するための Python 環境のセットアップ方法について説明します。

（注） Python を使用して展開できるのはリリース 1.2(1) 以降のみです。

始める前に

- Python 2.7.14 以降または Python 3.4 以降があることを確認します。

ステップ 1 **ACI Multi-Site Tools イメージ**を Cisco ACI Multi-Site ソフトウェア ダウンロード リンクからダウンロードします。

a) ソフトウェア ダウンロード リンクにアクセスします。

https://software.cisco.com/download/home/285968390/type

b) [ACI Multi-Site Software] をクリックします。

c) **ACI Multi-Site Tools イメージ**のリリース バージョンを選択し、ダウンロード アイコンをクリックします。

ステップ 2 ファイルを解凍して抽出します。

```
$ tar xvf tools-msc-<build_number>.tar.gz

msc_cfg_example.yml
msc_lib.py
msc_vm_clean.py
msc_vm_util.py
Node.py
python
README
requirements.txt
```

ステップ 3 tools-msc-<ビルド番号> ディレクトリに変更します。

```
$ cd tools-msc-<build_number>
```

ステップ 4 Python 2.7.14 以降または Python 3.4 以降のどちらかを実行していることを確認します。

```
$ python -V
Python 2.7.15
```

ステップ 5 Python パッケージをインストールする権限を持っていることを確認します。たとえば、シェルを変更してスーパーユーザになります。

```
$ sudo bash
```

ステップ 6 インターネット アクセスでプロキシを使用する予定の場合には、プロキシを次のように設定します。

```
$ export http_proxy=your_proxy_ip:your_proxy_port
$ export https_proxy=your_proxy_ip:your_proxy_port
```

ステップ 7 Python パッケージ インストーラをインストールします。

```
# python -m ensurepip
Collecting setuptools
Collecting pip
Installing collected packages: setuptools, pip
Successfully installed pip-9.0.3 setuptools-39.0.1
```

ステップ 8 Requirements.txt に従ってパッケージをインストールします。

virutalenv を使用してパッケージをインストールすることをお勧めします。この方法では、インストールしたパッケージがシステムにある既存のパッケージに影響を与えることがありません。virtualenv の使い方の詳細については、「pip と virtualenv を使用したパッケージのインストール」を参照してください。

```
# python -m pip install -r requirements.txt
```

（注） Python のインストールを正常に完了する必要があります。エラーが発生した場合は、必ずそれらを解決してから次のセクションに進んでください。解決しない場合、Cisco ACI Multi-Site Orchestrator の Python スクリプトが正しく機能しません。

ステップ 9 シェルを終了します。

```
# exit
$
```

次のタスク

このセクションの手順が完了すると、Python を使用した Cisco ACI Multi-Site Orchestrator の展開 （5 ページ）に進むことができます。

サンプルの msc_cfg.yml ファイル

このセクションでは、Python を使用して Cisco ACI Multi-Site を展開するためのサンプルの msc_cfg.yml ファイルを紹介します。

（注） Python を使用して展開できるのはリリース 1.2(1) 以降のみです。

次の構成ファイルの例では、すべての VM が同じホストの下で作成されます。構成ファイルの「host」パラメータを任意のノード レベルで指定して、Multi-Site の VM をさまざまなホストに作成できます。

```
# Vcenter parameters
vcenter:
  name: dev5-vcenter1
  user: administrator@vsphere.local

  # Host under which the MSC VMs need to be created
  host: 192.64.142.55

  # Path to the MSC OVA file
  msc_ova_file: ../images/msc-1.2.1g.ova

  #  Optional. If not given default library name of "msc-content-lib" would be used
  # library: content-library-name

  # Library datastore name
  library_datastore: datastore1

  # Host datastore name
  host_datastore: datastore1

  # MSC VM name prefix. The full name will be of the form vm_name_prefix-node1
  vm_name_prefix: msc-121g

  # Wait Time in seconds for VMs to come up
  vm_wait_time: 120


# Common parameters for all nodes
common:
  # Network mask
  netmask: 255.255.248.0

  # Gateway' IP address
  gateway: 192.64.136.1
```

```
  # Domain Name-Server IP. Leave blank for DHCP
  nameserver: 192.64.136.140

  # Network label of the Management network port-group
  management: "VM Network"


# Node specific parameters
node1:
  # To use static IP, please specify valid IP address for the "ip" attribute
  ip: 192.64.136.204

  # Node specific "netmask" parameter over-rides the comman.netmask
  netmask: 255.255.248.0

node2:
  # To obtain IP via DHCP, please leave the "ip", "gateway" & "nameserver" fields blank
  ip:
  gateway:
  nameserver:

node3:
  # To obtain IP via DHCP, please leave the "ip", "gateway" & "nameserver" fields blank
  ip:
  gateway:
  nameserver:
```

Python を使用した Cisco ACI Multi-Site Orchestrator の展開

このセクションでは、Python を使用した Cisco ACI Multi-Site の展開方法について説明します。

始める前に

- 『CiscoACI Multi-Site Hardware Requirements Guide』に記載されているハードウェア要件と互換性を満たしていることを確認します。
- Cisco ACI Multi-Site Orchestrator を展開するための Python 環境のセットアップ （2 ページ）の説明に従って Python 環境をセットアップします。
- ツールが実行されるサーバから vCenter に到達できることを確認します。

ステップ 1 `msc_cfg_example.yml` ファイルをコピーし、名前を `msc_cfg.yml` に変更します。

```
$ cp msc_cfg_example.yml msc_cfg.yml
```

ステップ 2 `msc_cfg.yml` ファイルを編集し、環境に合わせてすべてのパラメータを設定します。

必須パラメータはすべて大文字で記載しています（例：`<VCENTER_NAME>`）。

サンプルの `msc_cfg.yml` ファイルについては、「サンプルの msc_cfg.yml ファイル （4 ページ）」を参照してください。

ステップ 3 MSC VM を展開して準備するためのスクリプトを実行します。

```
$ python msc_vm_util.py
```

プロンプトが表示されたら、vCenter、node1、node2、node3 のそれぞれのパスワードを入力します。

サポートされているすべてのオプションを表示するには、次のように入力します。

```
$ python msc_vm_util.py -h
```

スクリプトにより、3 個の Multi-Site VM が作成され、初期展開スクリプトが実行されます。VM が作成されて展開スクリプトが実行されるまでに数分かかる場合があります。正常に実行されると、Multi-Site クラスタが使用できるようになります。Multi-Site GUI にアクセスして確認することができます。

ESX で VCenter を使用せずに Cisco ACI Multi-Site Orchestrator を直接展開する

このセクションでは、ESX で VCenter を使用せずに Cisco ACI Multi-Site Orchestrator を直接展開する方法について説明します。

ステップ 1 Cisco ACI Multi-Site ソフトウェア ダウンロード リンクから `msc-<バージョン>.ova` をダウンロードします。

a) ソフトウェア ダウンロード リンクにアクセスします。

https://software.cisco.com/download/home/285968390/type

b) [ACI Multi-Site Software] をクリックします。

c) リリース バージョンのイメージを選択し、ダウンロード アイコンをクリックします。

ステップ 2 新しい一時ディレクトリに OVA ファイルを展開します。

```
$ mkdir msc_ova
$ cd msc_ova
$ tar xvf ../msc-<version>.ova
esx-msc-<version>.ovf
esx-msc-<version>.mf
esx-msc-<version>.cert
msc-<version>.ovf
msc-<version>.mf
msc-<version>.cert
msc-<version>-disk1.vmdk
```

これで複数のファイルが作成されます。

ステップ 3 ESX vSphere クライアントを使用して OVF を展開します。

a) [File] > [Deploy OVF Template] に移動し、`esx-msc-<バージョン>.ovf` ファイルを選択します。

b) メニュー オプションの残りの部分を完了し、VM を展開します。

c) ステップ 3 を繰り返して、それぞれの Cisco ACI Multi-Site Orchestrator ノードを作成します。

ステップ 4 コマンドライン インターフェイス (CLI) またはテキスト ユーザ インターフェイス (TUI) ツールを使用して、各 VM のホスト名を設定します。

a) CLI の使用：

1 番目のノードで次のコマンドを入力します。

```
# hostnamectl set-hostname node1
```

2 番目のノードで次のコマンドを入力します。

```
# hostnamectl set-hostname node2
```

3 番目のノードで次のコマンドを入力します。

```
# hostnamectl set-hostname node3
```

TUI ツールを使用します。

nmtui コマンドを入力して、各 VM のホスト名を設定します。

b) VM ごとに、いったんログアウトしてから再度ログインする必要があります。

ステップ 5 node1 で、次の手順に従います。

a) SSH を使用して node1 に接続します。

b) /opt/cisco/msc/builds/<ビルド番号>/prodha ディレクトリに変更します。

```
[root@node1]# cd /opt/cisco/msc/builds/<build_number>/prodha
```

c) msc_cfg_init.py コマンドを実行します。

```
[root@node1 prodha]# ./msc_cfg_init.py
Starting the initialization of the cluster...
.
.
.
Both secrets created successfully.

Join other nodes to the cluster by executing the following on each of the other nodes:
./msc_cfg_join.py \
SWMTKN-1-4pu9zc9d81gxxw6mxec5tuxdt8nbarq1qnmfw9zcme1w1tljZh-7w3iwsddvd97ieza3ym1s5gj5 \
<ip_address_of_the_first_node>
```

d) 1 番目のノードの管理 IP アドレスを書き留めて、次のコマンドを入力します。

```
# ifconfig
inet 10.23.230.151  netmask 255.255.255.0  broadcast 192.168.99.255
```

ステップ 6 node2 で、次の手順に従います。

a) SSH を使用して node2 に接続します。

b) /opt/cisco/msc/builds/<ビルド番号>/prodha ディレクトリに変更します。

```
# cd /opt/cisco/msc/builds/<build_number>/prodha
```

c) ステップ 5c および 5d に示された 1 番目のノードの IP アドレスを使用して、msc_cfg_join.py コマンドを実行します。

例：

```
# ./msc_cfg_join.py \
SWMTKN-1-4pu9zc9d81gxxw6mxec5tuxdt8nbarq1qnmfw9zcme1w1tljZh-7w3iwsddvd97ieza3ym1s5gj5 \
10.23.230.151
```

ステップ 7 node3 で、次の手順に従います。

a) SSH を使用して node3 に接続します。

b) /opt/cisco/msc/builds/<ビルド番号>/prodha ディレクトリに変更します。

```
# cd /opt/cisco/msc/builds/<build_number>/prodha
```

c) ステップ 5c および 5d に示された 1 番目のノードの IP アドレスを使用して、msc_cfg_join.py コマンドを実行します。

例：

```
# ./msc_cfg_join.py \
SWMTKN-1-4pu9zc9d81gxxw6mxec5tuxdt8nbarq1qnmfw9zcme1w1tljZh-7w3iwsddvd97ieza3ym1s5gj5 \
10.23.230.151
```

ステップ 8 任意のノードで、ノードが正常に稼働していることを確認します。各ノードについて、[STATUS] が Ready、 [AVAILABILITY] が Active となっていて、[MANAGER STATUS] が 1 つのみ Leader と表示されることを除いて Reachable となっていることを確認します。

```
# docker node ls
```

サンプル出力：

```
ID                           HOSTNAME    STATUS    AVAILABILITY    MANAGER STATUS
g3mebdulaed2n0cyywjrtum31    node2       Ready     Active          Reachable
ucgd7mm2e2divnw9kvm4in7r7    node1       Ready     Active          Leader
zjt4dsodu3bff3ipn0dg5h3po *  node3       Ready     Active          Reachable
```

ステップ 9 任意のノードで、msc_deploy.py コマンドを実行します。

```
# ./msc_deploy.py
```

ステップ 10 任意のノードで、すべての [REPLICAS] が稼働していることを確認します。たとえば、3/3 (3 つのうち 3 つ) または 1/1 (1 つのうち 1 つ) と示されることを確認します。

例：

```
# docker service ls
```

サンプル出力：

```
ID           NAME                 MODE       REPLICAS  IMAGE                          PORTS
1jmn525od7g6 msc_kongdb           replicated 1/1       postgres:9.4
2imn83pd4l38 msc_mongodb3         replicated 1/1       mongo:3.4
2kc6foltcv1p msc_siteservice      global     3/3       msc-siteservice:0.3.0-407
6673appbs300 msc_schemaservice    global     3/3       msc-schemaservice:0.3.0-407
clqjgftg5ie2 msc_kong             global     3/3       msc-kong:1.1
j49z7kfvmu04 msc_mongodb2         replicated 1/1       mongo:3.4
lt4f2l1yqiw1 msc_mongodb1         replicated 1/1       mongo:3.4
mwsvixcxipse msc_executionengine  replicated 1/1       msc-executionengine:0.3.0-407
qnleu9wvw800 msc_syncengine       replicated 1/1       msc-syncengine:0.3.0-407
tfaqq4tkyhtx msc_ui               global     3/3       msc-ui:0.3.0-407
*:80->80/tcp,*:443->443/tcp
ujcmf70r16zw msc_platformservice  global     3/3       msc-platformservice:0.3.0-407
uocu9msiarux msc_userservice      global     3/3       msc-userservice:0.3.0-407
```

ステップ 11 ブラウザを開き、3 つのノードの任意の IP アドレスを入力して、Cisco ACI Multi-Site Orchestrator GUI を起動します。

例：

https://10.23.230.151

ステップ 12 Cisco ACI Multi-Site Orchestrator GUI にログインします。デフォルトのログインは **admin** で、パスワードは **we1come!** です。

ステップ 13 初めてログインしたときは、パスワードをリセットする必要があります。現在のパスワードと新しいパスワードを入力します。

新しいパスワードの要件は、次のとおりです。

- 6 文字以上
- 最低 1 つの英字
- 最低 1 つの数字
- * およびスペースとは異なる、少なくとも 1 つの特殊文字

初回操作の詳細については、「初回操作の概要」を参照してください。

OVA を使用した Cisco ACI Multi-Site Orchestrator の展開

このセクションでは、OVA を使用して Cisco ACI Multi-Site Orchestrator リリース 2.0(x) を展開する方法について説明します。

始める前に

- ハードウェア要件を満たしていることを確認します。詳細については、『Cisco ACI Multi-Site Hardware Requirements Guide』を参照してください。
- VMware vSphere の要件を満たしていることを確認します。詳細については、「Cisco ACI Multi-Site Orchestrator の展開に関するガイドライン （1 ページ）」を参照してください。

ステップ 1 仮想マシン（VM）をインストールします。

a) WebGUI か vSphere Client のどちらかを使用して OVA を展開します。

（注） ESX で直接、OVA を展開することはできません。vCenter を使用して展開する必要があります。ESX で直接、Cisco ACI Multi-Site Orchestrator を展開する必要がある場合は、vCenter を使用せずに OVA を展開して Multi-Site をインストールする手順について「ESX で VCenter を使用せずに Cisco ACI Multi-Site Orchestrator を直接展開する （6 ページ）」を参照してください。

[Properties] ダイアログボックスで、VM ごとに該当する情報を入力します。

- [Enter password] フィールドに、パスワードを入力します。
- [Confirm password] フィールドに、同じパスワードをもう一度入力します。

- [Hostname] フィールドに、それぞれの Cisco ACI Multi-Site Orchestrator ノードのホスト名を入力します。任意の有効な Linux ホスト名を使用できます。
- [Management Address]（ネットワーク アドレス）フィールドに、ネットワーク アドレスを入力するか、空欄のままにして DHCP 経由でネットワーク アドレスを取得します。
- [Management Netmask]（ネットワーク ネットマスク）フィールドに、ネットワーク ネットマスクを入力するか、空欄のままにして DHCP 経由でネットワーク ネットマスクを取得します。
- [Management Gateway]（ネットワーク ゲートウェイ）フィールドに、ネットワーク ゲートウェイを入力するか、空欄のままにして DHCP 経由でネットワーク ゲートウェイを取得します。
- [Domain Name System Server]（DNS サーバ）フィールドに、DNS サーバを入力するか、空欄のままにして DHCP 経由でDNS サーバを取得します。
- [Time-zone string (Time-zone)] フィールドに、有効なタイム ゾーン文字列を入力します。
- [NTP-servers] フィールドに、カンマで区切られたネットワーク タイム プロトコル サーバを入力するか、空欄のままにして NTP を無効にします。

  [Next] をクリックします。
- [Deployment settings] ペインで、指定したすべての情報が正しいことを確認します。
- [Power on after deployment] をクリックします。
- [Finish] をクリックします。
- 各 VM についてプロパティの設定を繰り返します。

b) 仮想マシンが互いに ping を実行できることを確認します。

ステップ 2 node1 で、次の手順に従います。

a) SSH を使用して node1 に接続します。

b) `/opt/cisco/msc/builds/<ビルド番号>/prodha` ディレクトリに変更します。

```
[root@node1]# cd /opt/cisco/msc/builds/<build_number>/prodha
```

c) `Msc_cfg_init.py` コマンドを実行します。

```
[root@node1 prodha]# ./msc_cfg_init.py
Starting the initialization of the cluster...
.
.
.
Both secrets created successfully.

Join other nodes to the cluster by executing the following on each of the other nodes:
./msc_cfg_join.py \
SWMTKN-1-4pu9zc9d81gxxw6mxec5tuxdt8nbarq1qnmfw9zcme1w1tljZh-7w3iwsddvd97ieza3ym1s5gj5 \
<ip_address_of_the_first_node>
```

d) 1 番目のノードの管理 IP アドレスを書き留めて、次のコマンドを入力します。

```
[root@node1 prodha]# ifconfig
inet 10.23.230.151  netmask 255.255.255.0  broadcast 192.168.99.255
```

ステップ 3 node2 で、次の手順に従います。

a) SSH を使用して node2 に接続します。

b) /opt/cisco/msc/builds/<ビルド番号>/prodha ディレクトリに変更します。

```
[root@node2]# cd /opt/cisco/msc/builds/<build_number>/prodha
```

c) ステップ 2c および d に示された 1 番目のノードの IP アドレスを使用して、msc_cfg_join.py コマンドを実行します。

例：

```
[root@node2 prodha]# ./msc_cfg_join.py \
SWMTKN-1-4pu9zc9d81gxxw6mxec5tuxdt8nbarq1qnmfw9zcme1w1tljZh-7w3iwsddvd97ieza3ym1s5gj5 \
10.23.230.151
```

ステップ 4 node3 で、次の手順に従います。

a) SSH を使用して node3 に接続します。

b) /opt/cisco/msc/builds/<ビルド番号>/prodha ディレクトリに変更します。

```
[root@node3]# cd /opt/cisco/msc/builds/<build_number>/prodha
```

c) ステップ 2c および d に示された 1 番目のノードの IP アドレスを使用して、msc_cfg_join.py コマンドを実行します。

例：

```
[root@node3 prodha]# ./msc_cfg_join.py \
SWMTKN-1-4pu9zc9d81gxxw6mxec5tuxdt8nbarq1qnmfw9zcme1w1tljZh-7w3iwsddvd97ieza3ym1s5gj5 \
10.23.230.151
```

ステップ 5 任意のノードで、ノードが正常に稼働していることを確認します。各ノードについて、[STATUS] が Ready、 [AVAILABILITY] が Active となっていて、[MANAGER STATUS] が 1 つのみ Leader と表示されることを除いて Reachable となっていることを確認します。

```
[root@node1 prodha]# docker node ls
```

サンプル出力：

```
ID                           HOSTNAME    STATUS    AVAILABILITY     MANAGER STATUS
g3mebdulaed2n0cyywjrtum31    node2       Ready     Active           Reachable
ucgd7mm2e2divnw9kvm4in7r7    node1       Ready     Active           Leader
zjt4dsodu3bff3ipn0dg5h3po *  node3       Ready     Active           Reachable
```

ステップ 6 任意のノードで、msc_deploy.py コマンドを実行します。

```
[root@node1 prodha]# ./msc_deploy.py
```

ステップ 7 任意のノードで、すべての [REPLICAS] が稼働していることを確認します。たとえば、3/3 (3 つのうち 3 つ) または 1/1 (1 つのうち 1 つ) と示されることを確認します。

例：

```
[root@node1 prodha]# docker service ls
```

サンプル出力：

```
ID            NAME                MODE       REPLICAS IMAGE                        PORTS
1jmn525od7g6  msc_kongdb          replicated 1/1      postgres:9.4
2imn83pd4l38  msc_mongodb3        replicated 1/1      mongo:3.4
```

```
2kc6foltcv1p  msc_siteservice     global     3/3      msc-siteservice:0.3.0-407
6673appbs300  msc_schemaservice   global     3/3      msc-schemaservice:0.3.0-407
clqjgftg5ie2  msc_kong            global     3/3      msc-kong:1.1
j49z7kfvmu04  msc_mongodb2        replicated 1/1      mongo:3.4
lt4f2l1yqiw1  msc_mongodb1        replicated 1/1      mongo:3.4
mwsvixcxipse  msc_executionengine replicated 1/1      msc-executionengine:0.3.0-407
qnleu9wvw800  msc_syncengine      replicated 1/1      msc-syncengine:0.3.0-407
tfaqq4tkyhtx  msc_ui              global     3/3      msc-ui:0.3.0-407
*:80->80/tcp,*:443->443/tcp
ujcmf70r16zw  msc_platformservice global     3/3      msc-platformservice:0.3.0-407
uocu9msiarux  msc_userservice     global     3/3      msc-userservice:0.3.0-407
```

ステップ 8 ブラウザを開き、3 つのノードの任意の IP アドレスを入力して、Cisco ACI Multi-Site Orchestrator GUI を起動します。

例：

https://10.23.230.151

ステップ 9 Cisco ACI Multi-Site Orchestrator GUI にログインします。デフォルトのログインは **admin** で、パスワードは **we1come!** です。

ステップ 10 初めてログインしたときは、パスワードをリセットする必要があります。現在のパスワードと新しいパスワードを入力します。

新しいパスワードの要件は、次のとおりです。

- 6 文字以上
- 最低 1 つの英字
- 最低 1 つの数字
- * およびスペースとは異なる、少なくとも 1 つの特殊文字

初回操作の詳細については、「初回操作の概要」を参照してください。