Deploying in VMware ESX

This chapter contains the following sections:

- Prerequisites and Guidelines, on page 1
- Deploying Cisco ACI Multi-Site Orchestrator Using Python, on page 3
- Deploying Orchestrator in vCenter, on page 9
- Deploying Orchestrator in ESX Directly, on page 12

Prerequisites and Guidelines

For all new deployments, we recommend using Cisco Application Service Engine as described in Deploying in Cisco Application Services Engine instead. However, if you still want to deploy the Orchestrator cluster in VMware ESX VMs directly, you can follow the guidelines and procedures in this chapter.

This chapter covers deployment of a 3-node Multi-Site Orchestrator cluster. If you want to set up a single-node Orchestrator (for example, for testing purposes), follow the instruction in the Installing Single Node Orchestrator chapter instead.

Deployment Method

When deploying in ESX VMs, you can choose one of the following 2 approaches:

- Use Cisco-provided Python scripts to deploy the entire Multi-Site Orchestrator cluster. The scripts allow you to execute the deployment and later upgrades remotely, for example from your laptop, as long as you have access to the vCenter where the Orchestrator VMs are to be deployed.

  This is the preferred approach when deploying an Orchestrator cluster in ESX VMs as it automates a number of manual steps and allows remote execution of Cisco ACI Multi-Site Orchestrator installation and subsequent software upgrades.

- Using an OVA image to deploy each Orchestrator VM individually. In this case you can also choose to deploy the image either using the vCenter or directly on the ESX server.

Docker Subnet Considerations

The Multi-Site Orchestrator application services run in Docker containers. When deployed, Docker uses a number of internal networks for its own application services (`bridge`, `docker_gwbridge`) as well as the Orchestrator services (`msc_msc`).

Prior to Release 2.2(2), an internal `10.0.0.0/24` network was used for the Docker swarm application services by default and could not be changed during the Multi-Site Orchestrator installation. This meant no other services in your fabric could reside on the same network.

Starting with Release 2.2(2), you can configure custom networks for the Docker services during Orchestrator deployment. Two additional parameters are now available in the Python configuration file or the OVA template:

Note When configuring these networks, ensure that they are unique and do not overlap with any existing networks in the environment.

- **Application overlay**: The default address pool to be used for Docker internal bridge networks.

  Application overlay must be a `/16` network. Docker then splits this network into two `/24` subnets used for the internal `bridge` and `docker_gwbridge` networks.

  For example, if you set the application overlay pool to `192.168.0.0/16`, Docker will use `192.168.0.0/24` for the `bridge` network and `192.168.1.0/24` for the `docker_gwbridge` network.

- **Service overlay**: The default Docker overlay network IP.

  Service overlay must be a `/24` network and is used for the `msc_msc` Orchestrator Docker service network.

Network Time Protocol (NTP)

Multi-Site Orchestrator uses NTP for clock synchronization, so you must have an NTP server configured in your environment. You provide NTP server information as part of the Orchestrator installation procedure.

Note VMware Tools provides an option to synchronize VMs' time with the host, however you should use only one type of periodic time synchronization in your VMs. Because you will enable NTP during Multi-Site Orchestrator deployment, ensure that VMware Tools periodic time synchronization is disabled for the Orchestrator VMs.

VMware vSphere Requirements

The following table summarizes the VMware vSphere requirements for Multi-Site Orchestrator:

- You must not enable vMotion for Multi-Site Orchestrator VMs.

  vMotion is not supported with docker swarm, which is used by the Multi-Site Orchestrator.

- You must ensure that the following vCPUs, memory, and disk space requirements are reserved for each VM and are not part of a shared resource pool:

Table 1: VMware vSphere Requirements

Orchestrator Version	Requirements
Release 2.2(4) or later	• ESXi 6.0 or later • 6 vCPUs (8 vCPUs recommended) • 48 GB of RAM • 64 GB disk • 10 GHz CPU reservation CPU cycle reservation is automatically applied when first deploying the Orchestrator VMs.
Release 2.2(1)-2.2(3)	• ESXi 6.0 or later • 6 vCPUs (8 vCPUs recommended) • 24 GB of RAM • 64 GB disk • 10 GHz CPU reservation CPU cycle reservation is automatically applied when first deploying the Orchestrator VMs.

Deploying Cisco ACI Multi-Site Orchestrator Using Python

The following sections describe how to prepare for and deploy Cisco ACI Multi-Site Orchestrator using Python.

Setting Up Python Environment

This section describes how to set up the Python environment for deploying Cisco ACI Multi-Site Orchestrator using Python. You must set up the Python environment on the laptop or server from which you will run the installation scripts.

Note If you have already set up your python environment, for example for another Multi-Site deployment or upgrade, you can skip this section.

Before you begin

you will need:

- A laptop or server from which you will run the scripts.

  You must not use any of the Multi-Site Orchestrator nodes for this purpose.

- Python 3.4 or later already installed on the system from which you will run the scripts.

Step 1 Download the **ACI Multi-Site Tools** image from Cisco ACI Multi-Site Software Download link.

a) Browse to the Software Download link:

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

b) Click **ACI Multi-Site Software**.

c) Choose the Cisco ACI Multi-Site Orchestrator release version.

d) Download the *ACI Multi-Site Tools Image* file (`tools-msc-<version>.tar.gz`).

Step 2 Extract the files.

```
# tar -xvzf tools-msc-<version>.tar.gz
```

Step 3 Change to the extracted directory.

```
# cd tools-msc-<version>
```

Step 4 Verify that you are running a correct version of Python.

Depending on your operating system and the versions of Python you have installed previously, the Python 3.x executable may be associated with `python` or `python3` command.

You can use one of the following examples to confirm which executable corresponds to Python 3.4 or later.

- If you have installed Python 2.x first, the `python` command will likely be associated with that version:

  ```
  # python -V
  Python 2.7.18
  ```

  In this case, you may need to use `python3` command instead:

  ```
  # python3 -V
  Python 3.4.5
  ```

- If you have installed only Python 3.x, you can use the `python` command as well:

  ```
  # python -V
  Python 3.7.7
  ```

Note The following steps use `python` for all commands. If your Python 3.x executable is associated with `python3`, use that command instead.

Step 5 If you plan to use a proxy to access the Internet, make sure to configure the proxy as follows:

```
# export http_proxy=<proxy-ip-address>:<proxy-port>
# export https_proxy=<proxy-ip-address>:<proxy-port>
```

Step 6 Install or update the Python package manager.

```
# python -m ensurepip
```

If the package is already installed, update it to the latest version:

```
# python -m ensurepip --upgrade
```

Step 7 (Optional) Set up Python virtual environment.

We recommend using `virutalenv` to install the packages, so they do not impact the existing packages in the system. The following steps provide a brief overview of how to set up `virutalenv`. For additional information on how to use `virtualenv`, see Installing packages using pip and virtualenv.

a) Install `virtualenv`.

   ```
   # python -m pip install --user virtualenv
   ```

b) Change into the directory where you want the virtual environment files to be created.

c) Create a virtual environment.

   In the following command, provide a name for the virtual environment, for example *mso-deployments*.

   ```
   # python -m venv <env-name>
   ```

d) Activate the virtual environment.

   You need to activate the virtual environment you created before installing the packages required for Orchestrator deployment or upgrade in the next step.

   For Windows:

   ```
   # .\<env-name>\Scripts\activate.bat
   ```

   For Linux:

   ```
   # source ./<env-name>/bin/activate
   ```

Step 8 Install the required packages.

The required packages are listed in the `requirements.txt` file.

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

Note The Python installation must complete successfully. If you encounter any errors, you must address them before proceeding to the next section or the Cisco ACI Multi-Site Orchestrator Python scripts will not work.

Step 9 If you used virtual Python environment, deactivate it now.

```
# deactivate
```

Sample Deployment Configuration File

When you deploy Multi-Site Orchestrator using Python, several required configuration details are specified in a YAML configuration file. This section provides a sample `msc_cfg.yml` file.

In the following sample configuration file all the VMs are created under the same host. The “host” parameter in the configuration file can be given as a node-level parameter instead if you want to create the Multi-Site VMs in different hosts.

```
# vCenter parameters
vcenter:
  name: 192.168.142.59
  user: administrator@vsphere.local

  # Host under which the Orchestrator VMs will be created
  host: 192.64.142.55

  # Path to the Orchestrator OVA file
  msc_ova_file: ../images/msc-2.1.1h.ova
```

```
  # (Optional) If not provided, default library name 'msc-content-lib' will be used
  #library: content-library-name

  # Library datastore name
  library_datastore: datastore1

  # Host datastore name
  host_datastore: datastore1

  # Prefix for Orchestrator VM names, full VM names will be '<vm_name_prefix>-node1',
  # '<vm_name_prefix>-node2', and '<vm_name_prefix>-node3'
  vm_name_prefix: msc

  # 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"

  # Time zone of the node, must be one of the values listed by 'timedatectl list-timezones'
 command
  time_zone: America/Los_Angeles

  # NTP (Network Time Protocol) servers, multiple servers can be listed separated by commas

  ntp_servers: ntp.company.com

  # Application Overlay IP for docker bridge type networks
  # Docker's bridge and docker_gwbridge networks are assigned addresses from this pool
  application_overlay: 192.168.0.0/16

  # Service Overlay IP for docker overlay type networks
  # Docker's msc_msc overlay network created at the time of deployment are assigned this
network address
  service_overlay: 2.1.1.0/24

# Node specific parameters over-ride the vCenter and common parameters
node1:
  # To use static IP, specify a valid IP address for the "ip" attribute
  # To obtain IP via DHCP, leave the "ip" field blank
  ip: 192.64.136.204

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

  # (Optional) If hostname is not specified, the VM name will be used
  hostname: mso-node1

node2:
  # To use static IP, specify a valid IP address for the "ip" attribute
  # To obtain IP via DHCP, leave the "ip" field blank
```

```
    ip:

    # (Optional) If hostname is not specified, the VM name will be used
    hostname: mso-node2

  node3:
    # To use static IP, specify a valid IP address for the "ip" attribute
    # To obtain IP via DHCP, leave the "ip" field blank
    ip:

    # (Optional) If hostname is not specified, the VM name will be used
    hostname: mso-node3
```

Deploying Multi-Site Orchestrator Using Python

This section describes how to deploy Cisco ACI Multi-Site Orchestrator using Python.

Before you begin

- Ensure that you meet the hardware requirements and compatibility that is listed in the *Cisco ACI Multi-Site Hardware Requirements Guide*.
- Ensure that you meet the requirements and guidelines described in Prerequisites and Guidelines, on page 1.
- Ensure that the NTP server is configured and reachable from the Orchestrator VMs and that VMware Tools periodic time synchronization is disabled.
- Ensure that the vCenter is reachable from the laptop or server where you will extract the tools and run the installation scripts.
- Ensure that your Python environment is set up as described in Setting Up Python Environment, on page 3.

Step 1 Download the Cisco ACI Multi-Site Orchestrator image and tools.

a) Browse to the Software Download link:

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

b) Click **ACI Multi-Site Software**.

c) Choose the Cisco ACI Multi-Site Orchestrator release version.

d) Download the *ACI Multi-Site Image* file (`msc-<version>.tar.gz`) for the release.

e) Download the *ACI Multi-Site Tools Image* file (`tools-msc-<version>.tar.gz`) for the release.

Step 2 Extract the `tools-msc-<version>.tar.gz` file to the directory from which you want to run the install scripts.

```
# tar -xvzf tools-msc-<version>.tar.gz
```

Then change into the extracted directory:

```
# cd tools-msc-<version>
```

Step 3 Create a `msc_cfg.yml` configuration file for your install.

You can copy and rename the provided `msc_cfg_example.yml` file or you can create the file using the example provided in Sample Deployment Configuration File, on page 5.

Step 4 Edit the `msc_cfg.yml` configuration file and fill in all the parameters for your environment.

The parameters that must be filled in are in all caps, for example *<VCENTER_NAME>*. You will also need to update *<MSC_TGZ_FILE_PATH>* with the path to the `msc-<version>.tar.gz` image file you downloaded in Step 1.

For a complete list of available parameters, see the sample `msc_cfg.yml` file is provided in Sample Deployment Configuration File, on page 5.

Step 5 Execute the script to deploy the Orchestrator VMs and prepare them:

```
# python msc_vm_util.py -c msc_cfg.yml
```

Step 6 Enter vCenter, `node1`, `node2` and `node3` passwords when prompted.

The script creates three Multi-Site Orchestrator VMs and executes the initial deployment scripts. This process may take several minutes to complete. After successful execution, the Multi-Site Orchestrator cluster is ready for use.

It may take several minutes for the deployment to complete.

Step 7 Verify that the cluster was deployed successfully.

a) Log in to any one of the deployed Orchestrator nodes.

b) Verify that all nodes are up and running.

```
# docker node ls
ID                            HOSTNAME        STATUS      AVAILABILITY     [...]
y90ynithc3cejkeazcqlu1uqs *   node1           Ready       Active           [...]
jt67ag14ug2jgaw4r779882xp     node2           Ready       Active           [...]
hoae55eoute6l5zpqlnxsk8o8     node3           Ready       Active           [...]
```

Confirm the following:

- The `STATUS` field is `Ready` for all nodes.
- The `AVAILABILITY` field is `Active` for all node.
- The `MANAGER STATUS` field is `Leader` for one of the nodes and `Reachable` for the other two.

c) Verify that all replicas are fully up.

```
# docker service ls
ID                  NAME                       MODE            REPLICAS    [...]
p6tw9mflj06u        msc_auditservice           replicated      1/1         [...]
je7s2f7xme6v        msc_authyldapservice       replicated      1/1         [...]
dbd27y76eouq        msc_authytacacsservice     replicated      1/1         [...]
untetoygqn1q        msc_backupservice          global          3/3         [...]
n5eibyw67mbe        msc_cloudsecservice        replicated      1/1         [...]
8inekkof982x        msc_consistencyservice     replicated      1/1         [...]
0qeisrguy7co        msc_endpointservice        replicated      1/1         [...]
e8ji15eni1e0        msc_executionengine        replicated      1/1         [...]
s4gnm2vge0k6        msc_jobschedulerservice    replicated      1/1         [...]
av3bjvb9ukru        msc_kong                   global          3/3         [...]
rqie68m6vf9o        msc_kongdb                 replicated      1/1         [...]
51u1g7t6ic33        msc_mongodb1               replicated      1/1         [...]
vrl8xvvx6ky5        msc_mongodb2               replicated      1/1         [...]
0kwk9xw8gu8m        msc_mongodb3               replicated      1/1         [...]
qhejgjn6ctwy        msc_platformservice        global          3/3         [...]
l7co71lneegn        msc_schemaservice          global          3/3         [...]
1t37ew5m7dxi        msc_siteservice            global          3/3         [...]
tu37sw68a1gz        msc_syncengine             global          3/3         [...]
8dr0d7pq6j19        msc_ui                     global          3/3         [...]
swnrzrbcv60h        msc_userservice            global          3/3         [...]
```

d) Log in to the Cisco ACI Multi-Site Orchestrator GUI.

You can access the GUI using any of the 3 nodes' IP addresses.

The default log in is **admin** and the default password is **We1come2msc!**.

When you first log in, you will be prompted to change the password.

What to do next

For more information about Day-0 Operations, see Adding Tenants and Schemas.

Deploying Orchestrator in vCenter

This section describes how to deploy Cisco ACI Multi-Site Orchestrator using an OVA in vCenter.

Before you begin

- Ensure that you meet the hardware requirements and compatibility that is listed in the *Cisco ACI Multi-Site Hardware Requirements Guide*.
- Ensure that you meet the requirements and guidelines described in Prerequisites and Guidelines, on page 1.
- Ensure that the NTP server is configured and reachable from the Orchestrator VMs and that VMware Tools periodic time synchronization is disabled.

Step 1 Download the Cisco ACI Multi-Site Orchestrator Image.

a) Browse to the Software Download link:

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

b) Click **ACI Multi-Site Software**.

c) From the left sidebar, choose the Cisco ACI Multi-Site Orchestrator release version.

d) Download the *ACI Multi-Site Image* file (`msc-<version>.ova`) for the release.

Step 2 Deploy the OVA using the VMware vCenter.

Note The OVA cannot be deployed directly in ESX, it must be deployed using vCenter. If you want to deploy Cisco ACI Multi-Site Orchestrator directly in ESX, see the "*Deploying Multi-Site Orchestrator in ESX Directly* " sections in this chapter.for instructions on how to extract the OVA and install the Orchestrator without vCenter.

Step 3 Configure the OVA properties.

In the **Properties** dialog box, enter the appropriate information for each VM:

- In the **Enter password** field, enter the root password for the VM.
- In the **Confirm password** field, enter the password again.
- In the **Hostname** field, enter the hostnames for each Cisco ACI Multi-Site Orchestrator node. You can use any valid Linux hostname.
- In the **Management Address** (network address) field, enter the network address or leave the field blank to obtain it via DHCP.

Note The field is not validated prior to installation, providing an invalid value for this field will cause the deployment to fail.

- In the **Management Netmask** (network netmask) field, enter the netmask netmask or leave the field blank to obtain it via DHCP.
- In the **Management Gateway** (network gateway) field, enter the network gateway or leave the field blank to obtain it via DHCP.
- In the **Domain Name System Server** (DNS server) field, enter the DNS server or leave the field blank to obtain it via DHCP.
- In the **Time-zone string (Time-zone)** field, enter a valid time zone string.

  You can find the time zone string for your region in the IANA time zone database or using the `timedatectl list-timezones` Linux command. For example, *`America/Los_Angeles`* .
- In the **NTP-servers** field, enter Network Time Protocol servers separated by commas.
- In the **Application overlay** field, enter the default address pool to be used for Docker internal bridge networks.

  Application overlay must be a `/16` network. Docker then splits this network into two `/24` subnets used for the internal `bridge` and `docker_gwbridge` networks.

  For example, if you set the application overlay pool to `192.168.0.0/16`, Docker will use `192.168.0.0/24` for the `bridge` network and `192.168.1.0/24` for the `docker_gwbridge` network.

  You must ensure that the application overlay network is unique and does not overlap with any existing networks in the environment.

  Note The field is not validated prior to installation, providing an invalid value for this field will cause the deployment to fail.
- In the **Service overlay** field, enter the default Docker overlay network IP.

  Service overlay must be a `/24` network and is used for the `msc_msc` Orchestrator Docker service network.

  You must ensure that the service overlay network is unique and does not overlap with any existing networks in the environment.

  Note The field is not validated prior to installation, providing an invalid value for this field will cause the deployment to fail.
- Click **Next**.
- In the **Deployment settings** pane, check all the information you provided is correct.
- Click **Power on after deployment**.
- Click **Finish**.

In addition to the above parameters, a 10GHz CPU cycle reservation is automatically applied to each Orchestrator VM when deploying the OVA.

Step 4 Repeat the previous two steps to deploy two more VMs.

The three VMs you deploy will join to form the Orchestrator cluster.

Step 5 Ensure that the virtual machines are able to ping each other.

Step 6 Initialize node1.

a) Connect to node1 using SSH.

b) Change to the initialization scripts directory.

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

c) Run initialization script and note the generated secret.

```
# ./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
 \
<node1-ip-address>
```

You will use the above token and IP address in the following steps to join `node2` and `node3` into the cluster.

d) Note the management IP address of the first node.

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

You will use this IP address in the following steps to join node2 and node3 into the cluster.

Step 7 Join node2 to the cluster.

a) Connect to node2 using SSH.

b) Change to the `/opt/cisco/msc/builds/<build_number>/prodha` directory.

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

c) Execute the `msc_cfg_join.py` command using the IP address of the first node.

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

Step 8 Join node3 to the cluster.

a) Connect to node3 using SSH.

b) Change to the `/opt/cisco/msc/builds/<build_number>/prodha` directory.

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

c) Execute the `msc_cfg_join.py` command using the IP address of the first node.

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

Step 9 On any node, make sure the nodes are healthy.

```
# docker node ls
ID                            HOSTNAME        STATUS      AVAILABILITY    [...]
```

```
y90ynithc3cejkeazcqlu1uqs *   node1           Ready       Active          [...]
jt67ag14ug2jgaw4r779882xp     node2           Ready       Active          [...]
hoae55eoute615zpqlnxsk8o8     node3           Ready       Active          [...]
```

Confirm the following:

- The `STATUS` field is `Ready` for all nodes.
- The `AVAILABILITY` field is `Active` for all node.
- The `MANAGER STATUS` field is `Leader` for one of the nodes and `Reachable` for the other two.

Step 10 On any node, execute the `msc_deploy.py` command:

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

Step 11 On any node, make sure that all REPLICAS are up.

```
# docker service ls
ID                  NAME                         MODE            REPLICAS     [...]
p6tw9mflj06u        msc_auditservice             replicated      1/1          [...]
je7s2f7xme6v        msc_authyldapservice         replicated      1/1          [...]
dbd27y76eouq        msc_authytacacsservice       replicated      1/1          [...]
untetoygqn1q        msc_backupservice            global          3/3          [...]
n5eibyw67mbe        msc_cloudsecservice          replicated      1/1          [...]
8inekkof982x        msc_consistencyservice       replicated      1/1          [...]
0qeisrguy7co        msc_endpointservice          replicated      1/1          [...]
e8ji15eni1e0        msc_executionengine          replicated      1/1          [...]
s4gnm2vge0k6        msc_jobschedulerservice      replicated      1/1          [...]
av3bjvb9ukru        msc_kong                     global          3/3          [...]
rqie68m6vf9o        msc_kongdb                   replicated      1/1          [...]
51u1g7t6ic33        msc_mongodb1                 replicated      1/1          [...]
vrl8xvvx6ky5        msc_mongodb2                 replicated      1/1          [...]
0kwk9xw8gu8m        msc_mongodb3                 replicated      1/1          [...]
qhejgjn6ctwy        msc_platformservice          global          3/3          [...]
l7co71lneegn        msc_schemaservice            global          3/3          [...]
1t37ew5m7dxi        msc_siteservice              global          3/3          [...]
tu37sw68a1gz        msc_syncengine               global          3/3          [...]
8dr0d7pq6j19        msc_ui                       global          3/3          [...]
swnrzrbcv60h        msc_userservice              global          3/3          [...]
```

Step 12 Log in to the Cisco ACI Multi-Site Orchestrator GUI.

You can access the GUI using any of the 3 nodes' IP addresses.

The default log in is **admin** and the default password is **We1come2msc!**.

When you first log in, you will be prompted to change the password.

What to do next

For more information about Day-0 Operations, see Adding Tenants and Schemas.

Deploying Orchestrator in ESX Directly

This section describes how to deploy Cisco ACI Multi-Site Orchestrator directly in ESX without using vCenter.

Before you begin

- Ensure that you meet the hardware requirements and compatibility that is listed in the *Cisco ACI Multi-Site Hardware Requirements Guide*.
- Ensure that you meet the requirements and guidelines described in Prerequisites and Guidelines, on page 1.
- Ensure that the NTP server is configured and reachable from the Orchestrator VMs and that VMware Tools periodic time synchronization is disabled.

Step 1 Download the Cisco ACI Multi-Site Orchestrator Image.

a) Browse to the Software Download link:

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

b) Click **ACI Multi-Site Software**.

c) Choose the Cisco ACI Multi-Site Orchestrator release version.

d) Download the *ACI Multi-Site Image (ESX Only)* file (`esx-msc-<version>.ova`) for the release.

Step 2 Untar the OVA file into a temporary directory:

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

Step 3 Use the ESX vSphere client to deploy the OVF.

a) Log in to vSphere.

b) Navigate to **File** > **Deploy OVF Template** > **Browse** and choose the `esx-msc-<version>.ovf` file.

c) Complete rest of the menu options and deploy the VM.

d) Repeat the steps for 2 additional Orchestrator nodes.

Step 4 Configure networking on Node2 and Node3.

After you configure networking on all 3 nodes, you will designate one of the nodes as `Primary` for the Docker swarm and use it to joint all 3 nodes into a cluster. Before you can do that, you must first configure networking on the two secondary nodes.

a) Log in to one of the secondary nodes (for example, `Node2`) as the `root` user.

The default password is `cisco`.

b) Change the default password.

The first time you log in, you will be prompted to change the default `root` password.

c) Run the Orchestrator setup utility.

```
# mso-setup
```

d) When prompted if it is a primary node, enter **n**.

```
When first deploying, one node must be designated as primary.
You must configure the other two nodes before configuring the primary node.
      If this is NOT the primary node, simply choose 'no' to proceed.
```

```
        If this is the primary node and the other nodes are ready, answer 'yes' to deploy.

Is this the primary node [y/N]? n
```

e) Confirm whether or not you will use a DHCP server to assign IP addresses to the node.

   If you choose to use a DHCP server, you will not be prompted for specific IP configuration, otherwise you will enter it in the next step.

```
Is this system going to get it's network configuration from a DHCP server [y/N]? n
```

f) Provide the required information.

   The setup utility will prompt for the following information:

   - **Management address**, for example `10.195.223.200`

     If you chose to use a DHCP server, this field is skipped.

   - **Management netmask**, for example `255.255.255.0`

     If you chose to use a DHCP server, this field is skipped.

   - **Management gateway**, for example `10.195.223.1`

     If you chose to use a DHCP server, this field is skipped.

   - **DNS server**, for example `171.70.168.183`

     If you chose to use a DHCP server, this field is skipped.

   - **Hostname**, for example `mso-node2`

     You can use any valid Linux hostname.

   - **Time zone string**, for example `America/Los_Angeles`

     You can find the time zone string for your region in the IANA time zone database or using the `timedatectl list-timezones` Linux command.

   - **NTP servers**, for example `ntp.esl.cisco.com`

     You can provide multiple NTP servers separated by commas.

   - **Application overlay network**, for example `192.168.0.0/16`

     Application overlay must be a `/16` network. Docker then splits this network into two `/24` subnets used for the internal `bridge` and `docker_gwbridge` networks.

     For example, if you set the application overlay pool to `192.168.0.0/16`, Docker will use `192.168.0.0/24` for the `bridge` network and `192.168.1.0/24` for the `docker_gwbridge` network.

     You must ensure that the application overlay network is unique and does not overlap with any existing networks in the environment.

   - **Service overlay network**, for example `1.1.1.0/24`

     Service overlay must be a `/24` network and is used for the `msc_msc` Orchestrator Docker service network.

     You must ensure that the service overlay network is unique and does not overlap with any existing networks in the environment.

g) Verify the provided information.

After you finish entering the information, you will be prompted to verify it. Reply **y** to confirm or **n** to re-enter the information.

```
   == Verify network configuration ==

        Management address: 10.195.223.200
        Management netmask: 255.255.255.0
        Management gateway: 10.195.223.1
        DNS server: 171.70.168.183
        Hostname: msc-node2
        Time zone string: America/Los_Angeles
        NTP servers: ntp.esl.cisco.com
        Application overlay network: 192.168.0.0/16
        Service overlay network: 1.1.1.0/24

Confirm the settings and proceed [Y/n]? y
```

Step 5 Repeat the previous step for the other secondary node (Node3).

Step 6 Configure the primary node (Node1) and deploy the cluster.

a) Log in to the primary node (Node1) as the `root` user.

   The default password is `cisco`.

b) Change the default password.

   The first time you log in, you will be prompted to change the default `root` password.

c) Run the Orchestrator setup utility.

```
# mso-setup
```

d) When prompted if it is a primary node, enter **y**.

```
  When first deploying, one node must be designated as primary.
  You must configure the other two nodes before configuring the primary node.
        If this is NOT the primary node, simply choose 'no' to proceed.
        If this is the primary node and the other nodes are ready, answer 'yes' to deploy.

Is this the primary node [y/N]? y
```

e) Confirm that the other two nodes have been configured.

   If you have not configured the other two nodes, you can respond **n** and re-run the setup utility at a later time.

```
Are other two nodes network configured [y/N]? y
```

f) Provide the network configuration information like you did for the other two nodes.

g) After you verify and confirm the network settings, provide other two nodes' information.

   You will be prompted to enter the IP addresses and `root` passwords for the other 2 nodes.

```
Confirm the settings and proceed [Y/n]? y
== MSO Network configuration done for node1 ==
== MSO Setup begins ==

Node2 IP address: 10.195.223.200
Node2 root password:
Node3 IP address: 10.195.223.201
Node3 root password:
msc_setup: Start
```

   If for any reason the setup does not complete, you can re-run just the deployment part without the full network configuration using the following command:

```
# mso-setup --install-mso
```

Step 7 Wait for the cluster to be deployed.

After you confirm the settings on the primary node, the setup utility

Step 8 Log in to the Cisco ACI Multi-Site Orchestrator GUI.

You can access the GUI using any of the 3 nodes' IP addresses.

The default log in is **admin** and the default password is **We1come2msc!**.

When you first log in, you will be prompted to change the password.