Deploying Hyper-Converged Ultra M Models Using UAS

This chapter provides information on the following topics:

Virtual Infrastructure Manager Installation Automation

Introduction

Leveraging RedHat and OpenStack's TripleO project concepts, UAS supports the ability to automate the deployment of both the virtual infrastructure manager (VIM, the Triple O Overcloud) and the VIM Orchestrator (the TripleO Undercloud).

Installing the VIM Orchestrator and the VIM involves deploying the following components as VMs on a RedHat Enterprise Linux (RHEL) server:

- AutoIT
- AutoDeploy
- OpenStack Platform Director (OSP-D)

VIM Orchestrator and VIM settings are maintained in configuration files which are used by AutoDeploy.

AutoDeploy processes the VIM Orchestrator configuration and works with AutoIT to automate the deployment of a VM running OSP-D which serves as the Undercloud. Once this operation is successful, AutoDeploy processes the VIM configuration and works with AutoIT to deploy the OpenStack Overcloud.

Notes:

- This functionality is supported only with Ultra M deployments based on OSP 10 and that leverage the Hyper-Converged architecture.
- Refer to Pre-Virtual Infrastructure Manager Installation Verification, on page 3 for pre-requisites pertaining to this feature.

VIM Installation Automation Overview

Figure 1: NFVI Deployment Automation Workflow, on page 2 provides an overview of the deployment automation process. Details are provided in Table 1: Virtual Infrastructure Manager Installation Automation Workflow Descriptions, on page 2. This information assumes that all prerequisite hardware has been installed, cabled, and configured.

Important The workflow information in this section assumes a new deployment scenario. If you are using this feature in relation with an upgrade process, please contact your support representative for complete details.

Figure 1: NFVI Deployment Automation Workflow

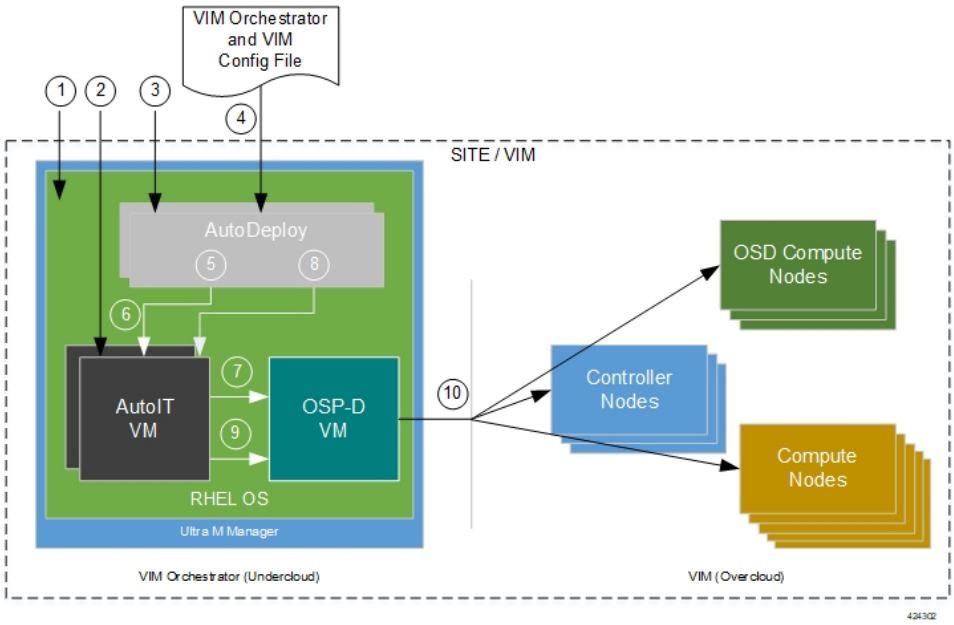


Table 1: Virtual Infrastructure Manager Installation Automation Workflow Descriptions

Callout	Description
1	Install RedHat Enterprise Linux (RHEL) operating system on bare metal hardware (Ultra M Manager Node).
2	Deploy the AutoIT VMs.
3	Deploy the AutoDeploy VMs.
4	Prepare the file containing the VIM Orchestrator and VIM. This file is used by AutoDeploy to initiate the OSP-D VM deployment process and to bring up the VIM. This file includes all the configuration information required to deploy OSP-D VM and VIM including configurations for constructs such as secure tokens, package images, NFVI point-of-presence descriptors (nfvi-popd), the VIM Orchestrator descriptor (vim-orchd), and VIM role and node information. Refer to Sample VIM Orchestrator and VIM Configuration File for more information.

Callout	Description
5	On the AutoDeploy VM, load, commit, and then activate the configuration file prepared in the previous step.
6	AutoDeploy passes data from the activated configuration to AutoIT requesting that it deploy the OSP-D VM for the Undercloud. Refer to Activate the VIM Orchestrator and VIM Deployment, on page 20 for more information.
7	AutoIT deploys the OSP-D VM which serves as the Undercloud.
8	AutoDeploy passes VIM data from the activated configuration to AutoIT for delivery to the OSP-D VM responsible for installing the VIM.
9	AutoIT initiates the VIM installation by passing parameters received from AutoDeploy to the OSP-D VM.
10	The OSP-D VM installs the VIM per the configuration requirements.

Once all the VIM servers have been successfully deployed, the process of deploying the VNF can begin as described in VNF Deployment Automation, on page 22.

Pre-Virtual Infrastructure Manager Installation Verification

Prior to installing the virtual infrastructure manager (VIM) and the VIM Orchestrator, please ensure that the following is true:

- Ensure that all required hardware is installed, powered on, cabled and configured according to the information and instructions in the *Ultra M Solutions Guide*. Refer to the following sections in that document:
  - *Hardware Specifications*
  - *Install and Cable the Hardware*
  - *Configure the Switches*
  - *Prepare the UCS C-Series Hardware*
- Ensure that all required software is available and that you have access to the Cisco-provided USP ISO image. See the *Software Specifications* section of the *Ultra M Solutions Guide* for more details.
- Ensure that the following repos are always enabled for Satellite Server and CDN Server:
  - rhel-7-server-rpms
  - rhel-7-server-rh-common-rpms
  - rhel-7-server-extras-rpms
  - rhel-ha-for-rhel-7-server-rpms
  - rhel-7-server-optional-rpms
  - rhel-7-server-rhscon-2-installer-rpms
  - rhel-7-server-openstack-10-rpms

- rhel-7-server-rhceph-2-mon-rpms
- rhel-7-server-rhceph-2-osd-rpms
- rhel-7-server-rhceph-2-tools-rpms

Install the VIM Orchestrator

The initial part of the Virtual Infrastructure Manager installation automation process is to install the VIM Orchestrator. You cannot install the VIM until after the VIM Orchestration installation is successful.

Important Before proceeding, ensure that all of the items in Pre-Virtual Infrastructure Manager Installation Verification, on page 3 have been verified.

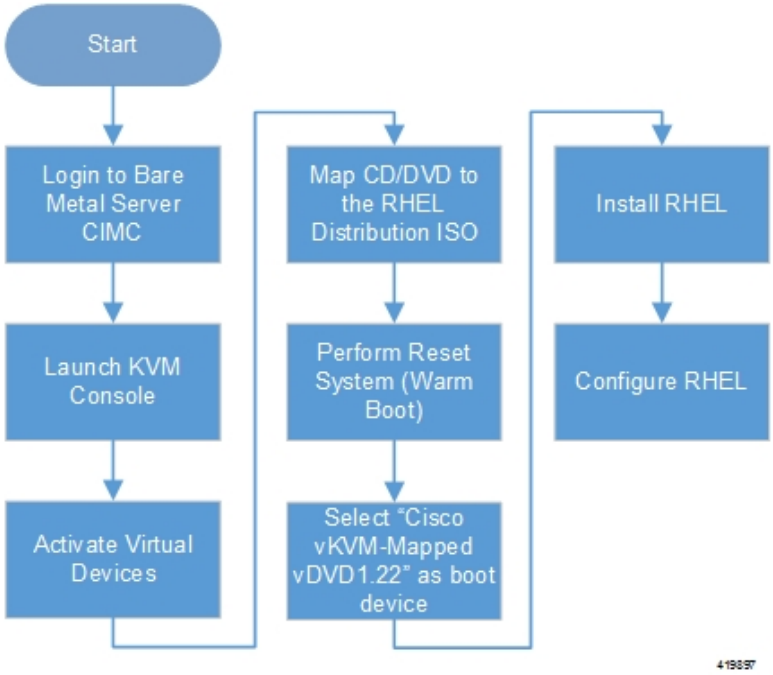
To install the VIM Orchestrator:

1. Install and Configure RHEL, on page 4.
2. Onboard the USP ISO, on page 10.
3. Extract the UAS Bundle, on page 12.
4. Deploy AutoIT, on page 13.
5. Deploy AutoDeploy, on page 16.
6. Prepare the VIM Orchestrator and VIM Configuration File, on page 20 based on your deployment requirements.
7. Activate the VIM Orchestrator and VIM Deployment, on page 20.

Install and Configure RHEL

As described in VIM Installation Automation Overview, on page 2, the VIM Orchestrator (OSP-D) is deployed as a VM on top of RHEL. Figure 2: Installation Process for RHEL Bare Metal Server, on page 5 illustrates the process for installing RHEL.

Figure 2: Installation Process for RHEL Bare Metal Server



General RHEL installation information and procedures are located in the product documentation:

- https://access.redhat.com/documentation/en/red-hat-enterprise-linux/

Prior to installing RHEL, refer to Table 2: Red Hat Installation Settings, on page 5 for settings required for the VIM Orchestrator installation in Ultra M.

Note Table 2: Red Hat Installation Settings, on page 5 assumes that you are using the product's graphical user interface (GUI) for Red Hat installation.

Table 2: Red Hat Installation Settings

Parameters and Settings	Description
Installation Summary > Language Support	
English > English (United States)	Sets the language to English and the region to United States.
Installation Summary > Software Selection	
Base Environment = Virtualization Host Add-Ons for Selected Environment = Virtualization Platform	
Installation Summary > Network & Host Name	

Parameters and Settings	Description
Host name	Configure the desired host name.
Installation Summary > Network & Host Name > Ethernet (eno2) > Configure > IPv4 Setting	
IP Address Netmask Gateway DNS Server Search Domain	Configure and save settings for the network interface by which the server can be accessed externally.
Installation Summary > Installation Destination > CiscoUCSC-MRAID12G (sda) > I will configure partitioning > Click here to create them automatically	
Select all partitions, then click "-" / = 100GB /var = 500GB /swap = 100GB /home = remaining space /boot = 1GB	Removes any previously configured partitions and creates partitions with the required sizes. **Note** You must use LVM-based partitioning.
Installation Summary > KDUMP	
kdump = disabled	It is recommended that kdump be disabled.
Installation Summary > Begin Installation > User Settings	
Root Password	Configure and confirm the root user password.
Create user "nfvi"	Creates a new user account. This account is used during the VIM Orchestration installation to log onto the Ultra M Manager Node. **Note** Ensure that a strong password is used. It must be a minimum of 8 alpha and/or numeric characters and must contain at least 1 uppercase letter, 1 lowercase letter, 1 number, and 1 special character (e.g. @, #, $, etc.).

To install and configure RHEL:

1. Follow the CIMC processes on the bare metal server as identified in Figure 2: Installation Process for RHEL Bare Metal Server, on page 5.
2. Select the option to install Red Hat Enterprise Linux to begin the installation.
3. Configure the settings identified in Table 2: Red Hat Installation Settings, on page 5.
4. Begin the installation and configure the User Setting identified in Table 2: Red Hat Installation Settings, on page 5.
5. Click **Reboot** once the installation is complete.
6. Log in to RedHat as the **nfvi** user.
7. Set password-less sudo access for **nfvi**.

   ```
   echo "nfvi ALL=(root) NOPASSWD:ALL" | tee -a /etc/sudoers.d/nfvi
   chmod 0440 /etc/sudoers.d/nfvi
   ```

8. Configure the network interfaces and network bridges.

Important If any of the network interface or bridge configuration files do not exist, create the related configuration files. Example configuration files are provided in Example RedHat Network Interface and Bridge Configuration Files.

a. Configure the eno2 interface by appending the following parameters to the */etc/sysconfig/network-scripts/ifcfg-eno2* file.

```
<--SNIP-->
DEVICE=eno2
ONBOOT=yes
BRIDGE=br-ex
NM_CONTROLLED=no
NETMASK=<netmask>
GATEWAY=<gateway_address>
```

b. Configure the eno1 interface by appending the following parameters to the */etc/sysconfig/network-scripts/ifcfg-eno1* file.

```
<--SNIP-->
DEVICE=eno1
ONBOOT=yes
BRIDGE=br-ctlplane
NM_CONTROLLED=no
```

c. Configure the br-ex network bridge by adding the following parameters to the */etc/sysconfig/network-scripts/ifcfg-br-ex* file.

```
<--SNIP-->
DEVICE=br-ex
DEFROUTE=yes
TYPE=Bridge
ONBOOT=yes
BOOTPROTO=static
NM_CONTROLLED=no
DELAY=0
IPADDR=<external_ip_address>
NETMASK=<netmask>
GATEWAY=<gateway_address>
PREFIX="24"
DNS1="<DNS_server_address>"
DOMAIN="<domain_name>"
IPV4_FAILURE_FATAL="yes"
```

d. Configure the br-ctlplane bridge by adding the following parameters to the */etc/sysconfig/network-scripts/ifcfg-br-ctlplane* file.

```
<--SNIP-->
DEFROUTE=yes
TYPE=Bridge
ONBOOT=yes
BOOTPROTO=static
```

```
NM_CONTROLLED=no
DELAY=0
DEVICE=br-ctlplane
```

Caution Once configured, it is recommended that you do not make any changes to the network interface or bridge configuration. Doing so will require that you redeploy AutoIT and AutoDeploy.

9. Create and prepare the directories required for installing the UAS components.

   ```
   sudo mkdir -p /var/cisco/isos
   sudo mkdir -p /var/cisco/disks
   sudo chmod 777 -R /var/cisco
   ```

10. Reboot the bare metal server.

    ```
    sudo reboot
    ```

11. Login as a root user upon reboot.

Important If the server is not accessible via the configured IP address, login into the server's KVM console and troubleshoot the configuration.

12. Validate the network configuration.

    ```
    ifconfig | more
    ```

    Example output:

```
br-ctlplane: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::22c:c8ff:fed9:f176  prefixlen 64  scopeid 0x20<link>
        ether 00:2c:c8:d9:f1:76  txqueuelen 1000  (Ethernet)
        RX packets 52  bytes 7044 (6.8 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 8  bytes 648 (648.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

br-ex: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.25.22.59  netmask 255.255.255.0  broadcast 172.25.22.255
        inet6 fe80::22c:c8ff:fed9:f177  prefixlen 64  scopeid 0x20<link>
        ether 00:2c:c8:d9:f1:77  txqueuelen 1000  (Ethernet)
        RX packets 1394  bytes 122906 (120.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 717  bytes 71762 (70.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eno1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::22c:c8ff:fed9:f176  prefixlen 64  scopeid 0x20<link>
        ether 00:2c:c8:d9:f1:76  txqueuelen 1000  (Ethernet)
        RX packets 57  bytes 8072 (7.8 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 16  bytes 1296 (1.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device memory 0xc7000000-c70fffff

eno2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
```

```
        inet6 fe80::22c:c8ff:fed9:f177  prefixlen 64  scopeid 0x20<link>
        ether 00:2c:c8:d9:f1:77  txqueuelen 1000  (Ethernet)
        RX packets 1497  bytes 148860 (145.3 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 726  bytes 72476 (70.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device memory 0xc6f00000-c6ffffff

enp6s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        ether 00:2c:c8:68:3b:ec  txqueuelen 1000  (Ethernet)
        RX packets 1  bytes 68 (68.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp7s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        ether 00:2c:c8:68:3b:ed  txqueuelen 1000  (Ethernet)
        RX packets 1  bytes 68 (68.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1  (Local Loopback)
        RX packets 84  bytes 6946 (6.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 84  bytes 6946 (6.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

virbr0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 192.168.122.1  netmask 255.255.255.0  broadcast 192.168.122.255
[root@rhel-baremetal nfvi]# brctl show
bridge name bridge id  STP enabled interfaces
br-ctlplane  8000.002cc8d9f176 no  eno1
br-ex  8000.002cc8d9f177 no  eno2
virbr0  8000.5254003d7549 yes  virbr0-nic
```

13. Perform the RHEL subscription-manager registration.

From Content Delivery Network (CDN) servers:

sudo subscription-manager config --server.proxy_hostname=*<proxy_url>* **--server.proxy_port=80**

subscription-manager register --username *<username>* **--password** *<password>*

subscription-manager attach -auto

sudo subscription-manager status

From Satellite Servers:

rpm -Uvh http://*<satellite_server_domain>***/pub/katello-ca-consumer-latest.noarch.rpm**

subscription-manager register --org="*<organization>***" --activationkey="***<activation_key>***"**

Example output:

```
+-------------------------------------------+
   System Status Details
```

```
+----------------------------------------+
Overall Status: Current
```

14. Install the virtualization packages.

    yum install virt-install -y

    Example output:

```
Loaded plugins: langpacks, product-id, search-disabled-repos, subscription-manager
rhel-7-server-rpms                                                    | 3.5 kB
 00:00:00
(1/3): rhel-7-server-rpms/7Server/x86_64/group                        | 709 kB
 00:00:01
(2/3): rhel-7-server-rpms/7Server/x86_64/updateinfo                   | 2.3 MB
 00:00:02
(3/3): rhel-7-server-rpms/7Server/x86_64/primary_db                   |  42 MB
 00:00:16
Resolving Dependencies
Loaded plugins: langpacks, product-id, search-disabled-repos, subscription-manager
rhel-7-server-rpms | 3.5 kB 00:00:00
(1/3): rhel-7-server-rpms/7Server/x86_64/group | 709 kB 00:00:01
(2/3): rhel-7-server-rpms/7Server/x86_64/updateinfo | 2.3 MB 00:00:02
(3/3): rhel-7-server-rpms/7Server/x86_64/primary_db | 42 MB 00:00:16
Resolving Dependencies
```

    yum install virt-viewer -y

```
Loaded plugins: langpacks, product-id, search-disabled-repos, subscription-manager
Resolving Dependencies
--> Running transaction check
---> Package virt-viewer.x86_64 0:5.0-7.el7 will be installed
```

15. Install the Python bindings to the OpenStack Compute API.

    yum install python-novaclient -y

16. Install the OpenStack networking API client.

    yum install python-neutronclient -y

17. Install the NETCONF client.

    yum install python-ncclient -y

18. Install the python library for XML and HTML processing.

    yum install python-lxml -y

19. Proceed to Onboard the USP ISO, on page 10.

Onboard the USP ISO

The files required to deploy the USP components are distributed as RPMs (called “bundles”) in a single ISO package. They are maintained using YUM on the Ultra M Manager Node. The following bundles are part of the ISO:

USP Bundle Name	Description
usp-em-bundle	The Element Manager (EM) Bundle RPM containing images and metadata for the Ultra Element Manager (UEM) module.

USP Bundle Name	Description
usp-uas-bundle	The Ultra Automation Services Bundle RPM containing AutoIT, AutoDeploy, AutoVNF, Ultra Web Services (UWS), and other automation packages.
usp-ugp-bundle	The Ultra Gateway Platform (UGP) Bundle RPM containing images for Ultra Packet core (VPC-DI). This bundle contains non-trusted images.
usp-vnfm-bundle	The VNFM Bundle RPM containing an image and a boot-up script for ESC (Elastic Service Controller).
usp-yang-bundle	The Yang Bundle RPM containing YANG data models including the VNFD and VNFR.
usp-auto-it-bundle	The bundle containing the AutoIT packages required to deploy the UAS.

Important Release 6.4 will not be backward compatible with previous releases, i.e., you cannot deploy a 6.4 ISO from an AutoDeploy/AutoIT/AutoVNF running a pre-6.4 release, and vice-versa.

In addition to the bundles, the ISO bundle also includes scripts used to deploy the bundles including UAS.

Important This procedure is not necessary if you are deploying a VNF on a Hyper-Converged Ultra M mode and have already deployed the VIM Orchestrator and the VIM using the information and instructions in Virtual Infrastructure Manager Installation Automation, on page 1.

Important Before attempting to deploy the Ultra M Manager Node, ensure that the USP Installation Prerequisites have been met.

To onboard the ISO package:

1. Log on to the Ultra M Manager Node.
2. Download the USP ISO bundle and related files pertaining to the release.
3. Create a mount point on the Ultra M Manager Node and mount the ISO package:

   ```
   mkdir /var/usp-iso
   ```

4. Mount the USP ISO.

   ```
   sudo mount -t iso9660 -o loop <ISO_download_directory>/<ISO_package_name>
   /var/usp-iso
   ```

   Example: The following command mounts the ISO bundle called *usp-5_5_0-1255.iso* located in a directory called *5_5_0-1283 to /var/usp-iso*:

   ```
   sudo mount -t iso9660 -o loop 5_5_0-1064/usp-5_5_0-1064.iso /var/usp-iso
   mount: /dev/loop1 is write-protected, mounting read-only
   ```

5. Verify the mount configuration.

df -h

Example output:

```
Filesystem Size Used Avail Use% Mounted on
/dev/sda2 187G 178G 316M 100% /
devtmpfs 63G 0 63G 0% /dev
tmpfs 63G 4.0K 63G 1% /dev/shm
tmpfs 63G 1.4M 63G 1% /run
tmpfs 63G 0 63G 0% /sys/fs/cgroup
/dev/sda1 477M 112M 336M 25% /boot
tmpfs 13G 0 13G 0% /run/user/0
/dev/loop1 4.2G 4.2G 0 100% /var/usp-iso >>>>>
```

6. Proceed to Extract the UAS Bundle, on page 12.

Extract the UAS Bundle

Once the USP ISO has been mounted, the UAS bundle must be extracted from the ISO in order to prepare the configuration files required for deployment.

Important These instructions assume you are already logged on to the server on which AutoIT, AutoDeploy, and VIM-Orchestrator VMs are to be installed and that the USP ISO has been mounted.

To extract the UAS bundle:

1. Navigate to the tools directory within the ISO mount.

   cd /var/usp-iso/tools/

2. Launch the *usp-uas-installer.sh* script.

   sudo ./usp-uas-installer.sh

   The script extracts the files that comprise the UAS bundle to */opt/cisco/usp/uas-installer*.

3. Verify that files have been extracted.

   Example output:

   ll /opt/cisco/usp/uas-installer

   ```
   total 20
   drwxr-xr-x 5 root root 4096 Aug 18 23:42 ./
   drwxr-xr-x 6 root root 4096 Aug 18 23:42 ../
   drwxr-xr-x 5 root root 4096 Aug 18 23:42 common/
   drwxr-xr-x 2 root root 4096 Aug 18 23:42 images/
   drwxr-xr-x 2 root root 4096 Aug 18 23:42 scripts/
   ```

   ll /opt/cisco/usp/uas-installer/images/

   ```
   total 711940
   drwxr-xr-x 2 root root      4096 Aug 18 23:42 ./
   drwxr-xr-x 5 root root      4096 Aug 18 23:42 ../
   -rw-r--r-- 1 root root 729010688 Aug 17 23:29 usp-uas-1.0.0-1074.qcow2
   ```

   ll /opt/cisco/usp/uas-installer/scripts/

   ```
   total 80
   -rwxr-xr-x. 1 root root   806 Aug 29 18:14 auto-deploy-booting.sh
   -rwxr-xr-x. 1 root root  5460 Aug 29 18:14 autoit-user.py
   -rwxr-xr-x. 1 root root   811 Aug 29 18:14 auto-it-vnf-staging.sh
   -rwxr-xr-x. 1 root root  4762 Aug 29 18:14 encrypt_account.sh
   ```

```
-rwxr-xr-x. 1 root root  3945 Aug 29 18:14 encrypt_credentials.sh
-rwxr-xr-x. 1 root root 14031 Aug 29 18:14 start-ultram-vm.py
-rwxr-xr-x. 1 root root 14605 Aug 29 18:14 boot_uas.py
-rwxr-xr-x. 1 root root  5384 Aug 29 18:14 uas-check.py
-rwxr-xr-x. 1 root root 11283 Aug 29 18:14 usp-tenant.py
```

4. Proceed to Deploy AutoIT, on page 13.

Deploy AutoIT

AutoIT deployment is facilitated through a script. The script relies on user inputs to perform pre-requisite configurations including whether or not to deploy with HA support and account encryptions. Additionally, the script removes existing AutoIT deployments that may already exist.

The following information is required to execute the script:

- **AutoIT VM Login Password for ID 'ubuntu':** The password for the default user account, which is named ubuntu.
- **AutoIT API Access password for 'admin':** The password for the ConfD administrator user, which is named admin.
- **AutoIT API Access password for 'oper':** The password for the ConfD operator user, which is named oper.
- **AutoIT API Access password for 'security-admin':** The password for the ConfD security administrator user, which is named security-admin.
- **Hostname:** The hostname assigned to the AutoIT VM.
- **Image (QCOW2):** The path and file name for the UAS qcow2 file. For example:

  /opt/cisco/usp/uas-installer/images/usp-uas-1.0.0-1074.qcow2

- **External Network HA VIP :** The VIP address to be assigned to AutoIT's external network interface.
- **External Network Details:**
  - **IP Address:** The IP address to be assigned to AutoIT VMs' external network interface. If AutoIT is deployed with HA support, you are prompted to enter separate external IP addresses for both the active and redundant VMs.
  - **Gateway:** The gateway assigned to AutoIT's external network interface.
  - **Netmask:** The mask to be assigned to AutoIT's external network interface.
- **Provisional Network HA VIP**: The VIP address to be assigned to AutoIT's provisional network interface.
- **Provisioning Network Details:**
  - **IP Address:** The IP address to be assigned to the provisioning network interface. Within Hyper-Converged Ultra M models, this interface is used by the Ultra M Health Monitoring function.

    If AutoIT is deployed with HA support, you are prompted to enter separate IP provisioning addresses for both the active and redundant VMs.
  - **Netmask:** The netmask to be assigned to the provisioning network interface.

Important

- All passwords must meet the requirements specified in Password Requirements and Login Security.
- You may be asked for some of the above pieces of information twice, once for each VM when AutoIT is deployed with HA support.

The script allocates the following resources to the AutoIT VM:

- 2 VCPUs
- 8 GB RAM
- 80 GB Root Disk

Important These instructions assume a bare-metal installation and that you are already logged on to the server on which AutoIT, AutoDeploy, and VIM-Orchestrator VMs are to be installed and on which the USP ISO has been mounted.

To deploy the AutoIT VM:

1. Navigate to the */opt/cisco/usp/uas-installer/scripts* directory:

   ```
   cd /opt/cisco/usp/uas-installer/scripts
   ```

2. Execute the *boot_uas.py* script with the desired options:

   ```
   ./boot_uas.py --kvm --autoit --ha
   ```

Important The above command deploys AutoIT with HA support which is recommended for use within Ultra M solutions. Remove the **--ha** if you do not wish to implement HA support for AutoIT.

There are a number of options that can be specified when deploying AutoIT. Refer to boot_uas.py Help for details. Some notes are below.

Note

- The above command deploys AutoIT with HA support. Remove the **--ha** if you do not wish to implement HA support for AutoIT.
- If you wish to configure syslog functionality for AutoIT, you must specify the IP address, TCP/UDP port, and severity level for one or more collection servers. The following command example configures two collection servers.

  ./boot_uas.py --kvm --autoit --ha --syslog-ip *192.168.2.1* **--port** *514* **--severity** *5* **--syslog-ip** *192.168.2.2* **--port** *514* **--severity** *5*

  You can set the severity level to one of the following values:

  - 0: emerg, panic
  - 1: alert
  - 2: crit
  - 3: err.error
  - 4: warning, warn
  - 5: notice
  - 6: info
  - 7: debug

3. Enter the information requested by the script for your deployment.

   The script displays progress information. For example:

```
2018-01-24 16:06:17,355 - '/home' disk capacity is 1807 GB Loaded plugins: langpacks,
product-id
2018-01-24 16:06:17,397 - Package 'virt-install' is present
2018-01-24 16:06:17,397 - Package 'libvirt' is present
2018-01-24 16:06:17,397 - Package 'virt-viewer' is present
2018-01-24 16:06:17,397 - Interface 'br-ex' is UP
2018-01-24 16:06:17,397 - Interface 'br-ctlplane' is UP
2018-01-24 16:06:17,398 - Removing old deployment 'AutoIT_instance_0', if it exists
2018-01-24 16:06:19,921 - Removing old deployment 'AutoIT_instance_1', if it exists
2018-01-24 16:06:19,946 - Using instance 'AutoIT_instance_0' at location
'/home/cisco/AutoIT/instance_0'
2018-01-24 16:06:19,946 - Staging configuration ISO
2018-01-24 16:06:19,951 - Completed configuration ISO
/home/cisco/AutoIT/instance_0/cfg.iso
2018-01-24 16:06:19,951 - Preparing root disk '/home/cisco/AutoIT/instance_0/uas.qcow2'
2018-01-24 16:06:20,378 - Resizing disk to '80GB'
2018-01-24 16:06:33,417 - Starting deployment 'AutoIT_instance_0'
2018-01-24 16:06:34,124 - Started deployment 'AutoIT_instance_0' successfully
2018-01-24 16:06:34,125 - Using instance 'AutoIT_instance_1' at location
'/home/cisco/AutoIT/instance_1'
2018-01-24 16:06:34,125 - Staging configuration ISO
2018-01-24 16:06:34,130 - Completed configuration ISO
/home/cisco/AutoIT/instance_1/cfg.iso
2018-01-24 16:06:34,130 - Preparing root disk '/home/cisco/AutoIT/instance_1/uas.qcow2'
2018-01-24 16:06:34,557 - Resizing disk to '80GB'
2018-01-24 16:06:42,629 - Starting deployment 'AutoIT_instance_1'
2018-01-24 16:06:43,360 - Started deployment 'AutoIT_instance_1' successfully
```

4. Verify that the AutoIT VM is running.

   virsh list -all

   Example command output:

```
 Id    Name                           State
----------------------------------------------------
 487   AutoIT_instance_0              running
 488   AutoIT_instance_1              running
```

5. Check the status of AutoIT.

   a. Log on to the master AutoIT VM.

      confd_cli -C -u admin

      Example command output:

```
Welcome to the ConfD CLI
admin connected from 127.0.0.1 using console on autoit1-0
```

   b. Enter the *admin* user password when prompted.

   c. View the status.

      show uas

      Example command output:

```
uas version               6.0.0
uas state                 active
uas external-connection-point 172.28.185.132
INSTANCE IP     STATE  ROLE
-------------------------------------
172.28.185.133  alive  CONFD-MASTER
172.28.185.134  alive  CONFD-SLAVE

NAME           LAST HEARTBEAT
-------------------------------------
AutoIT-MASTER  2018-01-24 21:24:30
USPCFMWorker   2018-01-24 21:24:30
USPCHBWorker   2018-01-24 21:24:30
USPCWorker     2018-01-24 21:24:30
```

6. Proceed to Deploy AutoDeploy, on page 16.

Deploy AutoDeploy

Important The information and instructions provided here are only applicable when AutoDeploy is used in the VIM Orchestrator installation process.

AutoDeploy deployment is facilitated through a script. The script relies on user inputs to perform pre-requisite configurations including whether or not to deploy with HA support and account encryptions. Additionally, the script removes existing AutoDeploy deployments that may already exist.

The following information is required to execute the script:

- **AutoDeploy VM Login Password for ID 'ubuntu'** The password for the default user account, which is named ubuntu.

- **AutoDeploy API Access password for 'admin':** The password for the ConfD administrator user, which is named admin.
- **AutoDeploy API Access password for 'oper':** The password for the ConfD operator user, which is named oper.
- **AutoDeploy API Access password for 'security-admin':** The password for the ConfD security administrator user, which is named security-admin.
- **Hostname:** The hostname assigned to the AutoDeploy VM.
- **Image (QCOW2):** The path and file name for the UAS qcow2 file. For example:

  /opt/cisco/usp/uas-installer/images/usp-uas-1.0.0-1074.qcow2
- **External Network HA VIP :** The VIP address to be assigned to AutoDeploy's external network interface.
- **External Network Details:**
  - **IP Address:** The IP address to be assigned to AutoDeploy VMs' external network interface. If AutoDeploy is deployed with HA support, you are prompted to enter separate external IP addresses for both the active and redundant VMs.
  - **Gateway:** The gateway assigned to AutoDeploy's external network interface.
  - **Netmask:** The mask to be assigned to AutoDeploy's external network interface.

Important

- All passwords must meet the requirements specified in Password Requirements and Login Security.
- You may be asked for some of the above pieces of information twice, once for each VM when AutoDeploy is deployed with HA support.

The script allocates the following resources to the AutoDeploy VM:

- 2 VCPUs
- 8 GB RAM
- 80 GB Root Disk

Important These instructions assume a bare-metal installation and that you are already logged on to the server on which AutoIT, AutoDeploy, and VIM-Orchestrator VMs are to be installed and on which the USP ISO has been mounted.

To deploy the AutoDeploy VM:

1. Navigate to the */opt/cisco/usp/uas-installer/scripts* directory:

   ```
   cd /opt/cisco/usp/uas-installer/scripts
   ```
2. Execute the *boot_uas.py* script:

   ```
   ./boot_uas.py --kvm --autodeploy --ha
   ```

Note The above command deploys AutoDeploy with HA support. Remove the **--ha** if you do not wish to implement HA support for AutoDeploy.

There are a number of options that can be specified when deploying AutoDeploy. Refer to boot_uas.py Help for details. Some notes are below.

Note

- The above command deploys AutoDeploy with HA support. Remove the **--ha** if you do not wish to implement HA support for AutoDeploy.
- If you wish to configure syslog functionality for AutoDeploy, you must specify the IP address, TCP/UDP port, and severity level for one or more collection servers. The following command example configures two collection servers.

  ./boot_uas.py --kvm --autodeploy --ha --syslog-ip *192.168.2.1* **--port** *514* **--severity** *5* **--syslog-ip** *192.168.2.2* **--port** *514* **--severity** *5*

  You can set the severity level to one of the following values:

  - 0: emerg, panic
  - 1: alert
  - 2: crit
  - 3: err.error
  - 4: warning, warn
  - 5: notice
  - 6: info
  - 7: debug

3. Enter the information requested by the script for your deployment.

   The script displays progress information. For example:

```
2018-01-24 16:28:05,095 - '/home' disk capacity is 1807 GB Loaded plugins: langpacks,
product-id
2018-01-24 16:28:05,134 - Package 'virt-install' is present
2018-01-24 16:28:05,135 - Package 'libvirt' is present
2018-01-24 16:28:05,135 - Package 'virt-viewer' is present
2018-01-24 16:28:05,135 - Interface 'br-ex' is UP
2018-01-24 16:28:05,135 - Interface 'br-ctlplane' is UP
2018-01-24 16:28:05,135 - Removing old deployment 'AutoDeploy_instance_0', if it exists
2018-01-24 16:28:06,980 - Removing old deployment 'AutoDeploy_instance_1', if it exists
2018-01-24 16:28:07,005 - Using instance 'AutoDeploy_instance_0' at location
'/home/cisco/AutoDeploy/instance_0'
2018-01-24 16:28:07,006 - Staging configuration ISO
2018-01-24 16:28:07,010 - Completed configuration ISO
/home/cisco/AutoDeploy/instance_0/cfg.iso
2018-01-24 16:28:07,010 - Preparing root disk
'/home/cisco/AutoDeploy/instance_0/uas.qcow2'
2018-01-24 16:28:07,450 - Resizing disk to '80GB'
```

```
2018-01-24 16:28:15,965 - Starting deployment 'AutoDeploy_instance_0'
2018-01-24 16:28:16,649 - Started deployment 'AutoDeploy_instance_0' successfully
2018-01-24 16:28:16,650 - Using instance 'AutoDeploy_instance_1' at location
'/home/cisco/AutoDeploy/instance_1'
2018-01-24 16:28:16,650 - Staging configuration ISO
2018-01-24 16:28:16,655 - Completed configuration ISO
/home/cisco/AutoDeploy/instance_1/cfg.iso
2018-01-24 16:28:16,655 - Preparing root disk
'/home/cisco/AutoDeploy/instance_1/uas.qcow2'
2018-01-24 16:28:17,106 - Resizing disk to '80GB'
2018-01-24 16:28:30,204 - Starting deployment 'AutoDeploy_instance_1'
2018-01-24 16:28:30,892 - Started deployment 'AutoDeploy_instance_1' successfully
```

4. Verify that the AutoDeploy VM is running.

   virsh list -all

```
 Id    Name                           State
----------------------------------------------------
 495   AutoDeploy_instance_0          running
 496   AutoDeploy_instance_1          running
```

Important It is recommended that you do not make any changes to the AutoIT network interface or bridge configuration. Doing so will require that you redeploy AutoDeploy.

5. Check the status of AutoDeploy.
   a. Log on to the master AutoDeploy VM.

      confd_cli -C -u admin

      Example command output:

```
Welcome to the ConfD CLI
admin connected from 127.0.0.1 using console on autodeploy-0
```

   b. Enter the *admin* user password when prompted.
   c. View the status.

      show uas

      Example command output:

```
uas version              6.0.0uas version              6.0.0
uas state                active
uas external-connection-point 172.28.185.132
INSTANCE IP     STATE  ROLE
-------------------------------------
172.28.185.133  alive  CONFD-MASTER
172.28.185.134  alive  CONFD-SLAVE

NAME               LAST HEARTBEAT
----------------------------------------
AutoDeploy-MASTER  2018-01-24 21:29:54
USPCFMWorker       2018-01-24 21:29:45
USPCHBWorker       2018-01-24 21:29:45
USPCWorker         2018-01-24 21:29:45
```

6. Choose the desired method by which to continue the deployment process:

- Use the ConfD CLI/APIs to continue the deployment process. To use this method, proceed to Prepare the VIM Orchestrator and VIM Configuration File, on page 20.

Important You will need access to both the OpenStack GUI and CLI to complete the configuration procedures.

Prepare the VIM Orchestrator and VIM Configuration File

As described in VIM Installation Automation Overview, on page 2, the VIM Orchestrator and VIM configuration file is used by AutoDeploy to activate the OSP-D VM and VIM deployment process.

This file includes all of the configuration information required to deploy OSP-D VM and VIM including configurations for constructs such as secure tokens, package images, NFVI point-of-presence descriptors (nfvi-popd), the VIM Orchestrator descriptor (vim-orchd), and VIM role and node information. Refer to Sample VIM Orchestrator and VIM Configuration File for more information. Additional information on the constructs and parameters used in this file are located in the *Cisco Ultra Services Platform NETCONF API Guide*.

You can also refer to RedHat user documentation for information on how to install the satellite server if your deployment requires:

https://access.redhat.com/documentation/en-US/Red_Hat_Network_Satellite/5.0/html/Installation_Guide/s1-intro-sat.html

Note These instructions assume you are already logged on to the AutoDeploy VM as the root user.

To prepare the VIM Orchestrator and VIM configuration file:

1. Create and edit your VIM Orchestrator and VIM configuration file according to your deployment requirements. Use the sample provided in Sample VIM Orchestrator and VIM Configuration File as a reference.
2. Save the VIM Orchestrator and VIM configuration file you have created to your home directory.
3. Proceed to Activate the VIM Orchestrator and VIM Deployment, on page 20.

Activate the VIM Orchestrator and VIM Deployment

Once you have completed preparing your VIM Orchestrator and VIM configuration file, you must load the configuration and activate the deployment in order to bring up the OSP-D VM and the VIM.

Important These instructions assume you are already logged on to the AutoDeploy VM as the root user and that your VIM Orchestrator and VIM configuration file has been prepared for your deployment as per the information and instructions in Prepare the VIM Orchestrator and VIM Configuration File, on page 20.

To activate the OSP-D VM and VIM deployment using AutoDeploy:

1. Login to the ConfD CLI as the *admin* user.

```
confd_cli -u admin -C
```

2. Enter the *admin* user password when prompted.
3. Enter the ConfD configuration mode.

   ```
   config
   ```

4. Load the VIM Orchestrator and VIM configuration file to provide the deployment artifacts to the VIM.

   ```
   load merge <your_config_file_name>.cfg
   commit
   end
   ```

Important If changes are made to the VIM Orchestrator and VIM configuration file after it was committed, you can apply the changes using the **load replace** command instead of the **load merge** command. You will also need to **commit** your changes.

5. Activate the VIM Orchestrator configuration aspects of the configuration file.

   ```
   activate nsd-id <nsd_name>
   ```

Important The output of this command is a transaction-id which can be used to monitor the deployment progress. If need be, the VIM deployment can be deactivated using the **deactivate** variant of this command.

6. Monitor the progress of the deployment.
   a. List the transactions.

      ```
      show transaction
      ```

      Example command output:

      ```
      TX ID              TX TYPE              ID            DEPLOYMENT TIMESTAMP
          STATUS     DETAIL STATUS
      ------------------------------------------------------------------------------
      1510448403-721303  activate-ns-deployment        test
      2017-11-12T01:00:03.721334-00:00  requested     -
      1510448404-104189  activate-vim-orch-deployment  ph-vim-orch
      2017-11-12T01:00:04.104204-00:00  requested     -
      ```

   b. Monitor the transaction log.

      ```
      show log tx-id
      ```

      Example command output:

      ```
      show log tx-id
      transaction 1510448403-721303
       tx-type       activate-ns-deployment
       deployment-id test
       timestamp     2017-11-12T01:00:03.721334-00:00
       status        success
      transaction 1510448404-104189
       tx-type       activate-vim-orch-deployment
       deployment-id ph-vim-orch
      ```

```
timestamp     2017-11-12T01:00:04.104204-00:00
status        success
```

c. Check the VIM Orchestrator status.

show vim-orchr

Example command output:

```
vim-orch status success
vim-orch steps-total 84
vim-orch steps-completed 84
vim-orch version "Red Hat OpenStack Platform release 10.0 (Newton)"
```

Important If there are any issues seen when executing the above commands, refer to Monitoring and Troubleshooting the Deployment for information on collecting logs.

7. Upon successful completion of the VIM deployment, proceed to VNF Deployment Automation, on page 22 for information and instructions on deploying your USP-based VNF.

VNF Deployment Automation

VNF Deployment Automation Overview

USP-based VNF deployment automation is performed through UAS.

Important This information in these sections assume that all of the USP Installation Prerequisites and/or that all requirements identified in the *Ultra M Solutions Guide* have been met.

Figure 3: UAS Deployment Automation Workflow for a Single VNF, on page 23 and Figure 4: UAS Deployment Automation Workflow for a Multi-VNF, on page 24 provide an overview of the VNF deployment automation process for Hyper-Converged Ultra M deployments. Details are provided in Table 3: VNF Deployment Automation Workflow Descriptions, on page 24.

Notes:

- The workflow described in this section is supported only with Ultra M deployments based on OSP 10 and that leverage the Hyper-Converged architecture.
- This information assumes that you have deployed the VIM Orchestrator and the VIM using the information and instructions in Virtual Infrastructure Manager Installation Automation, on page 1.

Figure 3: UAS Deployment Automation Workflow for a Single VNF

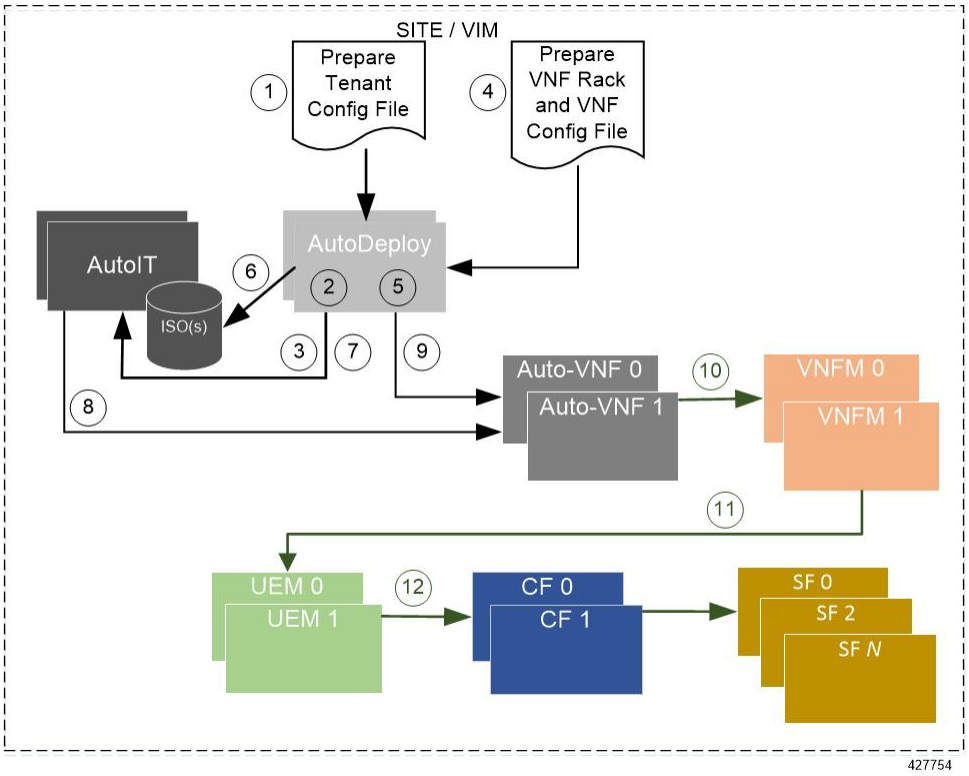


Figure 4: UAS Deployment Automation Workflow for a Multi-VNF

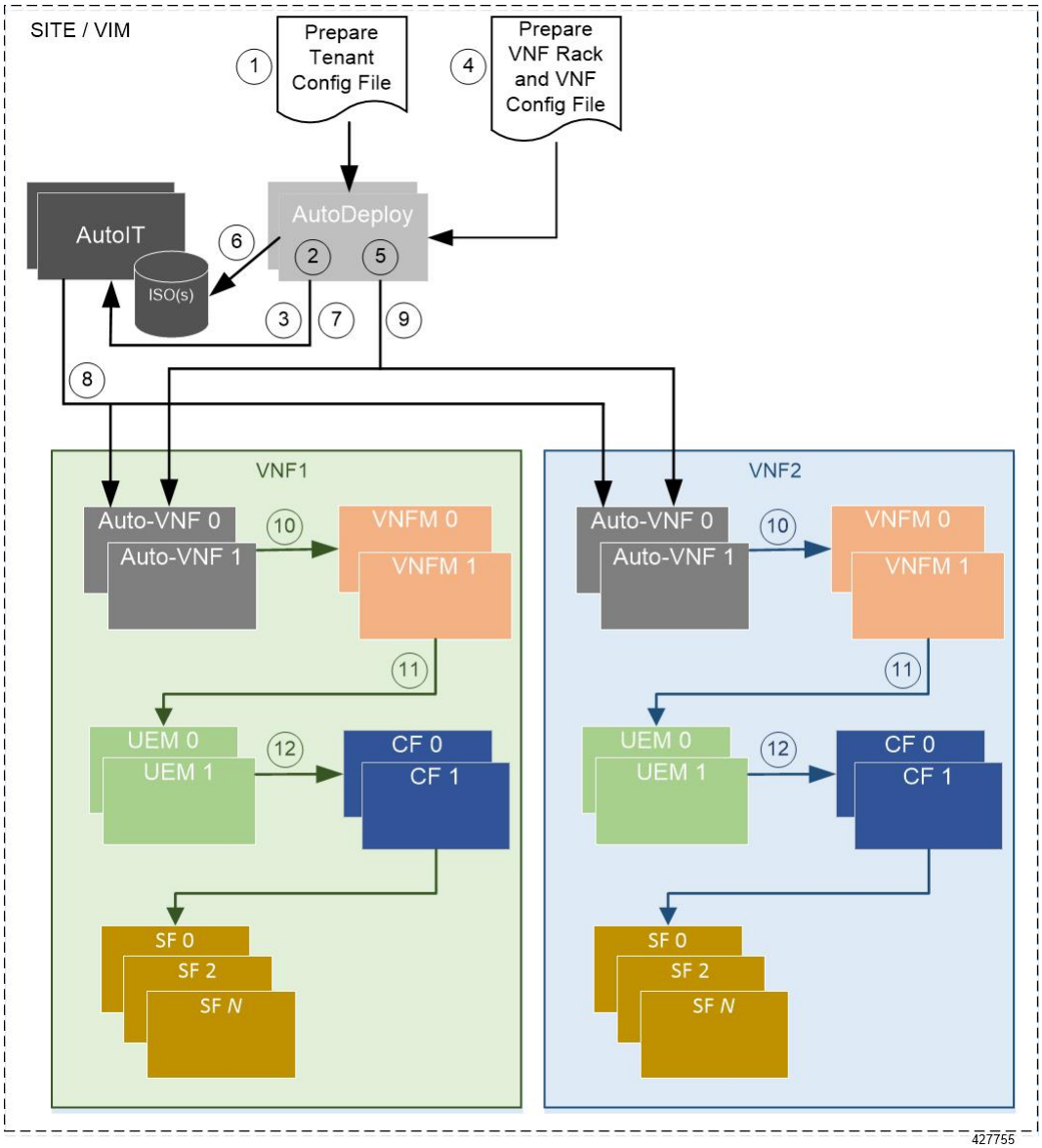


Table 3: VNF Deployment Automation Workflow Descriptions

Callout	Description
1	Prepare the tenant configuration file. Tenants define the physical resources allocated to specific user groups.
2	Load and activate the tenant configuration file through AutoDeploy. Tenants must be configured after the VIM has been deployed and before deploying the VNF.
3	AutoDeploy works with AutoIT to provision the tenants within the VIM.

Callout	Description
4	Prepare the VNF Rack and VNF descriptor configuration files. The parameters in this file identify the Compute Nodes to be used for VNF deployment and the VNFDs for AutoVNF, the VNFM, and for the UGP VNF. Refer to Sample VNF Rack and VNF Descriptor Configuration File for an example VNF Rack and VNF descriptor configuration file.
5	On the AutoDeploy VM, load and commit the configuration file prepared in previous step. Once committed, activate the previously loaded AutoDeploy configuration file. AutoDeploy processes this data to activate the Site and to deploy the functions needed to orchestrate the VNF deployment. Refer to Configure the VNF Rack and the VNF Descriptors, on page 30 for more information.
6	AutoDeploy loads the ISO on to AutoIT.
7	AutoDeploy passes data from the activated configuration to AutoIT requesting that it deploy the AutoVNF VM cluster for the initial VNF.
8	AutoIT deploys the AutoVNF VMs for the VNF.
9	Once the AutoVNF VMs are successfully deployed, AutoDeploy passes configuration information to the AutoVNF VMs. This is used by the AutoVNF to orchestrate the VNFM deployment. For deployments with multiple VNFs, AutoDeploy sends this information in parallel to the active AutoVNF VM for each VNF. **Important** In 6.0, concurrent VNF deployment functionality was not fully qualified and was made available only for testing purposes. In 6.1 and later releases, this functionality is fully qualified. For more information, contact your Cisco Accounts representative.
10	The active AutoVNF software module leverages the network service descriptor (NSD) information to work with the VIM to deploy the VNFM VMs. Once the VNFM VMs are successfully deployed, AutoVNF also ensures that the various VM catalogs pertaining to other VNFCs are on-boarded by the VNFM. It accomplishes this through a number of YANG-based definitions which are then used to configure various aspects of the virtualized environment using REST and NETCONF APIs. The VNFM mounts the VNFC catalogs and works with AutoVNF to deploy the various components that comprise the desired VNF use-case (e.g. UGP or USF).
11	The VNFM leverages the VNFD information to deploy the UEM VM cluster. Though the USP architecture represents a single VNF to other network elements, it is comprised of multiple VM types each having their own separate catalogs. The UEM component of the USP works with the VNFM to deploy these catalogs based on the intended VNF use case (e.g. UGP, USF, etc.).

Callout	Description
12	The UEM processes the Day-0 configuration information it received from the VNFM and deploys the Control Function (CF) and Service Function (SF) VNFC VMs. Once all of the VNF components (VNFCs) have been successfully deployed, AutoVNF notifies AutoDeploy.

Pre-VNF Installation Verification

Prior to installing the USP, please ensure that the following is true:

- The prerequisite hardware is installed and operational with network connectivity.
- The prerequisite software is installed, configured and functioning properly.
  - You have administrative rights to the operating system.
  - VIM Orchestrator is properly installed and operational.
  - VIM components are properly installed and operational. This configuration includes networks, flavors, and sufficient quota allocations to the tenant.

Important Supported and/or required flavors and quota allocations are based on deployment models. Contact your Cisco representative for more information.

  - You have administrative rights to the OpenStack setup.
- The Cisco USP software ISO has been downloaded and is accessible by you.

Deploy the USP-based VNF

The software roles that comprise the Ultra Automation Services (UAS) are used to automate the USP-based VNF deployment. UAS is designed to support deployment automation for all USP-based VNF scenarios.

Important Cisco's Elastic Services Controller (ESC) is the only VNFM supported in this release.

Important These instructions assume that you have verified the requirements identified in Pre-VNF Installation Verification, on page 26.

Configure VIM Tenants

VIM tenants define the physical resources allocated to specific user groups. They are provisioned by executing an API through AutoDeploy which processes parameters provided in a configuration file. The parameters are

grouped into a tenant descriptor which is referenced within the VIM Artifact descriptor. Tenants must be configured after the VIM has been deployed and before deploying the VNF.

In 6.3, the multi-tenants are supported per VNF but this feature was not fully qualified. It was made available only for testing purposes. In release 6.4, this functionality has been fully qualified. Refer to Sample Tenant Configuration File for an example tenant configuration file.

Important These instructions assume you are already logged on to the AutoDeploy VM as the root user.

To configure VIM tenants:

1. Prepare the tenant configuration file according to your deployment scenario.

Caution Ensure that the NSD name (nsd-id) specified in the configuration file is identical to the NSD name specified in the VIM Orchestrator and VIM configuration file. Additionally, ensure that only tenant information is referenced within the VIM artifact descriptor.

2. Login to the ConfD CLI as the *admin* user.

   confd_cli -u admin -C

3. Enter the *admin* user password when prompted.
4. Enter the ConfD configuration mode.

   config

5. Load the *addi-tenant.cfg* configuration file.

   load merge *<your_tenant_file_name>*.**cfg**

   commit

   end

6. Activate the tenant configuration.

   activate nsd-id *<nsd_name>*

Important The output of this command is a transaction-id which can be used to monitor the deployment progress. If need be, the tenant configuration can be deactivated using the **deactivate** variant of this command.

7. Monitor the progress of the tenant creation by viewing transaction logs:

   show log *<transaction_id>* **| display xml**

   transaction_id is the ID displayed as a result of the **activate** command executed in step 6, on page 27.

8. Verify that the tenants have been created properly by checking the network service record (NSR).

   show nsr

   Example command output:

```
                                VNF       VNF    VIM             VIM        VLR
NSR ID          NSD   VNFR      PACKAGE   RACK   ORCH     VIM    ARTIFACT   ID    NETWORK   VNFR
-------------------------------------------------------------------------------------------
sjc-instance    sjc   -         -         -      underc   overc  sjccore
```

9. Proceed to Configure OpenStack Prerequisites, on page 28.

Configure OpenStack Prerequisites

Prior to beginning the USP deployment process, there are several items as described in this section that must first be configured in OpenStack. The deployment automation process requires these items be configured in order for the UAS processes to run properly.

Important This procedure is not necessary if you are deploying a VNF on a Hyper-Converged Ultra M mode and have already deployed the VIM Orchestrator and the VIM using the information and instructions in Virtual Infrastructure Manager Installation Automation, on page 1.

Important The information in this section assumes that your Undercloud and Overcloud were previously installed and are operational as identified in Pre-VNF Installation Verification, on page 26.

Ensure that the following items are configured within the OpenStack CLI interface on the OSP-D Server / Staging Server:

1. Login to OSP-D and make sure to “su - stack” and “source stackrc”. Determine the name of the heat stack_name.

   heat stack-list

2. Source the rc file for the stack.

   source *~/<stack_name>* **rc**

3. Login to OpenStack Horizon with the tenant and username created in Configure VIM Tenants, on page 26 and download the credential file.

4. Source the “*stack_name***rc-core**” file.

   source *~/<stack_name>* **rc-core**

5. Create the volume type for your deployment.

   cinder type-create LUKS

6. Determine the tenant ID for the OpenStack *core* project.

   openstack project list | grep core

7. Create a neutron router for the core tenant called “main” and associate it with the *core* project tenant ID.

   neutron router-create main --tenant-id *<core_tenant_id>*

8. Create a public/“external” network.

```
neutron net-create public --router:external True
--provider:physical_network datacentre --provider:network_type vlan
--provider:segmentation_id <vlan_segmentation_id>
```

Important *<vlan_segmentation_id>* is based on your OpenStack configuration and must match the VLAN ID specified for the network.

```
neutron subnet-create public <network_address>/<mask-bits> --name
public-subnet --allocation-pool start= <start_address>, end=<end_address>
--disable-dhcp --gateway <gateway_address>
neutron router-gateway-set main public
```

Important It is recommended that you assign a static IP address to your router if your VNF configuration uses floating IP addresses in order to avoid potential IP address conflicts. The IP address is assigned based on the subnet created for floating IPs on the network. Floating IP address support is configured at the VNFD-level within the AutoDeploy configuration using the **floating-ip enabled** and **floating-ip ip-address** parameters. Static addresses can be assigned to the router using the following command:

```
neutron router-gateway-set main public --fixed-ip subnet_id=`neutron
subnet-list |grep public | awk '{print $2}'`,ip_address= <static_ip_address>
```

9. Create SRIOV networks for use with the DI-internal and Service networks.
   a. Create the SR-IOV network for DI-internal network.

      ```
      neutron net-create di-internal1 --provider:physical_network
      phys_pcie1_0 --provider:network_type flat --shared
      neutron subnet-create di-internal1 <network_address>/<mask-bits>--name
      di-internal1-subnet --enable-dhcp
      ```

   b. Repeat step 9.a, on page 29 for the redundant DI-network in the case where NIC bonding is used.

      ```
      neutron net-create di-internal2 --provider:physical_network
      phys_pcie4_1 --provider:network_type flat --shared
      neutron subnet-create di-internal2 <network_address>/<mask-bits>--name
      di-internal2-subnet --enable-dhcp
      ```

   c. Create the SR-IOV network for Service 1.

      ```
      neutron net-create service1 --provider:physical_network phys_pcie1_1
       --provider:network_type flat --shared
      neutron subnet-create service1 <network_address>/<mask-bits>--name
      service1-subnet --enable-dhcp
      ```

   d. Repeat step 9.d, on page 29 for the redundant Service in the case where NIC bonding is used.

      ```
      neutron net-create service2 --provider:physical_network phys_pcie4_0
       --provider:network_type flat --shared
      neutron subnet-create service2 <network_address>/<mask-bits>--name
      service2-subnet --enable-dhcp
      ```

e. Repeat steps 9.c, on page 29 and 9.d, on page 29 for each additional Service network required for your deployment.

10. Proceed to Configure the VNF Rack and the VNF Descriptors, on page 30.

Configure the VNF Rack and the VNF Descriptors

Once the VIM tenants and OpenStack prerequisites have been configured, the VNF Rack and VNF descriptors must be configured. Once these items are configured, your VNF can be deployed.

The VNF Rack descriptor is a logical grouping of Compute Nodes. It is used to map the nodes to specific VNFs. It is equivalent to Availability Zones and/or Host Aggregates in OpenStack. Like tenants, VNF Rack descriptors are configured at the network service descriptor (NSD) level and is referenced within the VIM artifact descriptor.

The VNF descriptor (VNFD) defines the deployment flavor for a specific VNF including all the aspects of VNF resources and associated networking. A single NSD can contain multiple VNFDs. For example, a configuration for deploying a UGP VNF on Ultra M will have separate VNFDs for:

- AutoVNF (one instance per VNF)
- VNFM
- UGP VNF

These VNFDs are defined under nested NSDs, one per VNF, within the file. Each VNF must be defined by its own NSD. The file also contains additional parameters related to and required by your specific deployment scenario. These are a mix of basic, operational paramters and enhanced features supported within the USP VNF deployment on the Ultra M solution. For more information on enhanced features, refer to:

- Monitoring and Recovering AutoVNF Through AutoIT
- Monitoring and Recovering VNFC Through AutoVNF
- Configuring Fully-Defined VM Names for ESC, on page 32
- The "Health Monitoring Within the Ultra M Solution" chapter of the *Ultra M Solutions Guide*

Refer to Sample VNF Rack and VNF Descriptor Configuration File for an example VNF Rack and VNF descriptor configuration file. Detailed information for the parameters used in the configuration constructs within the file is provided in the *Cisco Ultra Services Platform NETCONF API Guide*.

Important User credentials are configured through Secure Tokens specified in the configuration file. Ensure that passwords configured with Secure Token meet the requirements specified in Password Requirements and Login Security.

Important These instructions assume you are already logged on to the AutoDeploy VM as the root user.

To configure the VNF Rack and VNF descriptor file:

1. Prepare the VNF Rack and VNF descriptor configuration file according to your deployment scenario.

Caution Ensure that the NSD name (nsd-id) specified in the configuration file is identical to the NSD name specified in the VIM Orchestrator/VIM and tenant configuration files.

a. Add the VNF Rack descriptor parameter configuration to the file and reference it within the VIM Artifact descriptor.

b. Add the VNFD and other construct parameter configurations to the file as required for your deployment.

c. *Optional*. If required, add any other additional parameters to configure the enhanced features like - AutoVNF Health Check, VNFC Health Check, Passing Fully-Defined VM Names, syslog proxy functionality, and so on.

Example Configuration for Health Check features:

```
nsd <nsd_name>
…
vnfd <vnfd_name>
…
  vnfc <vnfc_name>
    health-check enabled
    health-check probe-frequency 10
    health-check probe-max-miss 6
    health-check retry-count 6
    health-check recovery-type restart-then-redeploy
    health-check boot-time 300
…
```

Example Configuration for Passing VM Names:

```
nsd <nsd_name>
<---SNIP--->
vnfd <uas_vnfd_name>
vnf-type usp-uas
…
configuration set-vim-instance-name true
<---SNIP--->
vnfd <vnfm_vnfd_name>
vnf-type esc
…
configuration set-vim-instance-name true
<---SNIP--->
```

Example syslog proxy functionality:

```
nsd <nsd_name>
…
vnfd <vnfd_name>
…
  vnfc <vnfc_name>
    uas-proxy
…
```

d. Save the file.

2. Login to the ConfD CLI as the *admin* user.

```
confd_cli -u admin -C
```

3. Enter the *admin* user password when prompted.

4. Enter the ConfD configuration mode.

```
config
```

5. Load the VNF Rack and VNF descriptor configuration file.

```
load merge <your_config_file_name>.cfg
commit
end
```

6. Activate the VNF rack and VNF descriptor configuration.

```
activate nsd-id<nsd_name>
```

Important The output of this command is a transaction-id which can be used to monitor the deployment progress. If needed, the VIM deployment can be deactivated using the **deactivate** variant of this command.

7. Monitor the progress of the tenant creation by viewing transaction logs:

```
show log <transaction_id> | display xml
```

transaction_id is the ID displayed as a result of the command executed in step 6, on page 32.

8. Verify that the VNFs have been created properly by checking the network service record (NSR).

```
show nsr
```

Example command output:

```
                                          VNF     VIM
                                VLR
NSR ID                    NSD                     VNFR
               VNF PACKAGE  VNF RACK  VIM ORCH  VIM ARTIFACT  NSR ID  VLR NETWORK  VNFR

-------------------------------------------------------------------------------------------------------------------------
ab-autovnf-instance   ab-autovnf      [ ab-autovnf-vnf1-em ab-autovnf-vnf1-esc
ab-autovnf-vpc-up1 ]   [ usp_6_0 ]     -         -       vim_art_rack    -        -
      -
ss-autoit-instance    ss-autoit       [ ss-autoit-f-autovnf ]
               [ usp_6_0 ]     -         -       vim_art_rack    -        -          -
```

Configuring Fully-Defined VM Names for ESC

Leveraging capabilities in the VNFM (ESC version 3.0 and later), UAS supports the ability to generate and pass VM names to the VNFM. This is applicable to all VMs deployed on OpenStack including ESC and AutoVNF.

VM name generation is based on known algorithms using the following parameters:

- VNF Component (VNFC) name
- Network Service Descriptor (NSD) name
- VNF Descriptor (VNFD) name

- VIM tenant name

VM instance names are assembled as follows:

For CF/SF:

```
<nsd_name>-<vnfd_name>-<vim_tenant_name>-<vdu_id>-X
```

For example: abcUAS-LBPCF401-UGP-core-LBPCF401-CF-VDU-0

For UEM:

```
<nsd_name>-<vnfd_name>-<vim_tenant_name>-<vnfc_instance_id>-X
```

For example: abcUAS-LBPCF401-UGP-core-EM1-1

Important There may be one or more VNFC instances depending on the redundancy and scaling.

This functionality is controlled in the UAS through a YANG-based configurable in the VNFD.

Important In 6.0, this functionality was not fully qualified and was made available only for testing purposes. In 6.1 and later releases, this functionality is fully qualified. For more information, contact your Cisco Accounts representative.

```
nsd <nsd_name>
<---SNIP--->
  vnfd <uas_vnfd_name>
    vnf-type          usp-uas
    …
configuration set-vim-instance-name true
<---SNIP--->
  vnfd <vnfm_vnfd_name>
    vnf-type          esc
    …
    configuration set-vim-instance-name true
<---SNIP--->
```

This functionality is enabled (e.g. set to “true”) by default.