Preparing for Installation on Servers Without Internet Access

Complete the following procedure if your monitoring node does not have access to the Internet. In this case, you must download the Cisco NFVI installation files to a 64 GB (minimum) USB 2.0 stick on a staging server that has Internet access, then use the USB stick to copy the files to the management node. If your management node has Internet access, you can skip this chapter.

Preparing to Install Cisco NFVI on Management Nodes Without Internet Access

The following procedure tells you how to download the Cisco NFVI installation files onto a USB stick mounted on a staging server with Internet access. You will then use the USB to load the Cisco NFVI installation files onto the management node that does not have Internet access.

Note We recommend you to use Virtual Network Computing (VNC), other terminal multiplexer, or similar screen sessions to complete these steps.

Before you begin

This procedure requires a CentOS 7 staging server (VM, laptop, or UCS server) with a 64 GB USB 2.0 stick. The staging server must have Internet access (wired access is recommended) to download the Cisco VIM installation files, which you will load onto the USB stick. You will then use the USB stick to load the installation files onto the management node. The installation files are over 25 GB in size; downloading them to the USB stick might take several hours depending on the speed of your Internet connection, so plan accordingly. Before you begin, disable the CentOS sleep mode.

Step 1 On the staging server, use yum to install the following packages:

- PyYAML (yum install PyYAML)
- python-requests (yum install python-requests)

Step 2 Connect to the Cisco VIM software download site using a web browser and login credentials provided by the account representative and download the **getartifacts.py** script from external registry:

```
# download the new getartifacts.py file (see example below)
curl -o getartifacts.py
https://username:password@cvim-registry.com/mercury-releases/cvim22-rhel7-osp10/releases/<2.2.n>/getartifacts.py



curl -o getartifacts.py-checksum.txt
https://username:passwordcvim-registry.com/mercury-releases/cvim22-rhel7-osp10/releases/<2.2.n>//getartifacts.py-checksum.txt
# calculate the checksum and verify that with one in getartifacts.py-checksum.txt
sha512sum getartifacts.py

# Change the permission of getartificats.py
chmod +x getartifacts.py
```

Step 3 Run **getartifacts.py**. The script formats the USB 2.0 stick and downloads the installation files. You must provide the registry username and password, tag ID, and USB partition on the staging server.

```
# cd <installer-xxx>/tools/
# ./getartifacts.py -h
usage: getartifacts.py [-h] -t TAG -u USERNAME -p PASSWORD -d DRIVE
                       [--proxy PROXY] [--retry]
                       [--artifacts [ARTIFACTS [ARTIFACTS ...]]]

Script to pull container images en masse.

optional arguments:
  -h, --help            show this help message and exit
  -t TAG, --tag TAG     installer version to pull
  -u USERNAME, --username USERNAME
                        Registry username
  -p PASSWORD, --password PASSWORD
                        Registry password
  -d DRIVE, --drive DRIVE
                        Provide usb drive path
  --proxy PROXY         https_proxy if needed
  --retry               Try to complete a previous fetch
  --artifacts [ARTIFACTS [ARTIFACTS ...]]
                        Artifact List values(space separated): core insight
                        all

This script pulls images from remote registry and copies the contents to usb drive
```

To identify the USB drive, execute the **lsblk** command before and after inserting the USB stick. The command displays a list of available block devices. The output data will help you to find the USB drive location. Provide the entire drive path in the –d option instead of any partition.

For example:

sudo ./getartifacts.py -t <tag_id> -u <username> -p <password> -d </dev/sdc> [--artifacts …] [--proxy proxy.example.com] –

For example: To download only the insight artifacts, execute the following command:

sudo ./getartifacts.py -t <tag_id> -u <username> -p <password> -d </dev/sdy/>-- artifacts insight

Note We recommend you not to remove the USB stick, while the synchronization is under way.

Note Sometimes executing getartifacts.py errors out with the following message: *stderr: mount: wrong fs type, bad option, bad superblock on /dev/sdy1, missing codepage or helper program, or other error.* This typically points to a bad superblock and the mount fails. Reformatting the drive might help in this case, use the fsck command to recover the drive: **fsck.ext4 -pv /dev/sdc .**

Note Given the size of the artificats (>25G) we recommend you to execute this step over a wired internet connection. This step typically takes few hours to download and populate data on USB stick, depending on the internet connectivity.

The getartifacts.py script downloads the following:

- Core Packages
  - buildnode-K9.iso
  - mercury-installer.tar.gz
  - registry-2.3.1.tar.gz
- Optional: Unified Management Package called Insight
  - insight-K9.tar.gz
  - mariadb-app-K9.tar.gz
- Respective checksums

Step 4 Verify the integrity of the downloaded artifacts and the container images by running the following command:

```
# create a directory
sudo mkdir -p /mnt/Cisco

# /dev/sdc is the USB drive, same as supplied in getartifacts.py python script
sudo mount /dev/sdc1 /mnt/Cisco
cd /mnt/Cisco

# execute the test-usb help to look at the options
./test-usb -h

usage: ./test-usb [-h] -- Show this program to check integrity of artifacts in this USB stick
                  [-c] -- Check integrity of only core artifacts in this USB stick
                  [-i] -- Check integrity of only insight artifacts in this USB stick
                  [-a] -- Check integrity of all (core and insight) artifacts in this USB stick
                  [-l] -- Location of artifacts

# execute the verification script
./test-usb

# failures will be explicitly displayed on screen, sample success output below
# sample output of ./test-usb execution with 2.2.x release
#./test-usb
INFO: Checking the integrity of this USB stick
INFO: Checking artifact buildnode-K9.iso
INFO: Checking artifact registry-2.3.1.tar.gz
INFO: Checking required layers:
INFO: 548 layer files passed checksum.

Following output shows the result when using -a option
# ./test-usb -a
INFO: Checking the integrity of this USB stick
INFO: Checking artifact buildnode-K9.iso
INFO: Checking artifact registry-2.3.1.tar.gz
INFO: Checking artifact mariadb-app-K9.tar.gz
```

```
INFO: Checking artifact haproxy-K9.tar.gz
INFO: Checking artifact insight-K9.tar.gz
INFO: Checking required layers:
INFO: 548 layer files passed checksum.
```

If a failure occurs, an error message is displayed.

For example:

```
# ./test-usb
INFO: Checking the integrity of this USB stick
INFO: Checking artifact buildnode-K9.iso
ERROR: Checksum for artifact buildnode-K9.iso does not match ('SHA512 (buildnode-K9.iso) =
96ec62a0932a0d69daf60acc6b8af2dc4e5eca132cd3781fc17a494592feb52a7f171eda25e59c0d326fbb09194eeda66036cbdc3870dafe74f59cf1f2dce225'
 != 'SHA512 (buildnode-K9.iso) =
a6a9e79fa08254e720a80868555679baeea2dd8f26a0360ad47540eda831617bea0514a117b12ee5f36415b7540afa112a1c904cd69e40d704a8f25d78867acf')
INFO: Checking artifact registry-2.3.1.tar.gz
ERROR: Artifact registry-2.3.1.tar.gz is not present
INFO: Checking required layers:
ERROR: Layer file sha256:002aa1f0fbdaea7ea25da1d906e732fe9a9b7458d45f8ef7216d1b4314e05207 has a bad
 checksum
ERROR: Layer file sha256:5be3293a81773938cdb18f7174bf595fe7323fdc018c715914ad41434d995799 has a bad
 checksum
ERROR: Layer file sha256:8009d9e798d9acea2d5a3005be39bcbfe77b9a928e8d6c84374768ed19c97059 has a bad
 checksum
ERROR: Layer file sha256:ea55b2fc29b95d835d16d7eeac42fa82f17e985161ca94a0f61846defff1a9c8 has a bad
 checksum
INFO: 544 layer files passed checksum.
```

Step 5 To resolve download artifact failures, unmount the USB and run the getartifacts command again with the --retry option.

```
sudo ./getartifacts.py -t <tag_id> -u <username> -p <password> -d </dev/sdc> --retry
```

Step 6 Mount the USB and then run the test-usb command to validate if all the files are downloaded:

```
# /dev/sdc is the USB drive, same as supplied in get artifacts.py python script
sudo mount /dev/sda1 /mnt/Cisco
cd /mnt/Cisco

# execute the verification script
./test-usb

# In case of failures the out of the above command will explicitly display the same on the screen
```

Step 7 After the USB integrity test finishes, unmount the USB stick by running the following command:

```
sudo umount /mnt/Cisco
```