Harmonic cOS VCMTS Image Documentation

The traditional cable modem termination system (CMTS) has played a key role in facilitating high speed internet access for millions of subscribers. The advent of cloud computing and containerization has caused a shift in access network solutions, with proposals to replace the traditional “integrated” CMTS (iCMTS) with a virtual CMTS (vCMTS), where the hardware-based iCMTS is replaced with open, interoperable, cloud-native and fully containerized applications that deliver the same functionality.

By decoupling the CMTS implementation from the underlying hardware, vCMTS was proposed as a way of providing cable providers the flexibility to update their networks to support next-gen cable modem technologies. (e.g.: DOCSIS 4.0 and Turbo DOCSIS 3.1) without significant hardware replacements, and to scale-up on demand to support increasing density and demand for higher throughput.

For example, when deployed in a distributed access architecture, where the digital nodes can be placed closer to the homes they serve instead of in the headend, vCMTS can provide improved service with significantly increased density while also reducing headend infrastructure costs.

vCMTS products are available from several companies (Cisco, Harmonic, Arcadyan, Casa Systems, CommScope, Sagemcom, Ubee, Vantiva, Vecima)

Harmonic vCMTS products have been on the market as early as 2018, with Harmonic’s CableOS platform already connected to about 18.4 million modems worldwide.

Image-Build Dependencies and Prerequisites

- A machine running Ubuntu 18.04+ with the ability to run KVM virtual machines.
- Packages:
  - qemu-system
  - qemu-utils
  - nbdkit
  - libnbd-bin
  - fuse2fs

    - ovmf
    - cloud-image-utils
    - parted
    - Packer, v1.7.0 or newer

Image-Deployment Requirements

- MAAS 3.0+
- Curtin 21.0+

Creating `harmonic-installer` Image

This Packer template will build a customized `harmonic-installer` image from an official Ubuntu Server cloud-enabled image.

The resulting image can be imported into MAAS and deployed to a bare-metal server, which bootstraps a fully automated CableOS installation.

Upon initial boot, a single-use (*oneshot*) SystemD service that downloads the latest `Apollo` (*Harmonic cOS*) .iso and executes the commands that will write the image content to the system's physical disk.

Building the image

The image can be built using the following packer commands, however the including Makefile simplifies the process:

- ```
  packer init .
  ```
- ```
  packer build \
    -var ubuntu_series=${SERIES:=jammy} \
    -var timeout=${TIMEOUT:=1h} \
    -var customize_script=${CUSTOMIZE_SCRIPT:=/dev/null}
  ```

To build the image using Make, simply run:

```
make all
```

Set variables as follows:

```
make all CUSTOMIZE_SCRIPT=<my_custom_commands.sh>
```

`CUSTOMIZE_SCRIPT` is an optional script to perform additional build steps inside the packer build target. This can be used to make and test changes to the build process without needing to modify the static build content.

Accessing external files from you script

Files can be included in your `CUSTOMIZE_SCRIPT` using the HTTP server that Packer creates at build time. The server IP and port can be accessed using the `PACKER_HTTP_IP` and `PACKER_HTTP_PORT` vars

```
!#/bin/bash
#
# Fetch a debian package from the Packer HTTP server and install it.
#

curl http://${PACKER_HTTP_IP}:${PACKER_HTTP_PORT}:/example.deb --output
/opt/example.deb

dpkg --install /opt/example.deb

rm -f /opt/example.deb
```

Makefile Parameters

- PACKER_LOG

Enable (1) or Disable (0) verbose packer logs. The default value is set to 0.

- SERIES

Specify the Ubuntu Series to build. The default value is set to Jammy.

- TIMEOUT

The timeout to apply when building the image. The default value is set to 1h.

- CUSTOMIZE_SCRIPT

Specify path to a script that will be executed inside the Packer build environment. The default is set to /dev/null

Default Credentials

The default username and password are set by the `user-data` file when the build VM is started. You generally will not need to log in to the image unless troubleshooting.

- User: `root`
- Password: `ubuntu`

Importing `harmonic-installer.tar.gz` into MAAS

- Copy the image file to the MAAS server (`44.10.4.101`) and SSH into it.
- Log into the MAAS API using the CLI

  ```
  maas login admin http://44.10.4.101:5240/MAAS
  ```

  (*The API Key can be found in the MAAS UI under User settings*)
- Run the command provided at the end of the Packer build process, whiuch will contain correct `sha256sum` and `byte size` values for the new image (*If MAAS has been deployed using **Snaps**, then the image file **must** be located in your user's home directory*)

The script runs the following command which prints out the exact command string you will need in order to import the image. (I.e.: Correct SHA256SUM, Byte Size, Filenames).

```
for image in *.tar.gz; do
if [[ -e ${image} ]]; then
  cat <<EOF
  MAAS IMPORT:
  Copy the image to the MAAS server then
  Import the image to MAAS with the following command:

  maas admin boot-resources create name='custom/$(echo "${image}"|sed -e
"s/\..*//")' title='$(echo "${image}" | sed -e "s/\..*//;s/\-/ /;s/\b\
(.\)/\u\1/g")' architecture='amd64/generic' filetype='tgz'
sha256='$(sha256sum "${image}" | cut -d ' ' -f1)' size='$(stat -c'%s'
"${image}")' base_image='ubuntu/jammy' content@='${image}'
EOF
fi
done
```

The resulting command string will look like this:

```
  MAAS IMPORT:
  Copy the image to the MAAS server then
  Import the image to MAAS with the following command:

  maas admin boot-resources create name='custom/harmonic-installer'
title='Harmonic Installer' architecture='amd64/generic' filetype='tgz'
sha256='0c1e7be460d391b0b8c297963609d9fbd4a658066818459a44fa37631954865d'
size='1311814444' base_image='ubuntu/jammy' content@='harmonic-
installer.tar.gz'
```

Main configuration script `harmonic-installer.sh`

This script is run on initial boot by the in-memory OS to provision the physical disks.

```
#!/bin/bash
#############################################################################
####
#
#   harmonic-installer.sh
#
#   This script is executed either by 'harmonic-install.service' or by a
#   user-provided 'cloud-init' configuration.
#
#   The harmonic-installer.sh script will then perform the following
steps:
#
#   1. Download and install the ostree-production .deb packages from the
#   MAAS webserver.
#   2. Download the latest Apollo (Harmony cOS) .iso from the MAAS
webserver.
#   3. Create a /data directory and move the Apollo .iso into it.
#   4. Run the 'ostree-production' commands to display and write the .iso
#   to the system's physical disk (/dev/sda)
#   5. Reboot the system to /dev/sda
#
#   The system will reboot into Harmony cOS
#
#############################################################################
####

# shellcheck disable=SC2312


export webserverHost="172.22.31.150"
export webserverPort="8080"
export apolloRelease="release-3.21.3.0-7+auto15"
export apolloISO="APOLLO_PLATFORM-${apolloRelease}.iso"
export ostreePackages="ostree-upgrade-bootstrap_2.0.41_all.deb ostree-
upgrade_2.0.41_all.deb"
export proxyURI="http://proxy4.spoc.charterlab.com:8080"
export
proxyIgnore="localhost,127.0.0.1,127.0.0.53,spoc.charterlab.com,nfv.charte
rlab.com,proxy4.spoc.charterlab.com,44.10.4.101/32,44.10.4.200/32,172.22.0
.0/16"
export workingDir="/media/root-rw"
export isoDir="/data"
export physicalDisk="/dev/sda"
export proxy=0
export download=0
export install=0

unset http_proxy
unset https_proxy
unset no_proxy

runPrint() {
cat <<EOF
```

```
================================================================
  $@
================================================================
EOF
}

if ! lsblk "${physicalDisk}" >/dev/null 2>&1; then
runPrint "Physical disk ${physicalDisk} not found.  Harmonic cOS
installation can not run on this host."
exit 0
else
runPrint "Physical disk ${physicalDisk} found.  Proceeding with Harmonic
cOS installation..."
fi


showHelp() {
cat << EOT
Usage: $0 [-p -v] [-i] [-h]

Image a physical server with Harmonic cOS

-p|     (OPTIONAL) Enable the HTTP Proxy
        Note: HTTP Proxy is disabled by default

-v|     (OPTIONAL) Enable verbose and xtrace mode (set -xv)

-i|     (REQUIRED FOR INSTALL) Install Apollo (Harmonic cOS) .iso located
in "${isoDir}" using ostree scripts

-h|     Display help

EOT
}

proxySetup() {

  if [[ ${proxy} == 1 ]]; then
    runPrint "Configuring HTTP(S) proxies"
    : "${http_proxy:=${proxyURI}}"  && export http_proxy="${http_proxy}"
    : "${https_proxy:=${proxyURI}}" && export https_proxy="${https_proxy}"
    : "${no_proxy:=${proxyIgnore}}" && export no_proxy="${no_proxy}"

    runprint "Proxy Information:
    http_proxy: ${http_proxy}
    http_proxy: ${https_proxy}
    no_proxy: ${no_proxy}
    "
  fi
  return
}

ostreeSetup() {
```

```
  runPrint "Installing 'ostree-production' provider packages"
    for debPkg in ${ostreePackages}; do

      runPrint "Downloading ${debPkg}"
      wget "http://${webserverHost}:${webserverPort}/packages/${debPkg}" -
O "${workingDir}/${debPkg}" 2>&1

      runPrint "Installing ${debPkg}"
      dpkg -i "${workingDir}/${debPkg}" 2>&1

    done

  return
}

harmonicSetup() {
  runPrint "Creating ${isoDir}"
  mkdir -p "${isoDir}"

  runPrint "Downloading ${apolloISO} to ${isoDir}"
  wget "http://${webserverHost}:${webserverPort}/apollo/latest" -O
"${isoDir}/${apolloISO}" 2>&1

  return
}

harmonicInstall() {
  runPrint "Listing .iso files located in ${isoDir}"
  ostree-production list-isos 2>&1

  runPrint "Installing ${isoDir}/${apolloISO} tdo ${physicalDisk}"
  ostree-production -D "${physicalDisk}" from "${isoDir}"/"${apolloISO}"
2>&1

  return
}

while getopts "h?vpi" o; do
    case "${o}" in
        h)
            showHelp
            exit 0
            ;;
        v)
            set -xv
            ;;
        p)
            proxy=1
            ;;
        i)
            download=1
            install=1
            ;;
        ?|*)
```

```
            showHelp
            exit 1
            ;;
    esac
done
shift $((OPTIND-1))


if [[ ${proxy} == 1 ]]; then
proxySetup
fi

if [[ "${download}" == 1 ]]; then
ostreeSetup
fi

if [[ "${install}" == 1 ]]; then
harmonicSetup
harmonicInstall
fi

exit 0
```

Ephemeral Deployment

A second deployment method that was recently made available (although only via the CLI is an 'ephemeral boot' deployment.

This method uses the default in-memory bootstrap OS (Ubuntu 22.04 at the time of this writing). The OS itself is unmodified, so how do we get it to exectute the scripts needed to image the hardware?

We use a custom *User-Data* configuration. The user-data is read only once per deploy, and we have fairly granular control over when the desired commands are executed. For our purposes, We want to run the commands a bit later in the boot process to ensure that all of the other MAAS provisioning scripts have completed successfully.

So we the 'run-commands' section. In cloud-init, run-commands will execute later in the boot process after all MAAS scripts have run (and the machine deployment is marked successfully deployed) -- but before the targeted OS is written to the physical drives.

Example User-Data script

- You will see here that we ensure that the default 'ubuntu' user is present with a standardized password.
- We ensure the user has the desired sudo privileges and SSHD configuration.
- The installer script (seen above) is downloaded and executed in the ephemeral environment
- Finally the host is rebooted.

```
#!cloud-init
users:
  - name: root
    lock_passwd: false
    plain_text_passwd: ubuntu
    ssh_redirect_user: false
    ssh_pwauth: true
    disable_root: false
    preserve_hostname: true
runcmd:
  - echo "---------------- Harmonic Installer - Update Sudoers (Add
'ubuntu' user with 'ALL=(ALL) NOPASSWD:ALL') -----------------"
  - echo 'ubuntu ALL=(ALL) NOPASSWD:ALL' > /etc/sudoers.d/ubuntu
  - echo "---------------- Harmonic Installer - Configure SSHd (Allow Root
Login) -----------------"
  - sed -i -e '/^[#]*PermitRootLogin/s/^.*$/PermitRootLogin yes/'
/etc/ssh/sshd_config
  - echo "---------------- Harmonic Installer - Configure SSHd (Allow
Password Login) -----------------"
  - sed -i -e '/^[#]*PasswordAuthentication/s/^.*$/PasswordAuthentication
yes/' /etc/ssh/sshd_config
  - echo "---------------- Harmonic Installer - Setup (RESTARTING SSHd) --
---------------"
  - systemctl restart ssh
  - echo "---------------- Harmonic Installer - Install Script (DOWNLOAD)
-----------------"
  - wget http://172.22.31.150:8080/scripts/harmonic-installer.sh -O
/media/root-rw/harmonic-installer.sh
  - chmod +x /media/root-rw/harmonic-installer.sh
  - echo "---------------- Harmonic Installer - Install Script (RUNNING) -
----------------"
  - /media/root-rw/harmonic-installer.sh -vi
  - echo "------------------- Harmonic Installer - Install Script
(COMPLETED) ------------------"
  - echo "------------------- Harmonic Installer - Rebooting System ------
------------"
  - shutdown -r now
```