Customizing Virtual Machine Images

Javier Fontán - OpenNebula Developer
There are other ways to create your images:

- virt-install
- packer.io
- foreman
- etc...

Sometimes modifying already created images is convenient.

Even if you use other image formats you can convert them.
qcow2 Format

$ qemu-img create -f qcow2 image.qcow2 10G
qcow2 Image With Parent

$ qemu-img create -f qcow2 -o backing_file=base.qcow2 image.qcow2
Consolidate qcow2 Image

$ qemu-img convert -O qcow2 image.qcow2 new_image.qcow2
qcow2 Image After Copy

$ cp base.qcow2 image.qcow2
Create Delta From 2 qcow2 Images

$ qemu-img rebase -b base.qcow2 image.qcow2

$ qemu-img convert -O qcow2 -o backing_file=base.qcow2 image.qcow2 new_image.qcow2
Mount Image

- Convert to raw and use mount -o loop
  - `mount -o loop,offset=32256 image.raw /mnt`

- Convert to raw and use losetup
  - `losetup /dev/loop0 image.raw`
  - `kpartx -a /dev/loop0`
  - `mount /dev/loop0p1 /mnt`

- Use nbd
  - `modprobe nbd`
  - `qemu-nbd -c /dev/nbd0 image.qcow2`
  - `mount /dev/nbd0p1 /mnt`
libguestfs

From its webpage http://libguestfs.org:

libguestfs is a set of tools for accessing and modifying virtual machine (VM) disk images. You can use this for viewing and editing files inside guests, scripting changes to VMs, monitoring disk used/free statistics, creating guests, P2V, V2V, performing backups, cloning VMs, building VMs, formatting disks, resizing disks, and much more.
guestfish - Read or Edit Files

$ guestfish -ia image.qcow2
<<fs> cat /var/log/service/error.log
<<fs> vi /etc/service.service.conf

$ guestfish -ia image.qcow2 <<EOF
upload service.conf /etc/service.conf
etc/sshd
EOF

$ guestfish --ro -i -c qemu:///system -d vm-name
<<fs> cat /var/log/service/error.log
Dojo Barcelona 2015

**virt-customize**

- Starts custom VM and attach disks and connects to network
- Change passwords, create users
- Move files
- Install packages
- Execute scripts

```
virt-customize [--options]
    [-d domname | -a disk.img [-a disk.img ...]] [-chmod PERMISSIONS:FILE] [--commands-from-file FILENAME]
    [--firstboot-command 'CMD+ARGS'] [--firstboot-install PKG,PKG...] [--hostname HOSTNAME] [--install PKG,PKG...]
    [--selinux-relabel] [--sm-credentials SELECTOR]
```
OpenNebula Marketplace Images

- Download CentOS images
- Create CDROM with OpenNebula context packages
- Create script to modify the image
  - Mount CDROM
  - Install context packages
  - Remove cloud-init and NetworkManager packages
  - Install EPEL repository
  - Install growpart packages
OpenNebula Images - Create CDROM

# Download context packages from github
curl -s https://api.github.com/repos/OpenNebula/addon-context-linux/releases | jq -r '.[0].assets[].browser_download_url' | xargs -L1 wget -P repo

# Create ISO image with label “EXTRA”
genisoimage -o extra-packages.iso -R -J -V EXTRA repo/
OpenNebula Images - Prepare Script

mkdir /tmp/mount
mount LABEL=EXTRA /tmp/mount

# Install opennebula context package
rpm -Uvh /tmp/mount/one-context*rpm

# Remove cloud-init and NetworkManager
yum remove -y NetworkManager cloud-init

# Install growpart and upgrade util-linux
yum install -y epel-release --nogpgcheck
yum install -y cloud-utils-growpart --nogpgcheck
yum upgrade -y util-linux --nogpgcheck
OpenNebula Images - Calling virt-customize

# Create an overlay to preserve original image
$ qemu-img create -f qcow2 -b $orig $image

# Run customization
$ virt-customize --attach $ISO_IMAGE --run $script --format qcow2
   -v -a $image --root-password disabled
Optimizing Images

- qemu-img does not know anything about filesystems
- Blocks not allocated (sparse files) or that contain zeroes are not copied
- Normal file deletion does not zero or deallocate blocks
- Swap partitions contain information if used
- This information can be stripped to make the images smaller
- **virt-sparsify to the rescue!**
Optimizing Images - virt-sparsify

There are two ways of doing sparsification:

- **Normal Sparsification:**
  - Occupies the maximum space of the image

- **In Place Sparsification:**
  - Create an sparse qcow2 file
Optimizing Images - Normal Sparsification

- Create overlay of the image
- Create a file in all filesystems and fill it with zeroes until there is not more space and delete file
- Fill swap partitions with zeroes
- Convert it to a new qcow2 file skipping zero blocks

```bash
$ TMPDIR=/var/tmp virt-sparsify original.qcow2 new.qcow2
```
Optimizing Images - In Place Sparsification

- Uses trim command, normally used for SSD disks
- Deallocates blocks from filesystem
- Does not require the total amount of disk space
- The qcow2 file contains holes and is not the best one for distribution
- Can be converted to a non sparse qcow2 file
- Can not be used with compression

```bash
$ virt-sparsify --in-place original.qcow2 new.qcow2
```
Optimizing Images - Compression

- qcow2 images can have the blocks compressed
- Compression rate is less that xz or bzip2
- Is more convenient as it can be directly used as is
- Use of these images trades disk space for CPU consumption
- Can be done directly in virt-sparsify with --compress (not In Place)
qemu-img tips

- There are two qcow2 file formats, pre version 0.10 and newer
  - CentOS 6 does not support post 0.10 version
  - On conversion or creation it can be specified with -o compat=0.10

- qemu-img < 2.4 does not support creation of delta images with compression
  - This tool can be easily compiled manually
  - Download qemu 2.4 code
  - ./configure
  - make qemu-img
Thank You!