virt_up

Release 2.1.0

Michael Meffie

Dec 28, 2021

CONTENTS:

`virt-up` is a command line tool to quickly create virtual machines local KVM hypervisor with `virt-builder` and `virt-sysprep`.

CHAPTER

ONE

INTRODUCTION

virt-up is a command line tool for creating virtual machines quickly on a libvirt based hypervisor. **virt-up** supports qemu/KVM and XEN virtualization.

virt-up runs the **libquestfs** tool **virt-builder** download (and cache) digitally signed virtual machine images. A *base virtual machine* is created from the downloaded image and is customized with **virt-sysprep**. Virtual machines are then cloned from the base virtual machine to quickly create new virtual machines.

virt-up automatically creates a login user and the ssh keys to connect to the new virtual machines. The login user is given sudo access. Connection information is stored in a json meta data file for each virtual machine created.

An ansible inventory file is created for the virtual machines to make it easier to run ansible playbooks for further configuration.

By default, **virt_up** will create image files in the **default** libvirt storage pool, e.g., `/var/lib/libvirt/images`. See the `pool` option Settings to change this. Be sure you have read and write access to the configured libvirt storage pool.

Normally, you want to run `virt_up` as a regular user, not root.

CHAPTER

TWO

USAGE

```
Usage: virt-up [OPTIONS] COMMAND [ARGS]...

Options:
  --version    Show the version and exit.
  -d, --debug
  -q, --quiet
  --help       Show this message and exit.

Commands:
  create    Create instances.
  destroy   Destroy instances.
  init      Initialize configuration files.
  list      List instances.
  login     Login to an instance.
  playbook  Run an ansible playbook on an instance.
  show      Show configuration information.
```

CHAPTER

THREE

INSTALLATION GUIDE

Verify your system supports virtualization. On Intel based systems, run `grep -c vmx /proc/cpuinfo` to verify the presence of the **vmx** flags. On AMD based systems, run `grep -c svm /proc/cpuinfo`. See KVM processor support for more information.

3.1 Debian/Ubuntu

This guide shows how to install KVM virtualization and **virt-up** on Debian and Ubuntu systems. Virtualization maybe installed on graphical desktop or a non-graphical server.

See Debian KVM for more information.

3.1.1 Installing KVM

Install virtualization packages with **apt**:

```
# apt install qemu-system libvirt-clients libvirt-daemon-system
```

When installing on a server, you can add the `--no-install-recommends` apt option, to prevent the installation of extraneous graphical packages:

```
# apt install --no-install-recommends qemu-system libvirt-clients libvirt-daemon-system
```

Install the **virt-install** and **libguestfs-tools** virtual machine management tools:

```
# apt install virtinst qemu-utils libguestfs-tools osinfo-db-tools
```

The graphical **virt-manager** tool is useful to have on a desktop system. If the kvm hypervisor is running on a server, you can install **virt-manager** on your desktop and connect to the server:

```
# apt install virt-manager
```

Add users to the **libvirt** group to grant them permission to manage virtual machines:

```
# adduser <username> libvirt
```

At this point, you should be able to login as a regular user and be able to create new guests with **virt-manager** and be able to manage the guests with **virsh**.

3.1.2 Installing virt-up

virt-up must be installed on the system running the KVM virtualization since it uses the **libguestfs** tools to prepare the virtual machine image files.

Install Python **pip**:

```
# apt install python3-pip
```

Install **virt-up** with Python **pip**. This can be installed as root for all users, or installed with **pip** as a regular user. If installed as a regular user, be sure `$HOME/.local/bin` is included in your `$PATH`:

```
$ pip3 install virt-up
```

Create the initial **virt-up** configuration files:

```
$ virt-up init config
```

The per-user configuration files are written to the directory `~/.config/virt-up/`. Set the `VIRTUP_CONFIG_HOME` environment variable to select an alternate location.

Run `virt-up show templates` to see the available template names.

Run `virt-up create <name> --template <name>` to create a virtual machine.

3.1.3 Linux kernel image permissions on Ubuntu

Linux images are not readable by regular users on Ubuntu distributions. This breaks the ability of libguestfs to modify guest images unless running as root.

Fix the kernel image permissions with the *dpkg-statoverride* command:

```
$ sudo dpkg-statoverride --update --add root root 0644 /boot/vmlinuz-$(uname -r)
```

To fix all of the installed images:

```
$ for i in /boot/vmlinuz-*; do sudo dpkg-statoverride --update --add root root 0644 $i;␣
↪done
```

To fix the permissions automatically with each new kernel version, create the file */etc/kernel/postinst.d/statoverride*:

```
#!/bin/sh
version="$1"
# passing the kernel version is required
[ -z "${version}" ] && exit 0
dpkg-statoverride --update --add root root 0644 /boot/vmlinuz-${version}
```

For more information see Ubuntu bug 759725.

CHAPTER

FOUR

CONFIGURATION

`virt-up` reads settings from INI formatted configuration files. The settings are divided into common settings and template definitions.

System defined configurations are located in the directory '/etc/virt-up'.

User defined configurations are located via a path set by an environment variable (see _virt_config_ below).

4.1 Common Settings

The `settings.cfg` file contains settings that are used when creating any virtual machine. The file should contain one section called `[common]`.

The following fields are supported:

pool The libvirt storage pool to write images. (default: `default`)

network The libvirt network, for example `bridge=br0`. (default: None)

username The username of the user account created by **virt-up** when creating new template instances (default: `virt`)

memory Instance memory, in KB. Default is 512.

vcpus Number of virtual cpus. Default is 1.

graphics Graphics type. Default is 1.

dns-domain The DNS domain used for new template instance hostnames. (default: None)

address-source The method used to detect the instance IP address. Supported values are:

- `agent` - Queries the qemu guest agent to obtain the IP address (`default`)
- `lease` - Parses the DHCP lease file to obtain the IP address (requires a libvirt managed DHCP server in the hypvervisor host)
- `arp` - Examines the arp table on the hypvervisor host
- `dns` - Uses the result of a DNS lookup for the guest host name.

image-format The image format. Supported values are `qcow2`, and `raw`. (default: `qcow2`)

virt-builder-args Extra arguments for `virt-builder`. (default: None)

virt-sysprep-args Extra arguments for `virt-sysprep`. (default: None)

virt-install-args Extra arguments for `virt-install`. (default: None)

template-playbook Optional ansible playbook to be executed on newly created template instances. (default: None)

instance-playbook Optional ansible playbook to be executed on newly created instances. (default: None)

These fields can be overridden by individual template definitions.

4.2 Template definitions

Template definitions are read from the files located in the `templates.d` sub-directory.

Provide one section for each template definition. The section name is the name for the template definition and is used for the **virt-up** `--template` option. The following fields are supported:

desc A text description, show by `--list-templates`.

os-version The **virt-builder** <os_version> name. See `virt-builder --list` for available names.

os-type The **virt-install** `--os-type`

os-variant The **virt-install** `--os-variant`. See `osquery-info os` for available names.

arch The target architecture.

memory Instance memory, in KB. Default is set in the common section.

vcpus Number of virtual cpus. Default is set in the common section.

graphics Graphics type. Default is set in the common section.

virt-builder-args Template specific extra arguments for `virt-builder`. (default: None)

virt-sysprep-args Template specific extra arguments for `virt-sysprep`. (default: None)

virt-install-args Template specific extra arguments for `virt-install`. (default: None)

template-playbook Optional ansible playbook to be executed on newly created template instances. (default: None)

instance-playbook Optional ansible playbook to be executed on newly created instances. (default: None)

In addition, the template configuration can override fields set in the `common` section of the settings.cfg file.

CHAPTER

FIVE

ENVIRONMENT VARIABLES

The following environment variables are used by `virt-up`.

LIBVIRT_DEFAULT_URI URI to access libvirt. Defaults to `qemu://session`

VIRTUP_CONFIG_HOME Path to `virt-up` configuration files. Defaults to `$XDG_CONFIG_HOME/virt-up`

XDG_CONFIG_HOME Path to `virt-up` configuration files. Defaults to the xdg standard location `$HOME/.local/share/virt-up`

VIRTUP_DATA_HOME Path to `virt-up` run-data files created by virt-up. Defaults to `$XDG_DATA_HOME/virt-up`

XDG_DATA_HOME Path to `virt-up` run-data files created by virt-up. Defaults to the xdg standard location `$HOME/.local/share/virt-up`

CHAPTER

SIX

FILES

The following files are created or referenced by **virt-up**

6.1 Configuration related

- /etc/virt-up/settings.cfg
- /etc/virt-up/templates.d/*
- /etc/virt-up/scripts/*
- /etc/virt-up/playbooks/*

The following override the files found in /etc/virt-up

- *virtup_config*/settings.cfg
- *virtup_config*/templates.d/*
- *virtup_config*/scripts/*
- *virtup_config*/playbooks/*

6.2 Runtime persistent data files

- *virtup_data*/sshkeys/ ``*name* ``
- *virtup_data*/macaddrs.json
- *virtup_data*/instance/ ``*name* ``.json
- *virtup_data*/inventory.yaml

6.3 Guest system image files

- *pool*/TEMPLATE-*template disk images*
- *pool*/*virtual guest disk images*

6.4 Transient runtime

- /var/run/user/*uid*/virt-up.lock If the above directory is not available
- /tmp/virt-up.lock

CHAPTER

SEVEN

OS INFO DATABASE

Operating system specific information is provided by the OS Info Database (`osinfo-db`) library. The OS Info Database provided by your package manager may be out of date and not provide definitions for recent operating system versions.

If you have already updated your system, and the osinfo-db is still to old, then you can use the `osinfo-db-import` tool with the `--local` option to install an up-to-date database in your home directory which will not conflict with your package manager installation. The `osinfo-db-import` tool is provided by the package name `osinfo-db-tools` on `yum` and `apt` managed systems.

Example:

```
$ wget https://releases.pagure.org/libosinfo/osinfo-db-<VERSION>.tar.xz
$ sudo osinfo-db-import --local osinfo-db-<VERSION>.tar.xz
```

See https://libosinfo.org/download for more information.

CHAPTER

EIGHT

XEN

virt-up can create and manage guests using the Xen hypervisor.

- To use a Xen hypervisor, set the LIBVIRT_DEFAULT_URI to use the xen system

  LIBVIRT_DEFAULT_URI=xen:///system

  and set `virt-install-args` to include '–hvm'.

  virt-install-args = '–hvm …'

- Xen does not support accessing guest information via the qemu-agent
- Some guest images are built with Xen support, but their device configurations are unloaded during initial boot processinmg. A boot parameter *xen_emul_unplug=never* must be added to the guest boot cmdline. This is usually done by updating the grub configuration when building the template.

  virt-builder-args = … –edit "/etc/default/grub:s/GRUB_CMDLINE_LINUX=""/GRUB_CMDLINE_LINUX="xen_emul
  –run-command 'grub-mkconfig -o /boot/grub/grub.cfg' …

CHAPTER

NINE

LICENSE

CHAPTER

TEN

INDICES AND TABLES

- genindex
- modindex
- search