Linux VM Provisioning

Here are some steps that need to be done when creating a VM template that will be used multiple times:

Template Usage

See this page to setup a deployed instance from a template VM: Ubuntu Host Setup

References

Here’s a good reference for template generation and cloud-init usage:

https://medium.com/@dsykes012/making-a-custom-ubuntu-20-04-lts-focal-fossa-vm-template-that-works-with-cloud-init-2cfffb6783b4

Important Notes

Anytime you power on a Linux VM template, the VM thinks it is a spawned instance.
And so, it will create new values, such as Machine-ID, SSH host keys, etc.
This is the normal functionality of a spawned clone.

However. If you are simply powering up a template VM, to make changes or updates to the template, you will need to rerun the reset script, to clear out any set values (due to VM startup).
To do this, follow the instructions at the bottom of this page, link here:

https://wiki.galaxydump.com/link/391#bkmrk-now.-each-time-you-s

> Warning: Forgetting to run the reset script, before shutting down the template VM, will result in clones having similar properties.

Host Template Setup

Here are steps for creating a golden template VM.

Create Template VM

Create a VM guest instance with minimal CPU, memory, and hard disk space for the template.

We go with minimal resources, as it’s easier to script additional resources, than it is to reduce them.

For example, it’s much faster and less risk to expand a drive than reducing one.

Assign the VM’s NIC to the provisioning VLAN, as this allows us to easily access and setup deployed clones.

DHCP Addressing

Setup the template with DHCP, so we don’t worry about address collisions when spawning multiple clones at once.

Perform Updates and Upgrades

To speed up the process of clone setup, we will do any updates and upgrades on the template.
Do this:

```
apt update && apt -y upgrade && apt -y autoremove && apt clean
```

SSH Server

Install the openssh server:

```
sudo apt install openssh-server
```

Verify it's running with this:

```
sudo systemctl status ssh
```

Configure it to listen on port 22 of all adapters.

Disable password authentication over SSH.

Firewall

Setup the ufw firewall rules with 22 as allowed:

```
sudo ufw allow ssh
```

Don’t enable the firewall on the template. We can script this as use cases require.

Switching to Root

In order to perform most of these setup tasks, you should be running as root.

To switch to root (from an existing user), use this:

```
sudo -i
Or
sudo -
```

Provisioning User Account

In order for us to manage and deploy to the host, it needs a user account.

Follow the instructions on this page, to create the user account and setup access to it: Managed Host User Setup

User Cleanup

Log into the template VM as root.

This allows us to scrub unnecessary users and groups from the system.

Reset Hostname

Every host needs a unique name in your network.

So, we will clear an default the hostname of a template.
The following will erase it:

```
truncate -s0 /etc/hostname
```

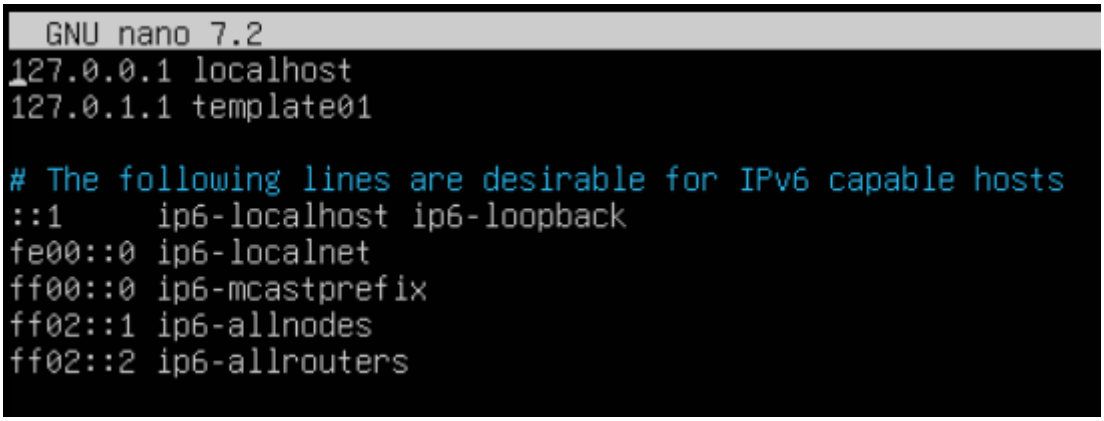
And, this will set it to 'localhost':

```
hostnamectl set-hostname template01
```

> NOTE: The above actions are performed by the reset script (bottom of this page).

Next, we need to clear the hostname in the /etc/hosts file.

Open /etc/hosts, and set the entry (127.0.1.1) to 'template01', like this:



Save and close the hosts file.

> WARNING: This last step, to update the hosts file is NOT performed by the reset script (bottom of this page).
> You will need to perform the above step, manually, when creating the template.

Clear the Machine-Id

Machine-Id is a string value that uniquely identifies a linux host, similar to how a Windows VM is uniquely identified by its machine sid.

Linux stores the machine-id is stored in a file at: /etc/machine-id.

So. If you are cloning VMs, you need to make sure this value is different for each VM clone.
The following will erase the machine-id, and any symbolic links to it:

```
truncate -s0 /etc/machine-id
rm /var/lib/dbus/machine-id
ln -s /etc/machine-id /var/lib/dbus/machine-id
```

The next time the VM reboots, the OS will generate a new machine-id during the boot process.

NOTE: Each time you boot the template, creates a new machine Id.
So, you will need to re-execute the above, after each time you boot the template VM for updates or changes.

NOTE: The above actions are performed by the reset script (bottom of this page).

Reset Host SSH Keypairs

Every linux host holds a set of SSH keypairs that identify the host, and allow SSH connections with it.

For some hosts, there may be more than one keypair, each supporting a different encryption type: RSA, ECDSA, ed25519, etc...

These host SSH keys need to be unique for every host.
But when cloning a VM, these keys get duplicated.
So, we need a way for the linux template to reset its host keypairs. And for the clone to create a new set when it first boots.

The easiest way, is for us to delete the SSH keys from the template.
And, to create a one-shot service that will create a new set on startup.

On the template, we need to delete all host SSH keys that it possesses, with this:

```
sudo rm -f /etc/ssh/ssh_host_*_key /etc/ssh/ssh_host_*_key.pub
```

NOTE: The above action is performed by the reset script (bottom of this page).

For a clone of the template to recreate host SSH keys, we need to create a new systemd daemon, that will create the new keys on startup, only if they don't yet exist.

We do so, by creating a systemd unit file, with this:

```
sudo nano /etc/systemd/system/ssh-hostkeys-generate.service
```

And, we populate the service unit file with this:

```
[Unit]
Description=Generate SSH host keys if missing
DefaultDependencies=no
After=local-fs.target
```

```
Before=sshd.service ssh.service
ConditionPathExistsGlob=!/etc/ssh/ssh_host_*_key

[Service]
Type=oneshot
ExecStart=/usr/bin/ssh-keygen -A
# If you only want ed25519 + ecdsa, comment the line above and use:
# ExecStart=/bin/sh -c '/usr/bin/ssh-keygen -t ed25519 -N "" -f /etc/ssh/ssh_host_ed25519_key; \
#                       /usr/bin/ssh-keygen -t ecdsa    -b 256 -N "" -f /etc/ssh/ssh_host_ecdsa_key'

[Install]
WantedBy=multi-user.target
```

The above service definition includes a check if any SSH keypairs exist in Line 6: `ConditionPathExistsGlob=!/etc/ssh/ssh_host_*_key`

This prevents the service from running more than once.

Once the systemd file is created, we need to reload the daemon and enable the service, with these:

```
sudo systemctl daemon-reload
sudo systemctl enable ssh-hostkeys-generate.service
```

Template Reset Script

Here's the definition of the template reset script that is used to clean up a linux template VM, for cloning.

The reset script will:

- Reset the machine-id (same as described above)
- Clear the host SSH keypairs
- Clear the hostid
- Clear the hostname
- Clear the hostname in /etc/hosts
- Clear existing DHCP leases
- Reset the system entropy seed
- Clear logs and history
- Clear user traces
- Clear package caches
- Clear any Docker clientid
- Remove any Docker cached layers

The script should be stored, here: /etc/template/reset.sh

To create it, first, create the template folder, with this:

```
sudo mkdir /etc/template
```

Once the folder exists, you can create the script file, with this:

```
sudo nano /etc/template/reset.sh
```

And, populate it with this code:

```
#!/bin/bash

# Name:          Ubuntu 24.04 Template Reset Script.
# Version:       1.0
# Date:          20250816
# See this page: https://wiki.galaxydump.com/link/391
# Description:   This script Will clear hostnames, leases, ids, keys, and other properties,
#                  before shutting down the template.
#               This allows a clone to be made, and started up, with its own unique Ids, keys, etc.

set -euo pipefail

echo "Clearing host SSH keys..."
rm -f /etc/ssh/ssh_host_*_key /etc/ssh/ssh_host_*_key.pub

echo "Clearing host machine-id..."
truncate -s0 /etc/machine-id
rm /var/lib/dbus/machine-id
ln -s /etc/machine-id /var/lib/dbus/machine-id

echo "Clearing host machine-id..."
# systemd will recreate on boot; or: dd if=/dev/urandom of=/etc/hostid bs=4 count=1
# 20250816 Update: We've disabled the removal of the hostid file, on Ubuntu24, as this distro does
#     NOT recreate the file on startup.
# As well, the hostid file is only used by some legacy apps/libraries.
# So, if a truly unique hostid is required, we will have to create a one-shot systemd service that
#     will generate it on first-boot of a clone.
#rm -f /etc/hostid || true
```

```
echo "Clearing hostname..."
truncate -s0 /etc/hostname
hostnamectl set-hostname localhost

echo "Clear Existing DHCP Leases..."
# Debian/Ubuntu dhclient
rm -f /var/lib/dhcp/* || true
# NetworkManager leases
rm -f /var/lib/NetworkManager/*lease* || true
# systemd-networkd leases
rm -f /var/lib/systemd/network/* || true

echo "Let systemd reseed entropy on first-boot..."
rm -f /var/lib/systemd/random-seed || true

echo "Clear logs and history..."
journalctl --rotate || true
journalctl --vacuum-time=1s || true
rm -f /var/log/wtmp /var/log/btmp || true
: | tee /var/log/lastlog >/dev/null
find /var/log -type f -name '*.log' -exec truncate -s0 {} +

echo "Clear User Traces..."
rm -f /root/.bash_history || true
find /home -maxdepth 2 -name .bash_history -exec rm -f {} + || true

echo "Clear Package Caches..."
# Debian/Ubuntu
apt-get clean || true;
# RHEL/CentOS/Alma/Rocky
#dnf clean all || yum clean all

echo "Clear any Docker ClientId..."
#Container runtime: if templating with Docker installed:
# Remove the client ID file...
rm -f /etc/docker/key.json
# Remove cached layers.
rm -f /var/lib/docker

echo "Template sealed. Power off, convert to template, and ensure first-boot key regen is enabled."
```

Save and close the script file.

Now, make sure it's executable, with this:

```
sudo chmod +x /etc/template/reset.sh
```

Now. Each time you shutdown the template VM, you can run the above script, to reset needed info, for cloning.

The script can be executed, with this:

```
sudo /etc/template/reset.sh
```

Revision #31
Created 3 August 2025 16:36:38 by glwhite
Updated 1 September 2025 05:45:17 by glwhite