A homelab rebuildable at any time

Fabio Alessandro “Fale” Locati

October 24, 2026

Senior Principal Architect, Red Hat

What we will cover

The problem

My homelab expectations

The operating system

Services

Configuration

Secrets

Network

Putting it together

Wrapping up

The problem

How a homelab gets out of hand

- It usually starts with one machine and a handful of commands.
- A few months later, you no longer remember all the changes you made.
- Devices all collect little bits of unique configuration.
- Updates get scary, and fixing a broken host means digging through history.
- Replacing the machine becomes a new project every time.

About me

- I have been working in IT since 2004, mostly in operations.
- I have been contributing to Fedora since 2010.
- I started using Ansible in 2013.
- I have been using immutable Linux since 2016.
- I work as a Senior Principal Architect at Red Hat.

My homelab expectations

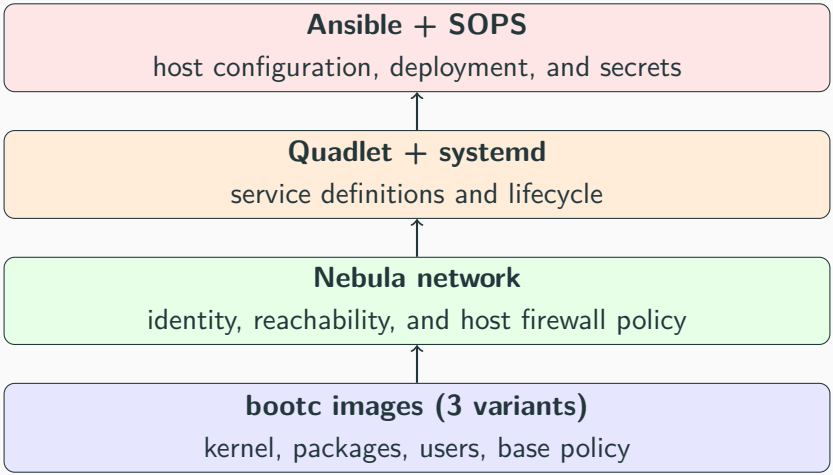
What I call a homelab

- In this talk, “homelab” covers more than a couple of boxes on a shelf.
- It includes NAS devices, routers, physical machines, virtual machines, and laptops.
- For a total of 12 systems in at least 6 different buildings, in 3 different countries.
- The laptop I am using for this presentation is managed this way.
- From my prospective, it is production, not a lab.

What I want

- Keep the description of each machine in git.
- Build the operating system from a container image.
- Run services with systemd, using Quadlet to describe the containers.
- Let Ansible put the machine-specific pieces in place.
- Keep secrets encrypted.
- Make the network work beyond my home LAN.

How I logically split the problem



The operating system

Why bootc?

- It is an OCI image, but boots as a complete operating system.
- I can use the same build and registry tools I already use for containers.
- An update swaps the whole OS at reboot.
- If an update is bad, the previous deployment is still there.
- Greenbot is able to auto-detect failed updates and rollback.

Why three images?

- Most of the setup is common to all my machines.
- Different base packages (e.g.: desktop environments) require new images.
- Different users configuration (e.g.: interactive login) require new images.

A small bootc image

```
FROM quay.io/fedora/fedora-bootc:latest

RUN dnf -y install podman quadlet nebula git
RUN dnf clean all

COPY etc/systemd/system/ /etc/systemd/system/
COPY usr/lib/sysusers.d/ /usr/lib/sysusers.d/
```

- Packages and base policies go into the image.
- Application data lives outside the immutable OS.

What an OS update looks like

1. Build the new image version (via Ansible).
2. Push it to the registry (via Ansible).

1. Tell the hosts to download the new version (via Ansible).
2. Reboot (via Ansible).
3. If something breaks, rollback (via Greenboot).
4. Check if services are up. If not, rollback (via Ansible).

Services

How I run services

- Podman runs the containers.
- systemd supervises the service.
- systemd starts, stops, restarts services.
- Quadlet allows me to have small declarative file.
- Journald manages the logs like any other systemd unit.

A Quadlet service

```
# /etc/containers/systemd/forgejo.container
[Unit]
Description=Forgejo Server
Requires=local-fs.target dmz.network
After=local-fs.target dmz.network

[Service]
Restart=always

[Install]
WantedBy=default.target

[Container]
ContainerName=forgejo
Network=dmz.network
Image=codeberg.org/forgejo/forgejo:17.0.0
PublishPort=2233:22
Volume=/home/forgejo:/data:Z
```

What goes in a Quadlet file?

- The image and the way it should run.
- Volumes, ports, networks, and dependencies.
- Restart behavior.
- Labels for specific configurations.
- Secrets are handled as environmental variables.

Configuration

Where Ansible fits

- Ansible is used to build the base system images.
- Ansible is used to trigger the base system images updates.
- Ansible manages Nebula certificates and configuration after bootstrap.
- Ansible installs Quadlet files, create directories, etc. for containerized services.
- All playbooks work on a newly provisioned system.
- All playbooks are fully idempotent.

A deliberately boring deployment task

```
- name: Install the service definition
  ansible.builtin.copy:
    src: homepage.container
    dest: /etc/containers/systemd/forgejo.container
    owner: root
    mode: '0644'
  notify: Reload systemd

- name: Ensure the data directory exists
  ansible.builtin.file:
    path: /home/forgejo
    state: directory
    owner: fale
    mode: '0750'
```

Secrets

Secrets stay encrypted

- Git can hold SOPS configuration.
- SOPS encrypts the files while keeping them usable from Ansible.
- SOPS uses GPG keys (living on a YubiKey).

An encrypted secret at deployment time

```
# .sops.yaml
creation_rules:
  - path_regex: inventory/(host|group)_vars/.*\.yaml$
    encrypted_regex: '^(.*_password|.*_secret|.*_key|.*_token)$'
    pgp: D880521E8D13FB5D0BCFC65817DBA3103879E9D9
```

```
# inventory/group_vars/git.sops.yml
psql_user: postgres
psql_password: ENC[AES256_GCM,data:...,iv:...,tag:...,type:str]
psql_db: postgres
```

Network

The network is part of the system

- Nebula is a Wireguard based mesh VPN.
- A lighthouse helps peers find each other.
- Every host has a certificate and explicit firewall policies.
- After the first bootstrap, Ansible owns the certificates and config.
- The traffic stays peer-to-peer, preferring local paths.
- Nebula traffic is rarely filtered even in public and corporate networks.

Putting it together

What is in the repository?

```
falesys/
|-- bootc/
|   |-- base/
|   |   |-- Containerfile
|   |   `-- etc/
|   |-- server/
|   |   |-- Containerfile
|   |   `-- etc/
|   `-- laptop/
|       |-- Containerfile
|       `-- etc/
`-- ansible/
    |-- main.yaml
    |-- inventory/
    `-- roles/
        |-- nebula/
        |   |-- files/
        |   `-- tasks/
        `-- forgejo/
            |-- files/
            `-- tasks/
```

Replacing a machine

1. Install one of the bootc image variants.
2. Give the new host its Nebula identity.
3. Run Ansible against it.
4. Restore the application data from backup.
5. Check that all services are up and running.

Backups are a different conversation

- Many applications need data on top of the configurations files.
- Backups are automatically done.
- Backups are automatically moved to other machines to prevent hardware-related issues.
- Backups automation, copy, and rotation is configured via Ansible.

Wrapping up

Wrapping up

- I treat all systems in the same way.
- bootc gives me versioned operating system images.
- Quadlet and systemd run the services in a very native way.
- Ansible installs the pieces and keeps hosts in line.
- SOPS protects secrets
- Nebula keeps the machines reachable.

Questions?

Email: mail@fale.io
Fediverse: @fale@fale.io



Links

- https://bootc-dev.github.io/bootc/
- https://docs.podman.io/en/latest/markdown/podman-systemd.unit.5.html
- https://docs.ansible.com/ansible/latest/
- https://github.com/getsops/sops
- https://nebula.defined.net/