

# Fedora Remix Lab

---

A collection of scripts to create and manage a Fedora-based virtual lab environment using KVM/libvirt. This lab creates two VMs that can communicate with each other, perfect for testing Ansible automation, clustering, or multi-node applications.

 **Using Fedora Remix Lab ISO?** See the [Quick Start Guide](#) for streamlined instructions when the lab is pre-installed.

## Prerequisites

---

Install the required packages:

```
sudo dnf install qemu-kvm libvirt virt-manager libguestfs-tools virt-viewer
```

Ensure libvirtd is running:

```
sudo systemctl enable --now libvirtd
```

## Quick Start

---

1. Download the base image (or place your own `Fedora43Lab.qcow2` in this directory):

```
./download-image.sh
```

2. Create the lab environment:

```
sudo ./create-lab-vms.sh
```

3. Start the VMs:

```
sudo ./start-lab-vms.sh
```

4. Add host entries to your local machine:

```
sudo ./manage-hosts.sh add
```

5. Check status:

```
sudo ./lab-status.sh
```

6. Connect to a VM:

```
sudo virt-viewer FedoraLab1
```

Or via SSH:

```
ssh ansibleuser@fedoralab1.example.com
```

## Lab Configuration

---

### Network

Setting	Value
Network Name	labnet
Subnet	192.168.100.0/24
Gateway	192.168.100.1
Domain	example.com

## Virtual Machines

VM Name	IP Address	FQDN	Hostname
FedoraLab1	192.168.100.10	federalab1.example.com	federalab1
FedoraLab2	192.168.100.11	federalab2.example.com	federalab2

## VM Specifications

Resource	Value
Memory	1 GB
vCPUs	2
Disk	Overlay on base image

## User Account

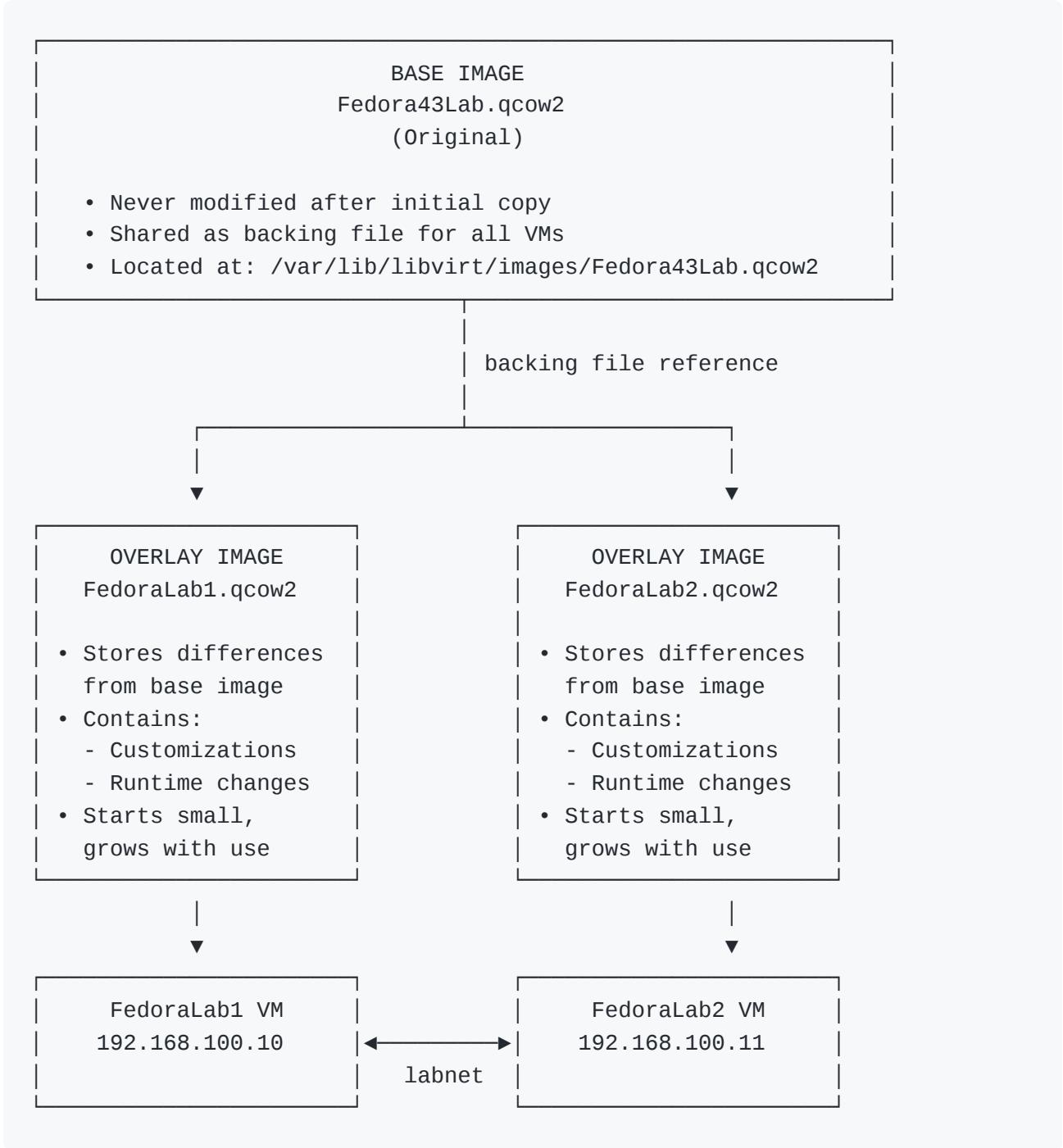
Setting	Value
Username	ansibleuser
Password	Automation!
Sudo	Passwordless ( NOPASSWD: ALL )

## How It Works

---

### Disk Image Architecture

The lab uses a **copy-on-write overlay** architecture to efficiently manage disk images. This approach keeps the base image pristine while allowing each VM to have its own independent storage.



## Creation Process

### 1. Copy base image to libvirt storage:

```
Fedora43Lab.qcow2 → /var/lib/libvirt/images/Fedora43Lab.qcow2
```

### 2. Create overlay images (copy-on-write):

```
qemu-img create -f qcow2 -b Fedora43Lab.qcow2 -F qcow2  
FedoraLab1.qcow2
```

The overlay references the base image and only stores blocks that differ.

### 3. Customize overlays using `virt-customize` :

- Create user account (`ansibleuser`)
- Set hostname, locale, timezone
- Configure `/etc/hosts`
- Disable initial-setup wizard
- SELinux relabel

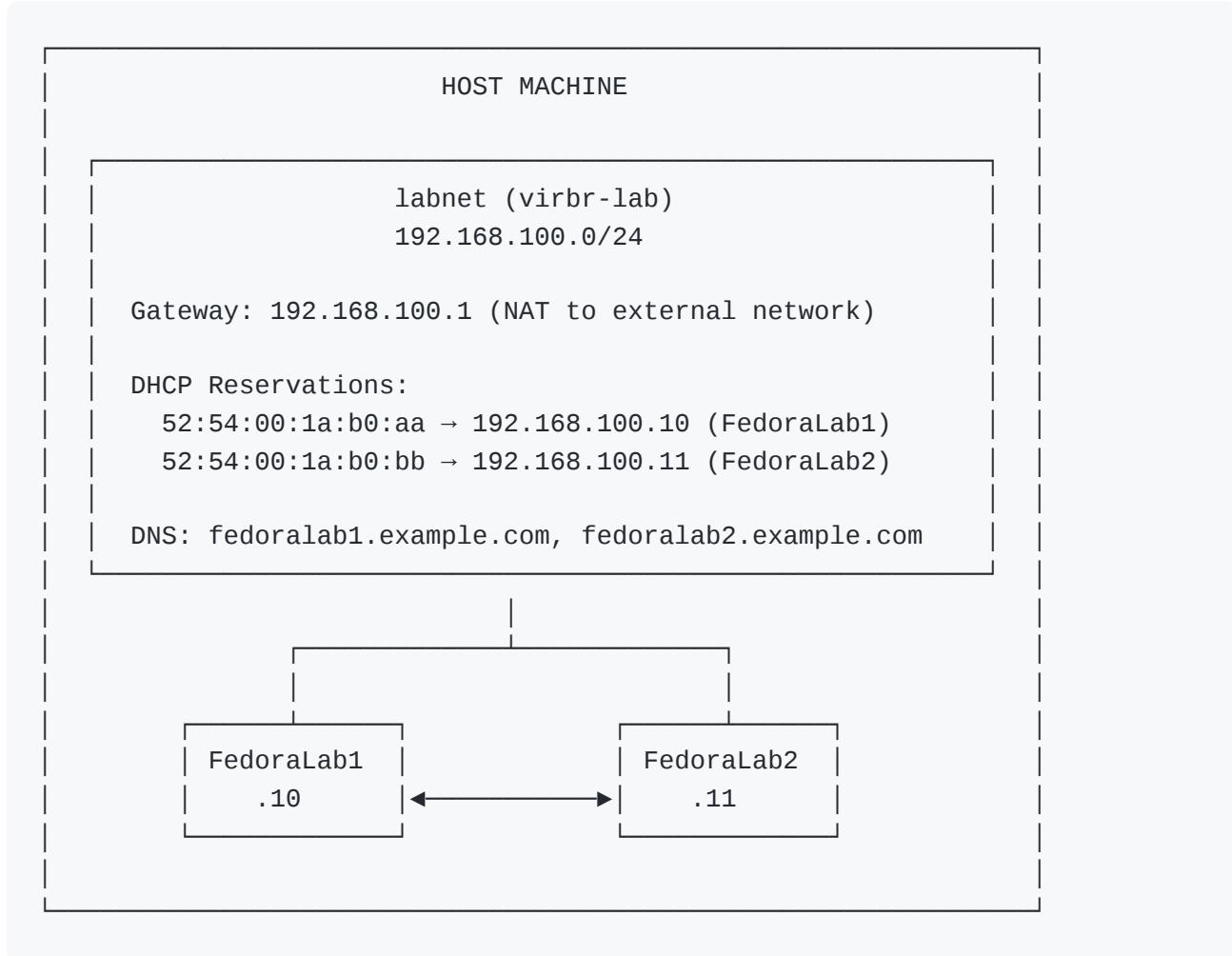
### 4. Generate libvirt XML definitions for each VM with:

- Static MAC addresses (for DHCP reservation)
- Network attachment to `labnet`
- UEFI firmware, TPM, etc.

## Benefits of This Architecture

Benefit	Description
<b>Disk Efficient</b>	Overlays only store differences (~500MB vs 10GB full copy)
<b>Fast Creation</b>	No need to copy the entire base image for each VM
<b>Base Image Preserved</b>	Original image never modified; easy to recreate VMs
<b>Independent VMs</b>	Changes in one VM don't affect the other
<b>Easy Reset</b>	Delete overlay and recreate for a fresh VM

# Network Architecture



The VMs use static IPs assigned via DHCP reservations based on MAC address. This ensures consistent IP addresses across reboots while still using DHCP for network configuration.

## Scripts

**Note:** Most scripts require `sudo` to interact with libvirt system domains.

### download-image.sh

Downloads the Fedora43Lab.qcow2 base image from Google Drive to  
`/var/lib/libvirt/images/`.

```
sudo ./download-image.sh
```

- Downloads directly to libvirt's standard image location
- Sets proper ownership ( `qemu:qemu` ) and permissions
- Uses `gdown` for reliable large file downloads from Google Drive
- Offers to install `gdown` if not present
- Requires `sudo` to write to `/var/lib/libvirt/images/`

## create-lab-vms.sh

Creates the complete lab environment including:

- Virtual network ( `labnet` ) with static IP assignments
- Overlay disk images for each VM (preserving the base image)
- Pre-configured VMs with user accounts, locale, and `/etc/hosts`
- XML definitions for libvirt

```
sudo ./create-lab-vms.sh
```

### What it configures on each VM:

- Hostname (FQDN)
- Timezone (America/New\_York)
- Locale (en\_US.UTF-8)
- User account with passwordless sudo
- `/etc/hosts` entries for all lab VMs
- Disables initial-setup wizard

## start-lab-vms.sh

Registers and starts the lab VMs. Ensures the lab network is active.

```
sudo ./start-lab-vms.sh
```

## lab-status.sh

Displays comprehensive status of the lab environment:

- Network status and DHCP leases
- VM states (running, stopped, not defined)
- IP addresses and hostnames
- Connectivity checks (ping and SSH)
- Host configuration status

```
sudo ./lab-status.sh
```

## manage-hosts.sh

Manages `/etc/hosts` entries on the host machine for easy VM access by hostname.

```
# Add entries
sudo ./manage-hosts.sh add

# Remove entries
sudo ./manage-hosts.sh remove

# Update entries (remove and re-add)
sudo ./manage-hosts.sh update

# Check status (can run without sudo)
./manage-hosts.sh status
```

## reset-lab.sh

Destroys and recreates the entire lab environment. Useful for starting fresh.

```
# Full reset with confirmation prompt
sudo ./reset-lab.sh

# Full reset without confirmation
sudo ./reset-lab.sh -y

# Reset only VMs, keep the network
sudo ./reset-lab.sh --vms-only

# Only destroy, don't recreate
sudo ./reset-lab.sh --destroy-only
```

## Options:

Option	Description
--full	Full reset including network (default)
--vms-only	Reset only VMs, keep the network intact
--destroy-only	Only destroy, don't recreate
-y, --yes	Skip confirmation prompt

## Common Tasks

---

### Start the Lab

```
sudo ./start-lab-vms.sh
```

### Stop the Lab

```
sudo virsh shutdown FedoraLab1
sudo virsh shutdown FedoraLab2
```

Or force stop:

```
sudo virsh destroy FedoraLab1  
sudo virsh destroy FedoraLab2
```

## Connect to VMs

Via Virt-Viewer (graphical console):

```
sudo virt-viewer FedoraLab1  
sudo virt-viewer FedoraLab2
```

Via SSH:

```
ssh ansibleuser@fedoralab1.example.com  
ssh ansibleuser@fedoralab2.example.com
```

Or by IP:

```
ssh ansibleuser@192.168.100.10  
ssh ansibleuser@192.168.100.11
```

## Reset the Lab

Use the reset script to completely recreate the lab (destroys all VM data):

```
# Full reset (with confirmation)  
sudo ./reset-lab.sh  
  
# Full reset (skip confirmation)  
sudo ./reset-lab.sh -y  
  
# Reset VMs only, keep network  
sudo ./reset-lab.sh --vms-only
```

Or manually:

```
# Stop VMs
sudo virsh destroy FedoraLab1 2>/dev/null
sudo virsh destroy FedoraLab2 2>/dev/null

# Undefine VMs
sudo virsh undefine FedoraLab1 --nvram
sudo virsh undefine FedoraLab2 --nvram

# Remove network
sudo virsh net-destroy labnet
sudo virsh net-undefine labnet

# Remove files
sudo rm -rf /var/lib/libvirt/images/fedora-lab

# Recreate
sudo ./create-lab-vms.sh
sudo ./start-lab-vms.sh
```

## Check VM Connectivity

From the host:

```
ping fedoralab1.example.com
ping fedoralab2.example.com
```

From FedoraLab1 to FedoraLab2:

```
ssh ansibleuser@fedoralab1.example.com
ping fedoralab2.example.com
ssh fedoralab2
```

# File Locations

---

File	Location
Base image	/var/lib/libvirt/images/Fedora43Lab.qcow2
VM overlay disks	/var/lib/libvirt/images/fedora-lab/*.qcow2
VM XML definitions	/var/lib/libvirt/images/fedora-lab/*.xml
Network XML	/var/lib/libvirt/images/fedora-lab/labnet.xml
Local hosts file	./hosts.local
Ansible inventory	./inventory

# Troubleshooting

---

## VMs show “not defined”

Run `sudo ./start-lab-vms.sh` to register and start the VMs.

## Network not found

Run `sudo ./create-lab-vms.sh` to create the network.

## Permission denied errors

All scripts must be run with `sudo` because the VMs use libvirt's system connection.

## Can't ping VMs from host

Ensure the lab network is active:

```
sudo virsh net-start labnet
```

Ensure `/etc/hosts` entries are added:

```
sudo ./manage-hosts.sh add
```

## SSH connection refused

The VM may still be booting. Wait a moment and try again, or check status:

```
sudo ./lab-status.sh
```

## Initial setup screen appears

If you see the Fedora initial setup screen, the virt-customize step may have failed. Recreate the VMs:

```
sudo ./create-lab-vms.sh
```

## Using with Ansible

---

The lab is designed for Ansible automation testing. Create an inventory file:

```
[lab]
fedoralab1.example.com
fedoralab2.example.com

[lab:vars]
ansible_user=ansibleuser
ansible_become=yes
```

Test connectivity:

```
ansible -i inventory lab -m ping
```

## License

---

This project is part of the Fedora Remix project.