**Jelly Concepcion**

Aspiring Cyber Defense Incident Responder | Future Chief Information Security Officer (CISO) | Passionate About Cybersecurity & Threat Detection

Email: jellydizonconcepcion@gmail.com | GitHub: https://github.com/jellyconcepcion/blue-team-homelab | LinkedIn: https://www.linkedin.com/in/jellyconcepcion/

---

**Introduction:**

This document provides a step-by-step guide to recreate the Kali Linux VM used in my Blue Team lab environment (Month 1). It covers VM setup, hostname configuration, network configuration, user setup, and essential verification commands. Screenshots and evidence files are included to ensure reproducibility.

**Goal:** Build a Kali Linux attacker VM for testing and interacting with lab endpoints (Ubuntu SIEM and Windows) while maintaining consistent host-only IPs and lab integrity.

**Audience:** Cybersecurity learners, red teamers, or penetration testers who want a reproducible lab setup to practice safely in an isolated environment.

---

Create a Kali VM using kali-linux-2025.2-virtualbox-amd64.vbox and kali-linux-2025.2-virtualbox-amd64.vdi.

- On Oracle VirtualBox, go to Machine → Add… → select kali-linux-2025.2-virtualbox-amd64.vbox
- After adding the .vbox, go to Settings → General → Basic → Rename it to `LAB-KALI-01`. Screenshot: `blue-team-homelab/01-lab-setup/03-kali-attacker/screenshots/kali-vm`-settings-general.png
- **Boot order:** Leave it as it is (Hard Disk first). You only need Optical first if you're installing from ISO. Since you're attaching a .vdi, the VM should boot straight into Kali. **EFI:** Keep EFI **disabled**. Kali boots fine in BIOS/Legacy mode, and you don't need EFI. Screenshot: kali-vm-settings-system.png

- **`Controller: SATA with .vdi attached`** `→ That's all you need. That .vdi is your Kali hard disk, and it should already have the OS installed. Screenshot:` kali-vm-settings-storage.png
- `Network: enable both:` **`Adapter 1 = NAT`** `(for internet access / updates) and` **`Adapter 2 = Host-only Adapter (VirtualBox Host-Only Ethernet Adapter)`** `(for isolated lab communication). Screenshot:` kali-vm-settings-adapter1.png and kali-vm-settings-adapter2.png
- **Before starting the Kali VM, screenshot the summary of Kali VM on vbox VMs List:** kali-vm-overall-setup-summary.png

---

## Step-by-step (run → verify → screenshot)

### 1) Boot & login

1. Start `LAB-KALI-01` in VirtualBox.
2. Login at the Kali login prompt with `kali` / `kali`.

**Screenshot (immediately after login):**

- **File:** `01-lab-setup/03-kali-attacker/screenshots/kali-first-login.png`
- **What to capture:** the VM window showing the Kali desktop (or login prompt) — include the VirtualBox title bar so it's obvious this is the VM.

---

### 2) Set the hostname (so it matches your naming standard)

Find the Terminal Emulator on the upper left. In a terminal run:

```
sudo hostnamectl set-hostname kali-lab-01
```

Confirm with:

```
hostnamectl
```

**Screenshot:**

- **File:** `01-lab-setup/03-kali-attacker/screenshots/kali-hostnamectl.png`
- **What to capture:** the `hostnamectl` output that clearly shows `Static hostname: kali-lab-01` and OS info. Make sure the terminal window and the output are readable.

---

### Identify NIC names and check current IPs

Run:

```
ip a
```

Find the interface name for the Host-Only adapter — it will usually be `eth1` or similar. Note which interface corresponds to Host-Only (it will not have an Internet IPv4 yet).

**Screenshot:**

- **File:** `01-lab-setup/03-kali-attacker/screenshots/kali-ip-a-before.png`
- **What to capture:** the `ip a` output (full screen of the terminal). Ensure the interface names are visible.

---

Before configuring the static ip, Edit `/etc/hosts` and add the mapping for your hostname:

```
sudo nano /etc/hosts
```

Make sure you have a line like this:

```
127.0.0.1   localhost
127.0.1.1   kali-lab-01
```

If you see the old hostname (`kali`), just replace it with `kali-lab-01`.

Save & exit (CTRL+O, ENTER, CTRL+X in nano).

Screenshot: **File name:**
`01-lab-setup/03-kali-attacker/screenshots/kali-etc-hosts-fix.png`

**What to capture:**
Show the `/etc/hosts` file open in `nano` (or after `cat /etc/hosts`) where `127.0.1.1 kali-lab-01` is clearly visible.

---

You need to give `eth1` the static IPv4 `192.168.56.30/24` (no gateway).

Run this (exactly):

```
# Create a persistent connection for eth1
sudo nmcli con add type ethernet ifname eth1 con-name hostonly-eth1
autoconnect yes ipv4.addresses 192.168.56.30/24 ipv4.method manual

# Ensure no gateway
sudo nmcli con modify hostonly-eth1 ipv4.gateway ""

# Set DNS if needed (optional)
sudo nmcli con modify hostonly-eth1 ipv4.dns "8.8.8.8"

# Activate it
sudo nmcli con up hostonly-eth1
```

## Screenshots you should take

- `nmcli con add` + `nmcli con up` (showing success)
  →
  `01-lab-setup/03-kali-attacker/screenshots/kali-nmcli-set-ip.png`

---

Then verify:

```
ip a          # eth1 should now show 192.168.56.30/24
ip route      # should NOT list a default route for eth1
```

### Screenshots you should take

- `hostnamectl` + `ip a` in one shot (eth1 with 192.168.56.30 visible)
  →
  `01-lab-setup/03-kali-attacker/screenshots/kali-hostname-ip-route.png`

---

Connectivity Test:

```
ping -c 2 kali-lab-01
ping -c 4 192.168.56.10   # Wazuh
ping -c 4 192.168.56.20   # Windows
```

### Screenshots you should take

Split them up for clarity (easier to reference later):

1. **Ping self (hostname resolution):**
   - **File name:**

     `01-lab-setup/03-kali-attacker/screenshots/kali-ping-self.png`

   - **What to capture:**
     `ping -c 2 kali-lab-01` output showing successful replies.

2. **Ping Wazuh (Ubuntu at 192.168.56.10):**
   - **File name:**

     `01-lab-setup/03-kali-attacker/screenshots/kali-ping-wazuh.png`

3. **Ping Windows (192.168.56.20):**
   - **File name:**

`01-lab-setup/03-kali-attacker/screenshots/kali-ping-win11.png`

---

**Creating a new user account (`labkali`), making it an admin, then deleting the default `kali` user is easier and safer than renaming.**

## 1) Create new user

```
sudo adduser labkali
```

- Password: `Dream!110100`
- Press **Enter** for all the extra info (Full Name, etc.) unless you want to fill them in.

**Screenshot:**

- `01-lab-setup/03-kali-attacker/screenshots/kali-adduser.png`
- Show the terminal output "Adding user `labkali` … Done".

---

## 2) Give `labkali` sudo/admin rights

```
sudo usermod -aG sudo labkali
```

Verify:

```
groups labkali
```

Output should include `sudo`.

**Screenshot:**

- `01-lab-setup/03-kali-attacker/screenshots/kali-user-sudo.png`
- Show `groups labkali` output with `sudo`.

---

## 3) Log in as `labkali`

1. Log out from the GUI.
2. On the login screen, you should now see `labkali`.
3. Log in with password `Dream!110100`.

**Screenshot:**

- `01-lab-setup/03-kali-attacker/screenshots/kali-labkali-login.png`
- Show login screen or desktop logged in as `labkali`.

---

To avoid encountering this error “userdel: user kali is currently used by process 1040.”:

**Reboot and log in only as `labkali`**

1. Log out of `kali` completely.
2. Reboot the VM.
3. At the login screen, log in as **`labkali`**.
    - Don’t log in as `kali`.

Open a terminal and run:

```
sudo deluser --remove-home kali
```

4. This time it should succeed because no `kali` processes are running.

**Screenshot:**

- `01-lab-setup/03-kali-attacker/screenshots/kali-deluser-kali.png`
- Show terminal where it says `Removing user 'kali' … Done`.

---

## Final evidence

Run:

```
whoami
hostnamectl
```

**Screenshot:**

- `01-lab-setup/03-kali-attacker/screenshots/kali-whoami-hostname.png`
- Show `whoami = labkali` and hostname = `kali-lab-01`.

---

### Evidence file (tie screenshots to repo commit)

Create a small file on Kali that contains the Git commit short SHA and timestamp for future proof.

Open **Git Bash** on your host and run:

```
cd "C:/Users/Concepcion/Documents"
git clone https://github.com/jellyconcepcion/blue-team-homelab.git
```

Now you'll have:

```
C:\Users\Concepcion\Documents\blue-team-homelab
```

Move your screenshots into:

```
C:\Users\Concepcion\Documents\blue-team-homelab\01-lab-setup\03-kali-a
ttacker\screenshots\
```

---

## 2. Run Git commands on host (not Kali)

From Git Bash (on host), run:

```
cd "C:/Users/Concepcion/Documents/blue-team-homelab"
git add 01-lab-setup/03-kali-attacker/screenshots/kali-*
git commit -m "M1: Add Kali screenshots"
git push
```

Then get the commit SHA:

```
git rev-parse --short HEAD
```

This gives something like `019fee6 M1: Add Kali screenshots`.

---

## 3. Create the evidence file on Kali

(Note: You first need to cd "C:/Users/Concepcion/Documents/blue-team-homelab on gitbash then run git log --oneline and find the SHA you need.)

Now switch to your Kali VM terminal and run:

```
echo "blue-team-homelab commit: 019fee6| $(date -u '+%Y-%m-%d %H:%M
UTC')" | sudo tee /etc/lab-evidence-kali.txt
cat /etc/lab-evidence-kali.txt
```

Take the screenshot (`01-lab-setup/03-kali-attacker/screenshots/kali-evidence-file.png`) of that output.

👉 Important: Kali doesn't need to know your GitHub repo at all. You're just manually pasting the SHA into the evidence file. That's enough to link your VM to your GitHub commit.

---

Upload that one screenshot (`kali-evidence-file.png`):

```
git add 01-lab-setup/screenshots/kali-evidence-file.png
git commit -m "M1: Add Kali evidence file screenshot"
git push
```

---

## Install Guest Additions (optional but recommended)

If you want clipboard & better display:

**Method (if Guest Additions not preinstalled):**

```
sudo apt update
sudo apt install -y virtualbox-guest-x11virtualbox-guest-x11
sudo reboot
```

## Verify it worked

After reboot, check modules:

```
lsmod | grep vboxguest
```

If you see `vboxguest 53248 6 vboxsf`, you're good.

**Screenshot:**

- **File:** `01-lab-setup/03-kali-attacker/screenshots/kali-guest-additions-installed.png`
- **What to capture:** `lsmod | grep vboxguest` after reboot showing modules loaded.

---

### Take a VirtualBox snapshot (preserve state)

In VirtualBox → select `LAB-KALI-01` → **Snapshots** → Take snapshot.

- Snapshot name: `M1-Kali-Evidence-019fee6`
- `Description: Kali Linux attacker VM installed, hostname kali-lab-01 set, Host-Only IP 192.168.56.30 configured. Evidence file created with GitHub commit 019fee6.`

**Screenshot:**

- **File:** `01-lab-setup/03-kali-attacker/screenshots/vb-kali-snapshot.png`

- **What to capture:** the snapshots pane where in the list it's showing `M1-Kali-Installed`.

---

# 🔎 Quick verification commands (run on the host / VMs to prove everything)

## Ubuntu VM — wazuh-lab-01

**Commands to run & capture:**

1. **Wazuh service status**

```
sudo systemctl is-active wazuh-manager wazuh-indexer wazuh-dashboard
```

2. **Wazuh API listening on 55000**

```
sudo ss -tlnp | grep 55000 || true
```

3. **Registered agents**

```
sudo /var/ossec/bin/agent_control -l
```

**Suggested filename if combined:**

`ubuntu-wazuh-services-api-agents.png`

---

## Host / Browser

**Verify Dashboard Access:**

- Open browser → `https://192.168.56.10` (self-signed cert warning is fine)
- Go to **Agents** → `WIN11-LAB-01` shows **Active**
- Go to **Threat Hunting** → **Events** → show Event ID **4625 entries**

**Screenshot filenames:**

1. `host-dashboard-login.png` → initial dashboard login page
2. `host-dashboard-agents.png` → Agents list with WIN11-LAB-01 Active
3. `host-dashboard-threat-events.png` → Threat Hunting showing 4625 events

---

## Windows VM — WIN11-LAB-01

**Commands to run & capture:**

1. **Check Wazuh service**

```
Get-Service WazuhSvc
```

✅ Capture output showing **Running**
**Screenshot filename:** `win11-wazuh-service.png`

2. **Tail the agent log**

```
Get-Content "C:\Program Files (x86)\ossec-agent\ossec.log" -Tail 20
```

✅ Capture connection / registration messages to Wazuh server
**Screenshot filename:** `win11-wazuh-log.png`

---

## Kali VM — LAB-KALI-01

**Commands to run & capture:**

1. **Check IP addresses**

```
ip a
```

✅ Capture output showing **192.168.56.30** assigned to host-only adapter
**Screenshot filename:** `kali-ip-a.png`