HACKRANGE

HackRange Local Lab: Installation Guide

for **Learning the Software Build Process: A Four-Week DevSecOps Intensive**

Please read this first: this is a cyber security training lab

The Local Lab is **not** normal software. It was built for practice, and on purpose it contains:

- **deliberately vulnerable code**, images and settings that you will attack, scan and then fix;
- **fake secrets** (fake passwords, tokens and keys) planted for you to find. None of them is real.

Install it **only inside a disposable virtual machine on a private network**, such as your home network. Never install it on a server, never on a computer you care about, and never anywhere the internet can reach it.

Author: Tim Rice
Course page: lms.hackrange.com/courses/view.php?c=8e09034afcde8ab12ca677a112846332
Lab repository: github.com/hackrange/lab_devsecops

What is inside this guide

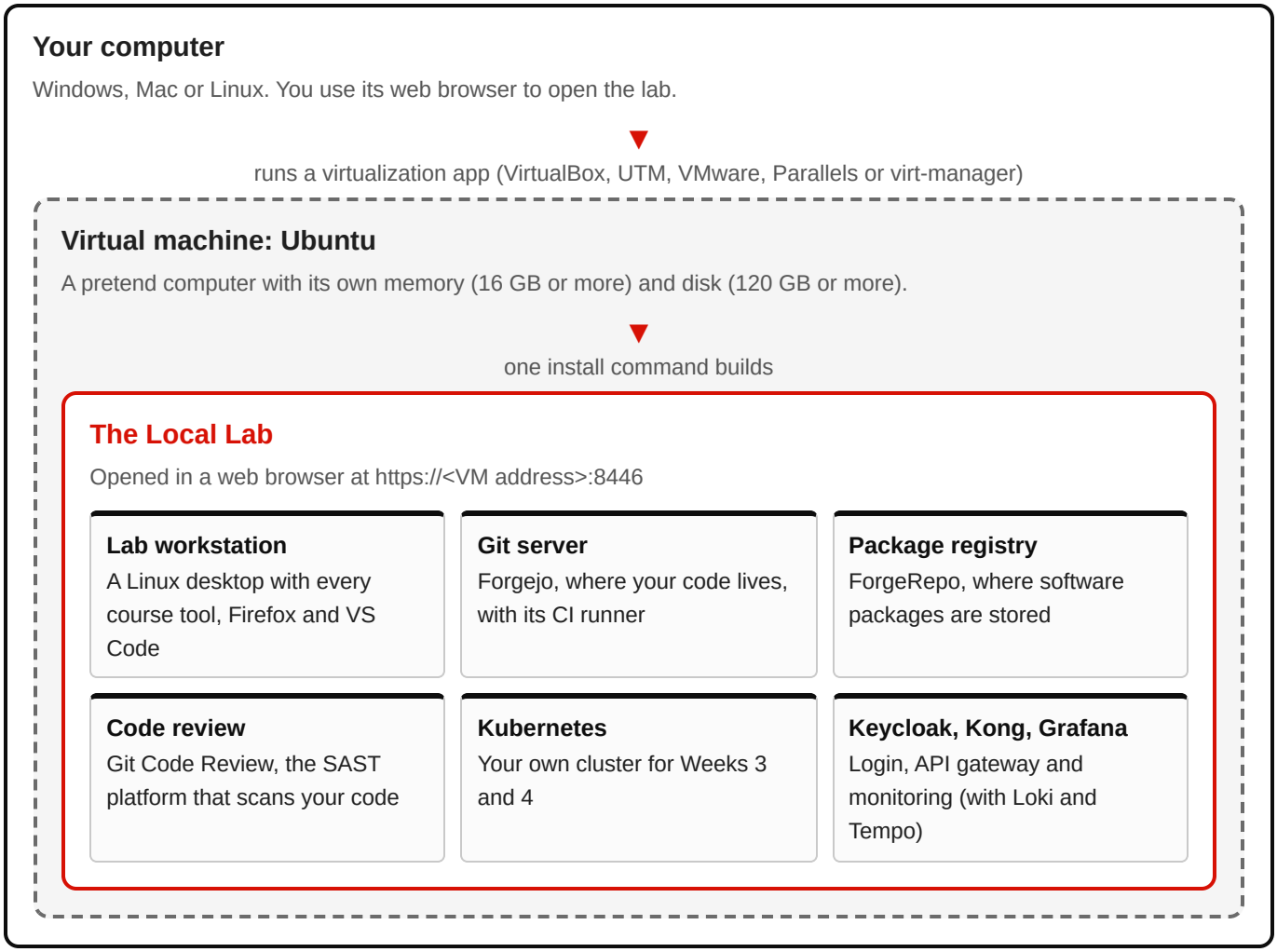
New to all of this? That is fine. This guide assumes nothing. Go one step at a time, and do not skip ahead. Words you may not know are explained in the Glossary (section 13).

1 What you are about to build

The course has labs: hands-on practice where you use real tools. Normally the course runs these labs for you in your web browser. The **Local Lab** is a second option: you run the very same lab on **your own computer**. It has the same workstation, the same tools and the same lab services, and the commands in the lessons work unchanged.

You will build it in three layers, like boxes inside boxes:

1. **Your computer.** The Windows PC, Mac or Linux computer you are using right now.
2. **A virtual machine running Ubuntu.** A virtual machine (VM) is a pretend computer that runs inside a window on your real computer. It has its own memory, its own disk and its own operating system. Here, that operating system is **Ubuntu**, a free version of Linux. Anything that happens inside the VM stays inside the VM, which is exactly why we use one.
3. **The Local Lab.** One command installs the whole lab inside the Ubuntu VM.



How long it takes

- Creating the virtual machine and installing Ubuntu: about **30 to 60 minutes**.
- Installing the Local Lab: about **30 to 90 minutes**, depending on your computer and your internet. It downloads about **20 GB**.

> **TIP**
>
> Plan to do this on a day when you have a couple of hours and a good internet connection. Keep your computer **plugged in**, and turn off sleep while you work, so it stays awake until the install is done.

2 Check your computer

The lab is big. Your computer runs its own system **and** the virtual machine at the same time, so it needs more than the VM itself. Check these first. It takes five minutes and can save you hours.

What your computer needs

Your computer	Minimum	Recommended
Memory (RAM)	24 GB (so the VM can have 16 GB)	32 GB or more
Free disk space	150 GB	200 GB or more, on an SSD
Processor	6 cores, with hardware virtualization turned on (Intel VT-x, AMD-V, or any Apple Silicon Mac)	8 cores or more

What you will give the virtual machine

For the virtual machine	Minimum	Recommended
Processor type	64-bit Intel/AMD (x86_64) **or** ARM64 (Apple Silicon M1 or newer, other ARM64)	same
Processor cores (vCPUs)	4	8 or more
Memory (RAM)	16 GB	24 GB or more
Virtual disk	120 GB (100 GB free after Ubuntu)	160 GB or more, on an SSD
Operating system	Ubuntu 22.04 LTS or newer	Ubuntu 24.04 LTS **Desktop**
Network	Internet access (NAT networking is best)	same
Screen	Any; a desktop lets you open the lab with one command	Ubuntu Desktop at 1920x1080

WATCH OUT

A Mac with Apple Silicon and 16 GB of memory **cannot** give a VM 16 GB. Use a computer with more memory, or use the hosted labs in the course instead. The hosted labs are a perfectly good choice.

On Windows: find your memory, cores and disk

1. Press `Ctrl` + `Shift` + `Esc` together. The **Task Manager** opens.
2. If you see a small window, click **More details** at the bottom.
3. Click **Performance** (on the left, or the tab at the top).
4. Click **Memory**. The large number at the top right is how much memory you have (for example, **32.0 GB**).
5. Click **CPU**. Under the graph, find **Cores** (for example, 8) and **Virtualization**. It should say **Enabled**.
6. For disk space, open **File Explorer** (the folder icon on the taskbar) and click **This PC**. Each drive shows how many GB are free.

On a Mac: find your chip, memory and disk

1. Click the **Apple menu** (the apple at the top left of the screen).
2. Click **About This Mac**.
3. Read **Chip** (or **Processor**). If it says **Apple M1**, M2, M3, M4 or newer, you have an **Apple Silicon** Mac. If it says **Intel**, you have an Intel Mac. Write this down: section 3 depends on it.
4. Read **Memory** (for example, **32 GB**).
5. For disk space, click the **Apple menu**, then **System Settings**, then **General**, then **Storage**. It shows how much space is available.

On Linux

Open a terminal and run these three commands. They show your memory, your number of processor cores, and your free disk space.

```
free -h
nproc
df -h ~
```

TIP

Hardware virtualization is usually already turned on. If it is off, it is changed in your computer's BIOS or UEFI settings (the setup screen you reach by pressing a key such as `F2`, `F10` or `Del` while the computer starts). The key and the menu names differ by brand, so search the web for your computer's model plus "enable virtualization". You do **not** need "nested virtualization".

3 Create the virtual machine

Pick the part below that matches your computer, and follow only that part:

- **3a.** Windows (Intel or AMD) with VirtualBox
- **3b.** Mac with Apple Silicon (M1 or newer) with UTM
- **3c.** Mac with Intel
- **3d.** Linux

In every case you will: download a free virtualization app, download Ubuntu, create a VM with the sizes from section 2, and install Ubuntu with the default options.

THE SIZES TO USE (FROM SECTION 2)

Memory: 16 GB minimum (16384 MB), 24 GB (24576 MB) or more if your computer has 32 GB or more.
Processors (CPUs): 4 minimum, 8 if your computer has 8 or more cores.
Disk: 120 GB minimum, 160 GB or more if you have the space.

WATCH OUT: NETWORK SAFETY

Use **NAT** networking (and, for VirtualBox, a second **host-only** adapter, explained below). The lab only answers computers on your own private network. But "private" still means *everyone* on the same network: at home that is your family's devices, and on a school, office or cafe network it can be hundreds of strangers. NAT and host-only keep the VM reachable from **your own computer only**. Never choose "Bridged" on a shared network.

3a. Windows (Intel or AMD) with VirtualBox

VirtualBox is free. (VMware Workstation Pro, free for personal use, or Hyper-V also work. This guide shows VirtualBox.)

WATCH OUT

Do **not** use WSL (Windows Subsystem for Linux). It cannot run the lab. You need a real virtual machine.

Download what you need

1. Open www.virtualbox.org in your web browser and click **Download**.
2. Click **Windows hosts**. Open the downloaded file and click **Next** and **Install** through the installer, keeping the defaults. If Windows asks "Do you want to allow this app to make changes?", click **Yes**.

3. Open ubuntu.com/download/desktop and download **Ubuntu 24.04 LTS** (Desktop). The file ends in `.iso` and is about 6 GB. Wait for it to finish.

Create the VM

1. Open **VirtualBox** and click **New**.
2. Under **Name**, type **HackRange Lab**.
3. Next to **ISO Image**, click the arrow, choose **Other...**, and pick the Ubuntu `.iso` file you downloaded (usually in your **Downloads** folder).
4. Tick **Skip Unattended Installation**, so you install Ubuntu yourself in the next part. (If you do not see this box, look for it on the next page, or open the **Unattended Install** section and turn it off.)
5. Click **Next** (or open the **Hardware** section).
6. Set **Base Memory** to **16384 MB** (or 24576 MB if your computer has 32 GB or more).
7. Set **Processors** to **4** (or 8 if your computer has 8 or more cores). Click **Next**.
8. Choose **Create a Virtual Hard Disk Now** and set the size to **120 GB** (or 160 GB). Leave **Pre-allocate Full Size** unticked. Click **Next**.
9. Check the summary and click **Finish**.

Set up the network (important for VirtualBox)

VirtualBox's default NAT network lets the VM reach the internet, but it hides the VM from your own computer. You add a second, private adapter so your computer can reach it.

1. Make sure the VM is **turned off** (it is, if you have not started it yet).
2. Select **HackRange Lab** in the list and click **Settings**.
3. Click **Network**. On **Adapter 1**, check that **Attached to** says **NAT**. Leave it.
4. Click the **Adapter 2** tab.
5. Tick **Enable Network Adapter**.
6. Set **Attached to** to **Host-only Adapter**. Leave the name it picks.
7. Click **OK**.

Host-only means only your own computer can reach the VM, which is exactly what you want.

Install Ubuntu

1. Select **HackRange Lab** and click **Start**. A window opens and Ubuntu starts from the `.iso` file.
2. If you see a black menu, choose **Try or Install Ubuntu** and press `Enter`.
3. When the installer appears, pick your language and click **Next** through the screens, keeping the **default options**.

4. When asked, choose **Install Ubuntu**, then **Interactive installation**, then the **Default selection** of apps.
5. On the disk screen, choose **Erase disk and install Ubuntu**. This only erases the *virtual* disk you just made, not your real computer.
6. Type your name, a computer name, a **username** and a **password**. Write the password down: this is your **Ubuntu password**, and you will need it often.
7. Pick your time zone, then click **Install**. Wait (usually 10 to 30 minutes).
8. When it says to, click **Restart Now**. If it asks you to remove the installation medium, just press `Enter`.
9. Log in with the username and password you chose. Click through any welcome screens.

> **TIP**
> If the Ubuntu window is tiny, open the VM window's **View** menu and try **Auto-resize Guest Display**, or change the size inside Ubuntu under **Settings**, **Displays**. If a small **green turtle** shows at the bottom of the VirtualBox window, VirtualBox is running in a slow mode because Windows' own virtualization features are on. The lab still works, but more slowly.

3b. Mac with Apple Silicon (M1, M2, M3, M4 or newer) with UTM

UTM is free. (VMware Fusion, free for personal use, or Parallels also work.) Apple Silicon Macs need the **ARM64** edition of Ubuntu. The steps below install **Ubuntu Server for ARM** and then add a small desktop to it.

> **TIP: A SHORTCUT**
> UTM's gallery and Parallels also offer ready-made Ubuntu ARM64 desktop VMs. If you use one, give it the memory, processors and disk from section 2, then skip to section 4.

Download what you need

1. Open mac.getutm.app and click **Download**. Open the downloaded `.dmg` file and drag **UTM** into **Applications**.
2. Open ubuntu.com/download/server/arm and download **Ubuntu Server 24.04 LTS for ARM**. The file name ends in `arm64.iso`.

Create the VM

1. Open **UTM** from Applications. If macOS asks whether to open it, click **Open**.
2. Click **Create a New Virtual Machine** (or the **+** button).
3. Click **Virtualize**. (Not "Emulate": that is much slower.)

4. Click **Linux**.
5. Leave **Use Apple Virtualization** unticked. Under **Boot ISO Image**, click **Browse** and choose the `arm64.iso` file. Click **Continue**.
6. Set **Memory** to **16384 MB** (or 24576 MB) and **CPU Cores** to **4** (or 8). Click **Continue**.
7. Set the storage size to **120 GB** (or 160 GB). Click **Continue**.
8. On **Shared Directory**, click **Continue** without choosing anything.
9. Name it **HackRange Lab** and click **Save**.

UTM's default network is **Shared Network**, which is a NAT network. It works as it is: leave it.

Install Ubuntu Server

1. Click the big **play** button to start the VM.
2. Choose **Try or Install Ubuntu Server** and press `Enter`.
3. The Server installer is text only. Use the **arrow keys** to move, `Space` to tick, and `Enter` to choose **Done**.
4. Keep the **default options** on every screen (language, keyboard, "Ubuntu Server", network, proxy, mirror).
5. On the storage screen, keep **Use an entire disk**, choose **Done**, then **Continue** to confirm. This only erases the *virtual* disk.
6. Type your name, a server name, a **username** and a **password**. Write the password down: this is your **Ubuntu password**.
7. Skip Ubuntu Pro and the extra software lists (choose **Done** or **Continue**).
8. Wait for **Installation complete!** and choose **Reboot Now**.
9. If it says to remove the installation medium: close the VM window, and in UTM select the VM, find the **CD/DVD** drive, choose **Clear**, then start the VM again.
10. Log in by typing your username, pressing `Enter`, then typing your password and pressing `Enter` (nothing appears as you type the password; that is normal).

Add a desktop

1. Type this command exactly and press `Enter`. Type your Ubuntu password when asked.

```
sudo apt update
```

2. Then type this one and press `Enter`. It takes about 5 to 20 minutes.

```
sudo apt install -y ubuntu-desktop-minimal
```

3. Restart the VM:

```
sudo reboot
```

4. A graphical login screen appears. Log in with your username and password.

The lab installer detects ARM automatically. You do not need to do anything special later.

3c. Mac with Intel

Use **VirtualBox**, VMware Fusion or Parallels, with the normal **Ubuntu 24.04 Desktop** ISO from ubuntu.com/download/desktop.

1. From www.virtualbox.org, click **Download** and choose **macOS / Intel hosts**. Open the `.dmg` and run the installer.
2. If macOS blocks it, open **System Settings**, **Privacy & Security**, and click **Allow** next to the message about Oracle, then restart the Mac if asked.
3. Download the **Ubuntu 24.04 LTS** Desktop `.iso`.
4. Follow **3a** from "Create the VM" onward. Every VirtualBox screen is the same on a Mac, including the **Adapter 2**, **Host-only Adapter** network step.

With VMware Fusion or Parallels, choose the same memory, processors and disk, and keep the default **Shared** (NAT) network. It works as it is.

3d. Linux

Use **virt-manager** (KVM), VirtualBox or VMware with the **Ubuntu 24.04 Desktop** ISO. These steps use virt-manager on Ubuntu or Debian.

1. Open a terminal and install virt-manager:

```
sudo apt install -y virt-manager
```

2. Log out and back in once, then open **Virtual Machine Manager** from your applications menu.
3. Download the **Ubuntu 24.04 LTS** Desktop `.iso` from ubuntu.com/download/desktop.
4. In virt-manager, click **File**, then **New Virtual Machine**.
5. Choose **Local install media (ISO image or CDROM)** and click **Forward**.
6. Click **Browse**, then **Browse Local**, pick the `.iso`, and click **Forward**. If you see "automatically detect", leave it ticked.
7. Set **Memory** to **16384** MiB (or 24576) and **CPUs** to **4** (or 8). Click **Forward**.
8. Set the disk to **120** GiB (or 160). Click **Forward**.

9. Name it **hackrange-lab**. Under **Network selection**, keep **Virtual network 'default': NAT**. Click **Finish**.
10. Install Ubuntu as in **3a**, "Install Ubuntu", keeping the default options.

> **WATCH OUT**
>
> Do not run the installer on your own Linux computer directly. It belongs in a VM of its own: the lab changes system settings and contains deliberately vulnerable software.

4 Install the Local Lab

Everything in this section happens **inside your Ubuntu virtual machine**, not on your own computer.

Before you start

- Plug your computer in, and turn off sleep for now. The install takes **30 to 90 minutes**.
- Make sure the VM has internet: open **Firefox** inside Ubuntu and visit any web page.

Step 1: Open a terminal

A **terminal** is a window where you type commands instead of clicking. It looks plain, but it is powerful.

1. Click inside the Ubuntu window so it has your keyboard.
2. Press `Ctrl` + `Alt` + `T` together. A terminal window opens.
3. If that does nothing, click the **Show Apps** button (the grid of dots at the bottom left), type **Terminal**, and click it.

Step 2: Paste the install command

This is the install command. It is one single line:

```
curl -fsSL https://raw.githubusercontent.com/hackrange/lab_devsecops/main/install.sh | sudo bash
```

1. Copy the command above. (Select it and press `Ctrl` + `C`.)
2. Click inside the terminal window.
3. Paste with `Ctrl` + `Shift` + `V`. (In the Ubuntu terminal, plain `Ctrl` + `V` does not paste. You can also right-click and choose **Paste**.)
4. Check that it is all on one line and matches exactly. Then press `Enter`.

> **TIP: COPYING INTO THE VM**
>
> Copy and paste between your own computer and the VM often does not work until extra tools are installed. The easiest fix: open **Firefox inside Ubuntu**, go to github.com/hackrange/lab_devsecops, and copy the command from that page. Or type it carefully by hand: every character matters, including the `|` (the "pipe", usually `Shift` + `\`).

Step 3: Type your Ubuntu password

The command ends in `sudo bash`. **sudo** means "run this as the administrator", so Ubuntu asks for your password first. You will see something like:

```
[sudo] password for yourname:
```

1. Type your **Ubuntu password** (the one you log in to Ubuntu with).
2. **Nothing appears on the screen while you type it**, not even dots. That is normal. Keep typing.
3. Press `Enter`. If you made a mistake, it asks again.

Step 4: Watch it start

The installer introduces itself, reminds you that this is a training lab, shows your Ubuntu version and processor type, and downloads the lab files to `/opt/hackrange-labs`. Then it works through its steps. Each one starts with `==>`, for example:

- `==> Checking this machine` (enough memory, cores, disk and internet)
- `==> Installing system packages` and `==> Installing Docker`
- `==> Setting up the lab network address` and `==> Creating the lab certificate authority`
- `==> Building the lab images (the first time takes 30 to 90 minutes)`: the longest step
- `==> Starting the lab services`, `==> Installing Kubernetes`, `==> Installing the CI runner`
- `==> Setting up the lab control panel` and `==> Creating your lab accounts and your Kubernetes cluster`

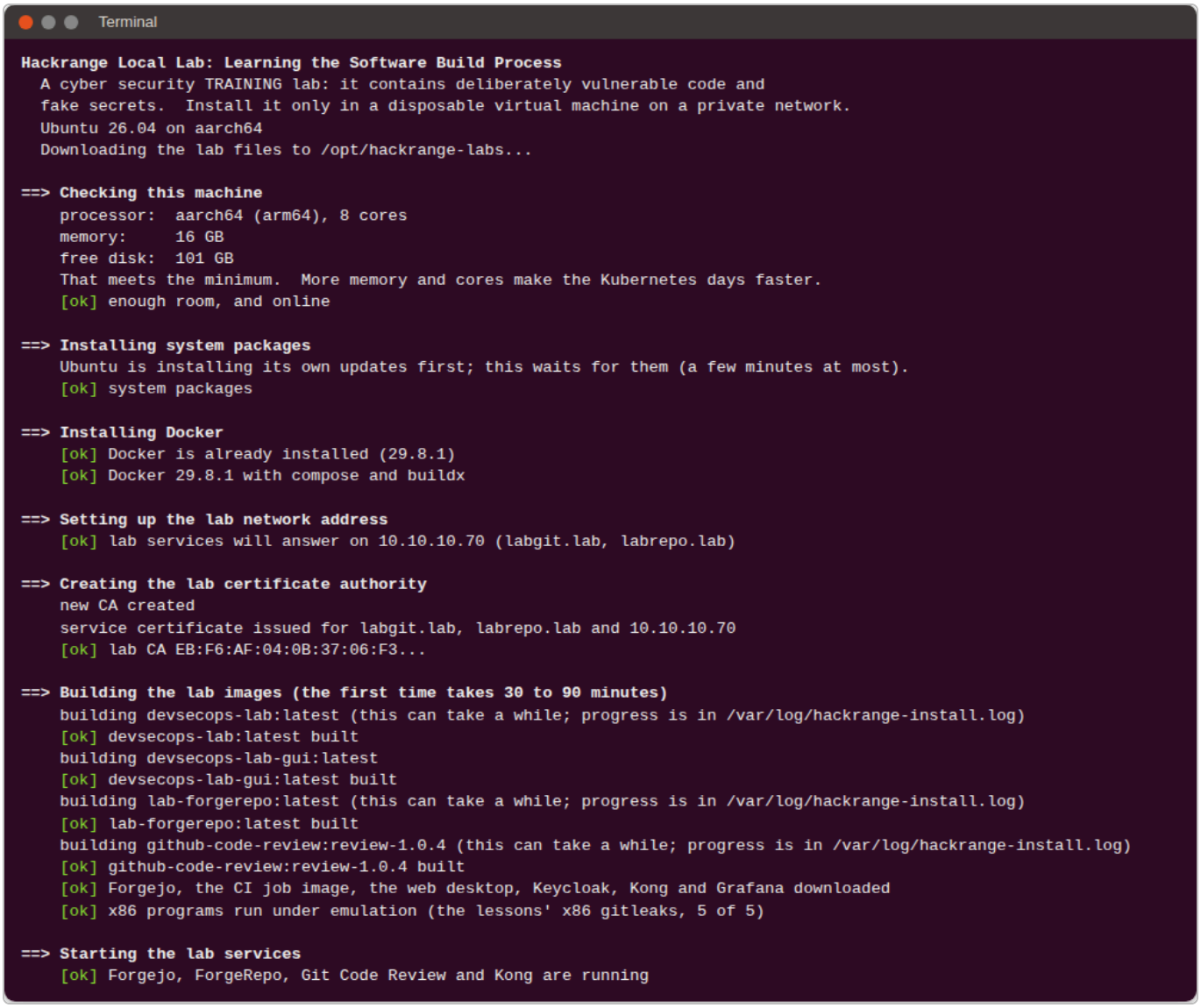


Figure 1. The installer has started. Look for the title **Hackrange Local Lab: Learning the Software Build Process**, the training lab notice, and the first `==>` steps with green `[ok]` lines under them.

Step 5: Wait

- The first install takes **30 to 90 minutes** and downloads about **20 GB**. You can leave it running.
- Some steps sit quietly for a long time. That is normal, as long as the computer stays on and awake.
- Do not close the terminal window, and do not shut down the VM while it runs.

WATCH OUT: IF IT STOPS

If the installer stops with a red `[stopped]` message, **read the message**. It says what went wrong and how to fix it (for example, "This virtual machine is smaller than the lab needs"). Fix it, then **run the same install command again**. It picks up where it left off. See section 11 for more help.

Step 6: What "done" looks like

When it finishes, you see a green banner: **The Local Lab is installed.** with the number of minutes it took. Right under it is a block called **Your Hackrange Local Lab** with everything you need to get in:

- the **web address** to open in your browser (from your own computer, and from the Ubuntu machine);
- the address of your **lab control panel**;
- the **username** (always `student`) and the **password**;
- the **SSH** command and the **Remote Desktop** address;
- the addresses of the lab services, and the lab certificate's **fingerprint**.

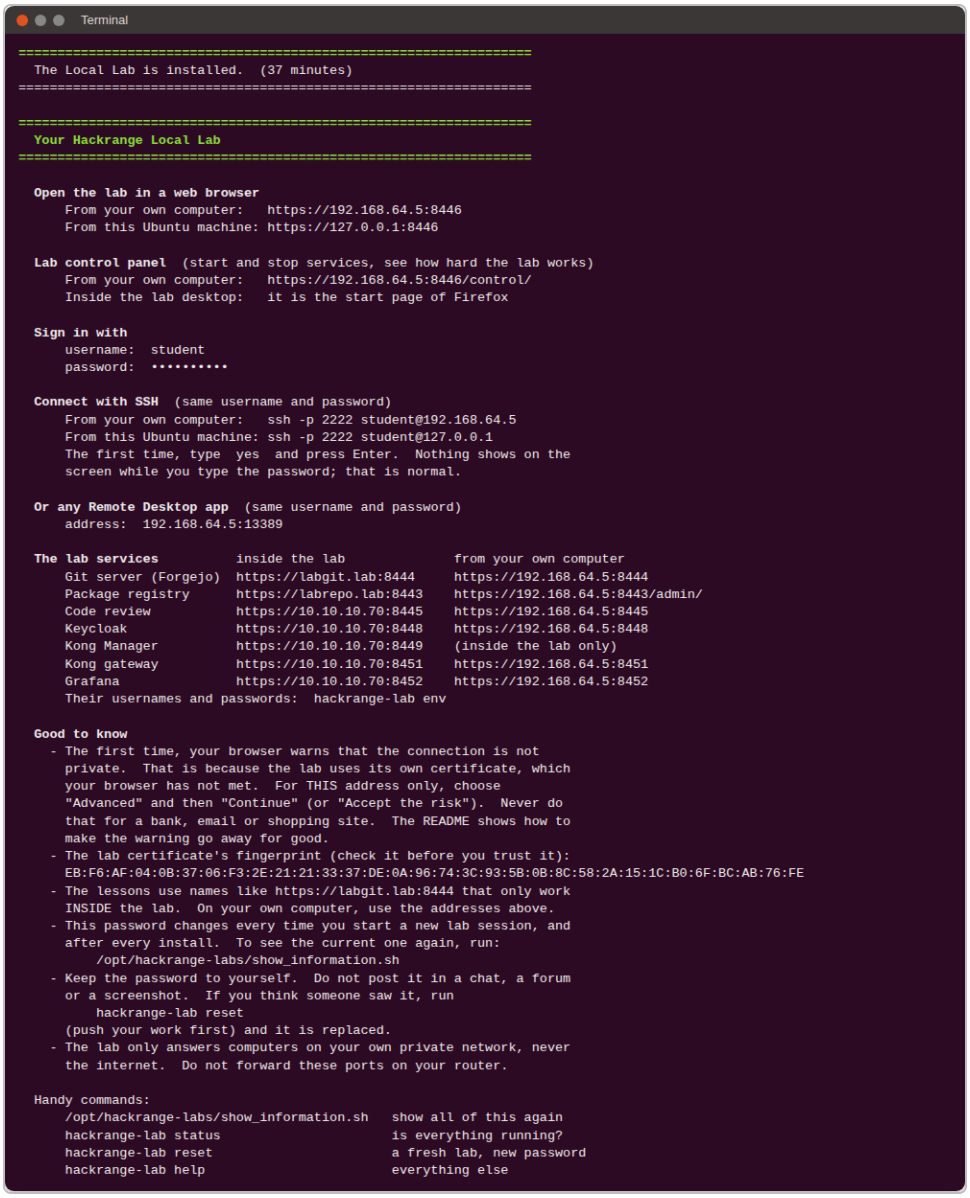


Figure 2. The install is done. Look for the green **The Local Lab is installed.** banner, then the **Open the lab in a web browser** addresses and the **Sign in with** username and password. The password and address in this picture are hidden; **yours will be different**.

IMPORTANT: SAVE THIS INFORMATION

Write down the web address, the username and the password, somewhere private (a notebook or a password manager). Do not post them anywhere.

The password changes every time you start a new lab session (and after every install or update). If a password stops working, just look it up again (section 5).

Step 7: Log out and back in once

1. Click the top-right corner of the Ubuntu screen (the power and network icons).
2. Click the **power** icon, then **Log Out**, and confirm.
3. Log back in with your Ubuntu username and password.

This lets the lab commands (such as `hackrange-lab status`) work without typing `sudo`.

> **TIP**
>
> Your applications menu now has a **Hackrange Lab** icon. Clicking it opens the lab desktop right inside Ubuntu.

5 Get your login information again, any time

Lost the password? Closed the terminal? No problem. The same information is always one command away. It lives in the folder `/opt/hackrange-labs`.

1. Open a terminal in Ubuntu (`Ctrl` + `Alt` + `T`).
2. Paste this command (`Ctrl` + `Shift` + `V`) and press `Enter`:

   ```
   /opt/hackrange-labs/show_information.sh
   ```

3. If it asks for a password, type your **Ubuntu password** (nothing appears as you type) and press `Enter`. It asks because the lab password is kept where only an administrator can read it.

You can also go into the folder first and run it from there. `cd` means "change directory" (move into a folder), and `./` means "the file in this folder":

```
cd /opt/hackrange-labs
./show_information.sh
```

Or use the lab command, which shows exactly the same thing:

```
hackrange-lab info
```

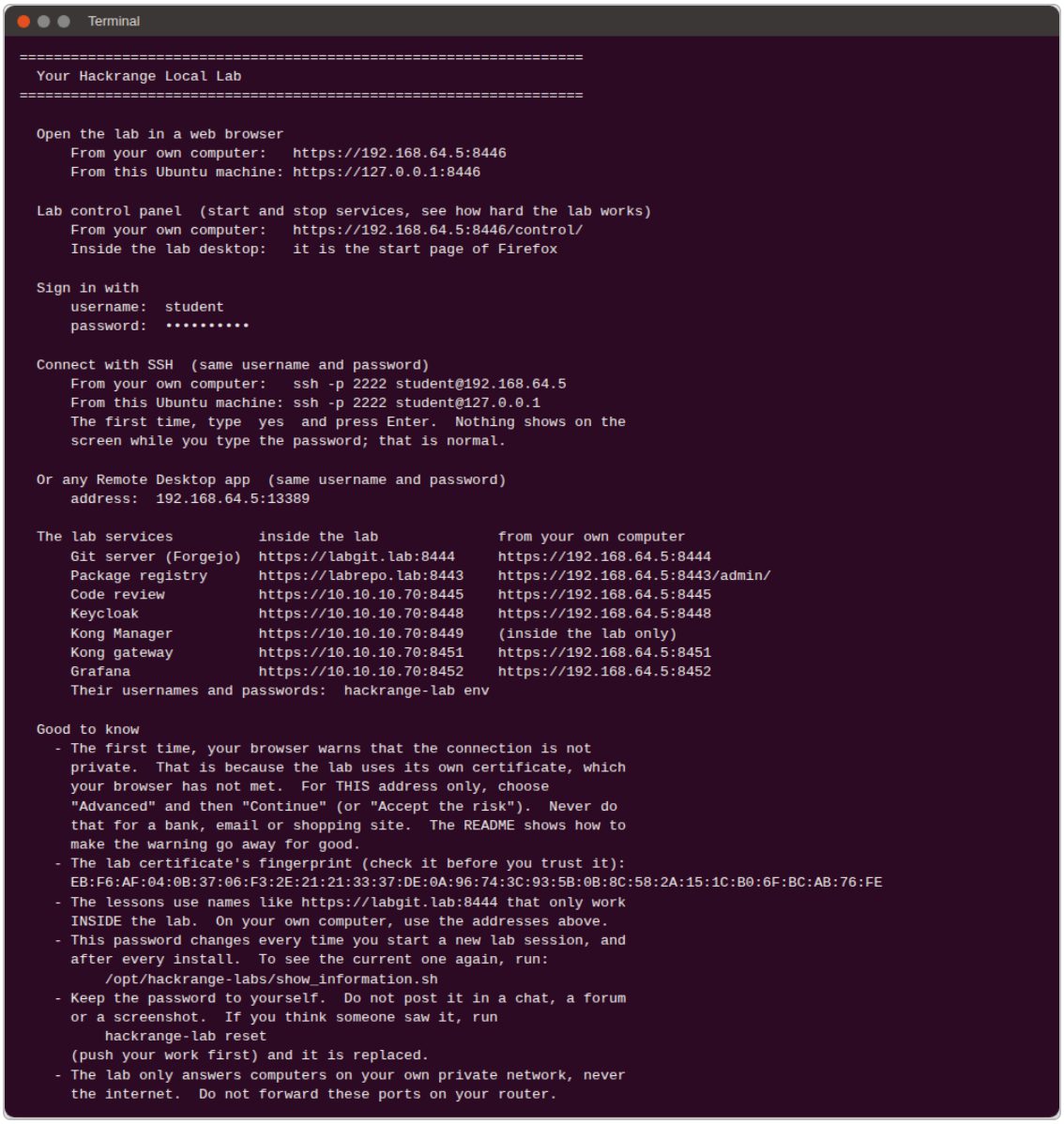


Figure 3. The information screen. From the top: **Open the lab in a web browser**, **Lab control panel**, **Sign in with** (username and password), **Connect with SSH**, **Or any Remote Desktop app**, **The lab services**, and **Good to know**. The password and addresses are hidden here; **yours will be different**.

Reading the information screen

Line	What it means
From your own computer	The address to type in the browser on your real computer, for example `https://192.168.64.5:8446`. The numbers are your VM's address.
From this Ubuntu machine	`https://127.0.0.1:8446`. `127.0.0.1` always means "this same machine".
username / password	One login for the web desktop, the control panel, SSH and Remote Desktop.
The lab services	Two columns. The "inside the lab" names (like `https://labgit.lab:8444`) are the ones the lessons use, and they only work inside the lab. From your own computer, use the right-hand column.
fingerprint	A long line of letters and numbers in pairs. You use it to check the lab certificate (section 6).

For the usernames and passwords of each lab service (Git server, registry, and so on), run:

```
hackrange-lab env
```

They are also shown, with Copy buttons, in the control panel (section 7).

6 Open your lab in the web browser

The lab desktop opens right in a web browser, the same as the hosted labs in the course. You can open it in two places.

Option A: from inside the Ubuntu VM

1. Open **Firefox** in Ubuntu.
2. Click the address bar at the top, type **https://127.0.0.1:8446**, and press `Enter`.

Option B: from your own computer (usually nicer)

1. Open your normal web browser (Chrome, Edge, Safari or Firefox) on your own computer.
2. Type the **From your own computer** address from section 5. It looks like **https://192.168.64.5:8446**, but with your VM's numbers.
3. Press `Enter`.

If Option B does not open but Option A does, the VM's network setting needs fixing. See section 11.

The "Your connection is not private" warning

The first time, your browser shows a scary-looking warning, such as **Your connection is not private** or **Warning: Potential Security Risk Ahead**.

Why? Websites prove who they are with a **certificate**. Your lab made its own certificate when it was installed, and your browser has never met it. So the browser cannot vouch for it, and warns you.

1. Check that the address bar shows **your lab's address** (the one from section 5) and nothing else.
2. Click **Advanced** (or **Show Details** in Safari).
3. Click **Continue to ...** or **Proceed to ...** (or **Accept the Risk and Continue** in Firefox, or **visit this website** in Safari).

> **WATCH OUT**
>
> Do this **for your lab's address only**. Never click past this warning on a bank, email or shopping site. There, the same warning can mean someone is intercepting your connection.

Sign in

1. On the login page, type **student** in the **Username** box.
2. Type the lab **password** from section 5 in the **Password** box.
3. Click **Login**.

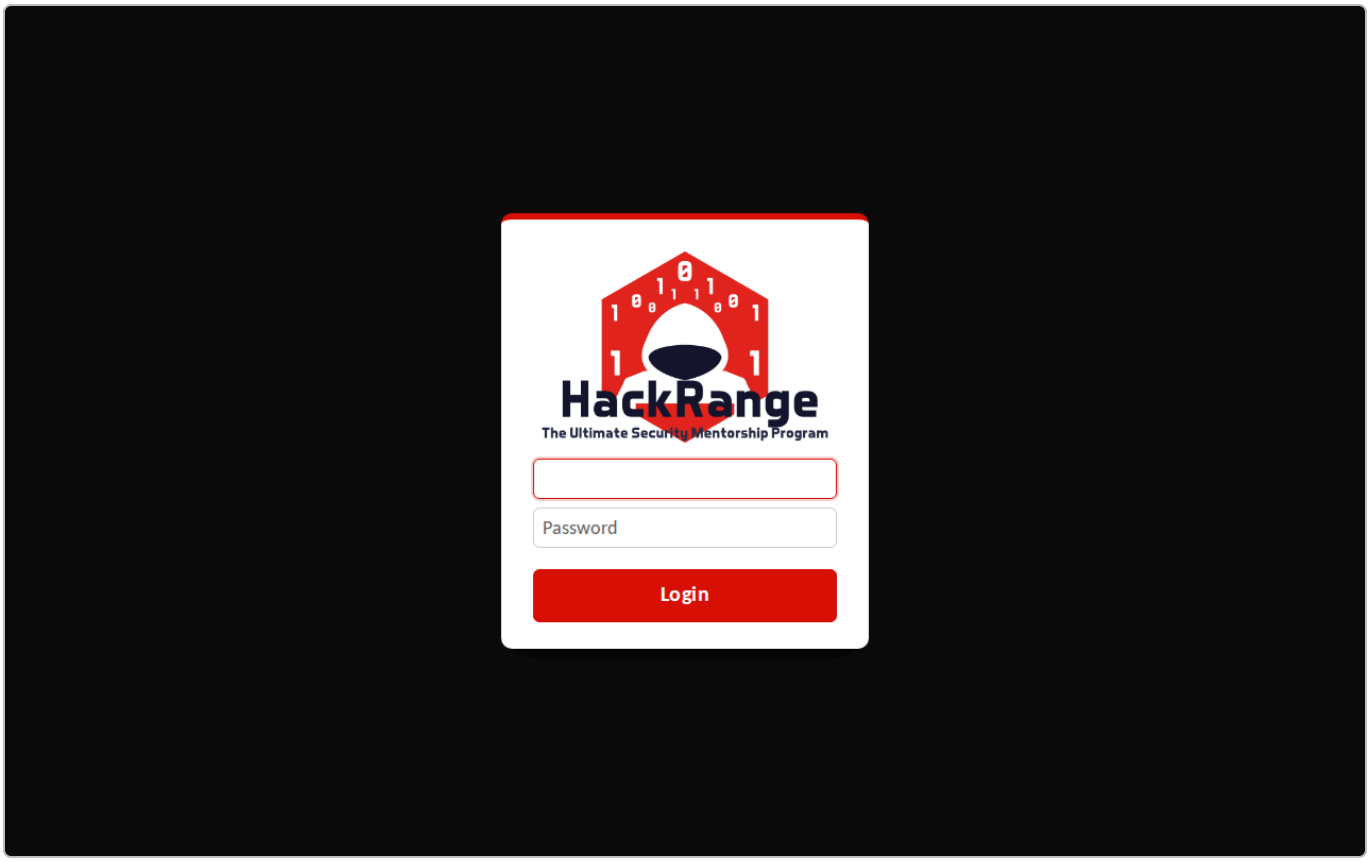


Figure 4. The web desktop's login page. Type **student** and your lab password. After five wrong tries, the page makes you wait five minutes before trying again.

The lab desktop opens in your browser. **Firefox** inside the lab opens on your lab control panel, called **Your Local Lab**, with every lab service and its username and password.

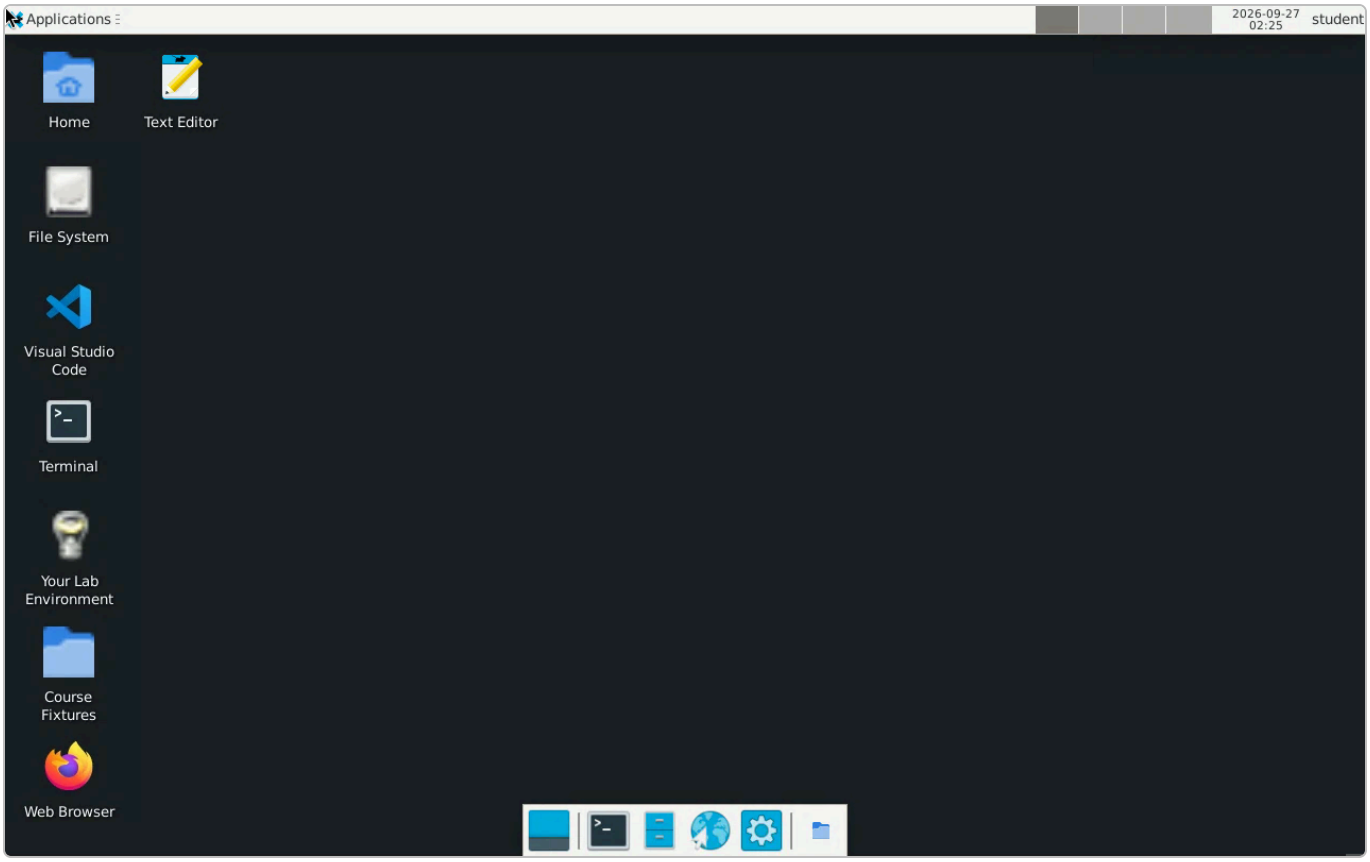


Figure 5. You are in. This is the lab workstation, a full Linux desktop inside your browser tab. Firefox shows **Your Local Lab**. Any passwords in this picture are hidden; **yours will be different**.

Optional: make the browser warning go away for good

You can tell your computer to trust your lab's own certificate. This is how real companies handle their private certificates. It is optional.

1. On your own computer, open **https://<your lab address>:8446/lab-ca.crt** (accept the warning one last time). It downloads a file called `lab-ca.crt`. This file is public: it holds no secret.
2. **Check the fingerprint. Do not skip this.** In Ubuntu, run `/opt/hackrange-labs/show_information.sh`. Near the end, it prints the certificate's fingerprint. On your own computer, open the downloaded file: Windows and Mac show a **SHA-256** fingerprint on its details page. Compare them pair by pair. **If they are not identical, delete the file and stop.** Something between your computer and the lab changed it.
3. Add it as a trusted certificate:
   - **Windows:** double-click the file, then **Install Certificate**, **Current User**, **Place all certificates in the following store**, **Browse**, **Trusted Root Certification Authorities**, **OK**, **Finish**.
   - **Mac:** double-click the file (it opens Keychain Access and adds it to **login**). Double-click the certificate in the list, open **Trust**, set **When using this certificate** to **Always Trust**, and close the window.

- **Firefox** (any system) keeps its own list: **Settings**, search for **certificates**, **View Certificates**, **Authorities**, **Import**, choose the file, tick **Trust this CA to identify websites**.

4. Close and reopen the browser.

The certificate belongs to this one lab. It can only vouch for the lab's own names and private network addresses, never for a real website. If you uninstall and reinstall the lab, it makes a new one and you repeat these steps. **When the course ends, remove it** (section 12 shows how).

7 The lab control panel

The control panel shows how your lab is doing and lets you start, stop and reset its parts.

- **Inside the lab desktop:** it is the start page of Firefox.
- **From your own computer:** open **https://<VM address>:8446/control/** (note the `/control/` at the end).

Sign in

1. The **Username** box already says **student**.
2. Type your lab password in **Password**. It is the same password as the web desktop.
3. Click **Sign in**.

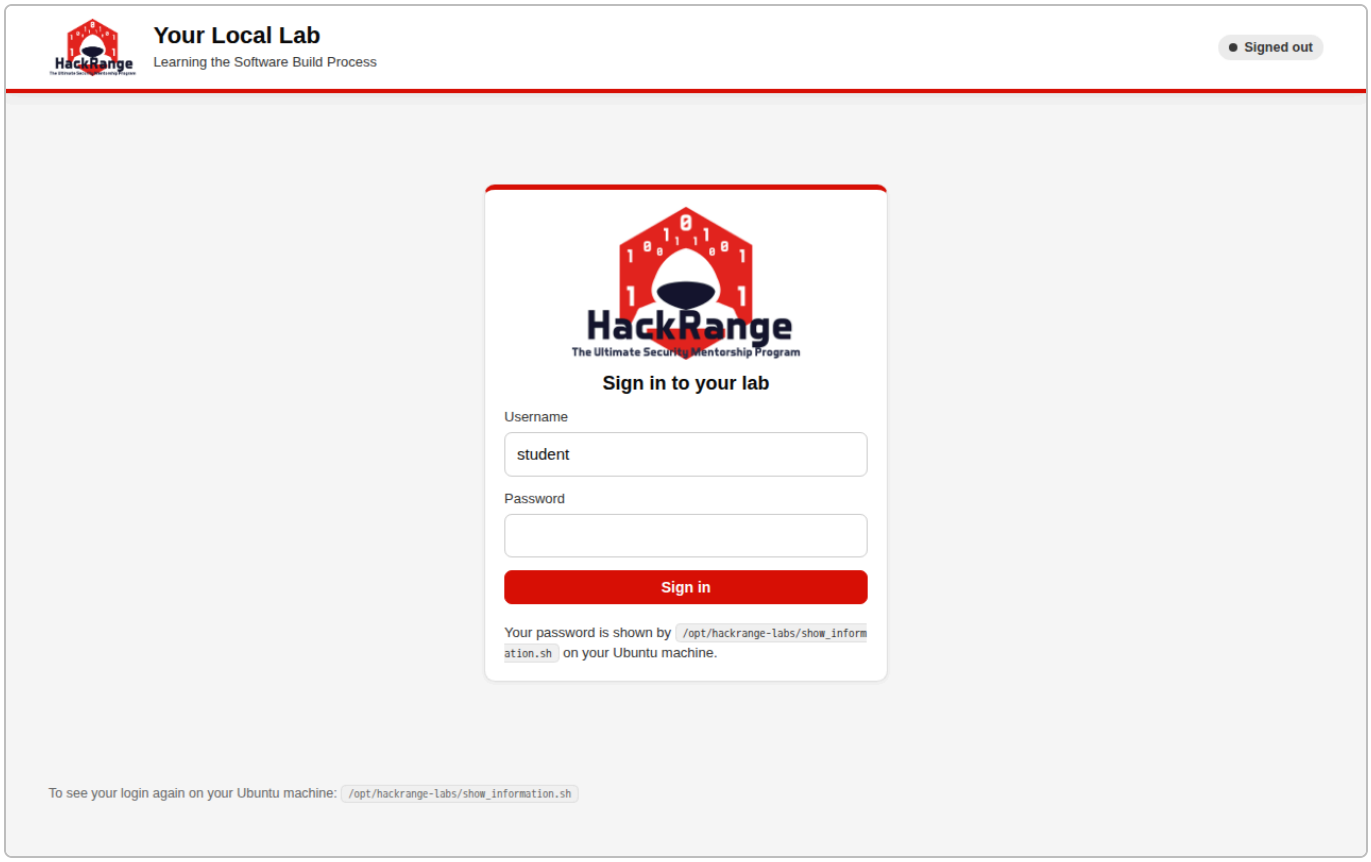


Figure 6. The control panel's **Sign in to your lab** page. The hint under the button reminds you where the password comes from: `/opt/hackrange-labs/show_information.sh`.

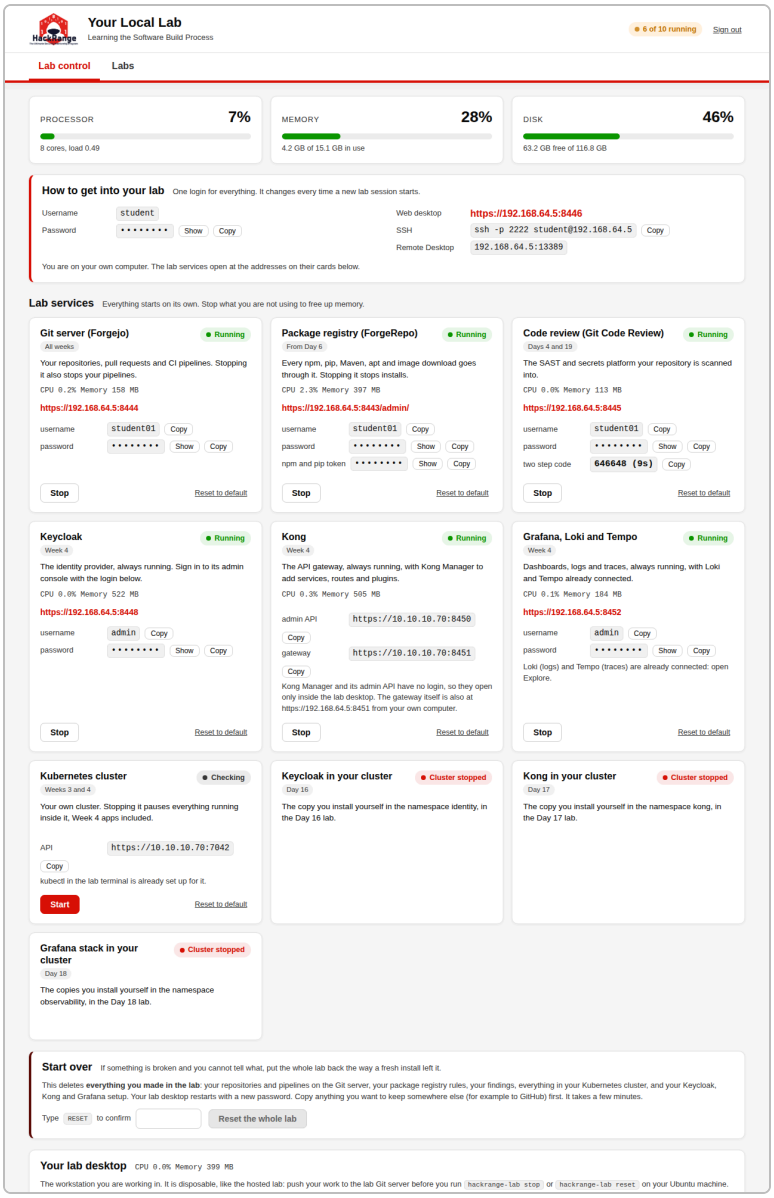


Figure 7. The **Lab control** tab. At the top are the **Processor**, **Memory** and **Disk** meters, then **How to get into your lab**, then one tile per lab service. Passwords and addresses are hidden here; **yours will be different**.

What each part does

Part	What it does
Processor, Memory, Disk	How hard the whole VM is working. Each service tile also shows the processor and memory that part is using.
How to get into your lab	Your username, your password (click **Show** to see it, **Copy** to copy it), the web desktop address, the SSH command and the Remote Desktop address.
Service tiles	One tile each for the Git server, the package registry, Git Code Review, Keycloak, Kong, Grafana (with Loki and Tempo), your Kubernetes cluster, and the copies of Keycloak, Kong and Grafana your Week 4 lessons install in your cluster. Each shows whether it is running, its address, and its username and password with **Copy** buttons.
Two-step code	Git Code Review asks for a two-step code when you sign in. Its tile shows the live code, how long it has left, and a **Copy** button. (The command `hackrange-lab totp` shows it too.)
Start / Stop	Stop what you are not using to free up memory. Everything starts again on its own when the VM restarts.
Reset to default	If you break a service, this deletes what you made in it and puts it back the way a fresh install left it (with a fresh account where it has one). Resetting the Git server also resets Git Code Review, which scans it.
Start over	At the bottom. Resets the **whole** lab: your repositories, pipelines, registry rules, findings, cluster, and Keycloak, Kong and Grafana setup. Type **RESET** in the box to confirm, then click **Reset the whole lab**. It takes a few minutes, and your lab desktop restarts with a new password.

WATCH OUT

Reset to default and **Start over** delete your work. Copy anything you want to keep somewhere else (for example, push it to GitHub) first.

The Labs tab: track and check your work

Click **Labs** at the top of the control panel. It lists every lab of the course, by day, and shows how many you have completed (for example, **3 of 20 labs complete**).

1. Do a lab in your lab desktop, following the lesson in the course.
2. In the **Labs** tab, click that day to open it. You see its checklist.
3. Some items are checked by the lab itself (for example, "your pipeline passed"). Others are things only you could have seen, such as an error message. **Tick** those yourself when you have done them.
4. Click **Check my work**. The lab looks at what you made and marks each item as passed or not yet, with a hint for anything still missing.

5. A lab is complete when every automatic check passes and every item you tick is ticked.

Some labs (Days 2, 3, 4, 9, 10 and 20) have GitHub parts. Type **Your GitHub username** in the box at the top of the tab and click **Save** so the lab can check them. Your progress is kept on your VM only, and it survives restarts and resets.

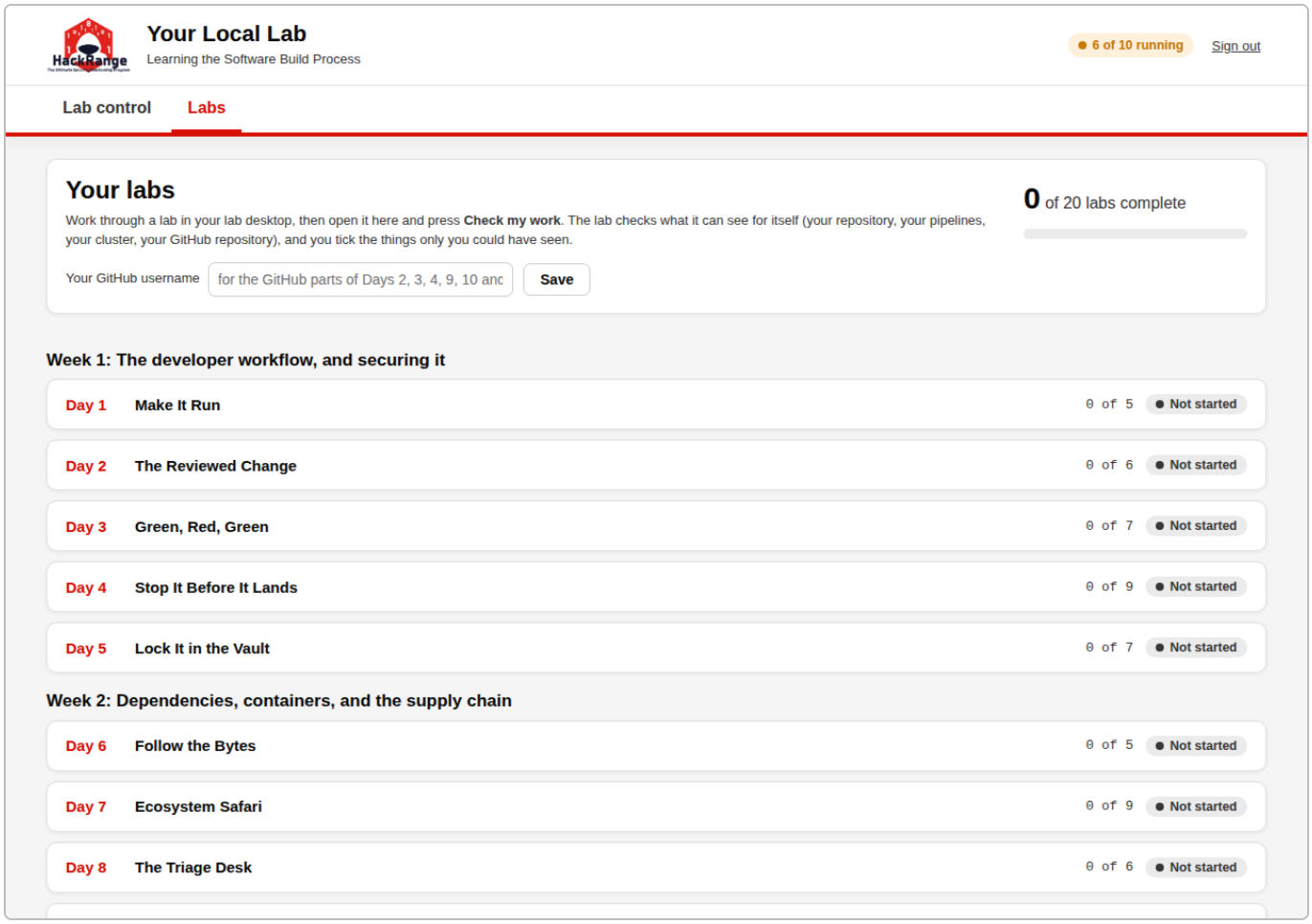


Figure 8. The **Labs** tab. Look for the **labs complete** count at the top right, the **Your GitHub username** box, and, inside an opened day, its checklist and the **Check my work** button.

8 Other ways in: SSH and Remote Desktop

The web browser is the easiest way into your lab. There are two more, and both use the **same username and password**.

SSH: a terminal on the lab

SSH (Secure Shell) is how people work on computers they cannot sit in front of. You type commands on your computer, and they run on the other machine. Many lessons assume you are comfortable with it, so this is a great place to practice.

1. **Open a terminal.** In Ubuntu: `Ctrl` + `Alt` + `T`. On Windows: open the Start menu, type **PowerShell**, press `Enter`. On a Mac: press `Command` + `Space`, type **Terminal**, press `Enter`.
2. **Find your password** with `/opt/hackrange-labs/show_information.sh` on the Ubuntu VM. It also prints the exact SSH command to use.
3. **Connect.** On the Ubuntu VM, type:

   ```
   ssh -p 2222 student@127.0.0.1
   ```

   From your own computer, use your VM's address instead of `127.0.0.1`:

   ```
   ssh -p 2222 student@<VM address>
   ```

   `-p 2222` is the port (the lab listens for SSH on 2222, not the usual 22). `student` is the user, and after the `@` is the machine to connect to.
4. **The first time only,** SSH asks *Are you sure you want to continue connecting (yes/no)?* Type **yes** and press `Enter`. It is making sure it is talking to the right machine.
5. **Type the password** and press `Enter`. Nothing appears while you type, not even dots. That is normal.
6. **You are in.** The prompt changes to `student@devsecops-lab`. Every command now runs inside the lab. Type **exit** and press `Enter` to leave.

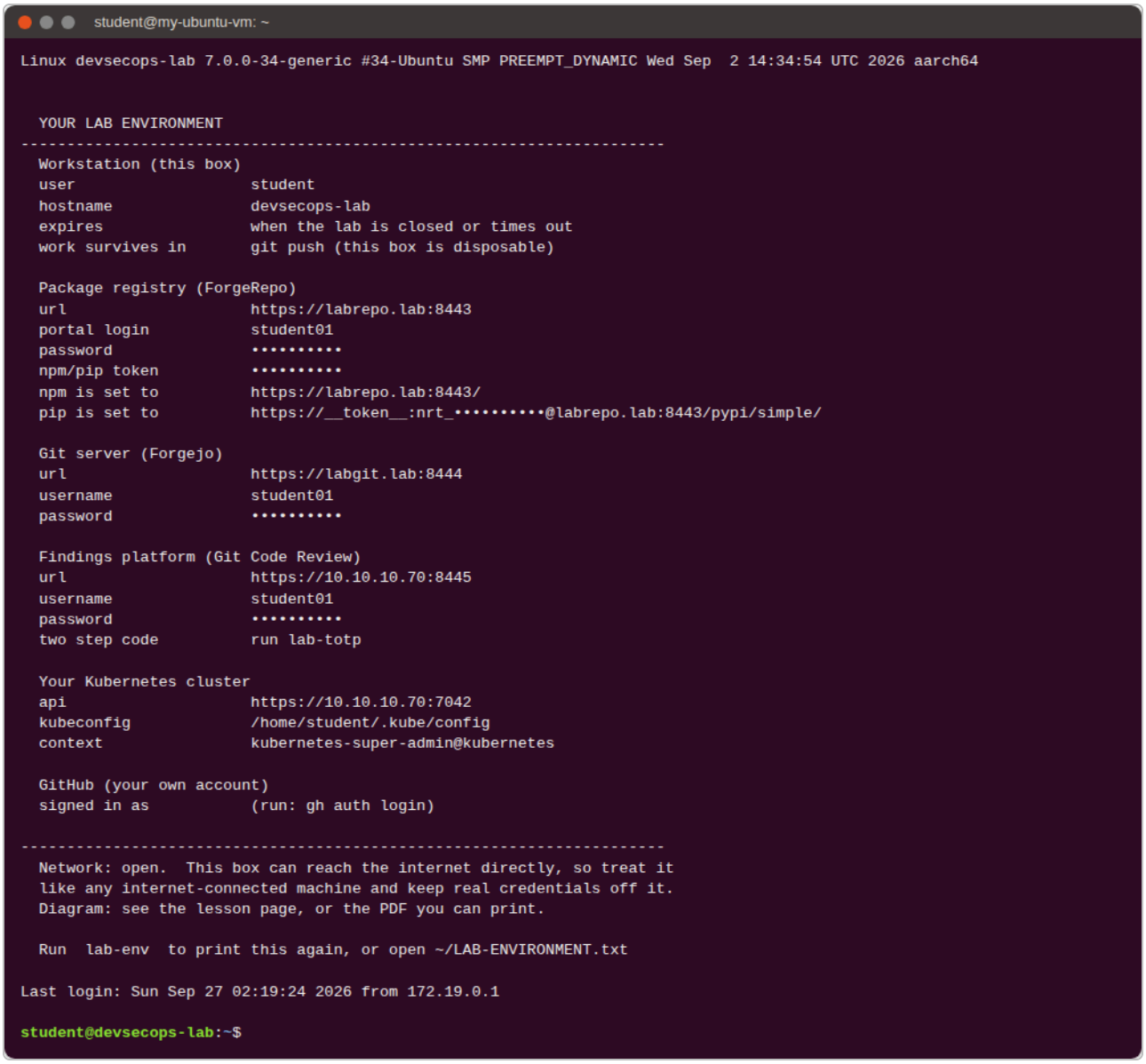


Figure 9. A successful SSH login. Look for the **yes** answer to the first-time question and the prompt changing to `student@devsecops-lab`. Addresses in this picture are hidden; **yours will be different**.

> **TIP: SHORTCUTS**
>
> `hackrange-lab ssh` shows the password and connects for you. `hackrange-lab shell` gives you a lab terminal with no password at all.

If SSH says **Connection refused**, the lab session is not running: type `hackrange-lab start` and try again. If it says **Permission denied**, the password has changed since you last looked: run `show_information.sh` again.

Remote Desktop

Remote Desktop shows the lab desktop in a separate app instead of a browser tab. Any Remote Desktop app works.

1. Open a Remote Desktop app. On Windows: open the Start menu, type **Remote Desktop Connection**, and open it. On a Mac, install Microsoft's free **Windows App** from the App Store.
2. For the computer address, type your VM's address followed by **:13389**, for example **192.168.64.5:13389**. (`show_information.sh` prints it under **Or any Remote Desktop app**.)
3. Sign in as **student** with your lab password.
4. If the app warns about the certificate, this is the same situation as the browser warning. Accept it for your lab only.

9 Everyday use

Starting and stopping the VM

- **To start:** open your virtualization app (VirtualBox, UTM, and so on), select **HackRange Lab**, and click **Start** (or the play button).
- **The lab starts by itself.** Give it a few minutes after Ubuntu starts. You do not need to run the installer again.
- **To stop:** inside Ubuntu, click the top-right corner, then the **power** icon, then **Power Off**. This is like shutting down a real computer, and it is the safest way.

Nothing in the lab times out. Your lab session keeps running, and comes back after a restart of the VM, until you stop it.

Useful lab commands

Type these in a terminal in Ubuntu. Some may ask for your Ubuntu password.

Command	What it does
`hackrange-lab info`	Shows the web address, username and current password (same as `show_information.sh`)
`hackrange-lab status`	Checks that every part of the lab is running. Each line that is down shows the command that fixes it.
`hackrange-lab desktop`	Opens the lab desktop in a window on the Ubuntu VM
`hackrange-lab start`	Starts a fresh lab session (with a new password)
`hackrange-lab stop`	Ends the session. The workstation is deleted, so push your work first.
`hackrange-lab reset`	Ends the session and starts a fresh one, with a new password
`hackrange-lab env`	Your lab addresses, usernames, passwords and tokens
`hackrange-lab totp`	The current two-step code for Git Code Review
`hackrange-lab update`	Gets the latest version of the lab
`hackrange-lab diag`	Saves a diagnostics file to send when you need help

Where your work lives

A lab session works like the hosted lab. The **workstation is disposable**: each session starts from a fresh one. Your work lives on the **lab Git server**. **Push your work before you run `hackrange-lab stop` or `hackrange-lab reset`.**

Your repositories, your package registry account, your findings and your Kubernetes cluster are **not** touched by `stop` or `reset`. They keep running in the background, and come back after a reboot on their own.

Check that everything is running

```
hackrange-lab status
```

Update the lab

Get the latest version of the lab. It is safe at any time and never touches a running session. It can take a while if there is a lot to download.

```
hackrange-lab update
```

After an update, the lab password is new. Run `show_information.sh` to see it.

Reset

- **A fresh workstation and a new password:** `hackrange-lab reset` (push your work first).
- **One broken service: Reset to default** on its tile in the control panel.
- **The whole lab, back to the beginning: Start over** at the bottom of the control panel.

How the Local Lab differs from the hosted labs

Almost nothing, on purpose. A few things you might notice:

- **Open internet.** The hosted labs allow only a short list of sites. Your Local Lab has normal internet access. On Day 6, `npm install --registry https://registry.npmjs.org left-pad` is meant to be refused. On the Local Lab it succeeds. Read the lesson's explanation of why it would fail in a locked-down environment.
- **Sessions do not expire.** A hosted lab closes after four hours. Yours runs until you stop it.
- **You are `student01`.** That matches the example output in the lessons.
- **ARM64 machines** (such as Apple Silicon Macs) show `arm64` where the lessons show `amd64` (for example on Days 8 and 9). On Days 5 and 20 the pipelines use an x86 tool on purpose; it works, a little slower.

10 Keep your lab safe

Your lab is full of deliberately weak software. That is what makes it a good place to learn, and also why it must stay private. Please read this once.

Who can reach it

The lab's doors (the web desktop on port 8446, SSH on 2222, Remote Desktop on 13389, and the lab services on 8443 to 8452) only answer **private** network addresses, never the internet. But "private" means everyone on the same network. At home, that is your family's devices. On a school, office or cafe network, it can be hundreds of strangers. On a shared network, give the VM a **NAT** or **host-only** network (section 3) so only your own computer can reach it.

Passwords

- The lab password changes at every new session and every install. The other services' passwords are made fresh for each install.
- None of them protects anything but this lab. Never reuse them anywhere else.
- Keep the password to yourself. Do not paste it into a chat, a forum, or a screenshot you share.
- If you think someone saw it, push your work and run `hackrange-lab reset`. You get a new one.

Never expose it to the internet

- Do not set up **port forwarding** on your home router for these ports.
- Do not put this VM on a public cloud server.
- If the Ubuntu VM ever sits directly on the internet, the lab refuses to answer your own computer at all. Open it from inside the VM instead, or ask your instructor about an SSH tunnel.

Apple Silicon and other ARM machines

The installer turns on a setting (`vm.legacy_va_layout`) that some x86 tools need to run on ARM. It weakens one of Linux's memory protections for the whole VM. That is one more reason the lab belongs in a virtual machine of its own.

The fake secrets

The lab contains fake passwords, tokens and keys, planted for you to find with secret scanners. None of them is real, and none grants access to anything. If a scanner flags them, that is the point of the exercise.

When you finish the course

Run the uninstaller, remove the lab certificate if you trusted it, and delete the virtual machine. Section 12 walks you through it.

11 Troubleshooting

Something went wrong? Do not worry: almost everything here has a simple fix. Find what you see in the left column.

If you see ...	Do this
The installer stopped with a red `[stopped]` message	Read the message and fix what it says. Then run the same install command again. It picks up where it left off. The full log is in `/var/log/hackrange-install.log`.
"This virtual machine is smaller than the lab needs"	Shut the VM down. In your virtualization app, give it more memory, cores or disk (section 2 has the sizes). Start it, and run the install command again.
Not enough memory on your computer to give the VM 16 GB	Close other apps, or use a computer with more memory. You can also use the hosted labs in the course, which need nothing on your computer.
"Cannot reach https://github.com" (or another site), or the VM has no internet	Open Firefox inside Ubuntu and try any web page. If it fails, check the VM's network: **Adapter 1** should be **NAT** (VirtualBox) or **Shared Network** (UTM). Check that your own computer is online. Then run the install command again.
The VM will not start, or says virtualization (VT-x, AMD-V) is not available	Hardware virtualization is turned off. On Windows, check Task Manager, **Performance**, **CPU**, **Virtualization**. Turn it on in your computer's BIOS/UEFI settings (search the web for your computer's model plus "enable virtualization").
"This looks like Windows Subsystem for Linux (WSL)"	WSL cannot run the lab. Create a real virtual machine (section 3a).
"Port ... is already used by ..."	Another program in the VM is using a port the lab needs. Stop or remove that program, and run the install again. A fresh VM avoids this.
The install seems stuck	Some steps take a long time with no new output, especially "Building the lab images". Wait. If the computer went to sleep or the VM was closed, run the install command again.
Nothing happens when I paste in the terminal	Use `Ctrl` + `Shift` + `V`, or right-click and choose **Paste**. If the command was copied on your own computer, copy it inside the VM instead (from the GitHub page in Ubuntu's Firefox).
Nothing appears when I type my password	That is normal for `sudo` and SSH. Type it and press `Enter`.
I forgot the lab password, or it does not work	It changes every time a fresh session starts. Run `/opt/hackrange-labs/show_information.sh` to see the current one. After five wrong tries the web desktop makes you wait five minutes.

"Your connection is not private" in the browser	Expected for your lab. For **your lab's address only**, click **Advanced**, then **Continue** (section 6). Never do this for other sites.
The web address does not open on my own computer	Try `https://127.0.0.1:8446` inside the Ubuntu VM first. If that works, fix the VM's network: in VirtualBox, add **Adapter 2** as **Host-only Adapter** (section 3a), then run `show_information.sh` to see the new address. VMware, Parallels and UTM work as they are; Hyper-V's **Default Switch** works too.
`https://labgit.lab:8444` does not open on my own computer	Those names only work inside the lab. On your own computer, use `https://<VM address>:8444` (run `show_information.sh` to see it).
"permission denied" from docker, or lab commands need sudo	Log out of Ubuntu and back in once after the install.
Something stopped working, or after a reboot	Run `hackrange-lab status`. Each line that is down shows the command that fixes it.
A lab says a command failed to reach `labgit.lab` or `labrepo.lab`	Run `hackrange-lab status`, then `hackrange-lab reset`.
SSH says "Connection refused"	The lab session is not running. Run `hackrange-lab start` and try again.

Still stuck? Ask for help

1. In a terminal in Ubuntu, run:

   ```
   hackrange-lab diag
   ```

2. It prints **Diagnostics saved to:** and a file name ending in `.tar.gz`. That file holds the install log and the state of every part of the lab, with every password, token and key removed.
3. Ask for help in the course (the course page), describe what you did and what you saw, and send that file.

12 Removing the lab when the course ends

Congratulations on finishing! Cleaning up takes three short parts.

> **WATCH OUT**
>
> Removing the lab deletes **everything** in it, including your repositories on the lab Git server. Copy anything you want to keep somewhere else (for example, to GitHub) first.

Part 1: Run the uninstaller

1. Open a terminal in Ubuntu.
2. Run this command, and type your Ubuntu password if asked:

   ```
   sudo /opt/hackrange-labs/uninstall.sh
   ```

3. It warns you that it removes everything. Type **REMOVE** (in capital letters) and press `Enter`. Anything else cancels, and nothing is removed.
4. It prints each thing it removes. When you see **The Local Lab has been removed. Docker is still installed.**, it is done.

Part 2: Remove the lab certificate (only if you trusted it)

If you followed the optional steps in section 6, remove the certificate from the same place you added it:

- **Windows:** press `Windows` + `R`, type **certmgr.msc**, press `Enter`. Open **Trusted Root Certification Authorities**, then **Certificates**. Right-click **Hackrange Local Lab CA** and choose **Delete**.
- **Mac:** open **Keychain Access**, click **login**, find **Hackrange Local Lab CA**, and delete it.
- **Firefox: Settings**, search for **certificates**, **View Certificates**, **Authorities**, select it, and click **Delete or Distrust**.

Part 3: Delete the virtual machine

- **VirtualBox:** make sure the VM is powered off. Right-click **HackRange Lab**, choose **Remove**, then **Delete all files**.
- **UTM:** make sure the VM is stopped. Right-click it and choose **Delete**, then confirm.
- **virt-manager:** make sure the VM is shut off. Right-click it, choose **Delete**, keep the disk ticked, and click **Delete**.
- **VMware or Parallels:** use its **Delete** (or **Move to Trash**) option for the VM.

Deleting the VM frees up the disk space on your computer. You can also delete the Ubuntu `.iso` file from your Downloads folder.

13 Glossary

ARM64 / Apple Silicon

A family of processors used in newer Macs (M1 and later) and some other computers. They need software built for ARM64.

BIOS / UEFI

The setup screen built into your computer that runs before the operating system. Hardware virtualization is turned on or off there.

Browser certificate

A digital ID a website shows to prove who it is. Your lab makes its own, so your browser warns you until you trust it.

Certificate authority (CA)

Something that vouches for certificates. Your lab creates its own private CA, which can only vouch for the lab itself.

CI runner / pipeline

CI (continuous integration) automatically builds and tests code every time it changes. A pipeline is the list of steps, and the runner is the program that runs them.

Command

An instruction you type into a terminal and run by pressing Enter.

Docker / container

Docker runs programs in containers: sealed boxes that hold a program and everything it needs. The lab's parts run in containers.

Fingerprint

A short, unique code calculated from a certificate. If two fingerprints match exactly, the certificates are the same.

Forgejo (Git server)

The lab's own code-hosting website, similar to GitHub. Your lab work is stored there.

ForgeRepo (package registry)

A storage place for software packages that programs download and reuse.

Git / repository

Git tracks changes to code. A repository (repo) is one project's folder of code and its history. To "push" is to send your changes to the Git server.

Git Code Review / SAST

SAST (static application security testing) scans source code for security bugs without running it. Git Code Review is the lab's SAST platform.

Grafana, Loki, Tempo

Monitoring tools. Grafana draws dashboards, Loki stores logs, and Tempo stores traces (the path a request takes through a system).

Hardware virtualization (VT-x, AMD-V)

A processor feature that lets virtual machines run fast. It must be turned on.

Host-only network

A VM network setting where only your own computer can reach the VM.

IP address

A number that identifies a computer on a network, such as `192.168.64.5`. `127.0.0.1` always means "this same computer".

ISO file

A single file that holds a whole installation disc, such as Ubuntu's installer.

Keycloak

A login system that handles usernames, passwords and sign-in for other apps.

Kong

An API gateway: a front door that checks and routes requests to other services.

Kubernetes (k3s, vcluster)

A system that runs and manages many containers across machines. k3s is a small Kubernetes, and vcluster gives you your own cluster inside it.

Linux

A free family of operating systems. Ubuntu is one kind of Linux.

NAT network

A VM network setting where the VM reaches the internet through your computer, but other devices cannot reach the VM.

Port

A numbered door on a computer. One computer can run many services, each answering on its own port, such as 8446 or 2222.

Port forwarding

A router setting that lets the internet reach a device inside your home. Never use it for this lab.

Private network

A network, such as your home or office network, that the internet cannot reach directly.

Remote Desktop (RDP)

A way to see and use another computer's desktop in a window on your own computer.

Session (lab session)
: One run of your lab workstation, from start to stop. Each new session gets a fresh workstation and a new password.

SSH
: Secure Shell: a safe way to type commands on another computer over a network.

sudo
: Short for "superuser do". Put in front of a command, it runs that command as the administrator, after asking for your Ubuntu password.

Terminal
: A window where you type commands instead of clicking. In Ubuntu, open it with Ctrl + Alt + T.

Two-step code (TOTP)
: A short code that changes every few seconds, used as a second proof of identity after a password.

Ubuntu
: A popular, free version of Linux. The lab runs inside an Ubuntu virtual machine.

URL / web address
: What you type in a browser's address bar, such as `https://192.168.64.5:8446`. The number after the colon is the port.

Virtual machine (VM)
: A pretend computer that runs inside an app on your real computer, with its own memory, disk and operating system.

Virtualization app
: The program that runs virtual machines, such as VirtualBox, UTM, VMware, Parallels or virt-manager.

WSL
: Windows Subsystem for Linux, a way to run some Linux programs on Windows. It cannot run this lab.

HackRange Local Lab: Installation Guide. Author: Tim Rice. Repository: github.com/hackrange/lab_devsecops