RIT | Golisano College of Computing and Information Sciences
School of Information

NSSA221 Systems Administration I

Lab 03: Git & Linux Local Account Management

INTRODUCTION

Modern systems administrators must be equally comfortable with version-control workflows and day-to-day host management. This lab combines both skill sets: you will configure Git and the GitHub CLI for script tracking, manage local accounts through Cockpit, and practice emergency password recovery from the GRUB prompt. By the end you will have a fully documented, end-to-end workflow that mirrors real-world administration on Rocky Linux 8.

LAB SUMMARY

This lab guides you through configuring Git + GitHub on Rocky Linux, managing local accounts locally and remotely with Cockpit, and performing an emergency *root*-password reset—equipping you with practical, end-to-end skills for real-world systems administration.

GOALS

Apply – Install Git on Rocky Linux and configure global defaults (user.name, user.email, main branch).

Create – Generate a 90-day GitHub Personal Access Token and store it securely outside any repository.

Apply – Install the GitHub CLI (gh), authenticate with the PAT, and verify access (gh auth status).

Create – Initialize a local repository, commit ping_test.py, and push the main branch to GitHub.

Apply – Use Cockpit's Accounts module to add a new local user through the web interface.

Apply / Analyze – Remotely connect to Cockpit from Windows 11, open the web terminal, and add a user with useradd, documenting network details.

Apply / Evaluate – Regain root access by booting into single-user mode (rd.break), resetting the password, and ensuring SELinux relabeling on reboot.

PREREQUISITES

Ensure the four VMs created, configured, and updated in Labs 1 & 2 are running correctly before beginning this lab. You do not need to create any new VMs.

Working Rocky Linux 9 (or newer) VM with internet access

Ability to reboot the VM and view the GRUB menu (VM console or local KVM)

Windows 11 host machine with a modern web browser

Basic command-line familiarity (dnf, systemctl, sudo)

Previous completion of Lab 02 networking connectivity tests

ACTIVITY SUMMARY

ACTIVITY 1: INSTALL GIT AND CONFIGURE OPTIONS

INTRODUCTION

In this first activity you will install Git—the industry-standard version-control tool—on your Rocky Linux virtual machine and configure a few global options. Setting your user name, email address, and default branch ensures every future commit is properly attributed and conforms to modern Git conventions (using main instead of the deprecated master). Completing these steps lays the foundation for all remaining scripting and source-control work in this course.

PROCEDURE

1. If you have not already done so, update the Rocky Linux virtual machine.

2. Install Git on your Rocky Linux VM.

```
sudo dnf install git -y
```

3. Confirm the Git version.

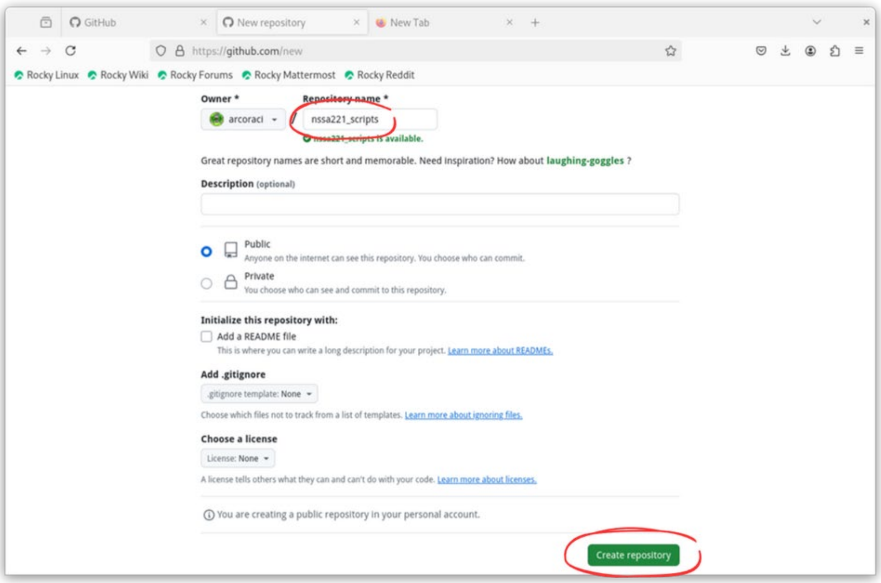
```
git --version
```

4. Configure your identity and set the default branch to main.

```
git config --global user.name “<full name>”
git config --global user.email <rit email>
git config --global init.defaultBranch main
git config --global push.default matching
```

5. Log in to GitHub on the VM and create a repository named ***nssa221_scripts***. Select **Create repository** (Figure 1). Please note Figure 1 shows public, you may opt to make it a **private** repository.

Figure 1 – Create the Git Repository



Owner * arcoraci / Repository name * nssa221_scripts

nssa221_scripts is available.

Great repository names are short and memorable. Need inspiration? How about **laughing-goggles** ?

Description (optional)

[✓ Public] Anyone on the internet can see this repository. You choose who can commit.

Private You choose who can see and commit to this repository.

Initialize this repository with:

☐ Add a README file
This is where you can write a long description for your project. Learn more about READMEs.

Add .gitignore

.gitignore template: None

Choose which files not to track from a list of templates. Learn more about ignoring files.

Choose a license

License: None

A license tells others what they can and can't do with your code. Learn more about licenses.

You are creating a public repository in your personal account.

CONCLUSION

At this point Git is fully installed, your personal identity is configured, and the default branch name is set to main. You are now ready to create repositories, commit code, and collaborate confidently for the rest of the semester.

DELIVERABLES

None.

ACTIVITY 2: CREATE A PERSONAL ACCESS TOKEN

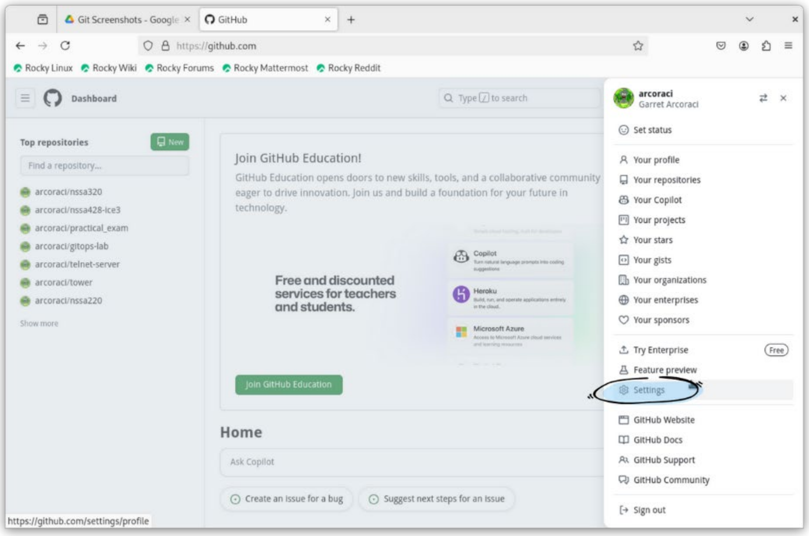
INTRODUCTION

Modern GitHub workflows rely on Personal Access Tokens (PATs) rather than account passwords for command-line and API authentication. In this activity, you will generate a PAT with broad scopes that will cover every task you perform in the course: pushing code, managing repositories, and interacting with GitHub-based automations. Storing the token in a local file (outside any repository) keeps the credential handy while protecting it from accidental exposure.

PROCEDURE

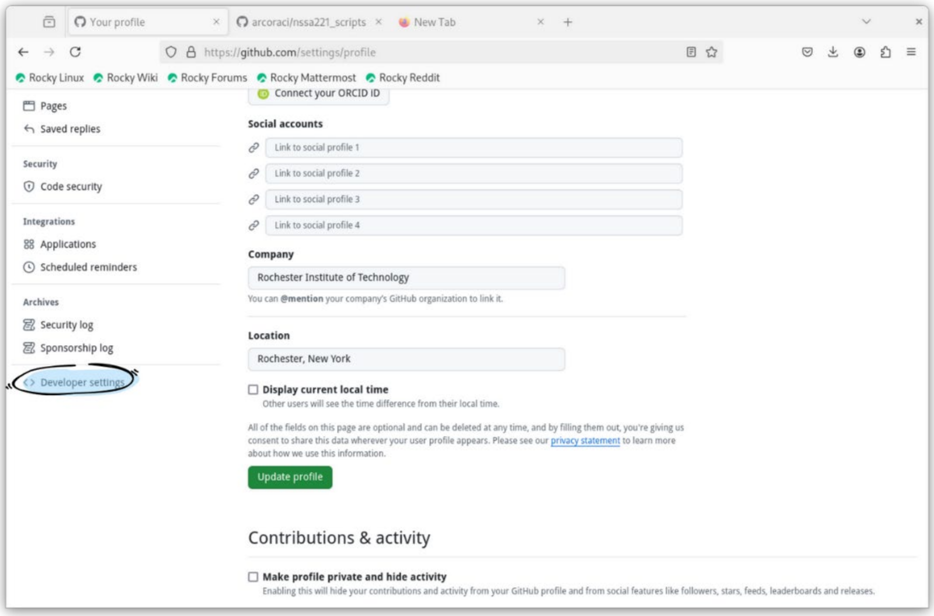
1. From the GitHub Dashboard, select your account and go to settings (Figure 2).

Figure 2 –Git Dashboard/Settings



2. Scroll to the bottom to find **Developer Settings** (Figure 3).

Figure 3 – Developer Settings



Company: Rochester Institute of Technology

You can @mention your company's GitHub organization to link it.

Location: Rochester, New York

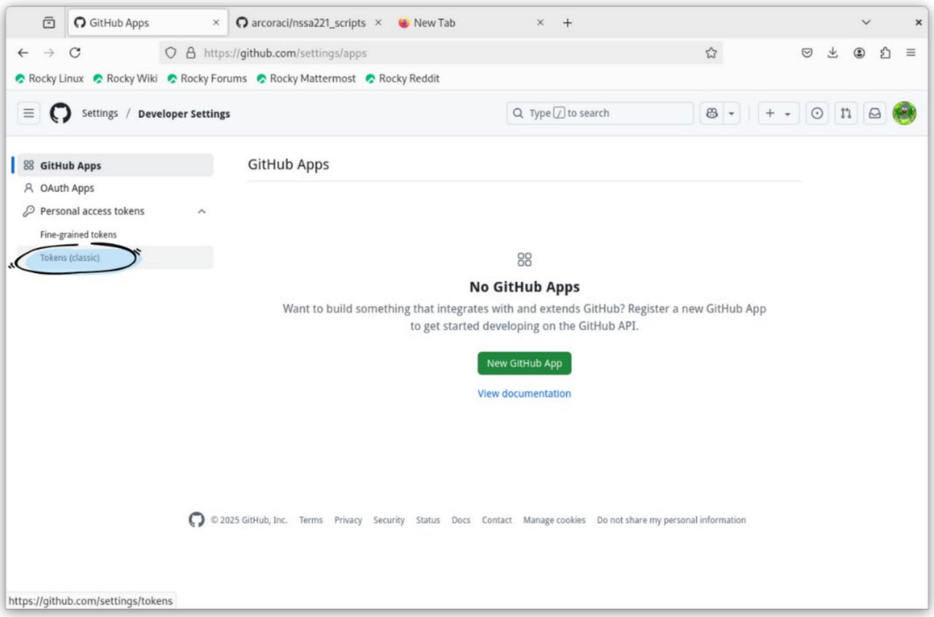
☐ Display current local time
Other users will see the time difference from their local time.

All of the fields on this page are optional and can be deleted at any time, and by filling them out, you're giving us consent to share this data wherever your user profile appears. Please see our privacy statement to learn more about how we use this information.

☐ Make profile private and hide activity
Enabling this will hide your contributions and activity from your GitHub profile and from social features like followers, stars, feeds, leaderboards and releases.

3. Open Developer settings and choose Tokens (classic), Figure 4.

Figure 4 – Token (classic)



GitHub Apps | arcoraci/nssa221_scripts | New Tab

https://github.com/settings/apps

Rocky Linux · Rocky Wiki · Rocky Forums · Rocky Mattermost · Rocky Reddit

Settings / Developer Settings

- GitHub Apps
- OAuth Apps
- Personal access tokens
  - Fine-grained tokens
  - Tokens (classic)

GitHub Apps

No GitHub Apps

Want to build something that integrates with and extends GitHub? Register a new GitHub App to get started developing on the GitHub API.

New GitHub App

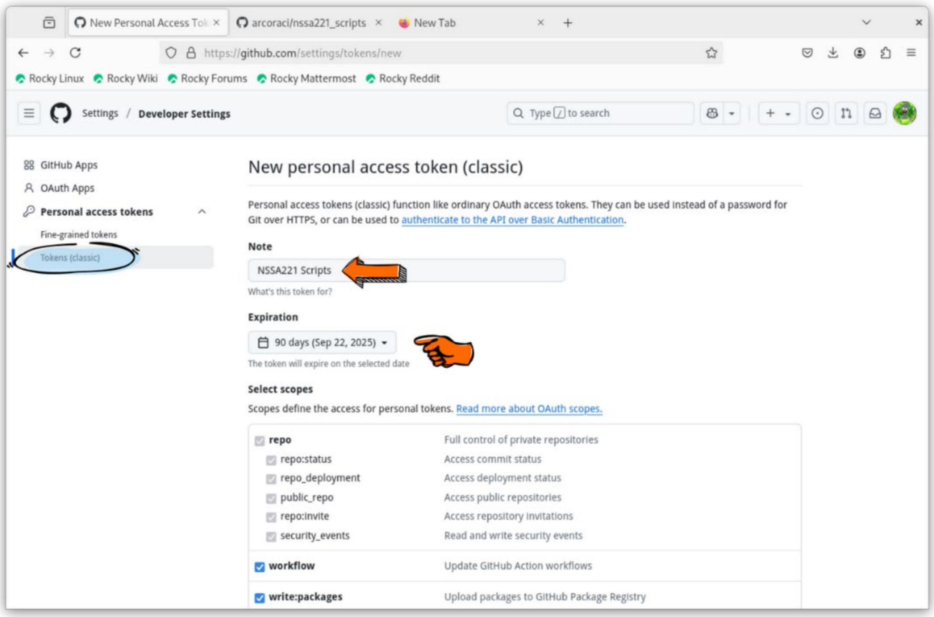
View documentation

© 2025 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information

https://github.com/settings/tokens

4. Configure your Personal Access Token (PAT).
   a. Optional: Add a descriptive note so you can recognize the token later.
   b. **Set expiration** to **90 days** (covers the whole semester).
   c. Under “Select scopes,” select All scopes to grant every required permission in one click.

Figure 5 – Configure Token



New Personal Access Tok | arcoraci/nssa221_scripts | New Tab

https://github.com/settings/tokens/new

Settings / Developer Settings

- GitHub Apps
- OAuth Apps
- Personal access tokens
  - Fine-grained tokens
  - Tokens (classic)

New personal access token (classic)

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.

Note: NSSA221 Scripts
What's this token for?

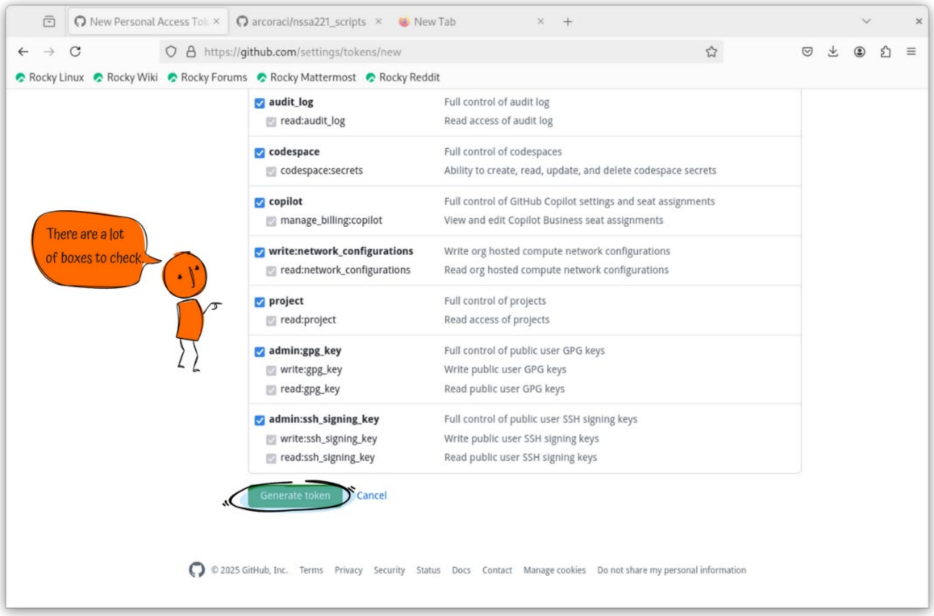
Expiration: 90 days (Sep 22, 2025)
The token will expire on the selected date

Select scopes
Scopes define the access for personal tokens. Read more about OAuth scopes.

☑ repo	Full control of private repositories
☑ repo:status	Access commit status
☑ repo_deployment	Access deployment status
☑ public_repo	Access public repositories
☑ repo:invite	Access repository invitations
☑ security_events	Read and write security events
☑ workflow	Update GitHub Action workflows
☑ write:packages	Upload packages to GitHub Package Registry

5. Scroll down and click **Generate token**, Figure 6.

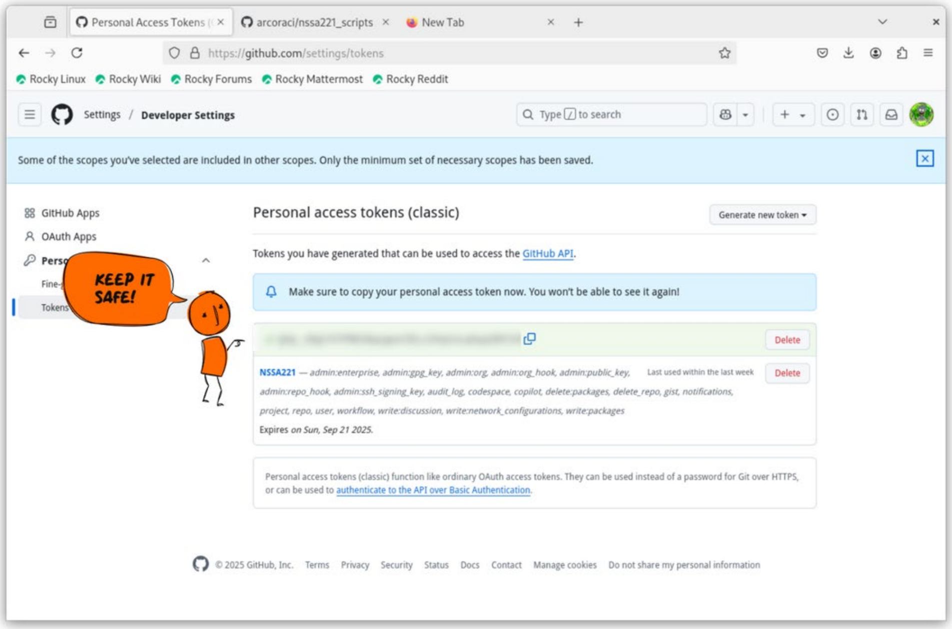
Figure 6 – Generate Token


New Personal Access Tok
arcoraci/nssa221_scripts
New Tab
https://github.com/settings/tokens/new
Rocky Linux
Rocky Wiki
Rocky Forums
Rocky Mattermost
Rocky Reddit
There are a lot of boxes to check
Generate token
Cancel
© 2025 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information

☑ **audit_log**	Full control of audit log
☑ read:audit_log	Read access of audit log
☑ **codespace**	Full control of codespaces
☑ codespace:secrets	Ability to create, read, update, and delete codespace secrets
☑ **copilot**	Full control of GitHub Copilot settings and seat assignments
☑ manage_billing:copilot	View and edit Copilot Business seat assignments
☑ **write:network_configurations**	Write org hosted compute network configurations
☑ read:network_configurations	Read org hosted compute network configurations
☑ **project**	Full control of projects
☑ read:project	Read access of projects
☑ **admin:gpg_key**	Full control of public user GPG keys
☑ write:gpg_key	Write public user GPG keys
☑ read:gpg_key	Read public user GPG keys
☑ **admin:ssh_signing_key**	Full control of public user SSH signing keys
☑ write:ssh_signing_key	Write public user SSH signing keys
☑ read:ssh_signing_key	Read public user SSH signing keys

6. Copy the token to a **git_tokens** file in your **home directory**—outside any repository.

Figure 7 – Personal Access Token



Some of the scopes you've selected are included in other scopes. Only the minimum set of necessary scopes has been saved.

Personal access tokens (classic)

Tokens you have generated that can be used to access the GitHub API.

Make sure to copy your personal access token now. You won't be able to see it again!

[redacted]

NSSA221 — *admin:enterprise, admin:gpg_key, admin:org, admin:org_hook, admin:public_key, admin:repo_hook, admin:ssh_signing_key, audit_log, codespace, copilot, delete:packages, delete_repo, gist, notifications, project, repo, user, workflow, write:discussion, write:network_configurations, write:packages* — Last used within the last week

Expires *on Sun, Sep 21 2025.*

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.

7. When you work in the terminal, you usually paste this token whenever you're prompted for a password. In the next activity, we'll cache the token so you won't need to paste it each time a password prompt appears.

CONCLUSION

You have successfully generated and securely stored a 90-day Personal Access Token that grants the required permissions for this course. This token will replace your GitHub password in terminal operations until it expires, streamlining authentication while maintaining security.

DELIVERABLES

Please share your token with the class if you want to receive a zero on this lab. 😈

ACTIVITY 3: GITHUB CLI AND INITIAL LOGIN

INTRODUCTION

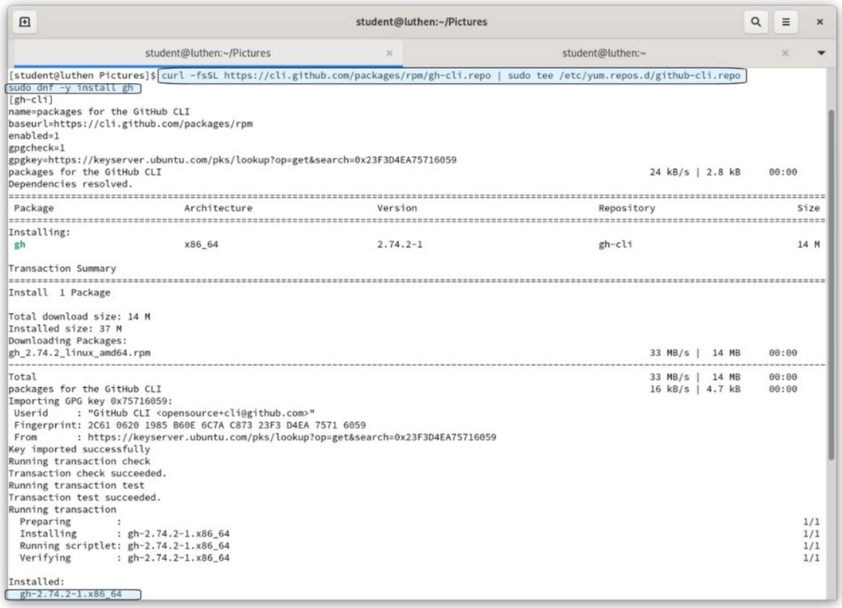
The GitHub CLI (gh) brings GitHub's web functionality—issues, pull requests, releases, and more—directly to your terminal. Installing it on Rocky Linux and authenticating with your PAT allows you to manage repositories and workflows without leaving the command line. This activity will guide you through adding the GitHub package repository, installing the CLI with **`dnf`**, and performing an initial login.

PROCEDURE

1. **Install** the GitHub CLI with the command below. (Copy it from Rocky docs if you want to avoid typos.)

```
curl -fsSL https://cli.github.com/packages/rpm/gh-cli.repo | sudo tee
/etc/yum.repos.d/github-cli.repo

sudo dnf -y install gh
```

Figure 8 – Installing GitHub CLI (gh) on Rocky



```
[student@luthen Pictures]$ curl -fsSL https://cli.github.com/packages/rpm/gh-cli.repo | sudo tee /etc/yum.repos.d/github-cli.repo
sudo dnf -y install gh
[gh-cli]
name=packages for the GitHub CLI
baseurl=https://cli.github.com/packages/rpm
enabled=1
gpgcheck=1
gpgkey=https://keyserver.ubuntu.com/pks/lookup?op=get&search=0x23F3D4EA75716059
packages for the GitHub CLI                                                         24 kB/s | 2.8 kB     00:00
Dependencies resolved.
=====================================================================================================================
 Package                 Architecture              Version                    Repository                        Size
=====================================================================================================================
Installing:
 gh                      x86_64                    2.74.2-1                   gh-cli                            14 M

Transaction Summary
=====================================================================================================================
Install  1 Package

Total download size: 14 M
Installed size: 37 M
Downloading Packages:
gh_2.74.2_linux_amd64.rpm                                                           33 MB/s |  14 MB     00:00
---------------------------------------------------------------------------------------------------------------------
Total                                                                               33 MB/s |  14 MB     00:00
packages for the GitHub CLI                                                         16 kB/s | 4.7 kB     00:00
Importing GPG key 0x75716059:
 Userid     : "GitHub CLI <opensource+cli@github.com>"
 Fingerprint: 2C61 0620 1985 B60E 6C7A C873 23F3 D4EA 7571 6059
 From       : https://keyserver.ubuntu.com/pks/lookup?op=get&search=0x23F3D4EA75716059
Key imported successfully
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                                             1/1
  Installing       : gh-2.74.2-1.x86_64                                                                          1/1
  Running scriptlet: gh-2.74.2-1.x86_64                                                                          1/1
  Verifying        : gh-2.74.2-1.x86_64                                                                          1/1

Installed:
  gh-2.74.2-1.x86_64
```

2. Verify the installation.

```
gh -version
```

3. Authenticate with GitHub:

```
gh auth login
```

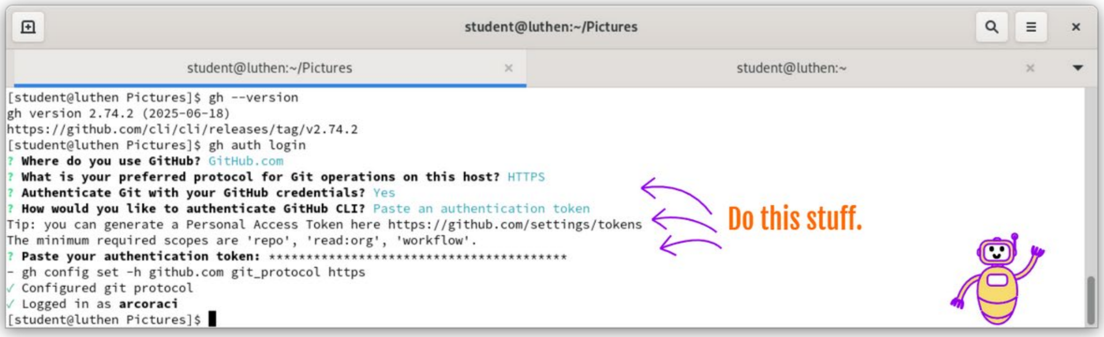
4. Respond to the **`gh auth login`** prompts.

Prompt	Response
Where do you use GitHub?	GitHub.com
Preferred protocol for Git operations on this host?	HTTPS
Authenticate Git with your GitHub credentials?	Yes
How would you like to authenticate GitHub CLI?	**Paste an authentication token** → paste the PAT you created in Activity 2.

5. After the last prompt you should see (See Figure 9):

```
Configured git protocol

Logged in as <your-GitHub-username>
```

Figure 9 – GitHub CLI Authentication Steps



```
[student@luthen Pictures]$ gh --version
gh version 2.74.2 (2025-06-18)
https://github.com/cli/cli/releases/tag/v2.74.2
[student@luthen Pictures]$ gh auth login
? Where do you use GitHub? GitHub.com
? What is your preferred protocol for Git operations on this host? HTTPS
? Authenticate Git with your GitHub credentials? Yes
? How would you like to authenticate GitHub CLI? Paste an authentication token
Tip: you can generate a Personal Access Token here https://github.com/settings/tokens
The minimum required scopes are 'repo', 'read:org', 'workflow'.
? Paste your authentication token: ****************************************
- gh config set -h github.com git_protocol https
✓ Configured git protocol
✓ Logged in as arcoraci
[student@luthen Pictures]$ █
```

CONCLUSION

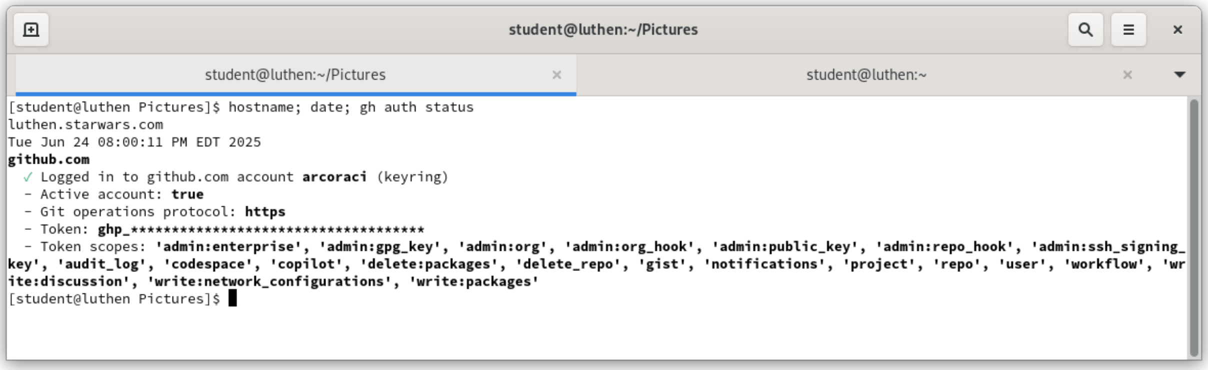
With **gh** installed and authenticated, you can now execute GitHub operations from the terminal just as easily as you use traditional Git commands. Verifying the login with **`gh auth status`** confirms that your CLI session is linked to your GitHub account and ready for upcoming automation and scripting tasks.

DELIVERABLES

For the report to validate completion of this activity enter the following command. See Figure 10 for expect output.

```
hostname; date; gh auth status
```

Figure 10 – Expected Output for GitHub CLI Authentication



```
[student@luthen Pictures]$ hostname; date; gh auth status
luthen.starwars.com
Tue Jun 24 08:00:11 PM EDT 2025
github.com
  ✓ Logged in to github.com account arcoraci (keyring)
  - Active account: true
  - Git operations protocol: https
  - Token: ghp_************************************
  - Token scopes: 'admin:enterprise', 'admin:gpg_key', 'admin:org', 'admin:org_hook', 'admin:public_key', 'admin:repo_hook', 'admin:ssh_signing_
key', 'audit_log', 'codespace', 'copilot', 'delete:packages', 'delete_repo', 'gist', 'notifications', 'project', 'repo', 'user', 'workflow', 'wr
ite:discussion', 'write:network_configurations', 'write:packages'
[student@luthen Pictures]$ 
```

ACTIVITY 4: GIT WORKFLOW

INTRODUCTION

In this activity you will complete the full Git workflow on your Rocky Linux VM: initialize a local repository, add an existing Python script, commit the change, and push the **main** branch to GitHub. These steps tie together everything you configured in the previous activities—Git, a Personal Access Token, and the GitHub CLI—so you can manage coursework files under version control from start to finish.

PROCEDURE

1. Create the project folder

   ```
   mkdir -p /home/student/nssa221_scripts
   ```

2. Enter the folder.

   ```
   cd /home/student/nssa221_scripts
   ```

3. Initialize the new Git repository.

   ```
   git init
   ```

4. Copy the script into the repo.

   ```
   cp /home/student/ping_test.py
   ```

5. Stage the script for commit.

   ```
   git add *
   ```

6. Verify the file is staged.

   ```
   git status
   ```

7. Commit the change with a message.

```
git commit -m “adding ping_test to repository.”
```

8. Rename the branch to main.

```
git branch -m main
```

9. Add the remote that points to your GitHub repository. Replace <your-username> with your actual GitHub user name.

```
git remote add origin https://github.com/<your-username>/nssa221_scripts.git
```

10. Push the commit to GitHub and set main as the upstream branch (running **`git status`** should report “nothing to commit, working tree clean.).

```
git push -u origin main
```

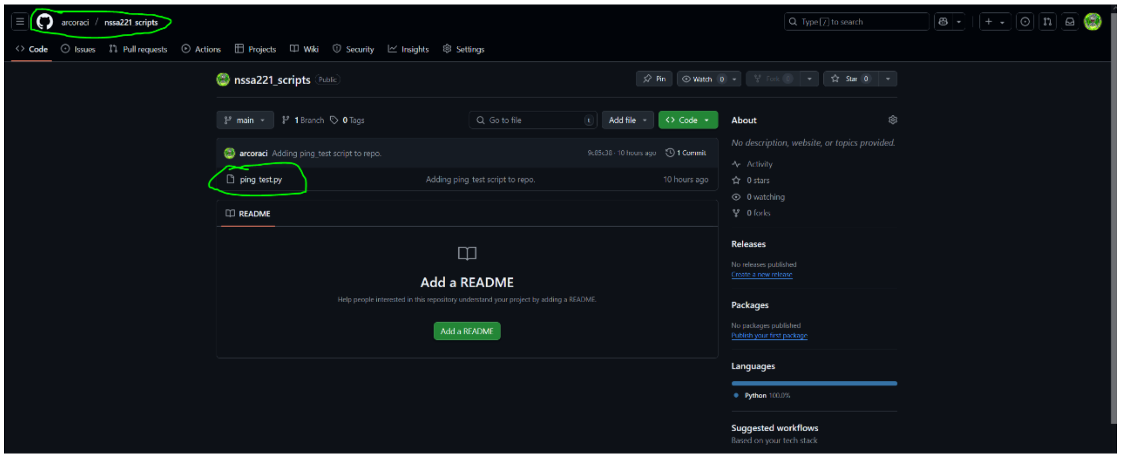
CONCLUSION

You have initialized a local repository, committed **ping_test.py**, and pushed the change to the remote **origin** on GitHub. Running git status now shows a clean working tree, confirming that your local and remote repositories are in sync. With this end-to-end workflow verified, you are ready to track and submit all future scripts for the remainder of the course.

DELIVERABLES

For the report include a screenshot with the GitHub page showing *ping_test.py* in the repository (see Figure 10 for example output). Your GitHub user name must be visible.

Figure 10 – *ping_test.py* Successfully Pushed to GitHub Repository



arcoraci Adding ping_test script to repo. 9c85c38 · 10 hours ago 1 Commit

ping_test.py Adding ping_test script to repo. 10 hours ago

ACTIVITY 5: USING COCKPIT

INTRODUCTION

Cockpit is a web-based interface that lets you perform common administration tasks on Rocky Linux without memorizing long shell commands. If Firefox blocks access with the ‘Advanced’ button not working, try using Chrome or Edge instead. In some cases, accessing via http:// works locally, while Chrome accepts https:// with the Advanced override. In this activity you will:

1. Verify that **Cockpit** is installed and running,
2. Enable its socket so the service starts automatically at boot,
3. Launch the web console, and
4. Create and test a new local user account—all from Cockpit's **Accounts** module.

Completing these steps gives you a quick, point-and-click alternative to dnf, systemctl, and user-management commands while reinforcing how those underlying tools work.

PROCEDURE

1. Check wether Cockpit is installed.

   ```
   dnf list installed cockpit\*     # lists any package whose name starts with "cockpit"
   dnf list installed | less        # fully scrollable list if you need it
   ```

2. Verify that the Cockpit socket is active.

   ```
   systemctl status cockpit.socket
   ```

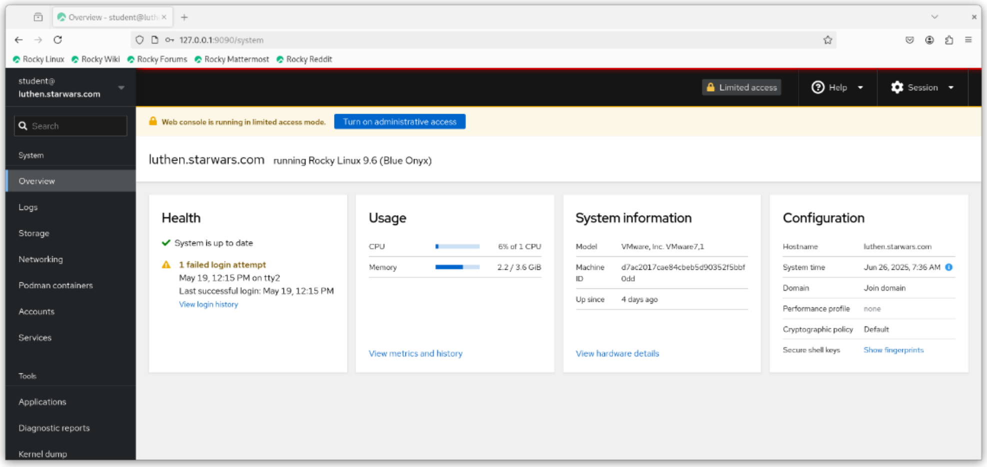
3. Enable Cockpit at boot and start it now.

   ```
   systemctl enable --now cockpit.socket
   ```

4. Need more information on dnf and systemctl?
   a. `man dnf`
   b. **man systemctl**
   c. Cheat-sheets are posted in MyCourses → Resources.

5. Launch the web console (Figure 11).
   a. Open a browser on the VM and visit https://127.0.0.1:9090.
   b. Click Turn on administrative access, log in with your system credentials.

Figure 11 – Cockpit Web Console



Web console is running in limited access mode.

luthen.starwars.com running Rocky Linux 9.6 (Blue Onyx)

Health

System is up to date

1 failed login attempt
May 19, 12:15 PM on tty2
Last successful login: May 19, 12:15 PM

Usage

CPU	6% of 1 CPU
Memory	2.2 / 3.6 GiB

System information

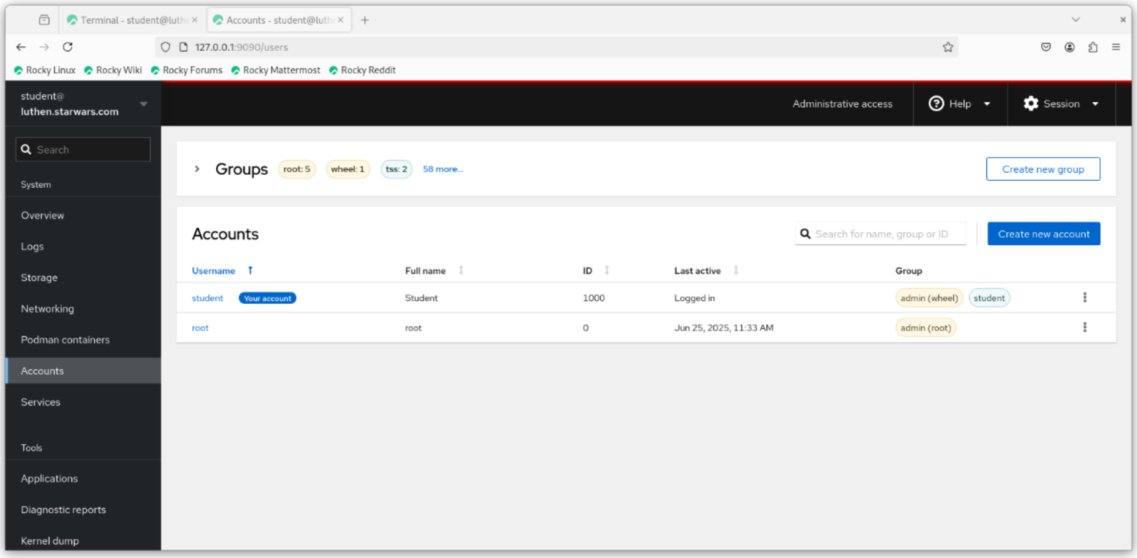
Model	VMware, Inc. VMware7,1
Machine ID	d7ac2017cae84cbeb5d90352f5bbf0dd
Up since	4 days ago

Configuration

Hostname	luthen.starwars.com
System time	Jun 26, 2025, 7:36 AM
Domain	Join domain
Performance profile	none
Cryptographic policy	Default
Secure shell keys	Show fingerprints

6. In the left-hand navigation pane, select **System → Accounts** (Figure 12).

Figure 12 - Accounts



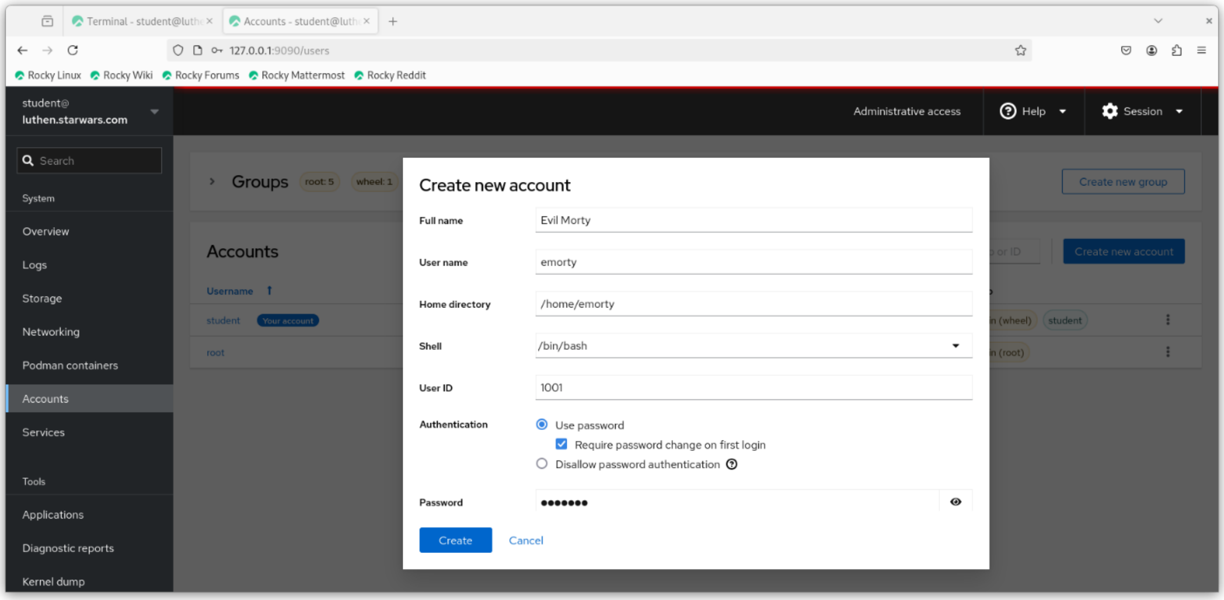
Accounts

Username	Full name	ID	Last active	Group
student Your account	Student	1000	Logged in	admin (wheel) student
root	root	0	Jun 25, 2025, 11:33 AM	admin (root)

7. Click Create new account.

8. Fill out the Create new account form.
   a. **Full name** – Human-readable name (e.g., "Evil Morty").
   b. **User name** – UNIX login (lowercase, no spaces).
   c. **Home directory** – Default is **/home/ *username***; adjust only if you have a custom path.
   d. **Shell** – Choose **/bin/bash**.
   e. **User ID** – Leave blank to auto-assign the next free UID.

Figure 13 – Create New Account Form


Create new account

Full name	Evil Morty
User name	emorty
Home directory	/home/emorty
Shell	/bin/bash
User ID	1001
Authentication	● Use password ☑ Require password change on first login ○ Disallow password authentication
Password	•••••••

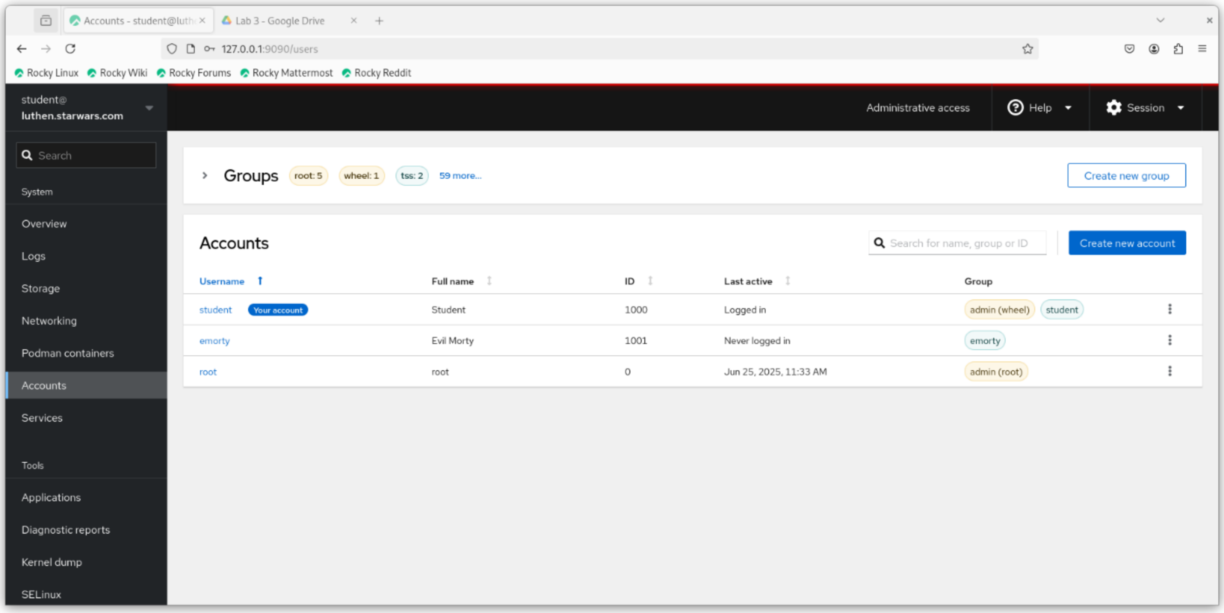
9. To configure authentication, select Use password.

10. Optionally, check **Require password change on first login**.

11. Enter and confirm password in the Password field.

12. Click Create. Cockpit returns you to the **Accounts** page, where the new user now appears in the list (Figure 14).

Figure 14 – Cockpit "Accounts" Page Showing Newly Created User


Groups root: 5 wheel: 1 tss: 2 59 more...

Accounts

Username	Full name	ID	Last active	Group
student Your account	Student	1000	Logged in	admin (wheel) student
emorty	Evil Morty	1001	Never logged in	emorty
root	root	0	Jun 25, 2025, 11:33 AM	admin (root)

13. Open **Terminal**. If you're currently logged in as **student**, switch to the new account:

```
su <new user>              # the dash (-) loads the new user's full environment
```

14. Enter **<newuser>**'s password. Verify the switch succeeded:

```
whoami
```

15. Type exit (or press **Ctrl-D**) to leave the new user's shell and return to **student**. Run `whoami` again if you want to confirm.

CONCLUSION

In this activity, you have confirmed that Cockpit is installed, enabled its socket, and accessed the web console at https://127.0.0.1:9090. Inside the interface you created a new user, assigned a password, and verified the account by switching to it in a terminal. Cockpit is now fully operational on your VM, providing an intuitive dashboard for future tasks such as monitoring logs, managing storage, and administering additional users.

DELIVERABLES

Include a screenshot of Cockpit's **Accounts** page in your lab report. The image must clearly show all of the following:

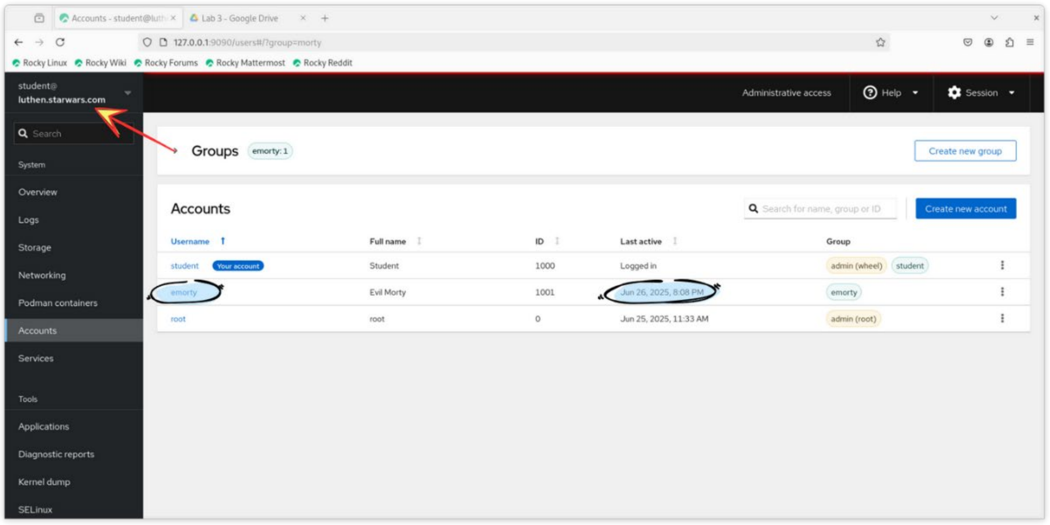
a) The newly created user listed in the table.

b) A visible **Last active** timestamp for that user.

c) The server's full **Rocky Linux FQDN** in the upper-left corner of the Cockpit interface.

(See Figure 15 for an example.)

Figure 15 – Account Creation Verification



Username	Full name	ID	Last active	Group
student Your account	Student	1000	Logged in	admin (wheel) student
emorty	Evil Morty	1001	Jun 26, 2025, 8:08 PM	emorty
root	root	0	Jun 25, 2025, 11:33 AM	admin (root)

ACTIVITY 6: CONNECT TO COCKPIT REMOTELY

INTRODUCTION

In this activity you will manage your Rocky Linux VM entirely from a Windows 11 host. After opening Cockpit in a browser (https://<vm-ip>:9090), you'll launch its built-in **Terminal** tool and create a new local user with classic command-line utilities (`useradd`, `passwd`, and `usermod`). Recall that Firefox may block access with the 'Advanced' button not working, try using Chrome or Edge instead. In some cases, accessing via http:// works locally, while Chrome accepts https:// with the Advanced override. This exercise reinforces three key skills:

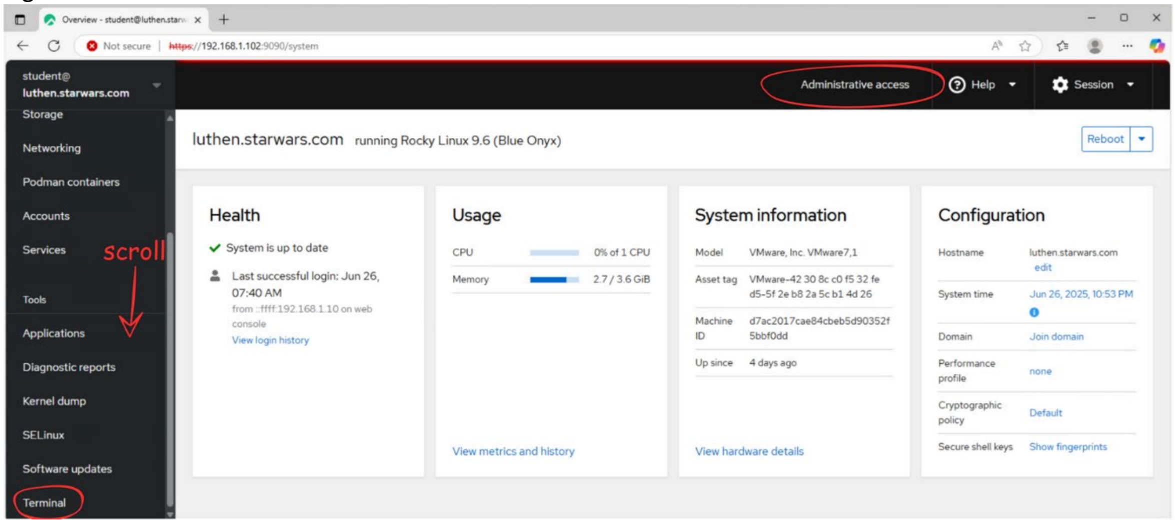
1. **Remote administration**—accessing Cockpit securely over the network.
2. **Web-terminal workflows**—running root-level commands without SSH.
3. **Linux account management**—adding users and granting optional sudo rights from the command line.

Completing these steps demonstrates that you can administer your VM from any system with a web browser, while still leveraging the full power of standard Linux commands.

PROCEDURE

1. **Open a browser** on Windows 11 (Edge, Chrome, Firefox).
2. Browse to https://<rocky linux IP>:9090. Ignore the HTTPS warning and click **Continue**; Cockpit uses a self-signed certificate by default.
3. Log in with your administrative Linux account (student), or any user who is a member of the wheel group. Confirm that the black banner shows administrative access.

Figure 16 – Administrative Access



luthen.starwars.com running Rocky Linux 9.6 (Blue Onyx)

Health
System is up to date
Last successful login: Jun 26, 07:40 AM
from ::ffff:192.168.1.10 on web console

Usage
CPU 0% of 1 CPU
Memory 2.7 / 3.6 GiB

System information

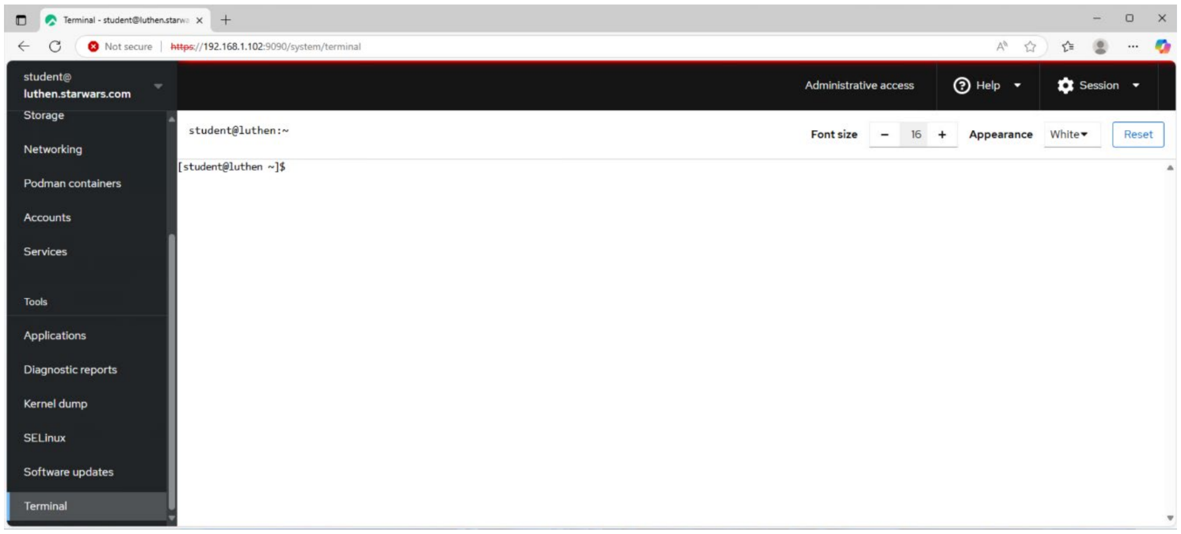
Model	VMware, Inc. VMware7,1
Asset tag	VMware-42 30 8c c0 f5 32 fe d5-5f 2e b8 2a 5c b1 4d 26
Machine ID	d7ac2017cae84cbeb5d90352f 5bbf0dd
Up since	4 days ago

Configuration

Hostname	luthen.starwars.com edit
System time	Jun 26, 2025, 10:53 PM
Domain	Join domain
Performance profile	none
Cryptographic policy	Default
Secure shell keys	Show fingerprints

4. In the left navigation menu, scroll down to **Tools** and click **Terminal**.
5. A full-screen shell opens (Figure 17); you are now on the Rocky Linux command line from your browser.

Figure 17 – Rocky Terminal



```
[student@luthen ~]$
```

6. Add the user using the **`useradd`** command (replace placeholders).

   ```
   sudo useradd -m -c “Full Name” -s /bin/bash <newuser>
   ```

7. Set the password.

   ```
   sudo passwd <newuser>
   ```

8. (Optional) Grant sudo rights by adding the user to the wheel group.

   ```
   sudo usermod -aG wheel <newuser>
   ```

9. Verify the account.

   ```
   id <newuser>                    # shows UID, groups
   su - <newuser>                  # switches to the new account
   whoami                          # prints <new user>
   exit                            # returns to original shell user account (student)
   ```

CONCLUSION

You successfully connected to Cockpit from Windows 11, verified administrative access, and used the web terminal to:

- Create a new user with useradd –m –c "<Full Name>" –s /bin/bash <username>,
- Set that user’s password,
- (Optionally) add the account to the **wheel** group for sudo privileges, and
- Confirm the new identity with id, su - <username>, and whoami.

The VM now contains the additional account, and Cockpit's terminal proves you can perform secure, remote command-line administration without relying on a separate SSH client. You are ready to employ Cockpit for other remote management tasks—log monitoring, storage checks, and service control—directly from your host workstation.

DELIVERABLES

For the report, submit one screenshot that clearly shows all of the items below:

a) The browser address bar showing the IP address of the Rocky Linux VM.

b) The Windows command prompt with output from **`ipconfig /all`**.

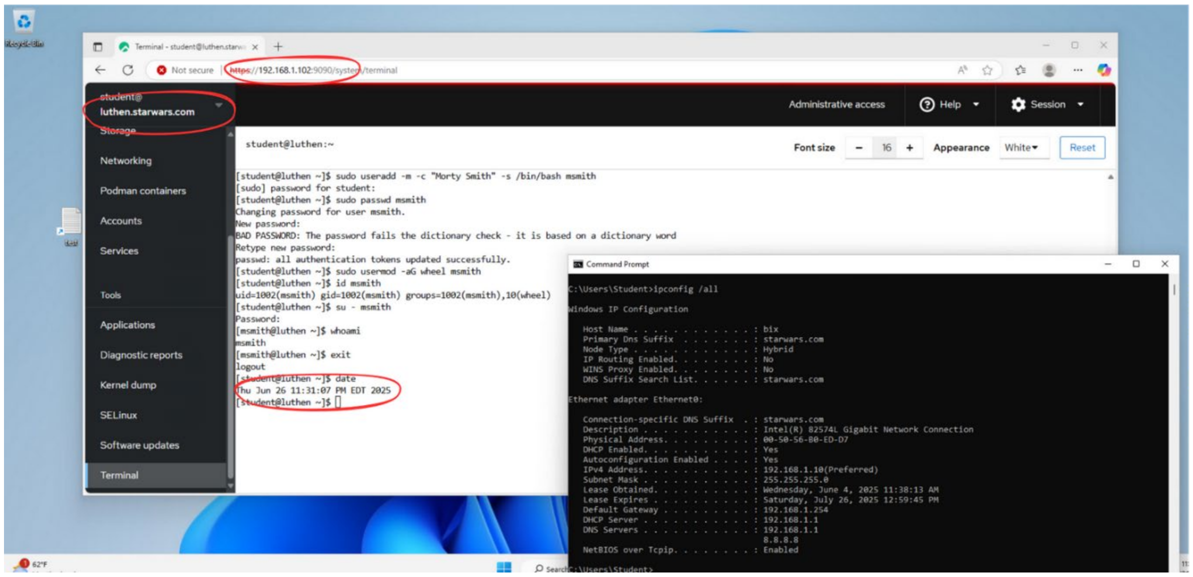
c) From the Cockpit terminal the commands used to add the user.

d) The server's full **Rocky Linux FQDN** in the upper-left corner of the Cockpit banner.

e) From the Cockpit terminal output from the **`date`** command.

(Figure 18 illustrates the expected output)

Figure 18 – Remote Account Creation Verification



```
[student@luthen ~]$ sudo useradd -m -c "Morty Smith" -s /bin/bash msmith
[sudo] password for student:
[student@luthen ~]$ sudo passwd msmith
Changing password for user msmith.
New password:
BAD PASSWORD: The password fails the dictionary check - it is based on a dictionary word
Retype new password:
passwd: all authentication tokens updated successfully.
[student@luthen ~]$ sudo usermod -aG wheel msmith
[student@luthen ~]$ id msmith
uid=1002(msmith) gid=1002(msmith) groups=1002(msmith),10(wheel)
[student@luthen ~]$ su - msmith
Password:
[msmith@luthen ~]$ whoami
msmith
[msmith@luthen ~]$ exit
logout
[student@luthen ~]$ date
Thu Jun 26 11:31:07 PM EDT 2025
[student@luthen ~]$
```

```
C:\Users\Student>ipconfig /all

Windows IP Configuration

   Host Name . . . . . . . . . . . . : bix
   Primary Dns Suffix  . . . . . . . : starwars.com
   Node Type . . . . . . . . . . . . : Hybrid
   IP Routing Enabled. . . . . . . . : No
   WINS Proxy Enabled. . . . . . . . : No
   DNS Suffix Search List. . . . . . : starwars.com

Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . : starwars.com
   Description . . . . . . . . . . . : Intel(R) 82574L Gigabit Network Connection
   Physical Address. . . . . . . . . : 00-50-56-B0-ED-D7
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   IPv4 Address. . . . . . . . . . . : 192.168.1.10(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Lease Obtained. . . . . . . . . . : Wednesday, June 4, 2025 11:38:13 AM
   Lease Expires . . . . . . . . . . : Saturday, July 26, 2025 12:59:45 PM
   Default Gateway . . . . . . . . . : 192.168.1.254
   DHCP Server . . . . . . . . . . . : 192.168.1.1
   DNS Servers . . . . . . . . . . . : 192.168.1.1
                                       8.8.8.8
   NetBIOS over Tcpip. . . . . . . . : Enabled

C:\Users\Student>
```

ACTIVITY 7: RECOVERING THE ROOT PASSWORD (SINGLE-USER MODE)

INTRODUCTION

Losing the *root* password effectively locks you out of every privileged operation on the system. In Rocky Linux the supported recovery method is to interrupt GRUB, boot into the **initramfs** rescue shell, and reset the password on the mounted *sysroot* filesystem.

In this activity you will:

1. Interrupt the normal boot process at the GRUB menu.
2. Append **rd.break enforcing=0** to the kernel line and enter the **initramfs** emergency shell.
3. Remount **/sysroot** read-write, chroot into it, and run passwd to set a new *root* password.
4. Trigger an SELinux relabel so the system boots cleanly with the new credentials.

These steps give you a repeatable, exam-ready procedure for regaining administrative control without reinstalling the OS.

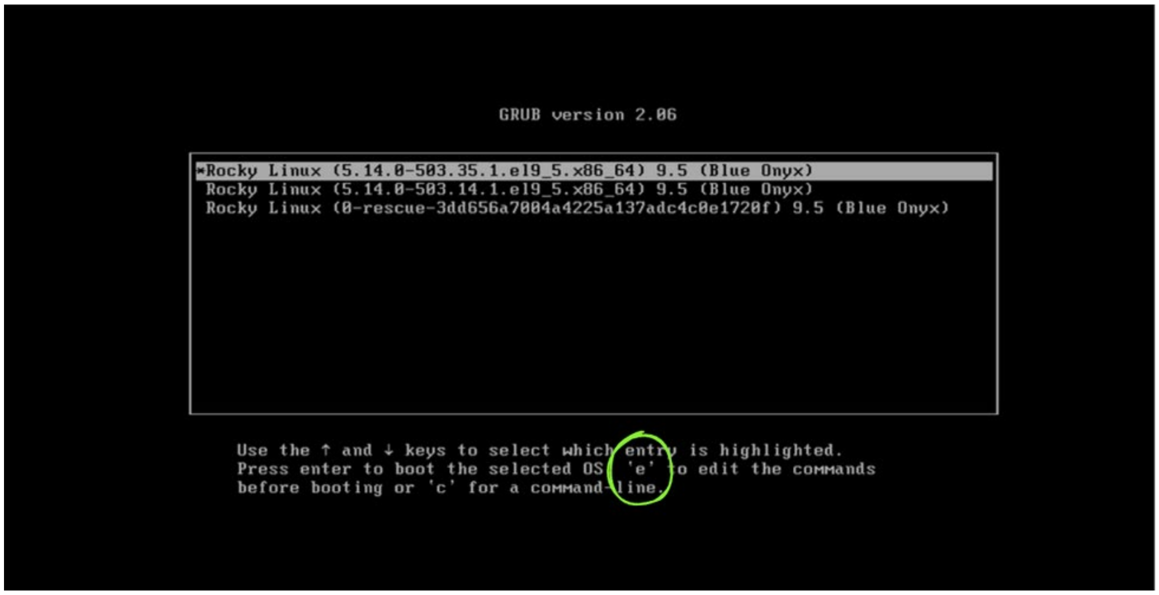
If pressing **<Esc>** or **e** does not interrupt the boot process, you may need to configure GRUB to display the menu. Edit /etc/default/grub and set GRUB_TIMEOUT=10. Then rebuild the configuration:

```
sudo grub2-mkconfig -o /boot/grub2/grub.cfg

sudo grub2-mkconfig -o /boot/efi/EFI/rocky/grub.cfg
```

PROCEDURE

1. Reboot the Rocky Linux system. At the GRUB menu, the first (default) kernel is already highlighted, so you can usually press **e** immediately (Figure 19). Tap an arrow key only if you need to stop the countdown or select a different kernel.

Figure 19 - GRUB Menu Highlighting the Default Kernel and the “e” Key for Editing Boot Parameters



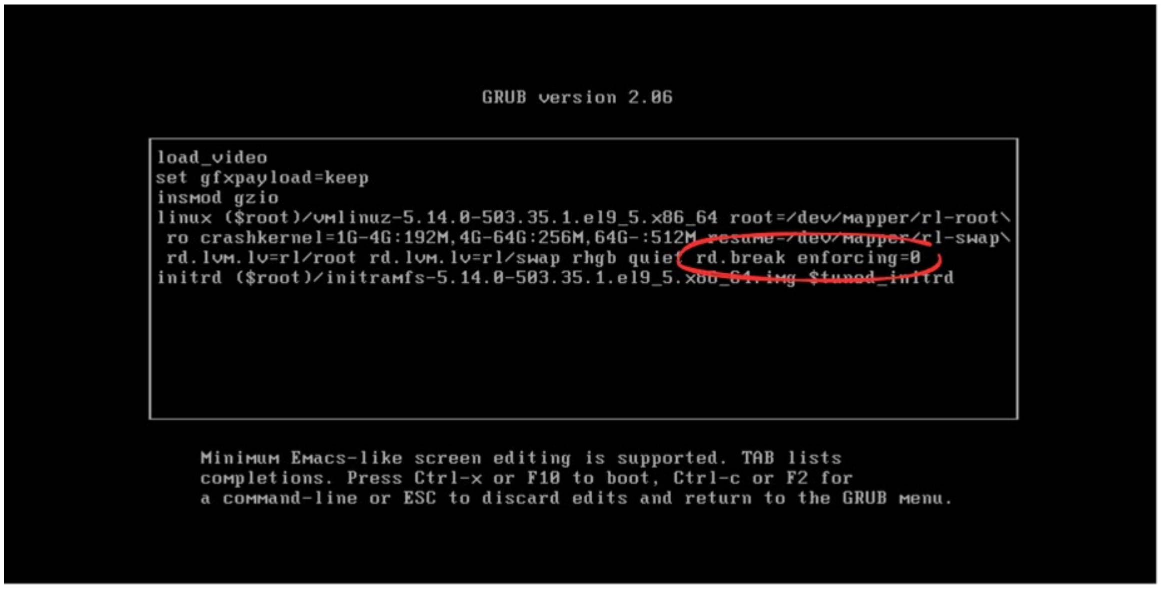
```
                    GRUB version 2.06

 *Rocky Linux (5.14.0-503.35.1.el9_5.x86_64) 9.5 (Blue Onyx)
  Rocky Linux (5.14.0-503.14.1.el9_5.x86_64) 9.5 (Blue Onyx)
  Rocky Linux (0-rescue-3dd656a7004a4225a137adc4c0e1720f) 9.5 (Blue Onyx)

     Use the ↑ and ↓ keys to select which entry is highlighted.
     Press enter to boot the selected OS, 'e' to edit the commands
     before booting or 'c' for a command-line.
```

2. Locate the long line that starts with **linux** (or linux16). Use the arrow keys to move to the **end** of that line (just after rhgb quiet). **Append** the following *single* fragment (Figure 20):

```
rd.break enforcing=0
```

This tells the kernel to break into the initramfs emergency shell and to disable SELinux enforcement temporarily. The **initramfs (initial RAM filesystem)** is a small, self-contained root filesystem that the bootloader loads into memory alongside the Linux kernel. When the kernel starts, it mounts this temporary filesystem and executes /init.

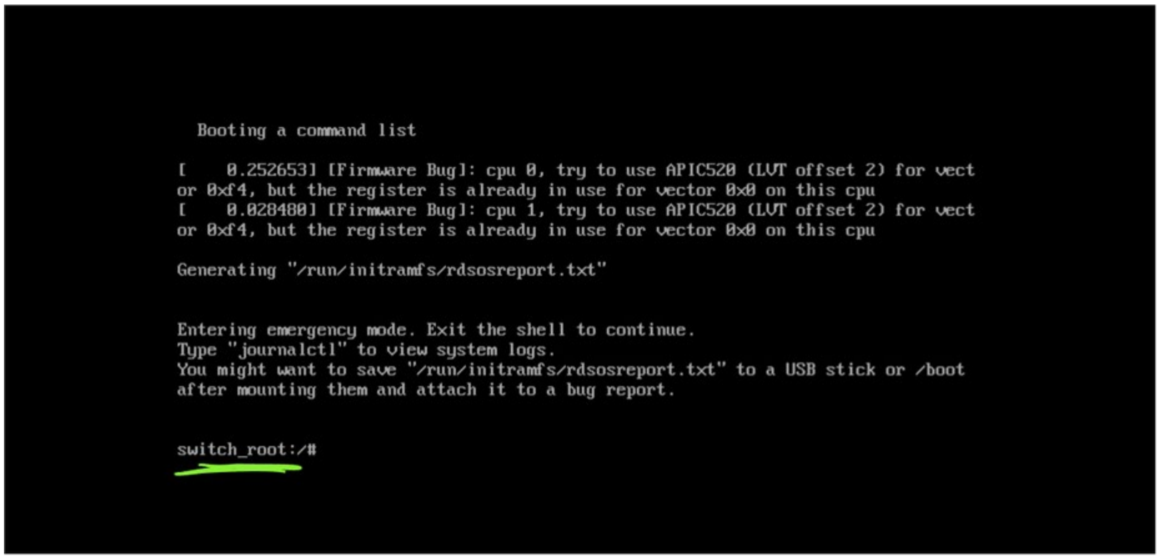
Figure 20 – GRUB Edit Screen with **`rd.break enforcing=0`** Appended to the Kernel Line



```
GRUB version 2.06

load_video
set gfxpayload=keep
insmod gzio
linux ($root)/vmlinuz-5.14.0-503.35.1.el9_5.x86_64 root=/dev/mapper/rl-root\
 ro crashkernel=1G-4G:192M,4G-64G:256M,64G-:512M resume=/dev/mapper/rl-swap\
 rd.lvm.lv=rl/root rd.lvm.lv=rl/swap rhgb quiet rd.break enforcing=0
initrd ($root)/initramfs-5.14.0-503.35.1.el9_5.x86_64.img $tuned_initrd

Minimum Emacs-like screen editing is supported. TAB lists
completions. Press Ctrl-x or F10 to boot, Ctrl-c or F2 for
a command-line or ESC to discard edits and return to the GRUB menu.
```

3. Press Ctrl + x (or F10) to boot with the modified options. This will drop you into a switch_root:/# prompt inside the initramfs (Figure 21).

Figure 21 – **`switch_root:/#`** Prompt in the initramfs Emergency Shell



```
Booting a command list

[    0.252653] [Firmware Bug]: cpu 0, try to use APIC520 (LVT offset 2) for vect
or 0xf4, but the register is already in use for vector 0x0 on this cpu
[    0.028480] [Firmware Bug]: cpu 1, try to use APIC520 (LVT offset 2) for vect
or 0xf4, but the register is already in use for vector 0x0 on this cpu

Generating "/run/initramfs/rdsosreport.txt"


Entering emergency mode. Exit the shell to continue.
Type "journalctl" to view system logs.
You might want to save "/run/initramfs/rdsosreport.txt" to a USB stick or /boot
after mounting them and attach it to a bug report.


switch_root:/#
```

4. **Remount /sysroot read-write** so you can edit files:

```
mount -o remount,rw /sysroot
```

5. **Chroot** into the real root file-system. The prompt will return the shell prompt (i.e., **`sh-5.1#`**).

```
chroot /sysroot
```

6. Set a new root password. Once the new password is entered successfully, you will see the message "*all authentication tokens updated successfully*."

```
passwd root
```

7. Force SELinux to relabel everything on the next boot (prevents login errors):

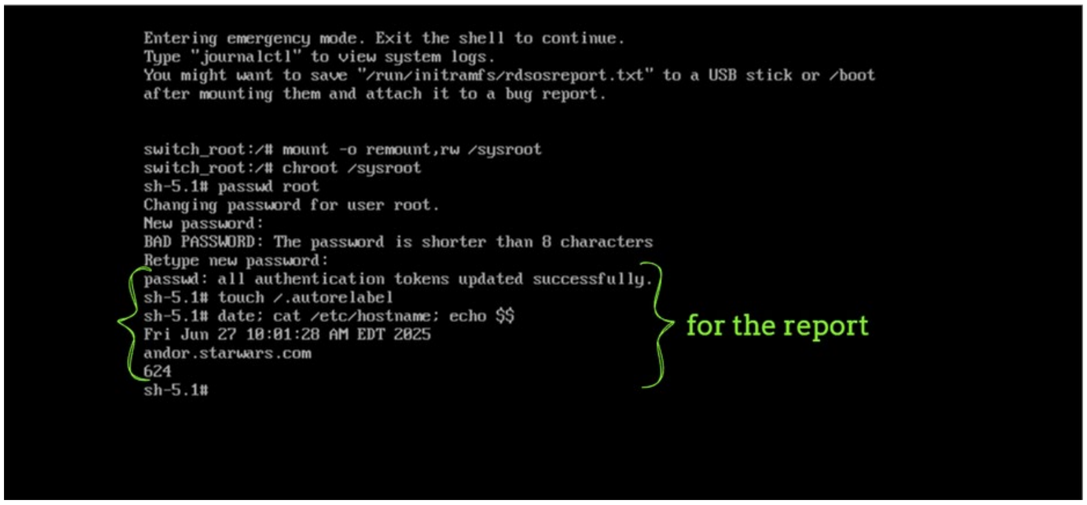
```
touch /.autorelabel
```

8. Before rebooting, obtain a screenshot that includes the "*all authentication tokens updated successfully*", and the output form the following commands:

```
date; cat /etc/hostname; echo $$
```

This is needed for the lab report (See Figure 22).

Figure 22 – Root Password Reset Verification



```
Entering emergency mode. Exit the shell to continue.
Type "journalctl" to view system logs.
You might want to save "/run/initramfs/rdsosreport.txt" to a USB stick or /boot
after mounting them and attach it to a bug report.


switch_root:/# mount -o remount,rw /sysroot
switch_root:/# chroot /sysroot
sh-5.1# passwd root
Changing password for user root.
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
passwd: all authentication tokens updated successfully.
sh-5.1# touch /.autorelabel
sh-5.1# date; cat /etc/hostname; echo $$
Fri Jun 27 10:01:28 AM EDT 2025
andor.starwars.com
624
sh-5.1#
```

9. Exit twice—first from the chroot, then from the initramfs—to continue the boot:

```
exit        # leave chroot
exit        # leave initramfs shell
```

10. The system reboots automatically, performs an SELinux relabel (this can take a minute), then shows the normal login screen. Log in as root with the new password to verify success.

CONCLUSION

You successfully:

- Entered the initramfs rescue environment via the rd.break option.
- Remounted **/sysroot** as read-write and changed the *root* password with passwd.
- Created the /.autorelabel flag to ensure SELinux contexts were fixed on reboot.
- Rebooted and verified *root* access with the new password.

The VM is now fully accessible again, and you have a proven recovery technique for any future password-loss scenario.

DELIVERABLES

Include in the report the screenshot from Step 8.

CONCLUSION

In this lab, you configured source-control tooling, practiced secure credential handling, and integrated GitHub workflows into your Rocky Linux environment. You also validated two essential administration techniques: graphical user management through Cockpit (local and remote) and emergency recovery of the *root* account. These combined tasks mirror real production scenarios and form the foundation for other systems administration tasks later in the course.

Successful completion—and submission of the required screenshots or technical write-ups—confirms that you can:

- Track and share scripts in a properly configured Git/GitHub pipeline.
- Manage local accounts interactively and through the command line.
- Regain privileged access to a locked-out system without data loss.

DELIVERABLES RUBRIC

Deliverable	**Evidence required for full credit**	**Points**
Activity 3	Screenshot of GitHub CLI Authentication from commands, **`hostname; date; gh auth status`**	15
Activity 4	Screenshot with the GitHub page showing *ping_test.py* in the repository including GitHub user name.	15
Activity 5	Screenshot from Cockpit Accounts showing the newly created user listed in the table. A visible **Last active** timestamp for that user. And the server's full **Rocky Linux FQDN** in the upper-left corner of the Cockpit interface.	20
Activity 6	Single composite screenshot that contains **all five items (five points each)**: • Browser bar https://<VM-IP>:9090 • Windows ipconfig /all • Cockpit Terminal commands (useradd, passwd, id) • Cockpit banner with full Rocky Linux FQDN • date output	25

Activity 7	Initramfs shell showing passwd root success message, **`cat /etc/hostname`**, **`date`**, and **`echo $$`**.	25
Screenshot quality	All images must be legible, unaltered, and include required elements; blurry or cropped images may receive **0 pts for that item**.	—
Written substitutions	If a screenshot is not possible ***for technical reasons***, a concise explanation of the problem and troubleshooting steps may earn up to **50 %** of that item's points. Generic excuses such as "ran out of time" or "I couldn't get it to work" receive **0 pts**.	—

Passing threshold: Each lab must score **≥ 70 pts**. Failing **three** labs (score < 70) results in automatic course failure.