termstyle backgroundcolor=TermBg, linecolor=TermBorder, linewidth=0.8pt, roundcorner=3pt, innertopmargin=10pt, innerbottommargin=10pt, innerleftmargin=10pt, innerrightmargin=10pt, frametitlebackgroundcolor=TermTitleBg, frametitlefont=

Linux System Administration & Security

Enterprise Laboratory Report

Lab A8 — Ansible Foundations

Secure Multi-Host Automation Setup

Name: Ayoub Zouargui
Degree: Master 1 — Networks and Embedded Systems
University: University of Algiers 1
Academic Year: 2025–2026
GitHub: https://github.com/AyoubSecurity

Date: February 12, 2026

1. Objective

In this lab, I established a professional Ansible foundation on RHEL to automate configuration across multiple systems. I configured a dedicated automation identity, deployed SSH key authentication, created a multi-host inventory, and enabled sudo privilege escalation to support non-interactive administration tasks.

The managed targets were:

localhost

rhel-server at **192.168.1.19**

2. Lab Environment

Control Node: RHEL system running Ansible under the `ansible` user.
Managed Node: RHEL server at `192.168.1.19` reachable via SSH and prepared for automation.

3. Work Performed

3.1 Installed Ansible on the Control Node

[style=termstyle,frametitle=Control Node — Install and Verify Ansible]

```
sudo dnf install -y ansible-core python3
ansible --version
```

3.2 Created the Automation Account

[style=termstyle,frametitle=Control Node / Managed Node — Create ansible User]

```
sudo useradd -m -s /bin/bash ansible
sudo passwd ansible
```

3.3 Enabled Passwordless sudo

[style=termstyle,frametitle=Control Node / Managed Node — Configure sudo Privileges]

```
echo 'ansible ALL=(ALL) NOPASSWD: ALL' | sudo tee /etc/sudoers.d/90-ansible
sudo chmod 440 /etc/sudoers.d/90-ansible
sudo visudo -cf /etc/sudoers.d/90-ansible
```

3.4 Enabled SSH on the Managed Server (192.168.1.19)

[style=termstyle,frametitle=Managed Node (192.168.1.19) — Enable and Check sshd]

```
sudo systemctl enable --now sshd
sudo systemctl status sshd --no-pager
```

3.5 Generated SSH Key and Distributed It to Targets

[style=termstyle,frametitle=Control Node — Switch to ansible and Generate SSH Key]

```
sudo -iu ansible
ssh-keygen -t ed25519
```

[style=termstyle,frametitle=Control Node — Install Public Key on Targets]

```
ssh-copy-id ansible@localhost
ssh-copy-id ansible@192.168.1.19
```

3.6 Created the Project Structure

[style=termstyle,frametitle=Control Node — Create Ansible Project Layout]

```
mkdir -p ~/ansible/{inventory,playbooks,roles,logs}
cd ~/ansible
```

3.7 Configured Inventory

[style=termstyle,frametitle=Control Node — Edit Inventory File]

```
nano inventory/hosts.ini
```

[style=termstyle,frametitle=inventory/hosts.ini — Final Content]

```
[servers]
localhost   ansible_host=127.0.0.1     ansible_user=ansible
rhel-server ansible_host=192.168.1.19  ansible_user=ansible
```

3.8 Configured ansible.cfg and Logging

[style=termstyle,frametitle=Control Node — Edit ansible.cfg]

```
nano ansible.cfg
```

[style=termstyle,frametitle=ansible.cfg — Final Content]

```
[defaults]
inventory = inventory/hosts.ini
host_key_checking = True
stdout_callback = yaml
log_path = logs/ansible.log

[privilege_escalation]
become = True
become_method = sudo
become_ask_pass = False
```

[style=termstyle,frametitle=Control Node — Secure Logs Directory]

```
chmod 700 logs
```

4. Validation Results

[style=termstyle,frametitle=Validation — Connectivity Test]

```
ansible servers -m ping
```

[style=termstyle,frametitle=Validation — Expected Output]

```
localhost   | SUCCESS => ping: pong
rhel-server | SUCCESS => ping: pong
```

5. Conclusion

I completed a secure Ansible multi-host foundation by establishing an automation identity, enabling SSH key authentication, defining a structured inventory, and configuring sudo-based privilege escalation. This environment is ready for subsequent automation labs involving system configuration, security hardening, and compliance enforcement.