**Day 35**

**Exploitation Analyst**

**User Management and PAM:**

**Sudo Access:**

**What is sudo?**

sudo (short for “superuser do”) is a Linux command that lets a permitted user run programs with the security privileges of another user, typically the root user. It allows users to perform administrative tasks without needing to log in as root.

**Why is it required?**

* **Security**: Instead of sharing the root password, users get limited administrative access.
* **Accountability**: Commands run via sudo are logged, helping track who did what.
* **Convenience**: Users can run specific commands with elevated privileges without switching users.

**Disadvantages**

* **Misconfiguration risks**: Incorrect sudoers file settings can give excessive privileges.
* **Potential abuse**: If a sudo user’s account is compromised, attacker gets root-level access.
* **Complexity**: Managing fine-grained sudo permissions requires care and knowledge.

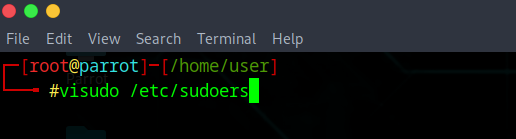
**Which file allows sudo users?**

* The **/etc/sudoers** file controls who can use sudo and what commands they can run.
* It should always be edited with visudo to prevent syntax errors.

**Setting Sudo Access rules:**

Steps:

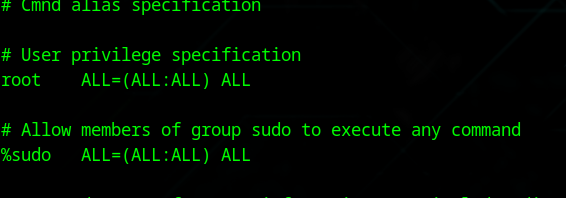
Open the /etc/sudoers with the visudo editor:



Following screen will appear:



Scroll down and reach this section:



This line in /etc/sudoers:

*root ALL=(ALL:ALL) ALL*

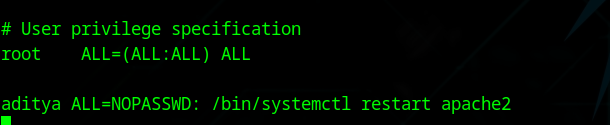
means that the **root user** has full administrative privileges on the system. Breaking it down:

* root — the username this rule applies to.
* ALL (first) — root can run commands from any host (useful in networked setups).
* (ALL:ALL) — root can run commands as any user and any group.
* ALL (last) — root can run any command.

What can we do here?

To give user aditya permission to restart the apache2 service using sudo without giving full root access, you can add a rule in the sudoers file like this:

*aditya ALL=NOPASSWD: /bin/systemctl restart apache2*



Save the file and exit.