**Day 36**

**Exploitation Analyst**

**User Management and PAM:**

**Disable root Login:**

**Why Root login is so crucial?**

Root login in Linux is important because it grants unrestricted access to the entire system, including all files, configurations, and administrative commands. It allows full control for system management, software installation, user management, and troubleshooting. However, misuse can cause security risks, system instability, or data loss.

**Which file contains the data for defining the root users?**

In Linux, the file `/etc/sudoers` controls who can execute commands as the root user. It defines permissions for specific users or groups to run administrative commands with `sudo`, effectively allowing them root privileges without direct root login.

**Which file contains the data to disable any root login?**

The file /etc/ssh/sshd\_config contains the setting PermitRootLogin that can be used to disable root login over SSH. Setting PermitRootLogin no in this file prevents remote root logins. root login can also be disabled by editing /etc/passwd. If you change the root user’s shell from something like /bin/bash to /sbin/nologin or /bin/false, it prevents interactive logins for root, both locally and remotely, while keeping the account itself present for system processes.

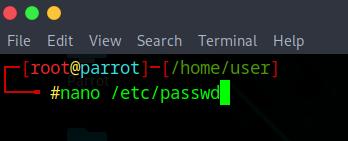
**Which one is better?**

Disabling root login via /etc/ssh/sshd\_config is generally better because it only blocks remote SSH access for root while keeping the account functional for local administrative use and automated processes.  
Editing /etc/passwd is riskier, as it blocks all interactive logins for root and could disrupt recovery or maintenance tasks that require direct root access.

**Disabling root Login:**

Steps: (by /etc/passwd)

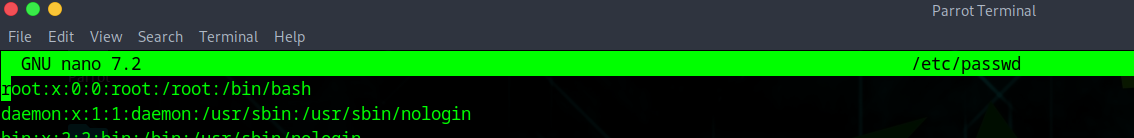
Open the file /etc/passwd using the nano command:

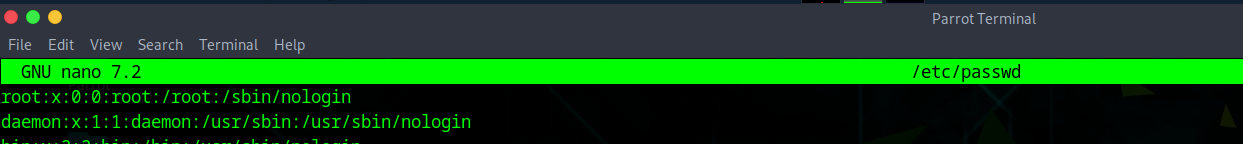


Following screen will appear:



Focus on the part mentioning ‘root’:



Edit that line from /bin/bash to /sbin/nologin:  


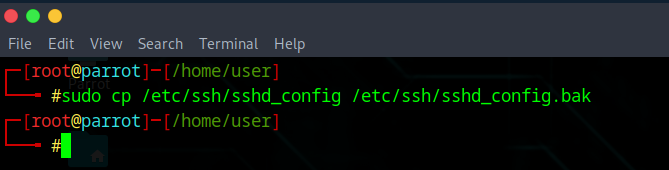
Save the file and exit.

**Disabling root Login:**

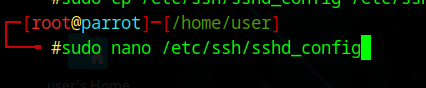
Steps: (by /etc/ssh/sshd\_config)

Ensure you have a sudo-enabled user so you don’t lock yourself out.

Backup the SSH config file: sudo cp /etc/ssh/sshd\_config /etc/ssh/sshd\_config.bak



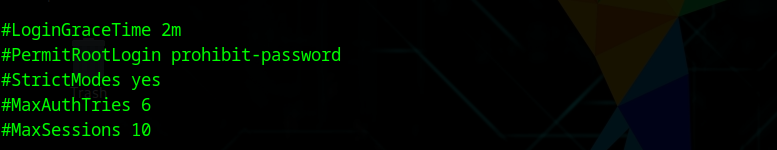
Open the file for editing: sudo nano /etc/ssh/sshd\_config



Following screen will appear:



Search for this section: PermitRootLogin



Change that section to: PermitRootLogin no



Save it and exit. Reload SSH service: sudo systemctl reload sshd

Test SSH as root to confirm it’s blocked.

**How does editing these two files differs?**

**1. Editing /etc/ssh/sshd\_config (PermitRootLogin)**

* This only affects SSH remote logins.
* When you set PermitRootLogin no and reload sshd, the SSH daemon refuses any connection attempts using the root username.
* The root account still exists and can log in locally (e.g., via terminal, recovery mode, or su).
* Safer because you can still use root if you have local machine access.

**2. Editing /etc/passwd (changing root’s shell)**

* /etc/passwd contains account information, including the shell to start after login.
* If you change root’s shell to /sbin/nologin or /bin/false, it prevents all interactive logins for root, both locally and remotely.
* The root account still exists for running system processes, but you can’t open a root shell directly.
* Riskier because if you lose your sudo access or other admin user, you could lock yourself out completely.

**What we can do make sure that these settings works fine?**

1. **Principle of Least Privilege** – Never allow direct root login over SSH unless absolutely necessary. Use a normal user with sudo for admin tasks.
2. **Fail-Safe Access** – Always have at least one other sudo-enabled account before disabling root login, so you don’t lock yourself out.
3. **Layered Security** – Disabling root SSH login is one layer; combine it with strong passwords, SSH key authentication, and firewall rules.
4. **Method Selection** – Prefer /etc/ssh/sshd\_config over /etc/passwd for root login restriction because it’s less disruptive and easier to reverse remotely.
5. **Testing** – Always test changes in a separate SSH session before closing your current one, so you can fix issues if something goes wrong.
6. **Logging** – Monitor /var/log/auth.log (Debian/Ubuntu) or /var/log/secure (RHEL/CentOS) for any failed root login attempts after changes.
7. **Recovery Knowledge** – Know how to re-enable root login via local console or recovery mode in case of emergencies.

Note:

The biggest difference is scope:

* /etc/ssh/sshd\_config → blocks only remote SSH root login.
* /etc/passwd → blocks all interactive root logins, both local and remote.

--The End--