🧠 DAY 9 — Privilege Escalation Awareness (DEFENSIVE):

CISO focus: “How does privilege escalation happen, and how do we see it before damage?”

🧱 LAB SAFETY RULES (READ ONCE)

- **Do NOT install exploit frameworks**
- **Do NOT modify system binaries**
- **Do NOT run random sudo commands blindly**
- **Everything is observe, test safely, and document**

🔹 PHASE 1 — Identity & Privilege Baseline (Foundation)

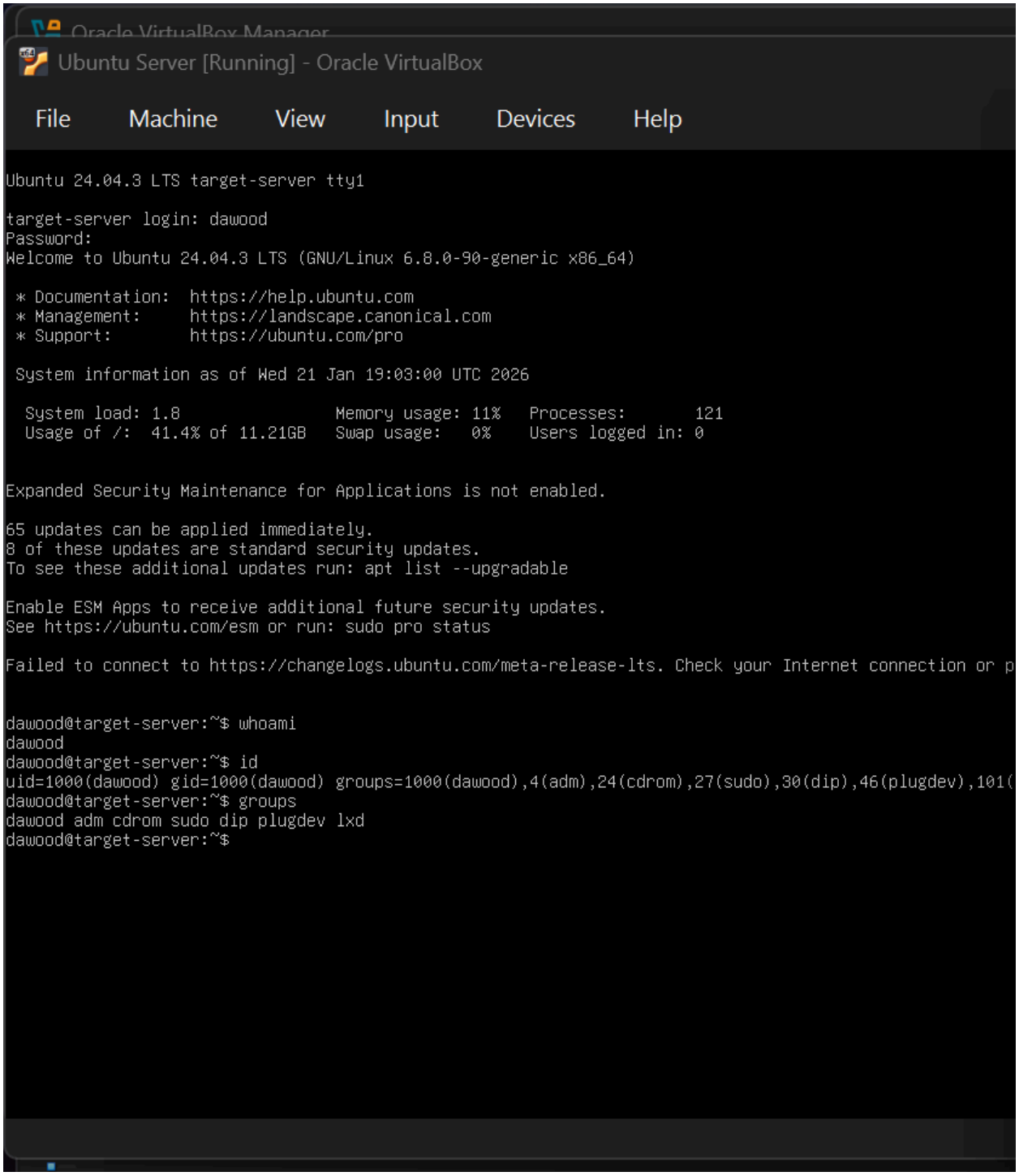
In this first phase we have to see where we are standing .

For foundation first we have check the username ,by typing this (whoami)command we will check the user name ,E.g dawood

After that we will type (id)command which tell us uid(user id) and gid(group id) And the last step of phase one we will type (groups)command ,it will tell us which clubs the user belongs to .

(Why we have to do this)
To know exactly what permissions a user has before an incident occurs.

Proof



🔹 PHASE 2 — Sudo Configuration Awareness

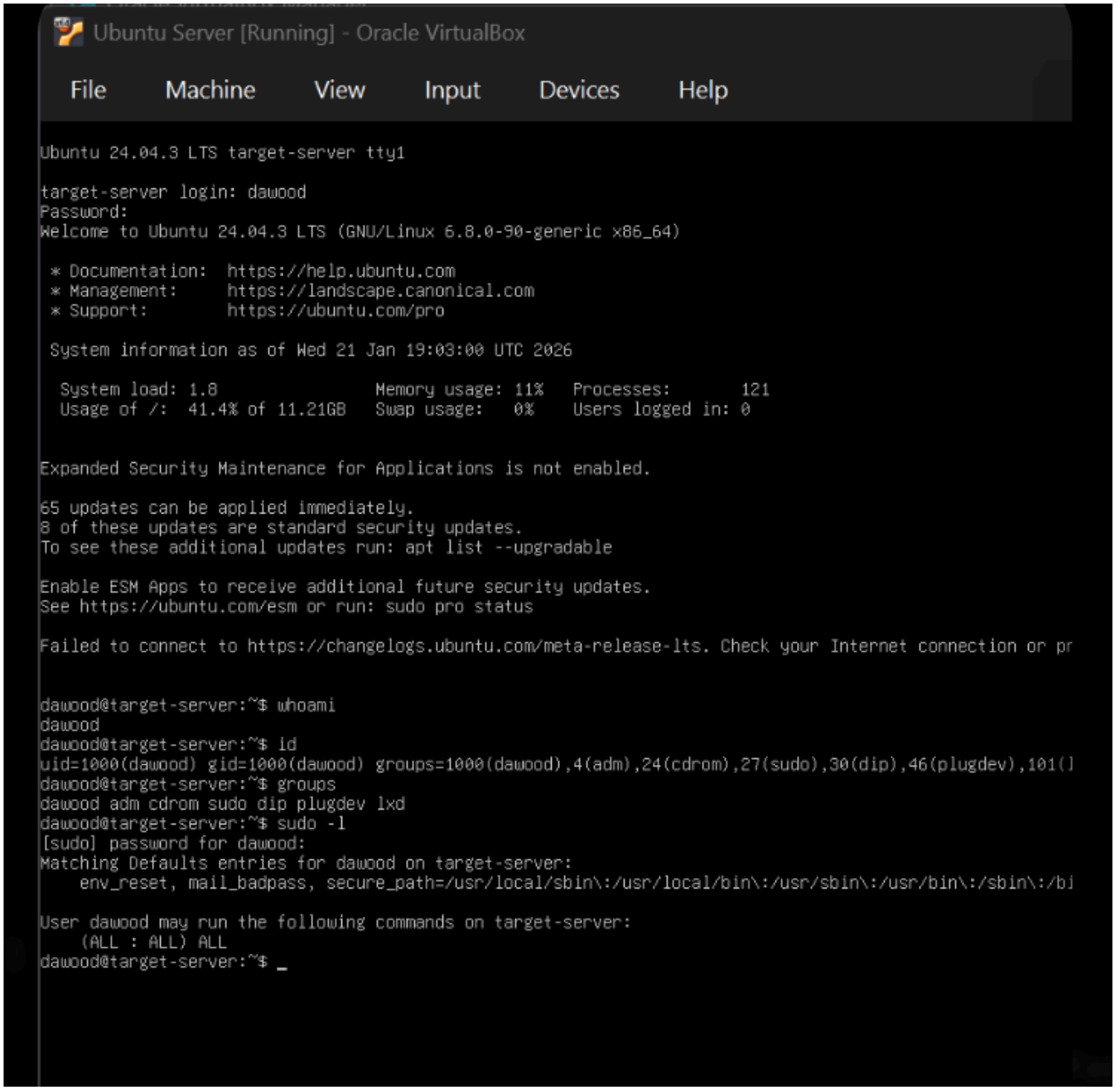
In this step we will check if our internal doors are locked or not .

In this step this(sudo -l) command will tell us about actual scenario

🛡️ Why do we do this?

Attackers love **"Misconfigurations."** If a lazy Administrator sets a user to `NOPASSWD` so they don't have to type their password every time, they just give a free "Master Key" to any hacker who slips into that account.

Proof



Step 2: View sudo configuration (Read-only)
In this step we will only inspect the logs ,table and master files
The command we will use in this is (sudo cat /etc/sudoers | less)

`sudo`: Required because this file is restricted to admins only.

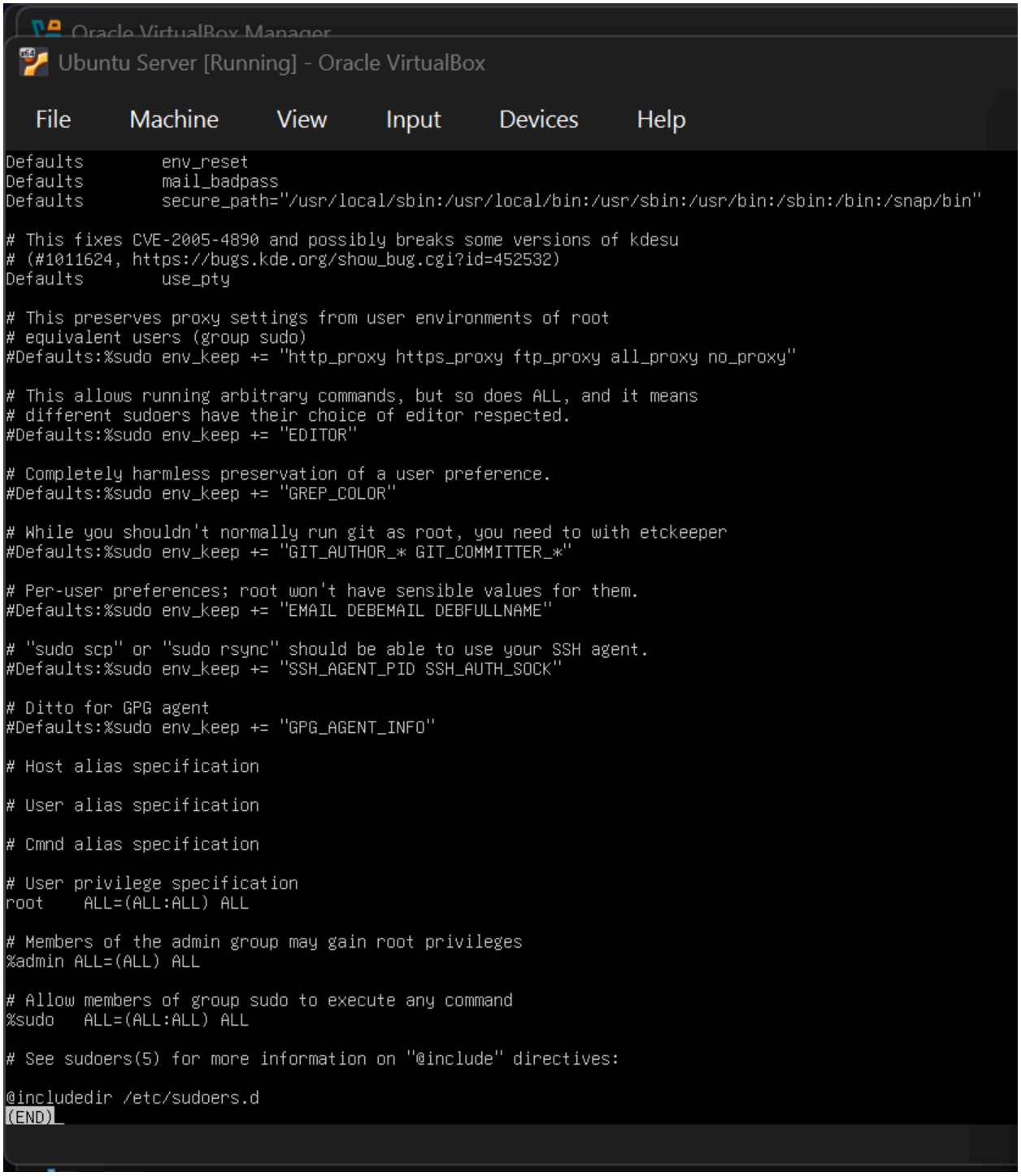
`cat`: Reads the file.

`|`: (The pipe you just fixed!) Send the text to the next tool.

`less`: Opens the file in a "viewer" mode so you don't accidentally change anything.

Why:
In a professional **SOC**, we use a "Look, Don't Touch" policy for audits. If you used a text editor like `nano` or `vi`, a single accidental keystroke could lock **everyone** (including root) out of the `sudo` command. Using `cat | less` is the safe, professional way to observe authority.

Proof



🔹 PHASE 3 — SUID Awareness (Critical Concept)

Step 3: Find SUID binaries

In step we will visualize everything
In this step the command we are using is (find / -perm -4000 -type f

2>/dev/null)
`find /`: Look everywhere on the entire server.

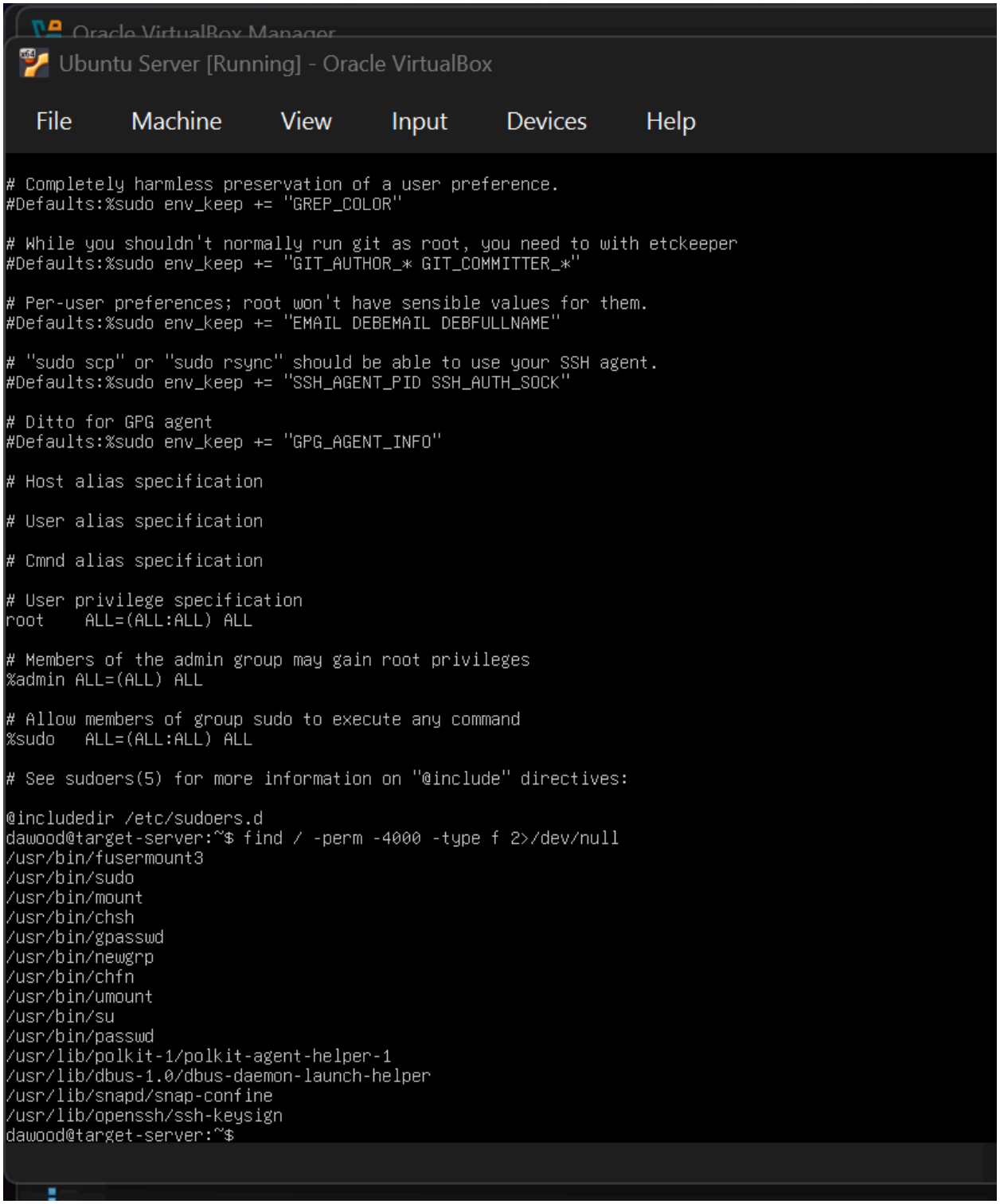
`-perm -4000`: This is the technical code for **SUID**. It tells Linux "show me programs that run with the owner's power."

`-type f`: Only look for files (not folders).

`2>/dev/null`: This hides "Permission Denied" errors so your screen stays clean.

Why:
By running this command, you are performing a **"Vulnerability Assessment."** You are making sure that the only programs with extra power are the ones that actually *need* it to keep the system running.

Proof


Oracle VirtualBox Manager
Ubuntu Server [Running] - Oracle VirtualBox
File
Machine
View
Input
Devices
Help
Completely harmless preservation of a user preference.
#Defaults:%sudo env_keep += "GREP_COLOR"
While you shouldn't normally run git as root, you need to with etckeeper
#Defaults:%sudo env_keep += "GIT_AUTHOR_* GIT_COMMITTER_*"
Per-user preferences; root won't have sensible values for them.
#Defaults:%sudo env_keep += "EMAIL DEBEMAIL DEBFULLNAME"
"sudo scp" or "sudo rsync" should be able to use your SSH agent.
#Defaults:%sudo env_keep += "SSH_AGENT_PID SSH_AUTH_SOCK"
Ditto for GPG agent
#Defaults:%sudo env_keep += "GPG_AGENT_INFO"
Host alias specification
User alias specification
Cmnd alias specification
User privilege specification
root ALL=(ALL:ALL) ALL
Members of the admin group may gain root privileges
%admin ALL=(ALL) ALL
Allow members of group sudo to execute any command
%sudo ALL=(ALL:ALL) ALL
See sudoers(5) for more information on "@include" directives:
@includedir /etc/sudoers.d
dawood@target-server:~$ find / -perm -4000 -type f 2>/dev/null
/usr/bin/fusermount3
/usr/bin/sudo
/usr/bin/mount
/usr/bin/chsh
/usr/bin/gpasswd
/usr/bin/newgrp
/usr/bin/chfn
/usr/bin/umount
/usr/bin/su
/usr/bin/passwd
/usr/lib/polkit-1/polkit-agent-helper-1
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/snapd/snap-confine
/usr/lib/openssh/ssh-keysign
dawood@target-server:~$

PHASE 4 — Log-Based Detection (MOST IMPORTANT)

Step 4: Monitor sudo usage

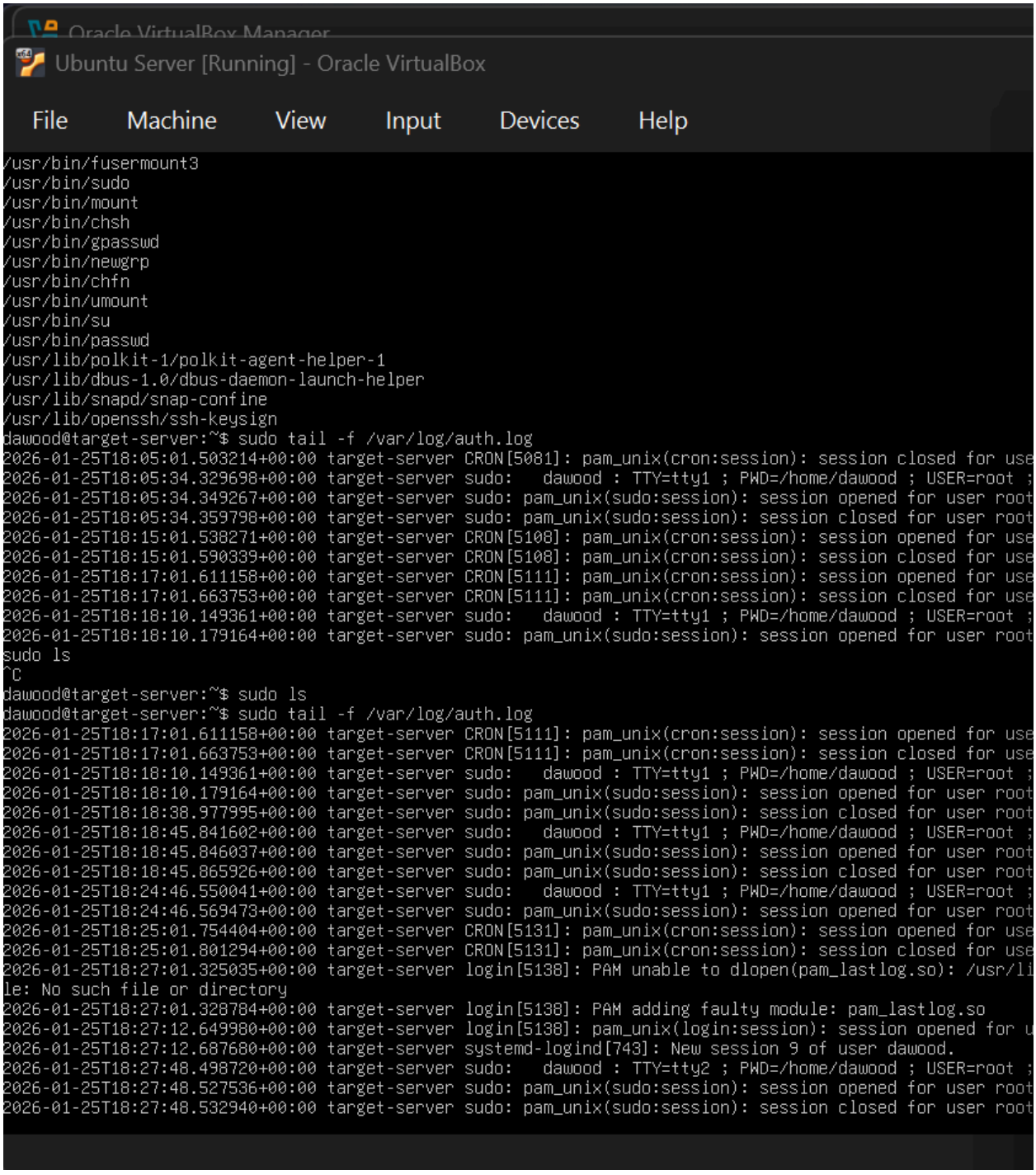
In this step we will move deeper in sudo with the help of auth.log we will monitor live logs
In this step the command we are using is (sudo tail -f /var/log/auth.log).

In the next step we have to open tty2 (second terminal)for observing live logs
We will type (alt+F2) for tty2 then enter (sudo ls).

Why:
If an attacker steals your password and runs `sudo cat /etc/shadow` to steal everyone's passwords, the **only** way the company will know is by looking at this log.

By watching `auth.log`, you are turning the server's memory into a weapon against the attacker.

Proof



🔹 PHASE 5 — Attack Simulation (Safe & Minimal)

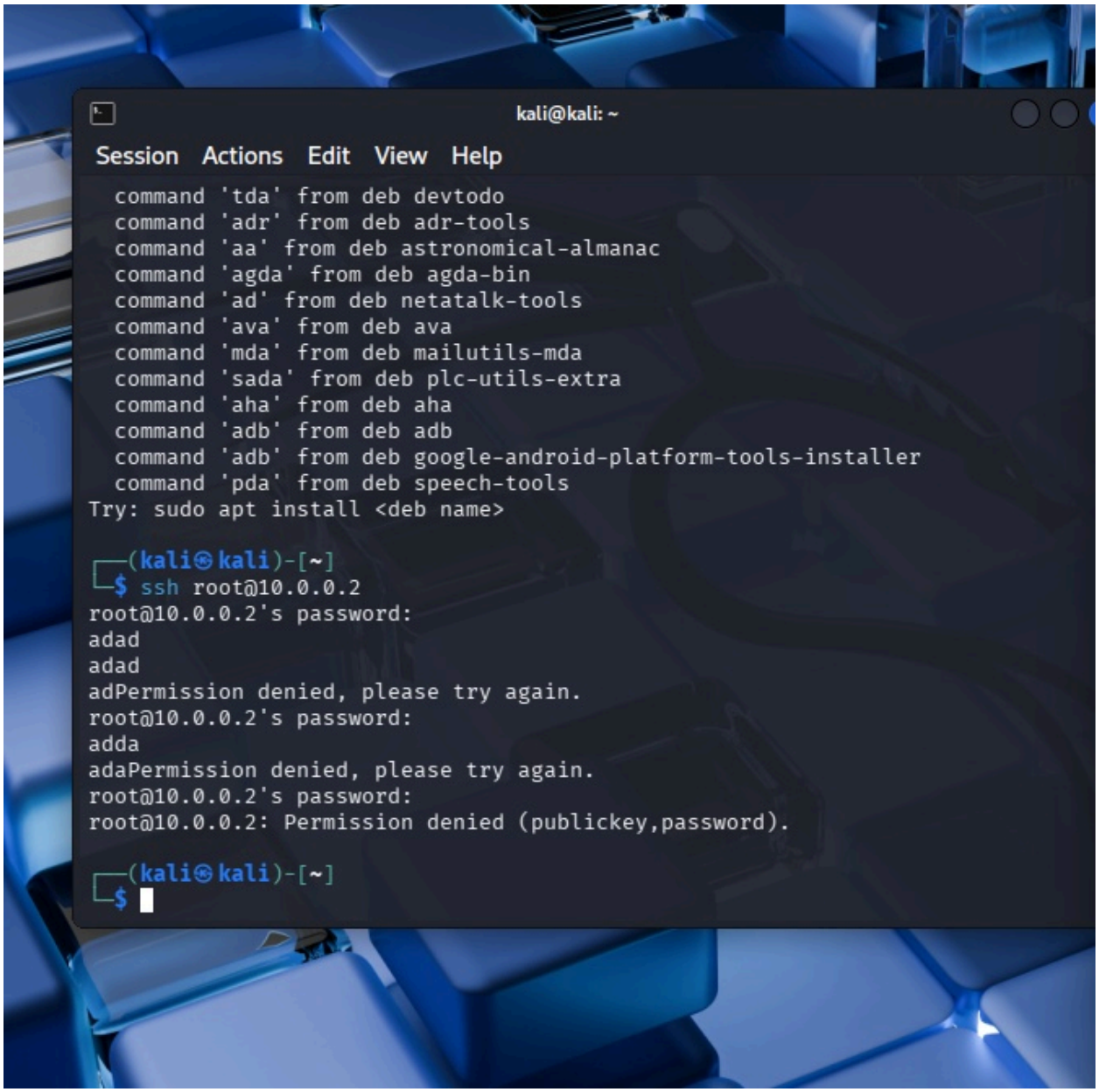
In this step we will run(ssh root@ubuntu_ip) this command ,and it is expected to be failure.
When Kali asks for a password, type `password123` (or any wrong password).

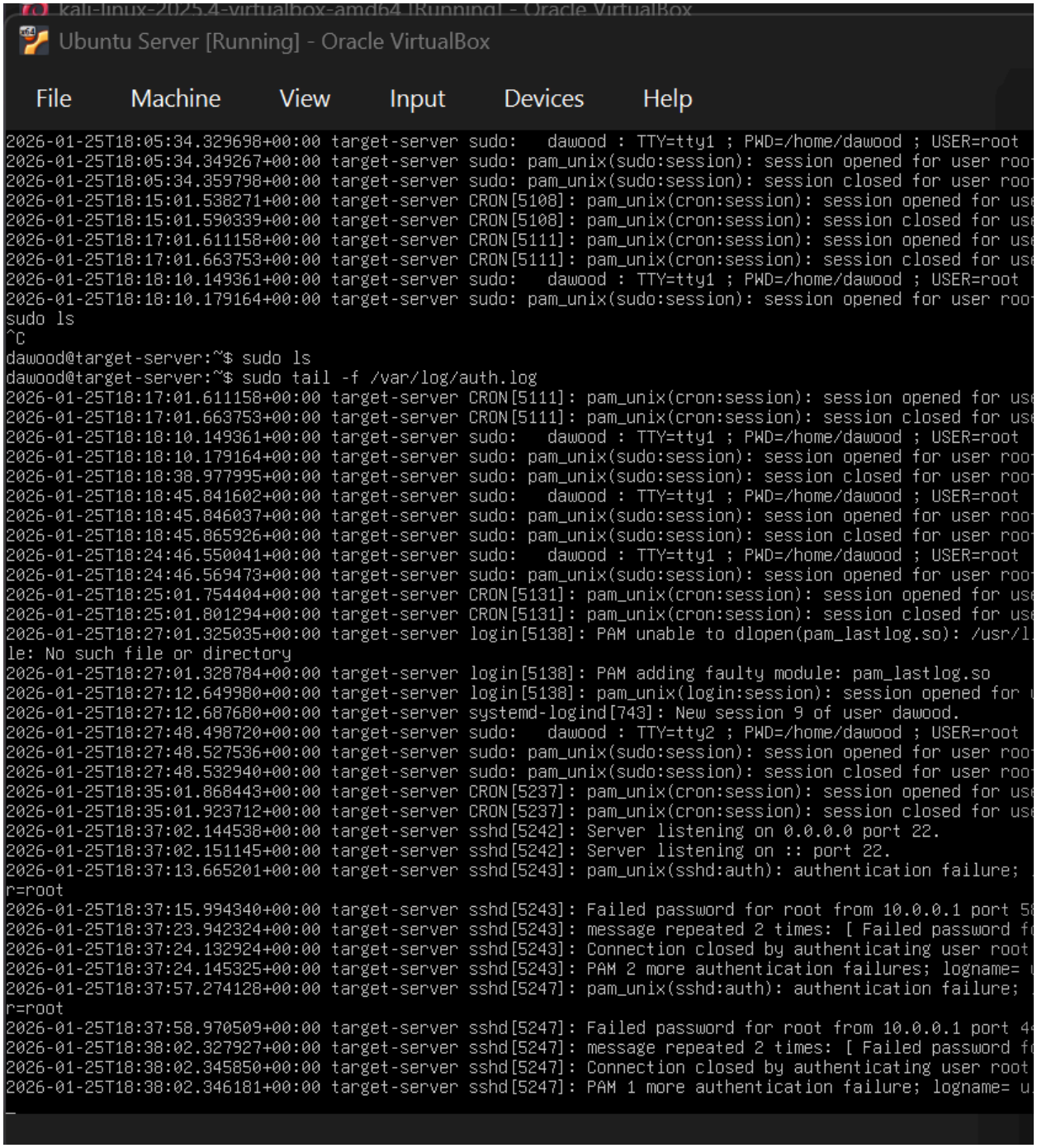
Do this **3 times**. SSH will eventually kick you out.

Why:

By seeing these logs, you have proven that:

> **Visibility**: You have a "camera" on your most sensitive account (root).
>
> **Attribution**: You know the attacker's IP is `10.0.0.1`.
>
> **Prevention**: Because you saw these failures, you could now block that IP before they ever get a chance to try a privilege escalation trick like `sudo` or exploiting a **SUID** binary.
> To prove the **SOC (Security Operations Center)** can detect Brute Force attacks before an attacker gets in.
>
> Proof


kali@kali: ~
Session Actions Edit View Help
command 'tda' from deb devtodo
command 'adr' from deb adr-tools
command 'aa' from deb astronomical-almanac
command 'agda' from deb agda-bin
command 'ad' from deb netatalk-tools
command 'ava' from deb ava
command 'mda' from deb mailutils-mda
command 'sada' from deb plc-utils-extra
command 'aha' from deb aha
command 'adb' from deb adb
command 'adb' from deb google-android-platform-tools-installer
command 'pda' from deb speech-tools
Try: sudo apt install <deb name>
(kali㉿kali)-[~]
$ ssh root@10.0.0.2
root@10.0.0.2's password:
adad
adad
adPermission denied, please try again.
root@10.0.0.2's password:
adda
adaPermission denied, please try again.
root@10.0.0.2's password:
root@10.0.0.2: Permission denied (publickey,password).
(kali㉿kali)-[~]
$



🔹 *PHASE 6 — Defensive Thinking (WRITE THIS)*

1)What misconfigurations enable privilege escalation?

ans)Giving users admin power without needing a password (NOPASSWD) or leaving "Master Keys" (SUID) on files that don't need them.

2)What log entries indicate escalation attempts?

ans) Repeated "Failed password" messages followed by someone successfully becoming "root" in the logs.

3)How would Fail2Ban or SIEM detect this?

ans) They act like a "Digital Security Guard" that counts failures and locks the door (blocks the IP) after too many bad guesses.

4)How does this map to cloud IAM misuse?

ans)It is also the same thing that we are doing here ,just the difference is it will be on cloud .

conclusion:
when a "bad guy" (Kali) tried to guess a password, your "Security Camera" (Ubuntu) caught them immediately.

Day 9: Command Table

Command	What it does (Easy Words)	Why we use it (Recruiter)

id	Shows your "Identity" number.	To see if you are a normal user or Root.
sudo -l	Lists your "Admin Powers."	To find security gaps in the rulebook.
sudo cat /etc/sudoers	Reads the "Master Law File."	To check if the system is locked properly.
find / -perm -4000	Searches for "Borrowed Keys."	To find hidden backdoors in programs.
tail -f /var/log/auth.log	Watches "Security Logs" live.	To catch attackers while they are moving.

Day 9: Full Form Keys

Short Form	**Full Form**	**Easy Meaning**

UID	**U**ser **ID**entifier	Your unique digital "ID Card" number.
GID	**G**roup **ID**entifier	The ID for your "Team" (like the Sudo team).
SU DO	**S**uper**U**ser **DO**	Acting like the "Big Boss" (Root) for one task.
SUI D	**S**et **U**ser **ID**	A file that "borrows" Root's power to run.
IAM	**I**dentity & **A**ccess **M**anagement	The rulebook for who gets to touch what.
SO C	**S**ecurity **O**perations **C**enter	The "Security Room" where we watch the logs.