

CIS 221 Group Project

Security Hardening of a Server

Prepared by: Group #3

Armaan Chima # 300211286

Derek Everard # 300126798

Zaid Al-Obaidi # 300225793

Date: July 30, 2025

CIS 221 Group Project Proposal & Report

Title: Security Hardening of a Server

1. Project Overview

Objective:

Deploy a vulnerable Ubuntu server and systematically secure it using industry best practices. This includes firewall configuration, intrusion detection, user and SSH hardening, regular patching, and continuous monitoring.

Significance:

Misconfigured or unpatched servers are common attack targets. This project demonstrates practical methods to secure a server against threats, aligning with CIS Benchmarks and standard security frameworks.

2. Requirements

Functional Requirements:

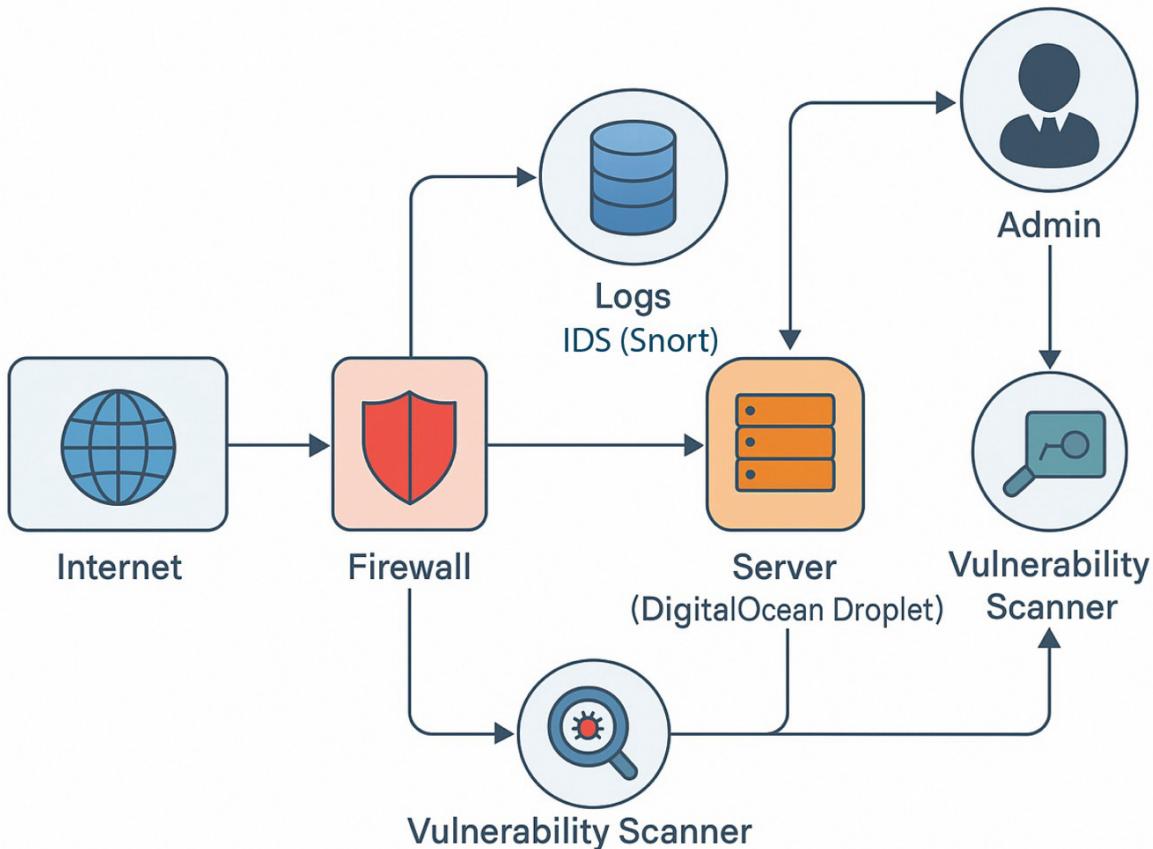
- Deploy an Ubuntu server with default configurations.
- Conduct baseline vulnerability scans (Nmap, Nessus/OpenVAS).
- Configure a firewall (UFW).
- Install & configure an IDS (Snort).
- Disable unnecessary services and restrict user access.
- Set up automatic security updates.
- Document every step and configuration.

Non-Functional Requirements:

- **Security:** Compliance with CIS Benchmarks.
 - **Performance:** Maintain normal server performance.
 - **Usability:** Configurations are manageable by system admins.
 - **Scalability:** Steps are repeatable for other servers.
-

3. Design

Data Flow Diagram (DFD):



Components of DFD:

- **Internet:** Source of all incoming and outgoing traffic.
- **Firewall (UFW):** Controls and filters traffic.
- **Server (Ubuntu on DigitalOcean Droplet):** Runs services.
- **Intrusion Detection System (Snort):** Monitors network traffic.
- **Logs:** Stores IDS and firewall logs.
- **Admin:** Reviews logs, configures security, and manages patches.
- **Vulnerability Scanner:** Tools: **Nmap** (port scanning) and **Snort** (IDS with custom scanning rules).
- Installed UFW: `sudo apt install ufw`

4. Implementation

- **Set Up:**
 - Deployed Ubuntu Server on DigitalOcean Droplet.
 - Updated OS: `sudo apt update && sudo apt upgrade`
- **Initial Scan:**
 - Installed **Nmap** on a different machine: `sudo apt install nmap`
 - Scanned server: `sudo nmap -sS 192.168.10.10 (Placeholder IP)`
 - Port scan before firewall configuration:

```
zaid@zubuntu:~$ sudo nmap -sS [REDACTED]
Starting Nmap 7.80 ( https://nmap.org ) at 2025-07-30 08:20 UTC
Nmap scan report for [REDACTED]
Host is up (0.074s latency).
Not shown: 997 filtered ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
8080/tcp  closed http-proxy

Nmap done: 1 IP address (1 host up) scanned in 6.61 seconds
```

- Ports detected: (22/tcp, 80/tcp, 8080/tcp)
- **Firewall Configuration:**
 - Installed **UFW**: `sudo apt install ufw`
 - Set defaults: `sudo ufw default deny incoming` and `sudo ufw default allow outgoing`
 - Allowed SSH: `sudo ufw allow ssh`
 - Enabled UFW: `sudo ufw enable`

```
root@ZaidUbuntu:~# sudo ufw status
Status: inactive
root@ZaidUbuntu:~# sudo ufw default deny incoming
Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)
root@ZaidUbuntu:~# sudo ufw default allow outgoing
Default outgoing policy changed to 'allow'
(be sure to update your rules accordingly)
root@ZaidUbuntu:~# sudo ufw allow 22/tcp
Rules updated
Rules updated (v6)
root@ZaidUbuntu:~# sudo ufw enable
[Command may disrupt existing ssh connections. Proceed with operation (y|n)? y
Firewall is active and enabled on system startup
```

- Block all unnecessary ports detected in the Baseline Vulnerability Scan. Keep only the SSH port

```
[root@ZaidUbuntu:~# ufw deny 80/tcp
Rule updated
Rule updated (v6)
[root@ZaidUbuntu:~# ufw deny 8080/tcp
Rule updated
Rule updated (v6)
root@ZaidUbuntu:~# ]
```

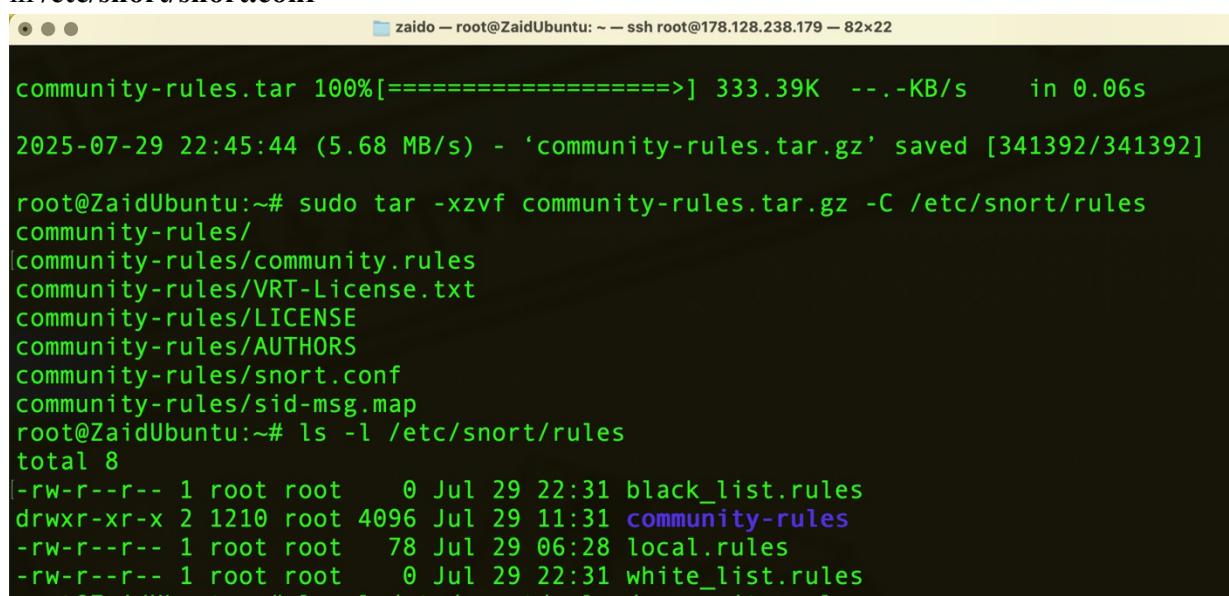
- **IDS (Snort) Installation & Setup:**

- Installed Snort: `sudo apt install snort`

```
root@ZaidUbuntu:~# snort -V

      _--> Snort! <--_
  o" )~ Version 2.9.20 GRE (Build 82)
    '--- By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
        Copyright (C) 2014-2022 Cisco and/or its affiliates. All rights reserved.
        Copyright (C) 1998-2013 Sourcefire, Inc., et al.
        Using libpcap version 1.10.5 (with TPACKET_V3)
        Using PCRE version: 8.39 2016-06-14
        Using ZLIB version: 1.3.1

root@ZaidUbuntu:~# ]
```
- Downloaded Snort community rules, configured Snort paths and rules in `/etc/snort/snort.conf`



The screenshot shows a terminal window titled "zaido — root@ZaidUbuntu: ~ — ssh root@178.128.238.179 — 82x22". The terminal output is as follows:

```
community-rules.tar 100%[=====] 333.39K ---KB/s in 0.06s
2025-07-29 22:45:44 (5.68 MB/s) - 'community-rules.tar.gz' saved [341392/341392]

root@ZaidUbuntu:~# sudo tar -xzvf community-rules.tar.gz -C /etc/snort/rules
community-rules/
|community-rules/community.rules
|community-rules/VRT-License.txt
|community-rules/LICENSE
|community-rules/AUTHORS
|community-rules/snort.conf
|community-rules/sid-msg.map
root@ZaidUbuntu:~# ls -l /etc/snort/rules
total 8
-rw-r--r-- 1 root root 0 Jul 29 22:31 black_list.rules
drwxr-xr-x 2 1210 root 4096 Jul 29 11:31 community-rules
-rw-r--r-- 1 root root 78 Jul 29 06:28 local.rules
-rw-r--r-- 1 root root 0 Jul 29 22:31 white_list.rules
```

- Verified rules: Created rules to detect ICMP (ping), SSH, HTTP, and FTP.

```

GNU nano 8.3                               /etc/snort/rules/local.rules
#Alert on ping attempts
alert icmp any any -> any any (msg:"ICMP test detected"; sid:1000001; rev:1;)
#Alert on FTP connection attempts
alert tcp any any -> $HOME_NET 21 (msg: "FTP Connection attempt"; sid:1000002; rev:1;)
#Alert for SSH connection
#threshold - limit alerts to 1 per source IP per 60 seconds.
#flags:S; → match only SYN packets (the TCP handshake start).
#Useful if you expect multiple sessions in a short time.
alert tcp any any -> $HOME_NET 22 (msg:"SSH connection attempt detected"; flags:S; sid:1000003; threshold: 1; rev:1;)

#Alert for HTTP Connection
alert tcp any any -> $HOME_NET 80 (msg:"HTTP connection attempt detected"; sid:1000004; rev:1;)


```

- Tested config: `sudo snort -T -c /etc/snort/snort.conf`

```

      === Initialization Complete ===

      .-'--> Snort! <*-_
o"   )~ Version 2.9.20 GRE (Build 82)
     '--- By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
          Copyright (C) 2014-2022 Cisco and/or its affiliates. All rights reserved.
          Copyright (C) 1998-2013 Sourcefire, Inc., et al.
          Using libpcap version 1.10.5 (with TPACKET_V3)
          Using PCRE version: 8.39 2016-06-14
          Using ZLIB version: 1.3.1

          Rules Engine: SF_SNORT_DETECTION_ENGINE Version 3.2 <Build 1>
          Preprocessor Object: SF_DNP3 Version 1.1 <Build 1>
          Preprocessor Object: SF_DNS Version 1.1 <Build 4>
          Preprocessor Object: SF_S7COMMPLUS Version 1.0 <Build 1>
          Preprocessor Object: SF_DCERPC2 Version 1.0 <Build 3>
          Preprocessor Object: SF_REPUTATION Version 1.1 <Build 1>
          Preprocessor Object: SF_IMAP Version 1.0 <Build 1>
          Preprocessor Object: SF_SMTP Version 1.1 <Build 9>
          Preprocessor Object: SF_MODBUS Version 1.1 <Build 1>
          Preprocessor Object: SF_GTP Version 1.1 <Build 1>
          Preprocessor Object: SF_SSLPP Version 1.1 <Build 4>
          Preprocessor Object: SF_SDF Version 1.1 <Build 1>
          Preprocessor Object: SF_SSH Version 1.1 <Build 3>
          Preprocessor Object: SF_POP Version 1.0 <Build 1>
          Preprocessor Object: SF_SIP Version 1.1 <Build 1>
          Preprocessor Object: SF_FTPTELNET Version 1.2 <Build 13>

          Total snort Fixed Memory Cost - MaxRss:57040
          Snort successfully validated the configuration!
          Snort exiting
          [07:14:44.115]

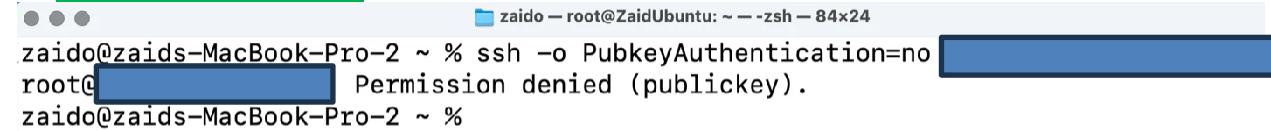
```

- **User & SSH Hardening:**

Root login on my Ubuntu server (hosted on a DigitalOcean Droplet) is secured by allowing SSH access only through verified SSH public keys. Password authentication for SSH is disabled, ensuring that only clients with authorized keys can connect as root.

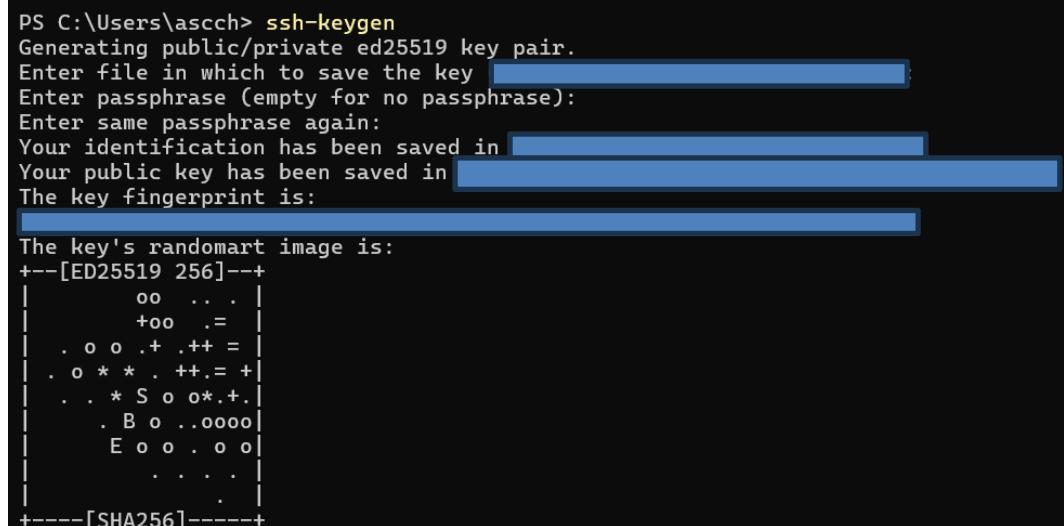
- Edited SSH config: `sudo nano /etc/ssh/sshd_config`
 - Set `PermitRootLogin prohibit-password`
 - Set `PasswordAuthentication no`

- Test using `-o PubkeyAuthentication=no root@192.168.10.10`
(Placeholder IP)



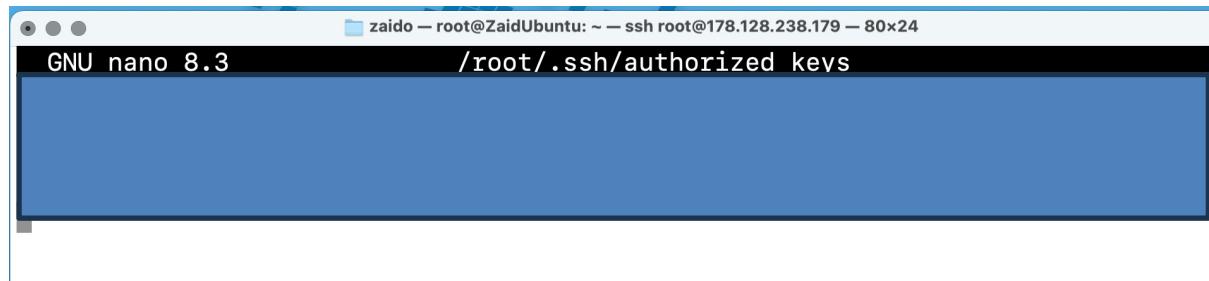
```
zaido@zaids-MacBook-Pro-2 ~ % ssh -o PubkeyAuthentication=no root@192.168.10.10
root@192.168.10.10 Permission denied (publickey).
zaido@zaids-MacBook-Pro-2 ~ %
```

-
- To Add Client's SSH keys:
 - User runs `ssh-keygen` locally.



```
PS C:\Users\ascch> ssh-keygen
Generating public/private ed25519 key pair.
Enter file in which to save the key [C:\Users\ascch\.ssh\id_ed25519]
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in [C:\Users\ascch\.ssh\id_ed25519]
Your public key has been saved in [C:\Users\ascch\.ssh\id_ed25519.pub]
The key fingerprint is:
The key's randomart image is:
+--[ED25519 256]--+
|   oo   . .
|   +oo   .=
|   . o o .+ .++ =
|   . o * * . ++.= +
|   . . * S o o * .+
|   . B o ..oooo
|   E o o . o o
|   . . . .
+---[SHA256]
```

- User sends public key to admin and Admin adds to `nano`
`~/ssh/authorized_keys`



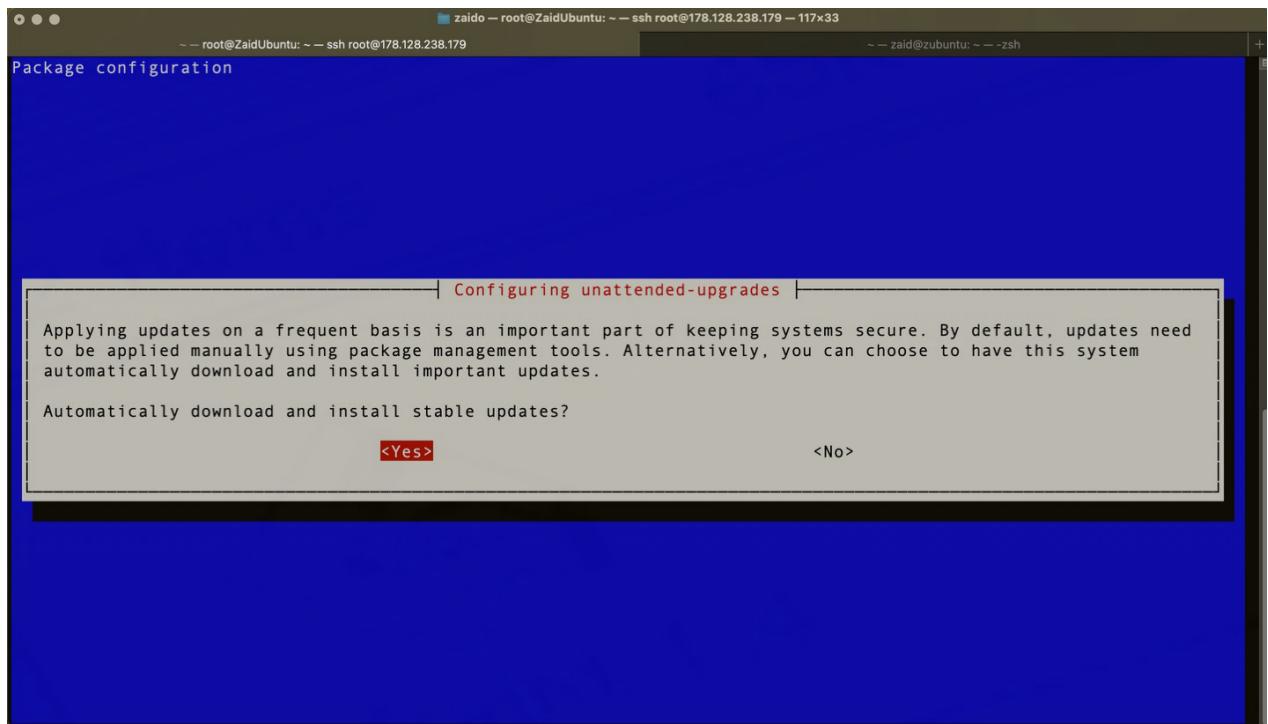
```
zaido@ZaidUbuntu: ~ -- zsh - 80x24
GNU nano 8.3          /root/.ssh/authorized_keys
```

- Restarted SSH: `sudo systemctl restart ssh`

- **Automatic Updates:**

The purpose is to keep your Ubuntu system secure by automatically installing security patches for the OS and installed packages without needing manual intervention. This helps (1) fix vulnerabilities quickly before attackers exploit them. (2) Reduce maintenance work — we don't have to remember to run `apt update` && `apt upgrade` manually every day.

- Installed tool: `sudo apt install unattended-upgrades`
- Configured: `sudo dpkg-reconfigure --priority=low unattended-upgrades`



- Confirming it's running automatically: `cat /var/log/unattended-upgrades/unattended-upgrades.log`.

```
root@ZaidUbuntu:~# cat /var/log/unattended-upgrades/unattended-upgrades.log
2025-07-29 06:56:25,743 INFO Starting unattended upgrades script
2025-07-29 06:56:25,744 INFO Allowed origins are: o=Ubuntu,a=plucky, o=Ubuntu,a=plucky-security, o=UbuntuESMAApps,a=plucky-apps-security, o=UbuntuESM,a=plucky-infra-security
2025-07-29 06:56:25,745 INFO Initial blacklist:
2025-07-29 06:56:25,745 INFO Initial whitelist (not strict):
2025-07-29 06:56:27,182 INFO No packages found that can be upgraded unattended and no pending auto-removals
2025-07-29 07:03:18,859 INFO Starting unattended upgrades script
2025-07-29 07:03:18,860 INFO Allowed origins are: o=Ubuntu,a=plucky, o=Ubuntu,a=plucky-security, o=UbuntuESMAApps,a=plucky-apps-security, o=UbuntuESM,a=plucky-infra-security
2025-07-29 07:03:18,860 INFO Initial blacklist:
2025-07-29 07:03:18,860 INFO Initial whitelist (not strict):
2025-07-30 06:29:55,872 INFO Starting unattended upgrades script
2025-07-30 06:29:55,873 INFO Allowed origins are: o=Ubuntu,a=plucky, o=Ubuntu,a=plucky-security, o=UbuntuESMAApps,a=plucky-apps-security, o=UbuntuESM,a=plucky-infra-security
2025-07-30 06:29:55,873 INFO Initial blacklist:
2025-07-30 06:29:55,874 INFO Initial whitelist (not strict):
2025-07-30 06:29:57,483 INFO No packages found that can be upgraded unattended and no pending auto-removals
root@ZaidUbuntu:~#
```

5. Testing:

- Ran second scan: `sudo nmap -sS 192.168.10.10 (Placeholder Address)`

```
zaid@zubuntu:~$ sudo nmap -sS [REDACTED]
Starting Nmap 7.80 ( https://nmap.org ) at 2025-07-30 08:48 UTC
Nmap scan report for [REDACTED]
Host is up (0.073s latency).
Not shown: 999 filtered ports
PORT      STATE SERVICE
22/tcp    open  ssh

Nmap done: 1 IP address (1 host up) scanned in 8.46 seconds
```

- Ports **22 (SSH)** and **80 (HTTP)** were open and reachable by Nmap in the Initial Scan. After configuring UFW to deny unnecessary ports, only the required **SSH port (22)** remained open, while **port 80** was successfully blocked, and **port 8080** is no longer detectable — **the firewall is working**.
- Snort alerts confirm Nmap scan detected — IDS is working.

```
Commencing packet processing (pid=69986)
07/30-09:21:08.766049  [**] [1:1000001:1] ICMP test detected [**] [Priority: 0] {ICMP} [REDACTED]
07/30-09:21:08.766111  [**] [1:1000001:1] ICMP test detected [**] [Priority: 0] {ICMP} [REDACTED]
07/30-09:21:08.771261  [**] [1:1000004:1] HTTP connection attempt detected [**] [Priority: 0] {TCP} [REDACTED]
07/30-09:21:08.771329  [**] [1:1000001:1] ICMP test detected [**] [Priority: 0] {ICMP} [REDACTED]
07/30-09:21:08.946512  [**] [1:1000002:1] FTP Connection attempt [**] [Priority: 0] {TCP} [REDACTED]
07/30-09:21:08.946609  [**] [1:1000004:1] HTTP connection attempt detected [**] [Priority: 0] {TCP} 184.65.1
07/30-09:21:10.301740  [**] [1:1000002:1] FTP Connection attempt [**] [Priority: 0] {TCP} [REDACTED]
07/30-09:21:10.301740  [**] [1:1000004:1] HTTP connection attempt detected [**] [Priority: 0] {TCP} [REDACTED]
07/30-09:21:10.654244  [**] [1:1000003:0] SSH connection attempt detected [**] [Priority: 0] {TCP} [REDACTED]
07/30-09:21:21.488834  [**] [1:1000003:0] SSH connection attempt detected [**] [Priority: 0] {TCP} [REDACTED]
```

6. Analysis:

Strengths:

- Significantly reduced attack surface.
- Continuous monitoring via Snort IDS.
- Automatic patching minimizes manual oversight.

Limitations:

- Does not protect against zero-day exploits.
- Requires regular review of logs and rule updates.

7. Conclusion:

The server was successfully hardened against common attacks by following CIS Benchmarks and best practices. The documented process can be used for securing similar deployments.

8. References:

- CIS 221 Lecture slides
- Installation and configuration of Snort:
<https://www.youtube.com/watch?v=cD-DoKLzq2s&t=201s>
- Server deployed with DigitalOcean:
<https://www.digitalocean.com/community/tutorials/how-to-set-up-ssh-keys-on-ubuntu-22-04>
- How to Set Up SSH Key-Based Authentication with Ubuntu Server: https://www.youtube.com/watch?v=Or3Sm_COFUY
- How to Install Homebrew on Mac OS:
<https://www.youtube.com/watch?v=Ml-j9qzb9xI>
- How to install and use NMAP on UBUNTU
<https://www.youtube.com/watch?v=WPej0xUmmu4>
- Helped make the data flow diagram
<https://online.visual-paradigm.com/app/diagrams/#diagram:proj=0&type=DataFlowDiagram&width=11&height=8.5&unit=inch>