Task 4: Setup and Use a Firewall on Linux using UFW
Objective:
The main objective of this task was to learn how to configure a firewall to allow or block network traffic on specific ports, and to gain hands-on experience with UFW (Uncomplicated Firewall), a user-friendly interface for iptables in Linux.

Tools Used:
OS: Ubuntu 22.04 LTS (Linux)

Terminal

UFW (Uncomplicated Firewall)

Telnet utility for testing blocked ports

GitHub for documentation and submission

Step-by-Step Implementation:
Step 1: Check Firewall Status
Before making any changes, I verified whether UFW was active and what rules were currently applied.

bash
sudo ufw status verbose
Output:

Status: inactive

So, I enabled UFW using:

bash
sudo ufw enable
Step 2: List Existing Rules
To get a baseline of current firewall rules, I ran:

bash
sudo ufw show added
This showed no rules had been added yet.

Step 3: Block Inbound Traffic on Port 23 (Telnet)
I blocked Telnet traffic to demonstrate firewall filtering. Telnet is known for being insecure since it transmits data in plaintext.

bash
sudo ufw deny 23
This added a rule that blocks all incoming traffic on port 23.

Step 4: Test the Rule
To test if the port was successfully blocked, I installed the telnet tool and attempted to connect to my own system on port 23.

bash
telnet localhost 23
Result:

Connection refused

This confirmed that the firewall rule was working as expected.

Step 5: Allow SSH (Port 22)
To avoid accidentally locking myself out (especially in a remote server scenario), I added a rule to allow SSH.

bash
sudo ufw allow 22
Step 6: Remove the Test Rule
After testing, I cleaned up by removing the Telnet rule to restore the system's previous state.

bash
sudo ufw delete deny 23
Step 7: Documented Everything
I took terminal screenshots of each step — enabling UFW, adding/removing rules, testing blocked ports — and added them to my GitHub repo.

Firewall Traffic Filtering - Summary in My Own Words:
A firewall acts like a security guard at the network entrance. It watches over data packets and decides what should be allowed in or out based on rules. UFW makes firewall configuration beginner-friendly by providing simple commands for tasks that usually require complex iptables syntax.

By blocking port 23, I learned how to defend against insecure services. Allowing port 22 helped me understand the importance of keeping critical access paths open. I

now know how to toggle rules, test connectivity, and manage traffic directionally (inbound/outbound).

Conclusion:
By completing this task, I developed hands-on skills in firewall configuration using UFW. I now understand how to manage ports, test connectivity, and write clear documentation. These skills are essential for real-world cybersecurity scenarios where controlling access is critical.