**INTERNSHIP TASK 4 REPORT**

**Setup and Use a Firewall on Linux using UFW**

**Intern Name:** Aryanshu Saha   
 **Internship Domain:** Cyber Security / Network Security  
 **Tool Used:** UFW (Uncomplicated Firewall)  
 **Operating System:** Kali Linux   
 **Date Performed:** 27 Jun. 2025

1. **Objective**

The primary objective of this task was to understand and practically implement basic firewall configurations using UFW on a Linux-based system. This includes enabling the firewall, listing active rules, blocking and allowing specific network ports, and testing rule enforcement through local connectivity attempts.

1. **Tools & Environment Details**

| **Component** | **Description** |
| --- | --- |
| OS | Kali Linux 2023.4 (or Ubuntu XX.XX) |
| Firewall Tool | UFW – Uncomplicated Firewall (frontend for iptables) |
| Terminal Shell | Bash (running as root or sudo) |
| Optional Tools | Telnet utility for testing port blocks |
| Network Status | Localhost (127.0.0.1) and system IP as target |

1. **Introduction to UFW (Uncomplicated Firewall)**

UFW is a simplified interface for managing firewall rules using iptables, the native Linux firewall. It allows administrators and users to allow or deny traffic on specific ports, interfaces, and protocols. By default, UFW denies all incoming traffic and allows all outgoing traffic, but it can be tailored to match security policies and protect endpoints against unauthorized access.

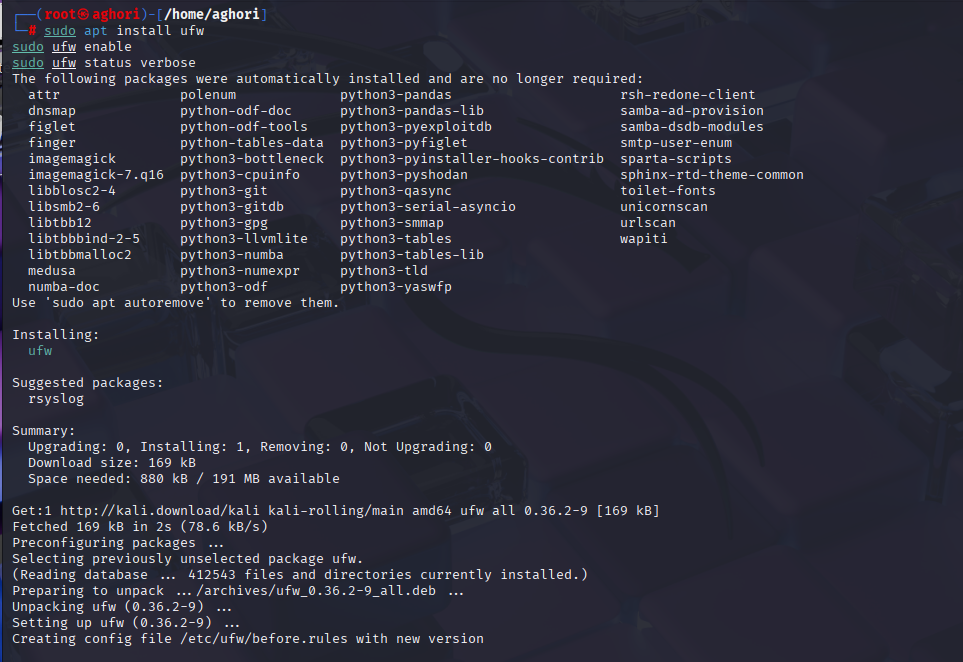
1. **Step-by-Step Implementation with Observations**

**🔹 Step 1: Checking UFW Installation and Enabling the Firewall**

sudo apt install ufw

sudo ufw enable

sudo ufw status verbose



**Observation:**  
UFW was successfully enabled. Default policy was:

* Incoming: Deny
* Outgoing: Allow

**🔹 Step 2: Listing Current Firewall Rules**

sudo ufw status numbered



**Observation:**  
No specific rules were present initially. The system was operating with only the default deny/allow policies.

**🔹 Step 3: Blocking Inbound Traffic on Port 23 (Telnet)**

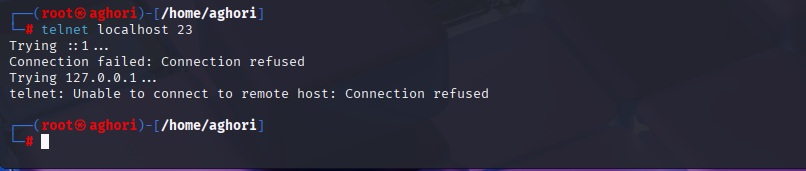
sudo ufw deny 23



**Explanation:**  
Port 23 is used by the legacy Telnet protocol, which transmits data in plain text and is considered insecure. Blocking this port helps mitigate unauthorized remote access risks.

**🔹 Step 4: Testing the Block with Telnet**

telnet localhost 23



**Result:**  
Connection to port 23 was successfully blocked, demonstrating that the UFW rule was enforced as intended.

**🔹 Step 5: Allowing SSH Access on Port 22**

sudo ufw allow ssh



**Rationale:**  
SSH is the default protocol used for secure remote shell access. Allowing port 22 ensures that legitimate administrators can still manage the system remotely.

**🔹 Step 6: Removing the Port 23 Block Rule**

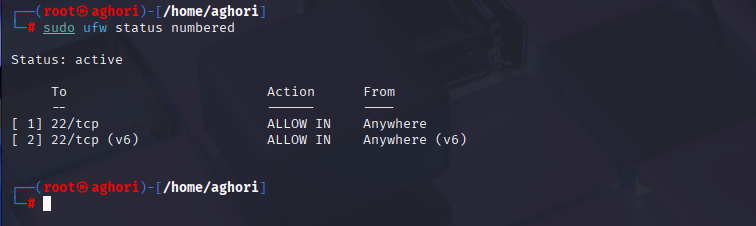
sudo ufw delete deny 23



**Outcome:**  
The firewall rule was successfully removed, restoring the system to its original state after testing.

**🔹 Step 7: Listing Final Firewall Rules**

sudo ufw status numbered



**Firewall Rules Summary**

| **Rule Action** | **Port** | **Protocol** | **Purpose** |
| --- | --- | --- | --- |
| Deny | 23 | TCP | Block insecure Telnet connections |
| Allow | 22 | TCP | Permit secure SSH access |

1. **What I Learned – Concepts & Real-World Application**

* Firewalls are essential first-line defenses in both enterprise and personal systems.
* UFW provides a user-friendly way to manage iptables without dealing with complex syntax.
* Ports represent services, and selective port access directly controls service availability.
* Blocking Telnet (port 23) reflects best practices for disabling insecure protocols.
* Enabling SSH (port 22) is necessary for secure remote administration, especially in cloud or server contexts.
* Testing firewall rules with tools like telnet, nmap, or netcat helps verify implementation success.

1. **Conclusion**

This task strengthened my understanding of Linux firewall management, rule application, and network traffic control. UFW proved to be an efficient tool to demonstrate basic firewall functionality, enabling me to apply industry-relevant practices such as:

* Port hardening
* Remote access control
* Secure protocol enforcement

I can now confidently configure basic firewall protections and verify rule behavior on both personal systems and enterprise Linux environments.