Cybersecurity Daily Dairy

Day 10: Firewall Implementation and SSH/FTP Control

Date: June 28, 2025

Topics Covered:

- Firewall implementation using a Python-based tool
- Managing open ports for system security
- Secure Shell (SSH) access management
- File Transfer Protocol (FTP) access management
- Identifying and closing unnecessary open ports

What I Did:

On Day 10, I implemented a **firewall** using a Python-based tool to manage and monitor port-based traffic. I learned how to define rules that allow or deny access based on IP addresses and ports. Additionally, I worked with **SSH** and **FTP** to connect to remote machines and tested how firewall rules affect these services. I also explored methods to detect and secure open ports using network scanning tools.

Firewall Implementation:

- A Python script was used to build a lightweight firewall.
- It monitored incoming and outgoing packets.
- Allowed configuration of **allow/deny rules** based on port numbers or IP addresses.
- Unauthorized traffic was detected and blocked.
- Logs were maintained for every access attempt or denial.
- Unused ports were blocked to prevent exploitation.

SSH and FTP Management:

SSH (Secure Shell):

- SSH was used to securely access a remote system from the terminal.
- I practiced remote login, logout, and tested how firewall settings impact SSH sessions.
- Verified that blocking port 22 (default for SSH) prevented remote access.

FTP (File Transfer Protocol):

- FTP was tested for file transfer between systems.

- Verified that when port 21 (default for FTP) was blocked by the firewall, connection was refused.
- Simulated attacks were blocked using deny rules in the firewall script.

Detecting and Closing Open Ports:

- Used tools like **netstat**, **ss**, and **nmap** to scan for open ports.
- Identified services that were unnecessarily running or listening.
- Updated firewall configurations to block unused or vulnerable ports.
- Re-ran scans to verify that ports were successfully closed or filtered.

Key Learnings:

- Firewalls are essential for **controlling access** and **blocking malicious traffic**.
- Python can be used to create custom, scriptable firewalls for simple use cases.
- SSH and FTP are common attack vectors and must be tightly controlled.
- Regular port scanning and cleanup improves overall system security.
- Logs help track intrusion attempts and support forensic investigation.