Introduction to Firewalls and Network Security

Cybersecurity

Networks Security Day 1

Today’s Objectives

By the end of class, you will be able to:

- ☐ Explain the relationship between ports and services.
- ☐ Describe how open ports contribute to a machine’s attack surface.
- ☐ Explain how firewalls can be used to protect a machine’s open ports.
- ☐ Develop and implement firewall policies with UFW and `firewalld`.

Restricting access to open ports is a fundamental skill for any technical security specialist.

Introduction and Professional Application

The ability to read and write firewall rules is expected for the following cyber professions:

- Help Desk / IT Specialists
- System / Network Administrators
- SOC Analysts
- Penetration Tester

Professional Application

Help Desk / IT Specialist

Entry-level IT roles focus more on troubleshooting user issues than implementing network controls. However, being able to determine if and how firewalls affect user traffic can help when troubleshooting issues like:

- a slow connection
- an unexpected lack of a connection
- broken networked applications such as Skype or Facebook Messenger

Professional Application

System/Network Administrator

System and Network Administrators are usually in charge of determining who should be allowed to access which devices on the network.

People in these roles must be able to develop firewall policies by determining which ports to open and which to close.

They must be aware that network controls can be implemented on hosts, using tools operating system-based tools `ufw` or `firewalld`.

Real world host side communication controls are now done with Host Intrusion Detection and Prevention software (HIDS and HIPS).

Professional Application

SOC Analyst

While SOC Analysts don't develop or implement firewall rules as a primary responsibility, they must be able to understand how firewalls on the network filter incoming and outgoing packets in order to accurately interpret the traffic logs they monitor for incidents.

Without this knowledge, they wouldn't be able to identify abnormal network traffic.

Professional Application

Penetration Tester

Penetration Testers also do not usually implement firewall rules themselves.

However, launching a successful attack against a network requires that they are able to determine whether a firewall sits between them and their target, if possible.

- If so, they must deduce which rules are enabled base on the results of their probing.
- Therefore, they can attempt to determine ways to work around a firewall and continue with their probing.

Ports, Services and Firewalls

Review of Ports and Services

Ports are essentially pathways into a machine's file system, allowing machines to send and receive data to each other.

Computers use ports by starting a program that needs to communicate on the network, then binding that program to its own specific port.

Open ports, like 80 (HTTP) and 443 (HTTPS), pose security vulnerabilities, as attackers can send malicious data containing SQL injections, XSS payloads, etc.

These threats can be mitigated through access controls.

How Ports and Services Work

Ports are essentially pathways into a machine’s file system, allowing machines to send and receive data to each other.

- Computers use these ports by initiating programs that need to communicate on the network and then **binding** those programs to their own specific port.

- **For example:** A user might start SSH on port 22.
  - The port number 22 acts like the “name” for the SSH server
  - When another device wants to communicate, it sends a message to `<IP Address>: 22`

- Ports can be **open** or **closed**.
  - When ports are not bound to service, they are considered **closed** and will not be able to receive data.
  - If a port is open, it will be able to send and receive data, but it also poses a security risk.

Open Port Vulnerability

Public servers must open `port 80 (HTTP)` and `443 (HTTPS)`.

HTTP server cannot whitelist IP addresses:

- Suppose a web server is running a vulnerable version of WordPress. In this case, anyone who requests the site will have access to the vulnerable application.

- Attackers can send requests to this site and use a SQL injection or something similar to take over the web server.
    - This is a common real-world scenario and ultimately possible because the web server had to open ports 80 and 443, but wasn't able to filter malicious users from genuine users.

- **SSH** is also vulnerable to exploits like brute-force passwords and stolen keys.

Securing the Server Through Access Controls

Access control can mitigate these threats to SSH and HTTP servers.

`The SSH server` can be protected by only allowing specific IP addresses to connect.

- **For example**: The server might only accept connections to port 22 from the administrator’s known IP address of `172.16.5.10` and deny attempts to connect from any other addresses.

- Protecting the SSH server by allowing only one IP address to connect is an access control because it restricts access to SSH to only a single host.

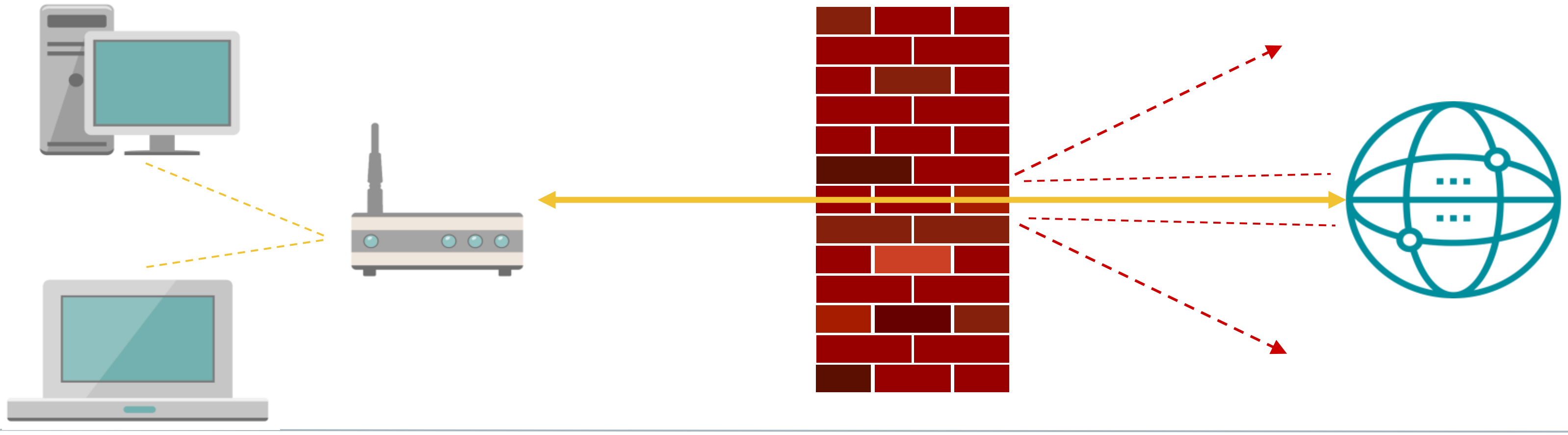
`The HTTP server` can be protected by detecting and dropping specific traffic.

- Dropping suspicious HTTP traffic is an access control because it restricts access to ports 80 and 443 to only hosts that send benign traffic.

Firewalls

Administrators would use **firewalls** to control access to SSH servers.

- Firewalls are devices or software that function as packet filters to intercept TCP and UDP packets as they enter a network, allow trusted packets through and then drop the rest.
  - Firewalls usually work by reading OSI layer 3 (Networking) and Layer 4 (Transport)



Firewalls

Administrators would use **a web application firewall (WAF)** to protect the HTTP server.

- A WAF is specifically designed to understand threatening activity.
    - It can identify suspicious HTTP traffic, log information about it, and drop it before it gets to the target server.
- Typically, firewalls only understand Layer 3-4 traffic. However, web applications communicate with HTTP, which is a Layer 7 protocol.
- Therefore, a firewall protecting a web server needs to understand application-layer (7) protocols.

Firewalls

Firewalls can be used to control access to either a single host (host-based firewall) or an entire network (network firewall).

- A **host-based firewall** runs on the computers that they are meant to protect and block traffic to/from that specific device.

- A **network firewall** is often placed in front of a router, in order to block malicious traffic from entering a private network.

These two types of firewall work in the same way, by:

- Intercepting traffic before it reaches a target host,
- Inspect the source and destination addresses / ports, TCP flags, and other features of the incoming packets.
- Allow packets that come from trusted sources and deny packets that don’t.

Firewall Rules

Firewalls decide which packets to allow or deny by applying rules to each packet.

A firewall rule is essentially a description of the kinds of packets that should be allowed or denied.

- Firewalls apply these rules by reading the source and destination information.
- It then determines whether to allow or reject the packet by comparing that information to the list of hosts it already knows to allow or deny.

For example: Administrator can protect the SSH server by restricting access to port 22 to only 172.16.5.5.

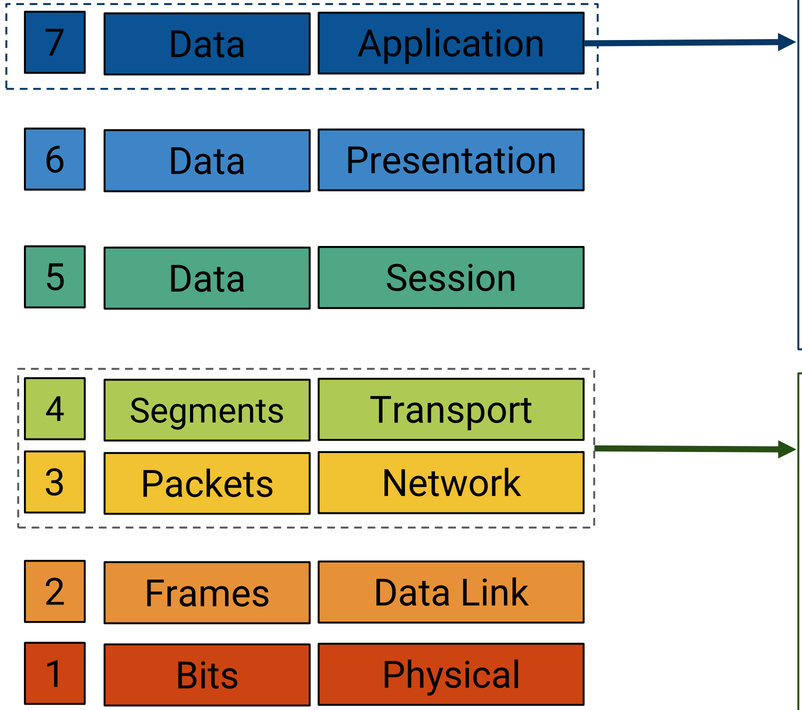
In this case, the rule would be: *"If the packet comes from 172.16.5.5 and is bound for 172.15.5.10, **allow** it into the network. Otherwise **reject** it.*

Firewall Rules

Other firewall rules include:

- *"If this TCP packet comes from the subnet 172.16.5.0/24, allow it into the network. Otherwise, reject it*."
- "*If this TCP packet comes from the IP address 216.58.193.175, reject it*."
- "*If this TCP packet has every TCP flag set, reject it*.”
- "*If this TCP packet was sent less than 1ms after another packet from the same host, reject it*."

Firewalls: Stateful and Stateless



Stateful firewalls monitor the whole TCP connection, instead of just its constituent packets.

A series of related TCP packets are a TCP Stream, which are transferred over TCP Connections.

Application protocols like HTTP involve transferring large amounts of data over TCP connections.

Stateless firewalls allow or reject traffic on a “per-packet” basis.

Applies rules by reading source and destination data of Layer 3 and Layer 4 level information

“State-less” because they do not “remember” anything about previous packets they’ve seen.

Firewalls can inspect almost any feature of incoming packets, the most important features being:

- **Source/Destination IP/MAC Addresses**
    - Does the packet comes from a device in a trusted subnet or is the response to a request made from a trusted device?
- **TCP Flags**
    - Malicious packets often have non-standard TCP flag settings.
    - For example: Hacking tools like Nmap's Xmas scan set non-standard combinations of TCP flags to bypass firewalls.
- **Malformed or Corrupt Data**
    - Attackers sometimes send specially crafted malformed or corrupted data as part of an attack. Firewalls can be configured to deny packets containing this type data.
- **Transmission Rate**
    - Firewalls can be configured to block traffic from machines that send too many packets too quickly.
    - For example: an administrator might configure a web server to accept traffic from any IP address, but block traffic from any host that sends more than 100 requests in one second.
    - This process is called **rate limiting** or **request throttling.**

Stateful Firewalls

Inspecting the whole data stream allows stateful firewalls to detect more information:

- If a packet is trying to establish a new connection. (New State)
- If a packet is part of an existing connection. (Established State)
- Is a packet is rouge. (Neither opening a new or part of an established)
- Which protocol is in use.

FTP / Stateful Firewall Process

File Transfer Protocol (FTP) is used to transfer files between machines, usually running on port 21.

01 FTP Command Channel: When a client wants to download a file, they connect to the server's port 21 and requests a file, with a command like GET `myfile.txt`.

02 The server opens a new connection back to the client, separate from the FTP Command Channel. Active vs. Passive

03 FTP Data Channel: To open this connection, the server chooses a random port on the client to send the file to. –Passive Mode FTP. Or opens port 21-Active Mode FTP

04 After the file download completes, the server and client tear down the FTP data channel.

05 From here, the user can continue to use the command channel to download/upload other files.

UFW

UFW

Most operating systems come with a built-in firewall solution. On Linux, the de facto firewall is **Uncomplicated FireWall (UFW).** Features include:

Configuring Allow/Deny rules

- For example: An administrator might automatically drop packets bound to any port they have not explicitly opened.

Logging Dropped and Suspicious Packets

- Helpful for determining if dropped packets were sent in honest error or as part of an attack.

Rate-Limiting Incoming Traffic

- Helpful for preventing denial-of-service (DoS) conditions and brute-force attacks.

Automatically Setting rules for Common Applications

- Admins can install "application profiles" for UFW, which are pre-made lists of common rules for common applications.

UFW Demo

1. Enabling UFW

UFW isn’t initiated by default. Admins need to enable it themselves.

Verify the status of the firewall (active or inactive) by running `sudo ufw status`

- UFW commands require sudo rights, because manipulating the firewall affects all users on the system.

Start UFW by running `sudo ufw enable`

UFW Demo

2. Setting Default Rules

Once UFW is running, admin’s typically set default rules.

Default rules are applied to packets that administrators have not applied specific rules to.

- For example: An administrator might set a rule that allows SSH traffic to a server’s port 22.
- Rather than write an explicit rule denying traffic to every other port, they can protect the machine by setting a “default deny” rule.
- This rule drops all incoming packets except for those bound for port 22.

A secure firewall policy should deny all incoming and outgoing packets by default. Then, administrators should open only the ports they know.

```
$ sudo ufw default deny incoming

$ sudo ufw default deny outgoing
```

UFW Demo

3. Allow and Deny Rules

Denying all incoming and outgoing traffic immediately breaks all networked applications, since it prevents them from sending or receiving data.

- To remedy this, administrators need to explicitly allow traffic to/from the ports they expect users to need.
- For example: enabling web browsing means allowing traffic to/from ports 80 and 443.

```
sudo ufw allow 80
sudo ufw allow 443


sudo ufw deny 80
```

UFW Demo

4. Deleting Rules

You can delete firewall rules once they are no longer necessary, or if there were erroneously configured.

To delete a rule, append **`delete`** to the beginning of the command.

```
sudo ufw delete deny 80
```

To see that the rule is gone, list all enabled rules: **`sudo ufw status verbose`**

You must start and restart the firewall for their changes to take effect:

```
$ sudo ufw disable
$ sudo ufw start
```

Activity: Configuring UFW

Suggested Time:

Take a Break!

firewalld

`firewalld` is a service to test and update firewall rules across interfaces without restarting the firewall to implement the new rule.

Firewalld

Firewalld does not require users to restart their firewall when implementing new rules.

- Like UFW, firewalld loads rules from a database before filtering packets.
- Unlike UFW, firewalld allows administrators to change those rules on the fly.
- When rules are changes, firewalld will immediately use the updated rules until it is changed again, or the server is shut down.
- Then, when firewall is restarted, it will default to the original rules of the database.

Firewalld Zones and Services

Firewalld incorporates **network zones** to simplify organizing rules.

- Different interfaces might need to implement different firewall policies, because each might provision different services.
  - It can be difficult to keep track of which rules correspond to which interfaces.

- Firewalld requires you to associate each interface with a **zone**.
  - Rules are written in a zone and only apply to the interface associated with the zone.

Firewalld uses services to act as shortcuts to configure rules for common services.

- For example: if you enable the SSH service in a zone, it will open port 22 without needing to specify the port number.

Firewalld Commands

List all zones: **`sudo firewall-cmd --list-all-zones`**

Assign an interface to a zone : **`sudo firewall-cmd --zone=<ZONE> --change-interface=eth1`**

List all service: **`sudo firewall-cmd --get-services`**

List all active rules in a zone: **`sudo firewall-cmd --zone=<ZONE> --list-all`**

Set a rule with a command like:

```
sudo firewall-cmd --zone=work --add-rich-rule=’rule family=”ipv4” source
address=”10.10.10.10” reject
```

- This rule blocks all traffic from 10.10.10.10 in the work zone.

Activity: Configuring firewalld

In this activity, you will continue to act as a system administrator tasked with updating a firewall to implement different rules for different zones.

Suggested Time:

Professional Application

Let’s review some of the professionals roles that are relevant to today’s lesson.

- Help Desk / IT Specialists
- System / Network Administrators
- SOC Analysts
- Penetration Tester

Today’s Objectives

By the end of class, you will be able to:

- ✓ Explain the relationship between ports and services.
- ✓ Describe how open ports contribute to a machine’s attack surface.
- ✓ Explain how firewalls can be used to protect a machine’s open ports.
- ✓ Develop and implement firewall policies with UFW and `firewalld`.