Network & Firewall

Life of a packet

Anish & Yubo

CS Club Cybsersecurity

Table of Contents

Logistics

Attendance

Please scan the QR code on the right and choose the seminar "CyberPatriot (Cybersecurity)"

The taking of attendance is required for administrative purposes.

https://www.tinyurl.com/gsmstcsclubattendance

Mock & QA

Mock State Round

- December 3 9:00 AM-3:00 PM
- Lower-level lecture hall
- No membership requirement

https://www.tinyurl.com/MockRoundRSVP

Q&A

Scan this QR code on the left if you have any questions. All the questions will be collected and answered collectively in the end.

https://www.tinyurl.com/cyberpatriotqa

Sticker Sell

Computer Science club is selling stickers next week in atrium and café, before and after school. Please check the CS Club channel on Discord for more information. Those stickers including

logos of various framework CSS (Cascading Style Sheets), HTML (HyperText Markup Language), and JavaScript, are great for decorating your laptop or notebook.

fun STEM stickers such as a periodic table, a DNA molecule, and a computer chip.

and more . . .

The stickers are $1.00 each.

Network

Server & Client

Consider the following scenario: as a Java developer, you made an extremely simple server and client, and you want to test them on your own computer. When the client receives the `"CyberPatriot"` from the server, what happened under the hood?

```java
public class Server {
    public static void main(String[] args) throws IOException {
        ServerSocket server = new ServerSocket(12345);
        Socket socket = server.accept();
        PrintWriter out = new PrintWriter(socket.getOutputStream(), true);
        out.println("CyberPatriot");
    }
}
```

```java
public class Client {
    public static void main(String[] args) throws IOException {
        Socket socket = new Socket("localhost", 12345);
        BufferedReader in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
        System.out.println(in.readLine()); // CyberPatriot
    }
}
```

IP

- In the form of `nnn.nnn.nnn.nnn`, where each `nnn` is 0 to 255.
- Connection through ISP (Internet Service Provider) is given temp IP (same thing with DHCP – Dynamic Host Configuration Protocol), while connection through LAN (Local Area Network) is given permanent IP Address
- Pinging server means sending ICMP (Internet Control Message Protocol) and echo response message to original computer. The original computer counts the time while it waits for the response.

MAC

- In the form of `nn:nn:nn:nn:nn:nn`, where each `nn` is 0 to 255.
- MAC address is a unique identifier assigned to network interfaces for communications on the physical network segment.
- MAC address is used to identify the device on the network.

ARP

- ARP (Address Resolution Protocol) is a protocol used to resolve IP addresses to MAC addresses.
- ARP is used to find the IP address of a device on the network.
- RARP (Reverse Address Resolution Protocol) is used to find the MAC address of a device on the network, given the IP address.

MAC vs. IP

- Although both MAC and IP are used to identify a device on the network, they are used for different purposes.
- MAC is like name in the classroom. It is used to identify a device on the network.
- IP is like classroom number in school. It is employed on a larger scale (entire internet) to identify a device on the network.

TCP vs. UDP

- TCP (Transmission Control Protocol) is a connection-oriented protocol, which means that the connection between the sender and receiver must be established before any data can be sent.
- UDP (User Datagram Protocol) is a connectionless protocol, which means that the connection between the sender and receiver does not need to be established before any data can be sent.

Structure of TCP Packet

- TCP packet is divided into 3 parts: header, data, and checksum.
- Header contains information about the packet, such as source and destination port, sequence number, acknowledgement number, and flags.
- Data contains the actual data being sent.
- Checksum is used to verify the integrity of the packet.

Structure of UDP Packet

- UDP packet is divided into 2 parts: header and data.
- Header contains information about the packet, such as source and destination port.
- Data contains the actual data being sent.

Session-based vs. Stateless

- Session-based protocol is a protocol that maintains a session between the sender and receiver. For example, TCP maintains a session between the sender and receiver, ensuring that the data is sent in order and that the data is not lost.
- UDP is a stateless protocol, which means that it does not maintain a session between the sender and receiver. It's like throwing data into the air and hoping that it lands on the receiver.

State Machine of TCP

1. When the client sends a SYN to the server, the client enters the SYN_SENT state.
2. Then, the server sends a SYN and ACK to the client, and the server enters the SYN_RCVD state.
3. Then the client sends an ACK to the server, and the client enters the ESTABLISHED state.
4. When the client sends a FIN to the server, the client enters the FIN_WAIT_1 state.
5. Then the server sends an ACK to the client, and the server enters the CLOSE_WAIT state.
6. Then the client sends an ACK to the server, and the client enters the FIN_WAIT_2 state.
7. Then, the server sends a FIN to the client, and the server enters the LAST_ACK state.
8. Finally the client sends an ACK to the server, and the client enters the CLOSED state. When server receives the ACK, it enters the CLOSED state. And the connection is closed.

State Machine of TCP

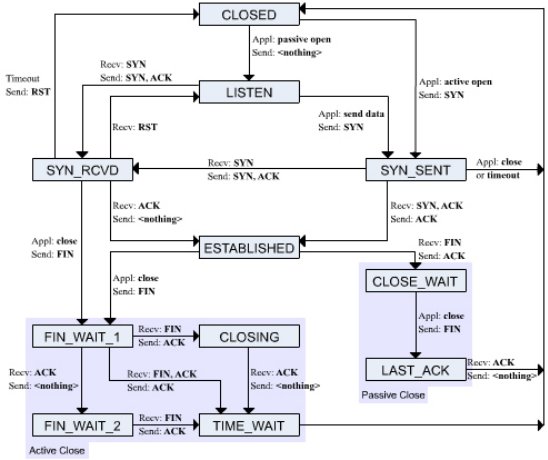

Figure: State Machine of TCP

Firewall

A firewall is a network security system that monitors and controls incoming and outgoing network traffic based on predetermined filters. Filters can be determined by:

- IP address (destination or source)
- Port number
- Protocol
- Application (e.g. HTTP, FTP, SSH)

Mainly determined by well-known port numbers

Figure: Firewall

Tools to Configure

A variety of tools can be used to configure a firewall in Linux. All those tools are simply interface/wrapper to the `netfilters` or !nftables! kernel modules. The most common tools are:

- `iptables` is the most commonly used firewall in Linux. It can be configured by editing the `/etc/sysconfig/iptables` file. It is pretty deprecated and not recommended to use.
- `firewalld` is a more recent firewall in Linux. It can be configured by running `firewall-cmd` commands. This is more a RedHat thing that are unavailable in Debian-based distributions.
- `ufw` is a simple firewall in Linux. It can be configured by running `ufw` commands.
- `nftables` is a new firewall in Linux. It can be configured by editing the `/etc/nftables.conf` file.

Learning Goals

Deploy

- `ufw`
- `iptables`
- `nftables`

Configure

- `ufw`
- `iptables`
- General idea about firewalls
- General introduction about network.

This will not be a comprehensive list of all the commands and features of the tools. We will only cover the most commonly used ones. Refer to manual pages for more information.

Firewall

Deployment

`ufw`, or Uncomplicated Firewall, is a simple firewall in Linux. It can be configured by running `ufw` commands.

- To install `ufw`, run:

```
sudo apt install ufw
```

- To enable `ufw`, run:

```
sudo ufw enable
```

- To disable `ufw`, run:

```
sudo ufw disable
```

- To check the status of `ufw` and display the rules (if any), run:

```
sudo ufw status
```

`iptables` originated from the `netfilter` kernel module. It is a very powerful, yet deprecated. Hence, it's still important to talk about it, at least.

`iptables` only manipulates the kernel module. It does not store any configuration. Hence, the configuration will be lost after a reboot. To make the configuration persistent, we need to save the configuration to a file and load it on boot. Further, because of this nature, `iptables` does not need to be enabled or disabled. It is always enabled.

To install `iptables`, run:

```
sudo apt install iptables
```

`nftables` is a new firewall in Linux. It works in an almost identical way to `iptables`, serving as it's replacement.

To install `nftables`, run:

```
sudo apt install nftables
```

Firewall

Usage of `iptables`

Chain & Rules

The firewall will read the configured policy rules from top to bottom, execute only the first matched rule. If no rule is matched, the default policy will be applied.

Chain is a collection of rules. There are three types of chains:

`INPUT` is the chain for incoming packets.
`OUTPUT` is the chain for outgoing packets.
`FORWARD` is the chain for forwarded packets.
`PREROUTING` is the chain for packets before routing.
`POSTROUTING` is the chain for packets after routing.

Rule is a set of conditions and actions. The following actions is possible for a rule:

`ACCEPT` will accept the packet.
`DROP` will drop the packet. No response will be sent.
`REJECT` will reject the packet. A response telling the packet is rejected will be sent.
`LOG` will log the packet.

List of Rules I

To list all the rules, run:

```
sudo iptables -L
```

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

List of Rules II

To list all the rules in a specific chain, run:

```
sudo iptables -L INPUT
```

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
```

Remove all Rules

To remove all rules, run:

```
sudo iptables -F
```

Set INPUT default to DROP

To set the default policy of the INPUT chain to DROP, run:

```
sudo iptables -P INPUT DROP
```

-P is used to set the default policy of a chain.

DISABLE `ping`

To disable ICMP (ping), run:

```
sudo iptables -A INPUT -p icmp
↪  --icmp-type echo-request -j
↪  DROP
```

To allow ICMP (ping), run:

```
sudo iptables -D INPUT -p icmp
↪  --icmp-type echo-request -j
↪  DROP
```

- **`-A`** will append the rule to the end of the chain.
- **`-I`** will insert the rule to the beginning of the chain. This would allows our drop rule to be executed first.
- **`-D`** will delete the rule from the chain.
- **`-p`** will specify the protocol. `ping` use ICMP.
- **`--icmp-type`** will specify the type of ICMP.
- **`-j`** will specify the action.

Only Allow SSH From Certain Host

To only allow SSH from certain host, run:

```
sudo iptables -A INPUT -p tcp --dport 22 -s 192.168.10.0/24 -j ACCEPT
```

- `-s` will specify the source IP address. We use CIDR notation to specify the range of IP address. Here `192.168.10.0/24` represents all the host with in network `192.168.10.*`
- `-d` will specify the destination IP address.
- `-dport` will specify the destination port. In this case, it is `22` as SSH uses port `22` by default.
- `-sport` will specify the source port.

Ban some host from accessing the server

Assuming this is an HTTP server with port `80` open, to ban some host from accessing the server, run:

```
iptables -I INPUT -p tcp -s 192.168.10.5 --dport 80 -j REJECT
```

The above command will prevent `192.168.10.5` from accessing `80` port of the server. The `REJECT` action will send a response to the host, telling it that the connection is rejected.

Save

To save the rules, run:

```
sudo iptables-save
```

To save the rules to a file, run:

```
sudo iptables-save > /etc/iptables/rules.v4
```

To load the rules from a file, run:

```
sudo iptables-restore < /etc/iptables/rules.v4
```

`iptables`'s rules will be lost after reboot. Restore command must be added to some on-boot service to restore the rules after reboot.

THANK YOU!