Spoofing

When a normal user sends out a packet, operating systems usually do not allow the user to set all the fields in the protocol headers (such as TCP, UDP, and IP headers). OSes will set most of the fields, while only allowing users to set a few fields, such as the destination IP address, the destination port number, etc. However, if users have the root privilege, they can set any arbitrary field in the packet headers. This is called packet spoofing, and it can be done through *raw sockets*.

Raw sockets give programmers the absolute control over the packet construction, allowing programmers to construct any arbitrary packet, including setting the header fields and the payload. Using raw sockets is quite straightforward; it involves four steps:

(1) create a raw socket

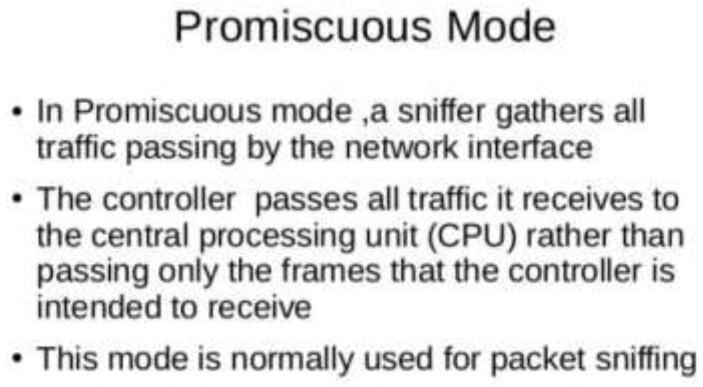
(2) set socket option

(3) construct the packet

(4) send out the packet through the raw socket.

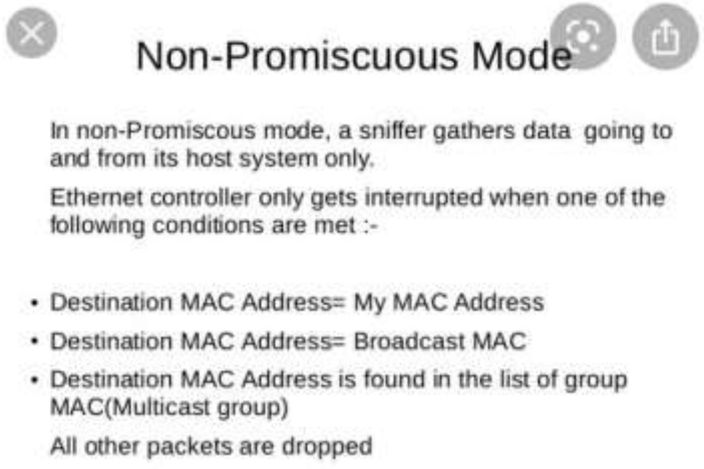
Why do you need the root privilege to run sniffex? Where does the program fail if executed without the root privilege?

Pcap_lookupdev() function needs root access because it wants to access network interfaces and it is impossible without root access in linux. Sniffer programs need raw sockets that allow direct sending of packets by the applications bypassing all applications in network software of operating system. And we need to be a root to create raw socket as we can’t discover NIC until we are root.



Promiscuous Mode

- In Promiscuous mode ,a sniffer gathers all traffic passing by the network interface
- The controller passes all traffic it receives to the central processing unit (CPU) rather than passing only the frames that the controller is intended to receive
- This mode is normally used for packet sniffing



Non-Promiscuous Mode

In non-Promiscous mode, a sniffer gathers data going to and from its host system only.

Ethernet controller only gets interrupted when one of the following conditions are met :-

- Destination MAC Address= My MAC Address
- Destination MAC Address= Broadcast MAC
- Destination MAC Address is found in the list of group MAC(Multicast group)

All other packets are dropped

LAB 2:

3 parameter to hijack session

1-Cookie

2-Session

3-TS Value

Cross-site scripting (XSS) is a type of vulnerability commonly found in web applications. This vulnerability makes it possible for attackers to inject malicious code (e.g. JavaScript programs) into victim’s web browser. Using this malicious code, the attackers can steal the victim’s credentials, such as session cookies.

```
<script>alert(’XSS’);</script>
```

Script attack agar input field m imited character allow ho or code bara ho toh we can make separate file for it

```
<script type="text/javascript" src="http://www.example.com/myscripts.js">
</script>
```

The `Elgg` server cannot distinguish whether the request is sent out by the user’s browser or by the attacker’s Java program.we provide you with a sample java program that does the following:

1. Open a connection to web server.

2. Set the necessary HTTP header information.

3. Send the request to web server.

4. Get the response from web server.

LAB 3:

Behavioral targeting is a type of online advertising where ads are displayed based on the user's web-browsing behavior. The user leaves a trail of digital foot prints moving from one website to the other. Behavioral targeting anonymously monitors and tracks the sites visited by a user. When a user surfs internet, the pages they visit, the searches they make, location of the user browsing from, device used for browsing and many other inputs are used by the tracking sites to collect data.

Cookies:

Cookies are created when a user's browser loads a particular website. The website sends information to the browser which then creates a text file. Every time the user goes back to the same website, the browser retrieves and sends this file to the website's web server. Computer Cookies are created not just by the website that the user is browsing but also by other websites that run ads, widgets, or other elements on the web page which are being loaded. These cookies regulate the ad display and functioning of other elements on the web page.

Third party `cookies` are `cookies` that are set by web site with a domain name other than the one the user is currently visiting. For example, user visits website abc.com, say the web page abc.com has an image to fetch from xyz.com. That image request can set `cookie` on domain xyz.com, and the `cookie` set on xyz.com domain is known as a `third-party cookie`. Some advertisers use these types of `cookies` to track your visits to the various websites on which they advertise.

Lab 4:

Netwox **Tools.** We need tools to send out network packets of different types and with different contents.

ARP cache poisoning
The ARP cache is an important part of the ARP protocol. Once a mapping between a MAC address and an IP address is resolved as the result of executing the ARP protocol, the mapping will be cached. Therefore, there is no need to repeat the ARP protocol if the mapping is already in the cache. However, because the ARP protocol is stateless, the cache can be easily poisoned by maliciously crafted ARP messages. Such an attack is called the ARP cache poisoning attack.
In such an attack, attackers use spoofed ARP messages to trick the victim to accept an invalid MAC-to-IP mapping, and store the mapping in its cache. There can be various types of consequences depending on the motives of the attackers. For example, attackers can launch a DoS attack against a victim by associating a nonexistent **MAC address to the IP address of the victim's default gateway; attackers can also redirect the traffic to and from the victim to another machine, etc**.

SYN Flooding Attack:

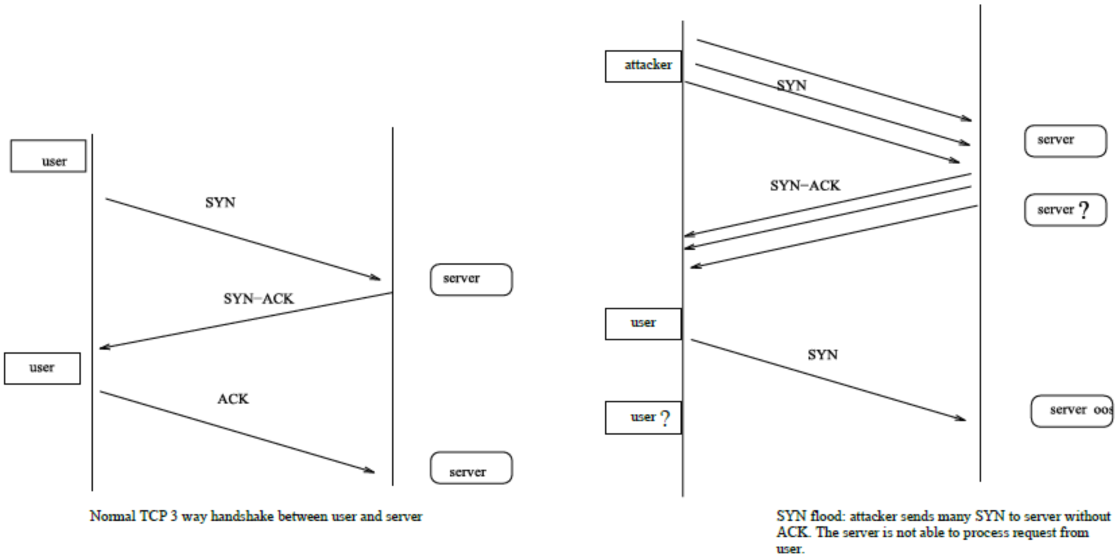

user
SYN
server
SYN–ACK
user
ACK
server
Normal TCP 3 way handshake between user and server
attacker
SYN
server
SYN–ACK
server ?
user
SYN
user ?
server oo
SYN flood: attacker sends many SYN to server without ACK. The server is not able to process request from user.

Fig 4.1: SYN Flood

SYN flood is a form of DoS attack in which attackers send many SYN requests to a victim's TCP port, but the attackers have no intention to finish the 3-way handshake procedure. Attackers either use spoofed IP address or do not continue the procedure. Through this attack, attackers can flood the victim's queue that is used for half-opened connections, i.e. the connections that has finished SYN, SYN-ACK, but has not yet got a final ACK back. When this queue is full, the victim cannot take any more connection. Figure6.1illustrates the attack.

Lab Session 06

In-band SQLi

The attacker uses the same channel of communication to launch their attacks and to gather their results. In-band SQLi's simplicity and efficiency make it one of the most common types of SQLi attack. There are two sub-variations of this method:

- **Error-based SQLi**—the attacker performs actions that cause the database to produce error messages. The attacker can potentially use the data provided by these error messages to gather information about the structure of the database.

- **Union-based SQLi**—this technique takes advantage of the UNION SQL operator, which fuses multiple select statements generated by the database to get a single HTTP response. This response may contain data that can be leveraged by the attacker.

Inferential (Blind) SQLi

The attacker sends data payloads to the server and observes the response and behavior of the server to learn more about its structure. This method is called blind SQLi because the data is not transferred from the website database to the attacker, thus the attacker cannot see information about the attack in-band.

Blind SQL injections rely on the response and behavioral patterns of the server so they are typically slower to execute but may be just as harmful. Blind SQL injections can be classified as follows:

- **Boolean**—that attacker sends a SQL query to the database prompting the application to return a result. The result will vary depending on whether the query is true or false. Based on the result, the information within the HTTP response will modify or stay unchanged. The attacker can then work out if the message generated a true or false result.

- **Time-based**—attacker sends a SQL query to the database, which makes the database wait (for a period in seconds) before it can react. The attacker can see from the time the database takes to respond, whether a query is true or false. Based on the result, an HTTP response will be generated instantly or after a waiting period. The attacker can thus work out if the message they used returned true or false, without relying on data from the database.

Out-of-band SQLi

The attacker can only carry out this form of attack when certain features are enabled on the database server used by the web application. This form of attack is primarily used as an alternative to the in-band and inferential SQLi techniques.

Out-of-band SQLi is performed when the attacker can’t use the same channel to launch the attack and gather information, or when a server is too slow or unstable for these actions to be performed. These techniques count on the capacity of the server to create DNS or HTTP requests to transfer data to an attacker.