# CSE406
# Computer Security sessional

## Lab Group B1
## Group 2

| Student ID | Project Name |
|---|---|
| 1605062 | ARP cache poisoning + Man-in-the-middle attack |
| 1605069 | DoS attack to the DNS server (using spoofed IP address) |
| 1605072 | DHCP starvation |
| 1605078 | TCP reset attack on video streaming |

# CSE 406

# Computer Security Sessional

## Report on Project No. 1

## ARP Cache Poisoning with Man in the Middle Attack

Submitted by-

Raihan Rasheed

Student ID. 1605062

Lab Group. B1

Project Group No. 2

Level 4 Term 1

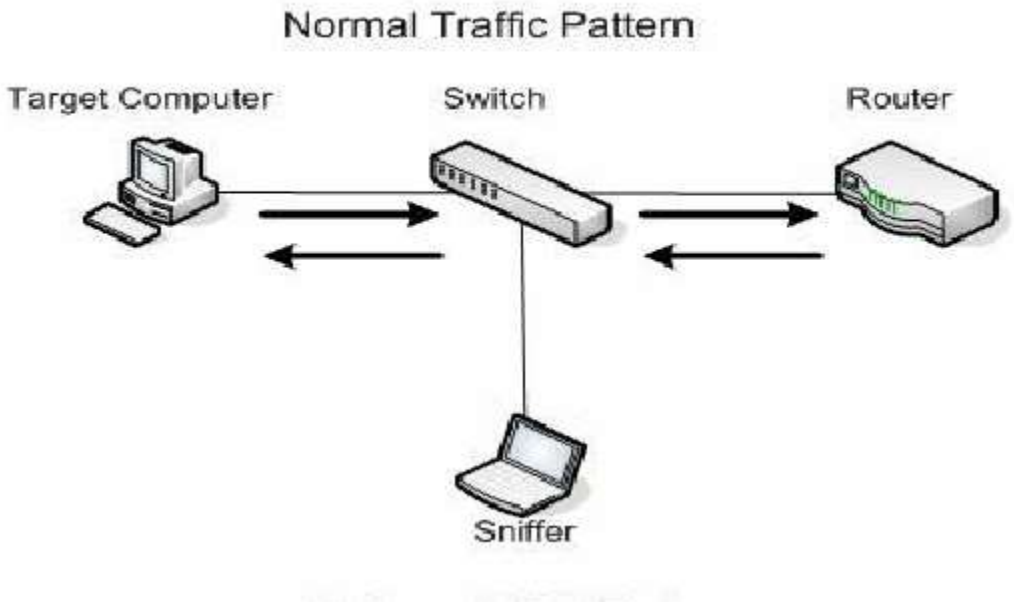
Department of CSE, BUET

# Definition of the attack

ARP spoofing, ARP cache poisoing, or ARP poison routing, is a technique by which an attacker sends (spoofed) Adress Resolution Protocol (ARP) messages onto a local area network. Generally, the aim is to associate the attacker's MAC address with the IP address of another host. This host can be host computer on this network or it can be the default gateway to other networks.

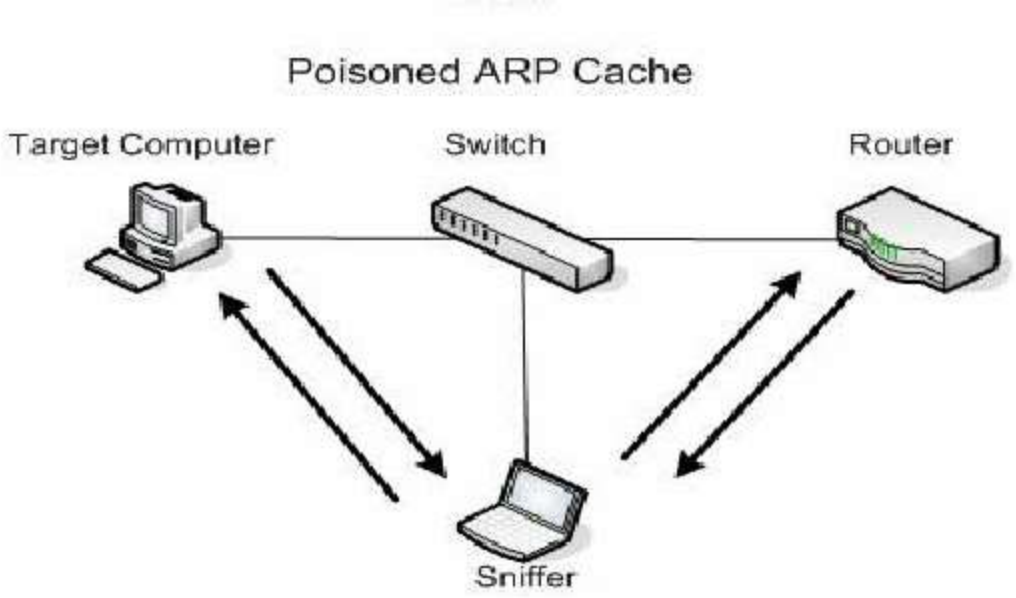
If attacker is successful, then any network traffic meant for that IP address would be sent to the attacker instead of the machine that uses that IP address. This allows attacker to intercept data frames on a network and then attacker can perform all sort of things such as modify the traffic, or stop all traffic. This attack is often used as an opening for other attacks, such as man in the middle or session hijacking attacks. The attack can only be used on networks that use ARP, and requires attacker have direct access to the local network segment to attacked.

# Topology Diagram




As previously stated, this attack requires attacker have direct access to the local network. In the figure above we can see sniffer (attacker) is connected with the same switch where target computer performs its normal communication with a router. This router can be the default gateway or gateway to a particular network.

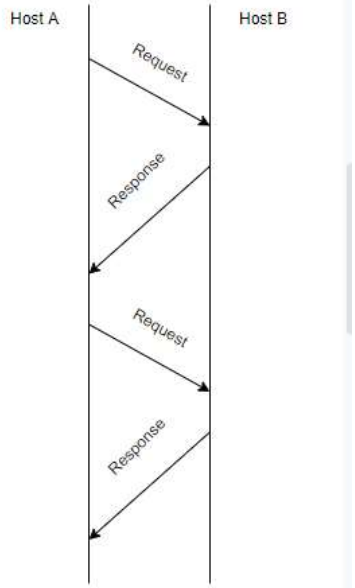
Before sniffer starts attacking the expected behaviour of this network is that target computer communicates back and forth with router via a network switch where target computer, router and attacker are connected. Then after successful attack the communication flow becomes like below -



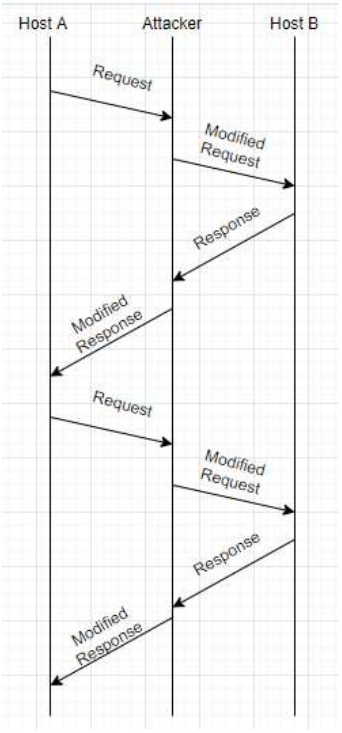

After the arp cache of both target computer and router is poisoned all the communication that is happening between target computer and router goes between the sniffer. Hence the sniffer computer gets opportunity to read data between target computer and router. Even the sniffer can modify the communication as intended or can completely hijack the session if desires.

# Timing Diagram and Attack Strategy

In the figure drawn in the right-hand side we can see the normal network flow where Host A sends requests and gets back response from Host B. Both Host A or Host B can be gateway server or just a normal machine who wants to communicate with the other one.




When the attacker poisons the cache of both Host A and Host B it can now seat between both of these hosts to sniff what they are sending each other and if wishes the attacker can modify those packets as well. In order to create this situation. We have to poison both ARP caches situated in Host A and Host B. In Host A's ARP cache we are going to place Host B's IP address with Attacker's MAC address and inside Host B's ARP cache we will map Host A's IP address with Attacker's MAC address so that both of them send their message to attacker when they actually want to send message between themselves. So how we will modify ARP message and poison cache of Host A and Host B will be discussed on next sections.



# Frame Details

ARP protocol works between various link layer and network layer protocol. Hence the size of an ARP message varies depending on the size of link layer and network layer address sizes. So, we are going to represent the structure of ARP message assuming it works with ipv4 network layer and ethernet data link layer. Here is a good time to mention the fact that, ARP is a data link layer protocol. So, ARP message structure shown below will go as a payload of Ethernet frame like IP packet. But it is does not work across networks hence its not network layer protocol.

**Internet Protocol (IPv4) over Ethernet ARP packet**

| Octet offset | 0 | 1 |
|---|---|---|
| 0 | Hardware type (HTYPE) | |
| 2 | Protocol type (PTYPE) | |
| 4 | Hardware address length (HLEN) | Protocol address length (PLEN) |
| 6 | Operation (OPER) | |
| 8 | Sender hardware address (SHA) (first 2 bytes) | |
| 10 | (next 2 bytes) | |
| 12 | (last 2 bytes) | |
| 14 | Sender protocol address (SPA) (first 2 bytes) | |
| 16 | (last 2 bytes) | |
| 18 | Target hardware address (THA) (first 2 bytes) | |
| 20 | (next 2 bytes) | |
| 22 | (last 2 bytes) | |
| 24 | Target protocol address (TPA) (first 2 bytes) | |
| 26 | (last 2 bytes) | |

ARP message shown above consists of total 28 bytes indexing from 0 to 27. We are assuming network layer address is IPv4 hence 32 bits or 4 bytes. Similarly, we are assuming link layer mac address is 48 bits or 6 bytes.

First two information stored are HTYPE and PTYPE indicate link layer protocol number and network layer protocol number respectively. In our case HTYPE would have 1 as its value denoting Ethernet link layer protocol and PTYPE value would be Ox0800 denoting IPv4 protocol at network layer.

Next two information HLEN and PLEN denotes address length in bytes for link layer and network layer respectively. For us that would be 6 and 4.

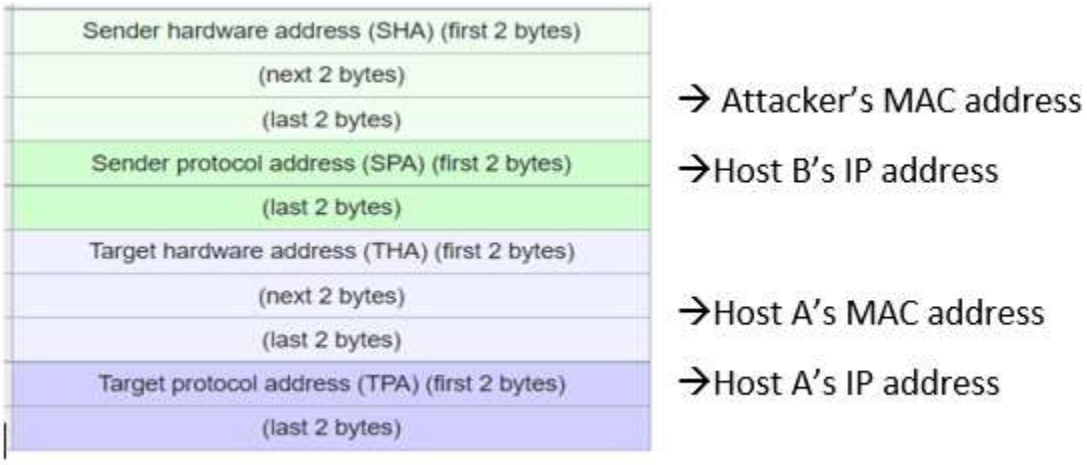
Operation field stores the type of ARP message being sent. 1 for request and 2 for reply.

The next 4 fields are important to us. Here we are going to make our modification actually. First two field is Sender Hardware Address (SHA) and Sender Protocol Address (SPA) that holds sender's MAC address and IP address respectively. Final two fields are concerned with receiver's information. Receiver's MAC address is placed on Target Hardware Address. In ARP request this field is ignored as it is not known at that time. Receiver's IP address is placed on Target Protocol Address.

# Modification of ARP Message

In last section we studied different fields of ARP message. We are going to modify a ARP message in a way that it helps to store wrong information in victim's ARP cache.

Below is our proposed modification-




Here instead of sending attacker's IP address to Host A we are going to send Host B's IP address with attacker's MAC address.

We are going to send similar ARP message to Host B as well where we are going to put attacker's MAC address and Host A's IP address in sender information fields.

If we continue this process after a while and send this spoofed ARP message Host A and Host B will have poisoned cache throughout all of their communication time.

# Justification

In last section we saw how are we going to modify ARP message fields so that Host A and Host B get wrong information about the MAC address where they want to send their frame. As we are going to keep sending ARP response with miss-information time after time Host A and Host B will never get correct IP address to MAC address mapping in their ARP cache. Hence they will continue to send their requests and responses to attacker and our attack will be successful. We hope that based on theories discussed in this report we will be able to demonstrate this attack in due time.

# CSE 406

# Computer Security

# Project Design Report

**DoS attack to the DNS server (using spoofed IP address)**

Name : Abdur Rahman Fahad
Std ID : 1605069
Group : 02
Lab Group : B1

Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology

## Definition of the attack

DoS attack (Denial-of-Service attack) is a cyber-attack in which the perpetrator seeks to make a machine or network resource unavailable to its intended users by temporarily or indefinitely disrupting services of a host connected to the Internet. Denial of service is typically accomplished by flooding the targeted machine or resource with superfluous requests in an attempt to overload systems and prevent some or all legitimate requests from being fulfilled.

Here we will attack the DNS (Domain Name Server) using spoofed IP address. We will perform the attack by sending lots of meaningless DNS query to the DNS server.

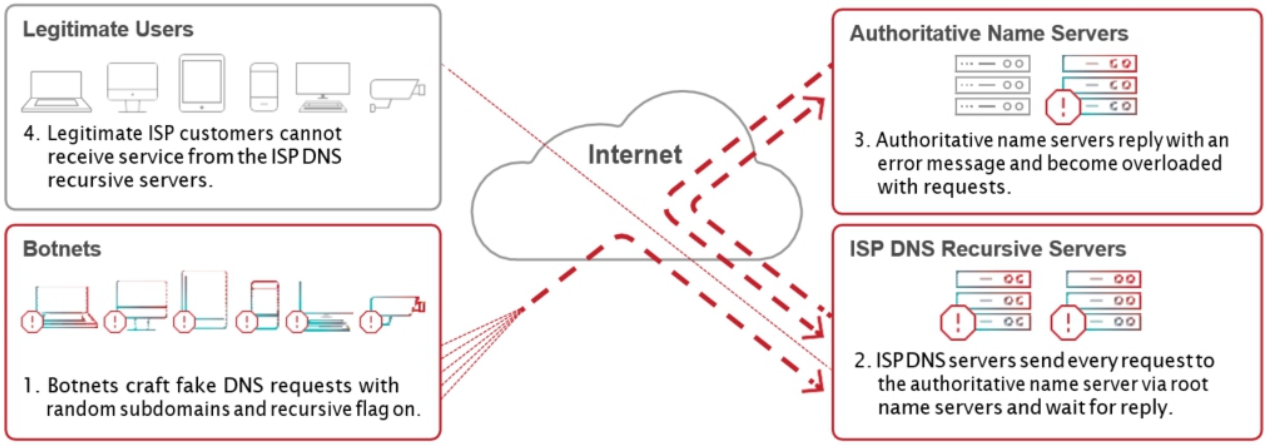



Figure : Topology Diagram of the attack

## Timing Diagrams

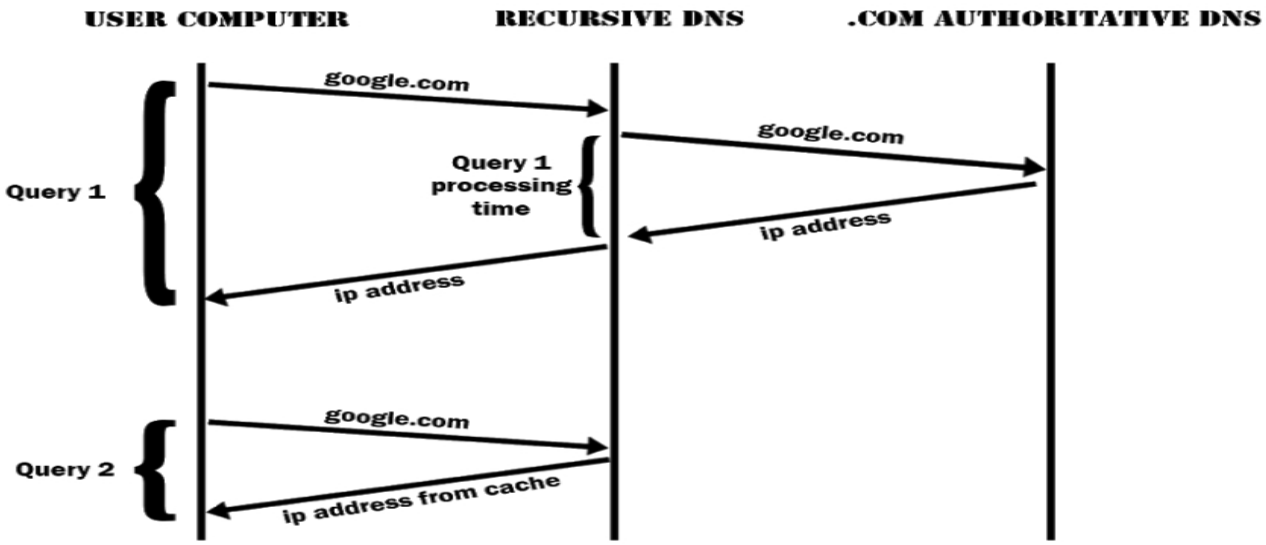



Figure : Timing Diagram of a DNS Query

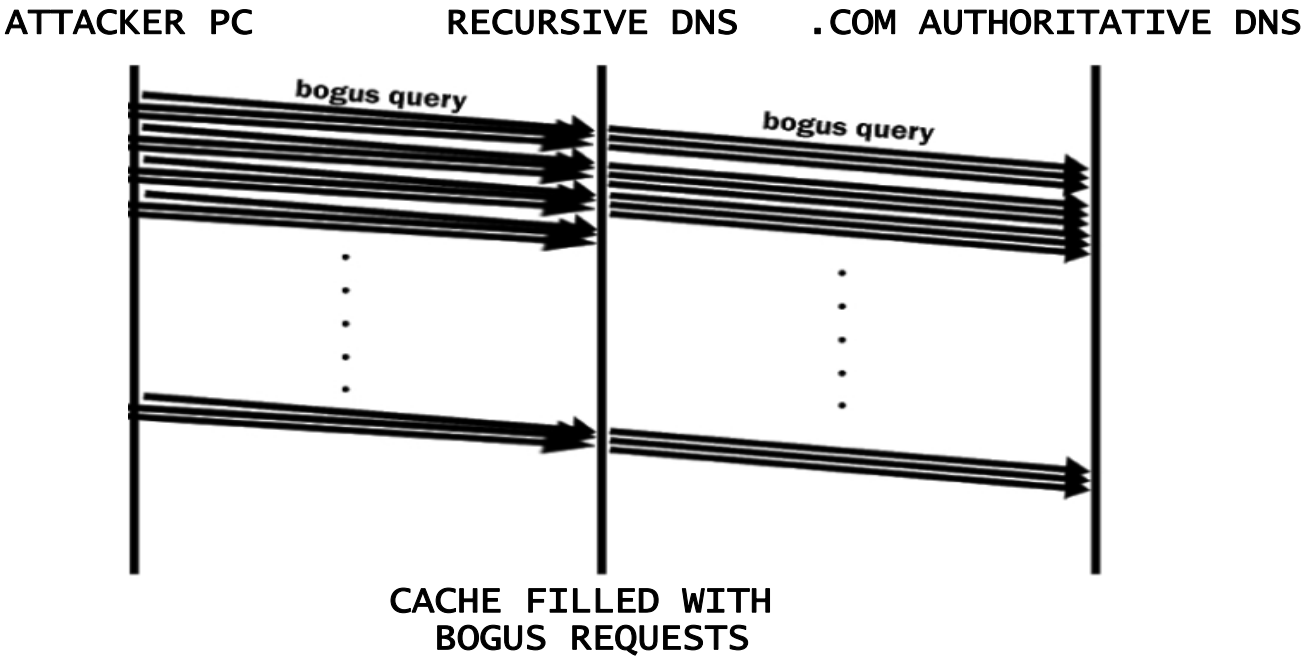



Figure : Timing Diagram of a DOS Attack on DNS Server

## Attack Strategy

- We'll be using a local DNS server, which we'll attack. We'll configure a custom DNS server with a tool called Bind9.
- We'll flood the server with bogus DNS queries with unlimited UDP requests through our script.
- We'll change our IP continuously with the help of our script.
- We'll make our own packets for the DNS queries in which we'll set IP header as we wish, thus attack with spoofed IP will be possible.
- Thus the DNS server will run out of resources and any legit user will not be able to use DNS server.

## Packet details and IP header modification

A standard DNS query packet looks like this,

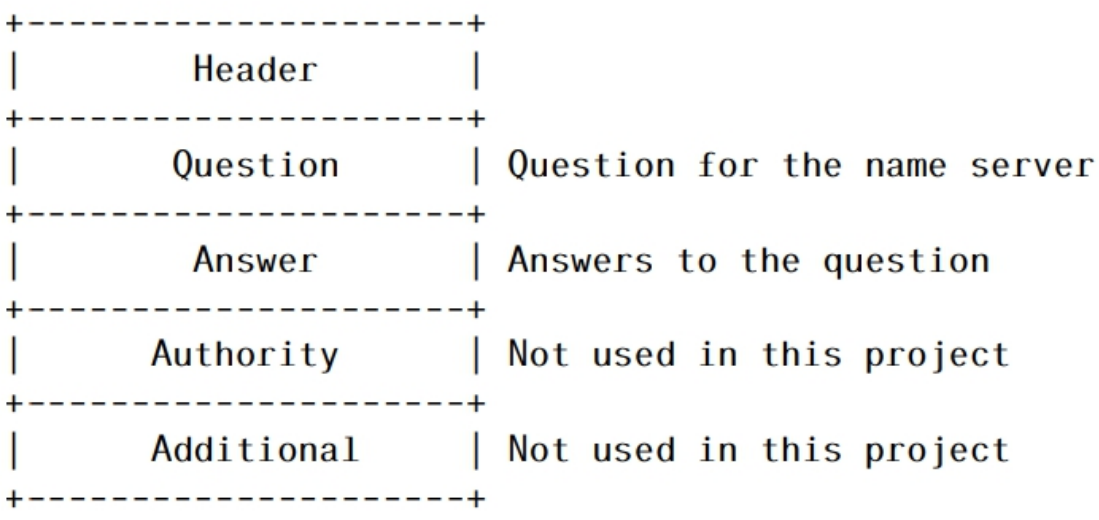



Figure : DNS Packet Structure

DNS packets have a header that is shown below.

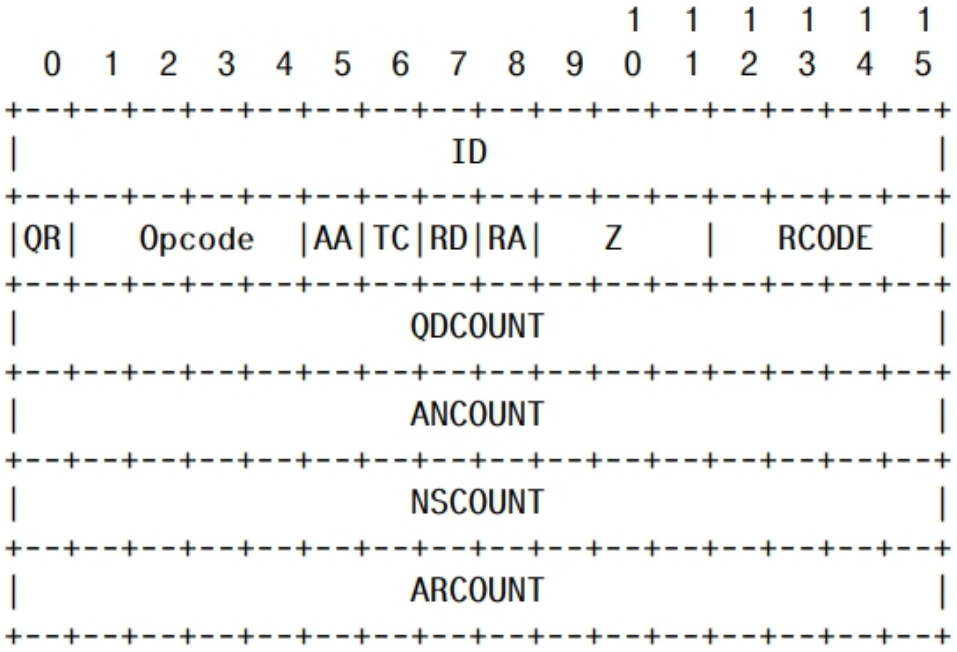



Figure : DNS Packet Header Structure

A DNS question has the format

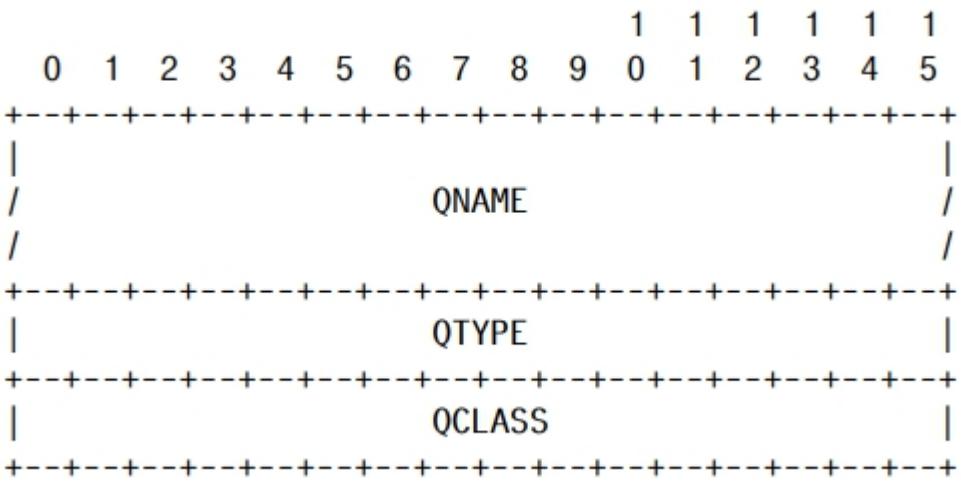



Figure : DNS Question Structure

IP Header has the following format

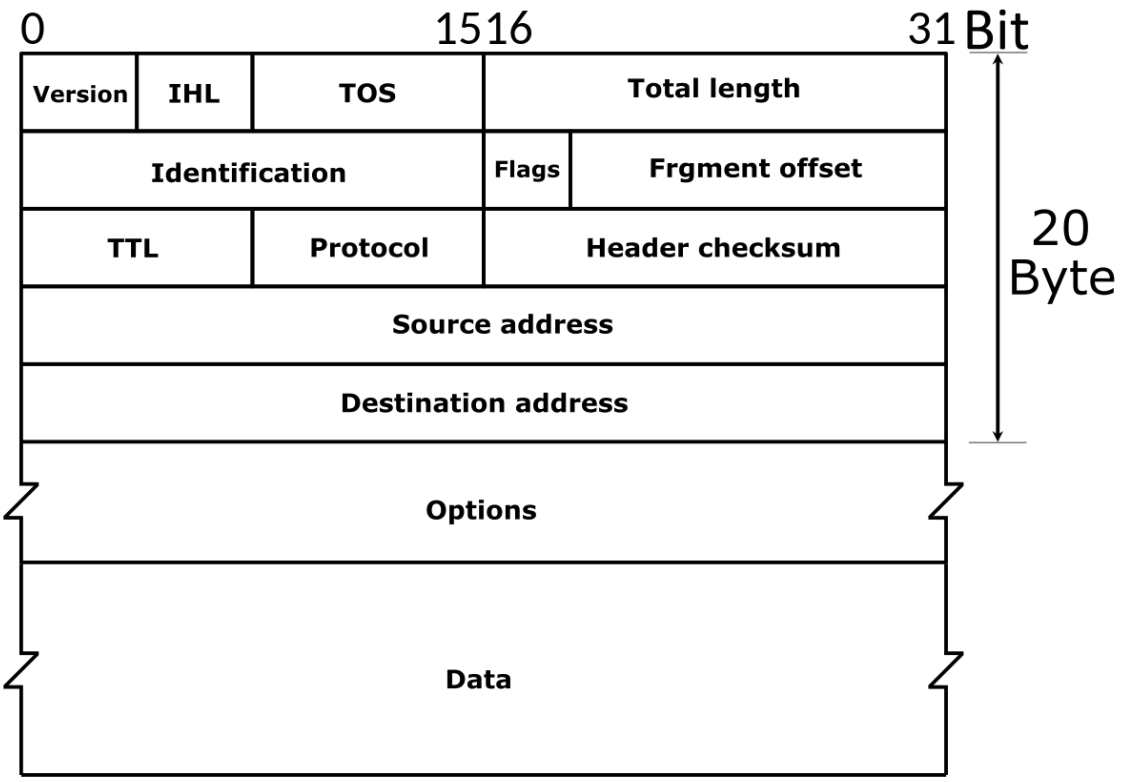



Figure : IP Header Structure

We'll modify the Source Address in the IP header to implement spoofed IP address.

## Justification

We will send bogus queries to DNS Server with a spoofed IP address in the source IP address field of IP Header. It'll fail to find a valid entry in cache and so, the DNS server will send the query to authoritative DNS Servers and wait for the result, which will also eventually be failed.

By doing this with many infinite loop and many more requests than usual, eventually the cache of DNS server will be filled with bad requests. This way, it is possible to flood the targeted DNS and the server will deny any further service from any legit user and our attack will be successful.

# DHCP Starvation Attack

Name: Binoy Kumar Sutradhar
Roll: 1605072

# DHCP Starvation Attack

## Introduction:

A DHCP starvation attack is a malicious digital attack that targets DHCP servers. During a DHCP attack, a hostile actor floods a DHCP server with bogus DISCOVER packets until the DHCP server exhausts its supply of IP addresses. Clients looking for IP addresses find that there are no IP addresses for them, and they're denied service.

## Basic DHCP Process:

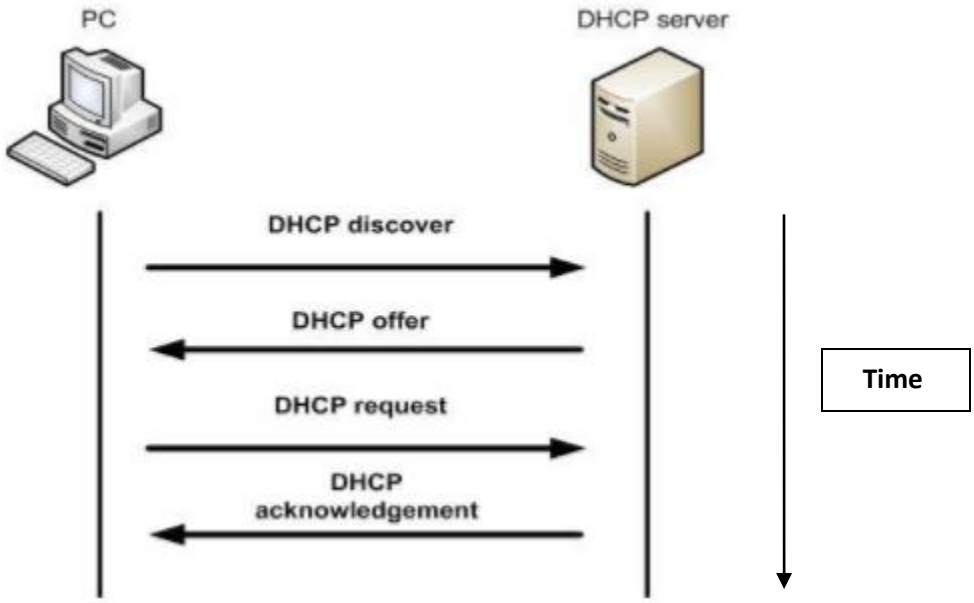
A Dynamic Host Configuration Protocol server is responsible for issuing IP addresses to devices on its network. This is done through a series of packet exchanges between individual DHCP clients and DHCP servers.

A DHCP IP address allocation transaction depends on four types of packets: DISCOVER, OFFER, REQUEST, and ACKNOWLEDGEMENT. These stages are often abbreviated as DORA for discovery, offer, request and acknowledgement.

1. When a PC boots up on the network, if it's a DHCP client, it's going to broadcast a DHCP DISCOVER packet asking for an IP address.
2. DHCP server responds to the DISCOVER message by sending an OFFER packet. And in that offer, it's going to offer an IP address that the client is allowed to use.
3. After the client receives OFFER packet, client sends a REQUEST packet requesting for exclusives rights to the offered IP address.
4. When the DHCP server receives the DHCP REQUEST message from the client, server sends an ACKNOWLEDGEMENT packet to the client. This packet includes the lease duration and any other configuration information that the client might have requested. At this point, the IP configuration process is completed.

Now, the client can use the assigned IP address.

## Basic DHCP Process Diagram:




## DHCP Starvation Attack Strategies:

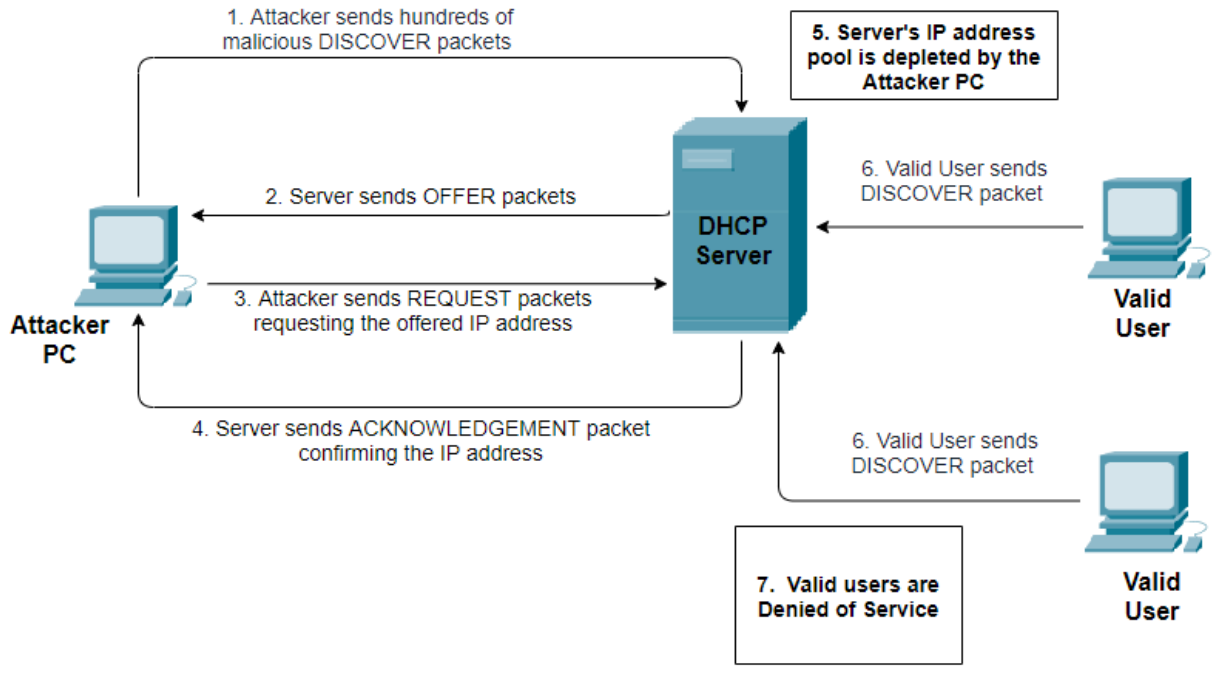
1. At first, I will open a socket.
2. Then I need to perform the following steps for the attack to take place.
    a. I will generate a fake MAC address.
    b. Using that fake MAC address as the Client Hardware Address, a DISCOVER packet is broadcasted on the network.
    c. Receiving the DISCOVER packet, the DHCP server will send an OFFER packet. I will define a fixed time interval. DHCP OFFER packet must be received at the client end within that time.
    d. In response to that OFFER packet, that fake client will send a REQUEST packet.
    e. Then the server will send an ACKNOWLEDGEMENT packet issuing an IP address for that fake client.

3. On a /24-bit network, the max number of IP addresses that could be in a DHCP pool of addresses would be 254. Additionally, it's very likely that a few of those addresses are saved for static router addresses and so forth. So the pool of available addresses the DHCP server can draw from may only be about 252 IP addresses.

   So, if we perform the 2nd point at least 252 times, the DHCP server will run out of IP addresses and will fail to provide for any valid client that asks for IP addresses in future.

4. I will keep performing the 2nd Point until any DHCP OFFER packet won't be received at the client end within a specified time interval. It will mean that the DHCP server has run out of IP addresses.

## DHCP Starvation Attack Diagram:



## DHCP Packet Details:

| Field Name | Size (Bytes) | Description |
|---|---|---|
| Operation Code | 1 | Set to 1 in messages sent by a client (requests) and 2 in messages sent by a server (response). |
| Hardware Type | 1 | Set to 1 for Ethernet |
| Hardware Address Length | 1 | Defines the length of MAC address in the chaddr field. Set to 6 for Ethernet. |
| Hops | 1 | Set to 0 by a client before transmitting a request |
| Transaction Identifier | 4 | Generated by the client to match up the request with replies received from DHCP servers. |
| Seconds | 2 | Elapsed time (in seconds) since the client began the DHCP process. |
| Flags | 2 | Flags field is called the Broadcast bit. It can be set to 1 for Broadcasting. |
| CIADDR | 4 | Client's IP Address; set by the client when the client has confirmed that it's IP address is valid. |
| YIADDR | 4 | Client's IP Address; set by the server to inform the client of the client's IP address. |
| SIADDR | 4 | IP address of the next server for the client to use in the configuration process (for example, the server to contact for TFTP download of an operating system kernel). |
| GIADDR | 4 | Relay agent (gateway) IP address; filled in by the relay agent with the address of the interface through which Dynamic Host Configuration Protocol (DHCP) message was received. |
| CHADDR | 16 | The hardware (layer two) address of the client. The fake MAC address is set here. |
| Magic Cookie | 4 | 0x63825363 |

In DISCOVER packet, CIADDR, YIADDR, SIADDR, GIADDR are set to 0 by attacker PC.

In OFFER Packet, YIADDR & SIADDR are set by server.

## Ethernet Header:

Source MAC Address = Fake MAC Address
Destination MAC Address = FF:FF:FF:FF:FF:FF

## IP Header:

Source IP Address = 0.0.0.0
Destination IP Address = 255.255.255.255

## UDP Header:

Source Port = 68
Destination Port = 67

## Justification:

I will create hundreds and hundreds of fake MAC addresses. Using those fake MAC addresses as source addresses, I will create malicious DHCP DISCOVER packets. When I will send these packets to server, DHCP server will response to these malicious packets and assign each of the fake MAC addresses with a valid IP addresses. Thus the entire IP address pool of the DHCP server could be depleted. Now, if any valid user pc asks for an IP address from the server, server will fail to provide. Thus the valid users would be denied of services.

So, my attack would fulfill its target.

# CSE 406

# TCP Reset Attack On Video Streaming Application

Submitted by-
1605078
Md.Mohib Hossain

# 1. TCP Reset Attack

TCP reset attack also known as “forged TCP resets” or “spoofed TCP reset packets” is a way to tamper and terminate an established internet connection between a server and a client by sending a forged TCP reset packet.

A TCP connection can be terminated in two ways

i. A **FIN** Packet
ii. A **RST** Packet

A **FIN** Packet contains a packet with the **FIN** bit set in TCP header and used in normal condition for terminating the connection.

A **RST** packet contains a packet with the **RST** bit set in TCP header and used to terminate the connection immediately.

**RST** packets are necessary for a firewall to use in goodwill, but they can also be abused by attackers to interrupt internet connections.

# 2. RST Timing Diagram

A RST packet is sent to terminate the established connection immediately.

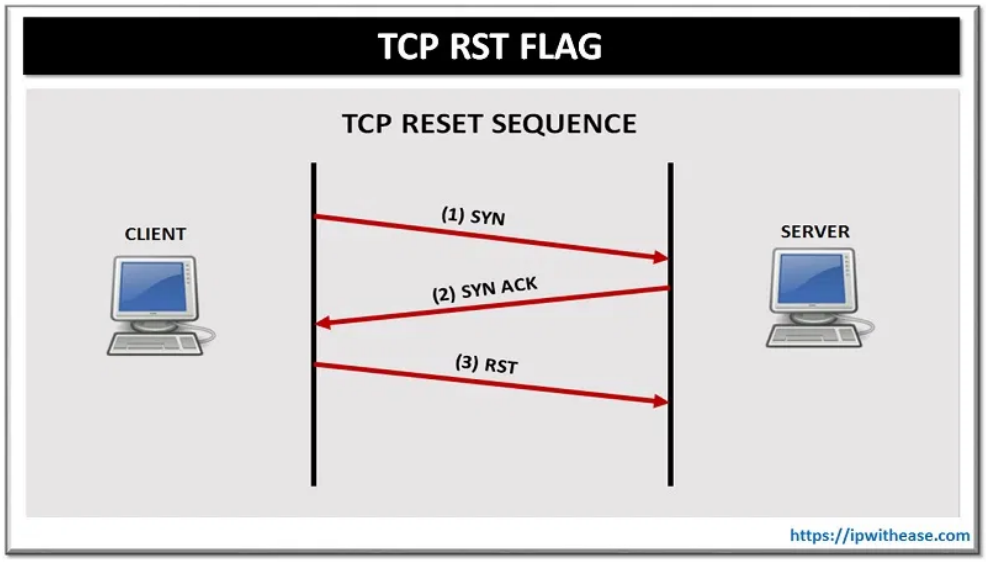



Fig: RST Timing diagram

# 3. Network Topology

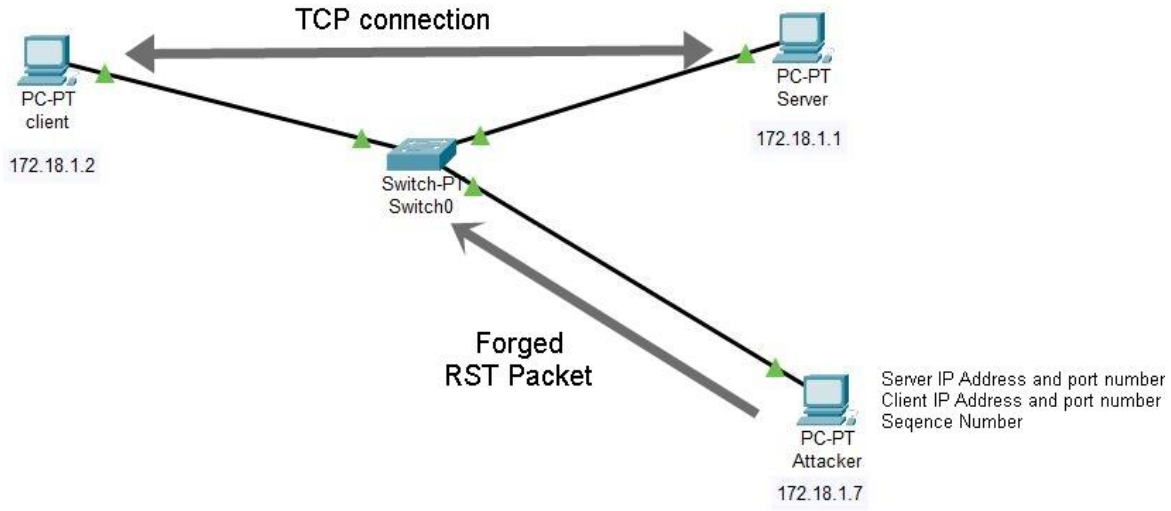



Fig: TCP reset attack

Here the server, client(victim) and the attacker are in the same LAN. Server will stream the video and the client will view the video using a video streaming application. The attacker will forge a **RST** packet and send it to the client, upon receiving the **RST** packet, the client will terminate the connection.

# 4. Attack Strategy

For a TCP reset attack, an attacker would need to know few information

i. Source (Victim) IP Address
ii. Destination (Server) IP Address
iii. Source Port Number
iv. Destination Port Number
v. Sequence Number in TCP header

The Attack Procedure is given bellow:

i. **ARP Spoofing:** ARP Spoofing to update ARP tables of both the server and client so that all traffic between the server and the client (victim) are passed through the attacker.

ii. **Packet Sniffing:** After ARP Spoofing, the attacker would be able to intercept the packets between the server and the client (victim). Sniffing the packets, the attacker would be able collect the above required information.

iii. **Packet Spoofing:** After collecting the required information, the attacker would be able to forge a **RST** TCP packet with the server's IP as the source IP Address and the victim's IP as the destination IP Address and setting the **RST** bit in TCP header. Then the attacker would spoof these forged TCP packets to the client. Upon receiving the **RST** packet, the client will terminate the TCP connection

# 5. Packet details

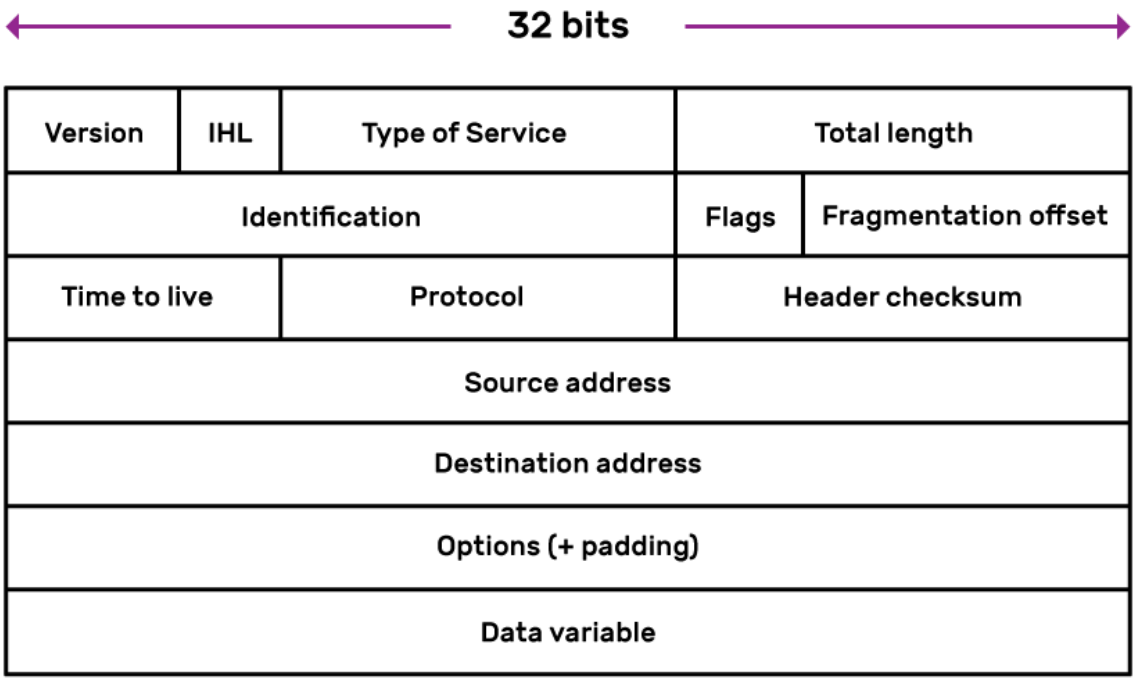



Fig: IP Header

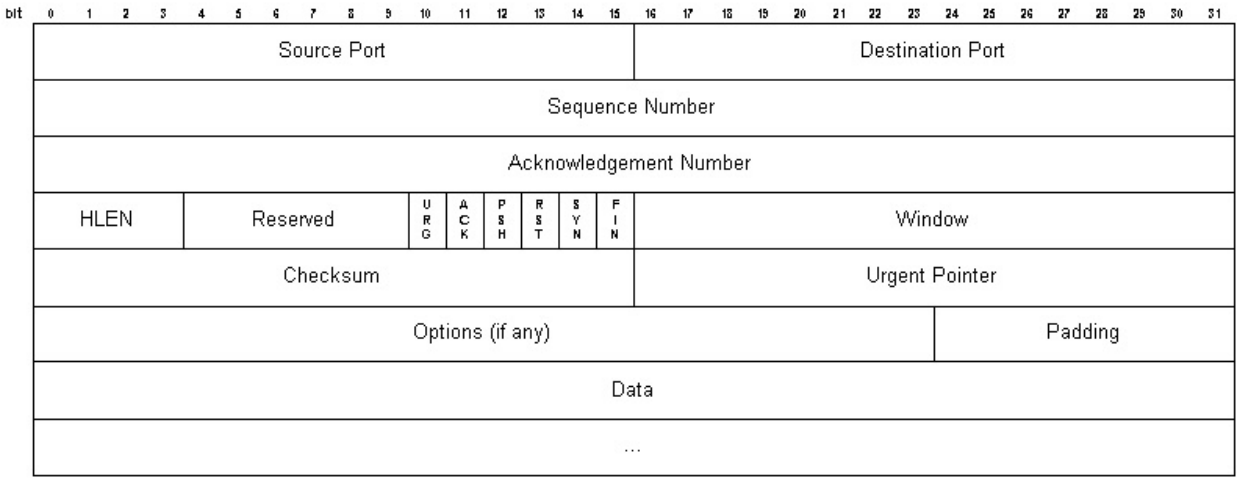



Fig: TCP Header

The attacker will sniff packets and retrieve IP address of source and victim from IP header and Source Port Number, Destination Port Number and Sequence Number from TCP header and then forge a packet following the above mentioned procedure.

# 6. Conclusion

As the attacker acts as a gateway for the victim, the victim has no way to differentiate between an actual **RST** packet and a forged **RST** packet. So it will terminate the connection regardless.

Hence the attack should be successful.