FINAL REPORT

DHCP Starvation Attack

Name: Binoy Kumar Sutradhar
Roll: 1605072
Lab Group: B1
Project Group: 2

DHCP Starvation Attack

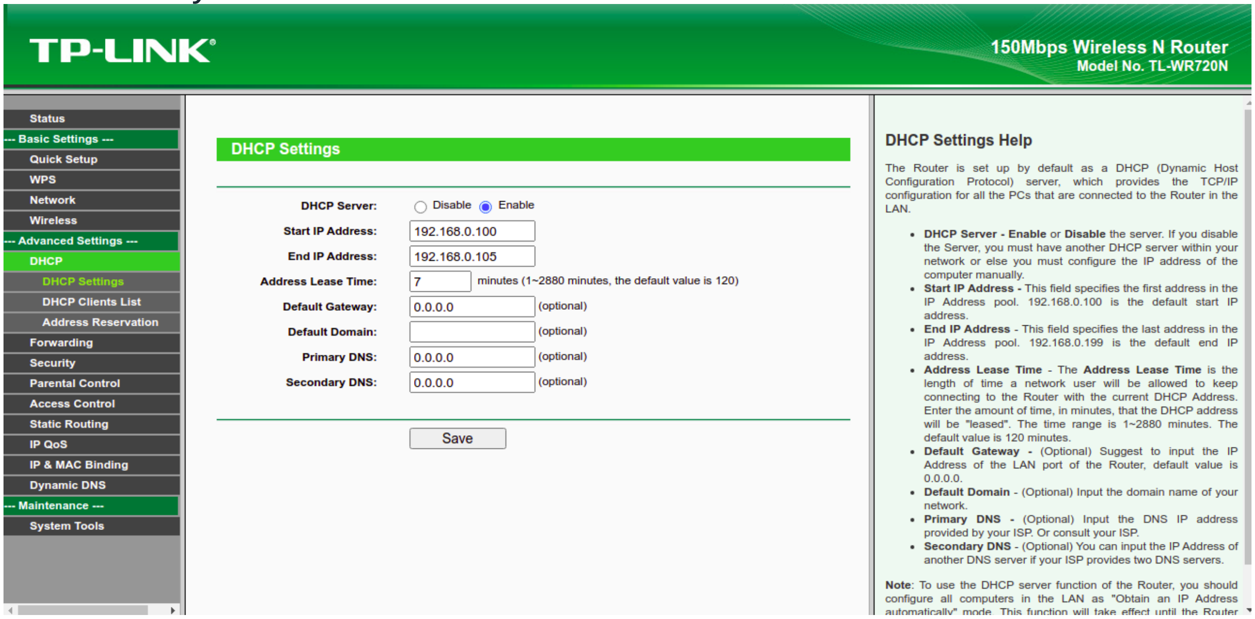
Introduction:

A DHCP starvation attack is a malicious digital attack that targets DHCP servers. During a DHCP attack, an attacker broadcasts large number of DHCP DISCOVER messages with spoofed source MAC addresses. If the legitimate DHCP Server in the network starts responding to all these bogus DHCP DISCOVER messages, available IP Addresses in the DHCP server scope will be depleted within a very short span of time. Clients looking for IP addresses find that there are no IP addresses for them, and they're denied service.

Steps of DHCP Starvation Attack:

I will be performing the attack on my TP-LINK router.

1. At first, I manipulated the DHCP settings on my router. I reduced size of the IP address pool of the DHCP server from 200 to 6 to understand the attack clearly.



DHCP Settings

DHCP Server: ○ Disable ● Enable
Start IP Address: 192.168.0.100
End IP Address: 192.168.0.105
Address Lease Time: 7 minutes (1~2880 minutes, the default value is 120)
Default Gateway: 0.0.0.0 (optional)
Default Domain: (optional)
Primary DNS: 0.0.0.0 (optional)
Secondary DNS: 0.0.0.0 (optional)

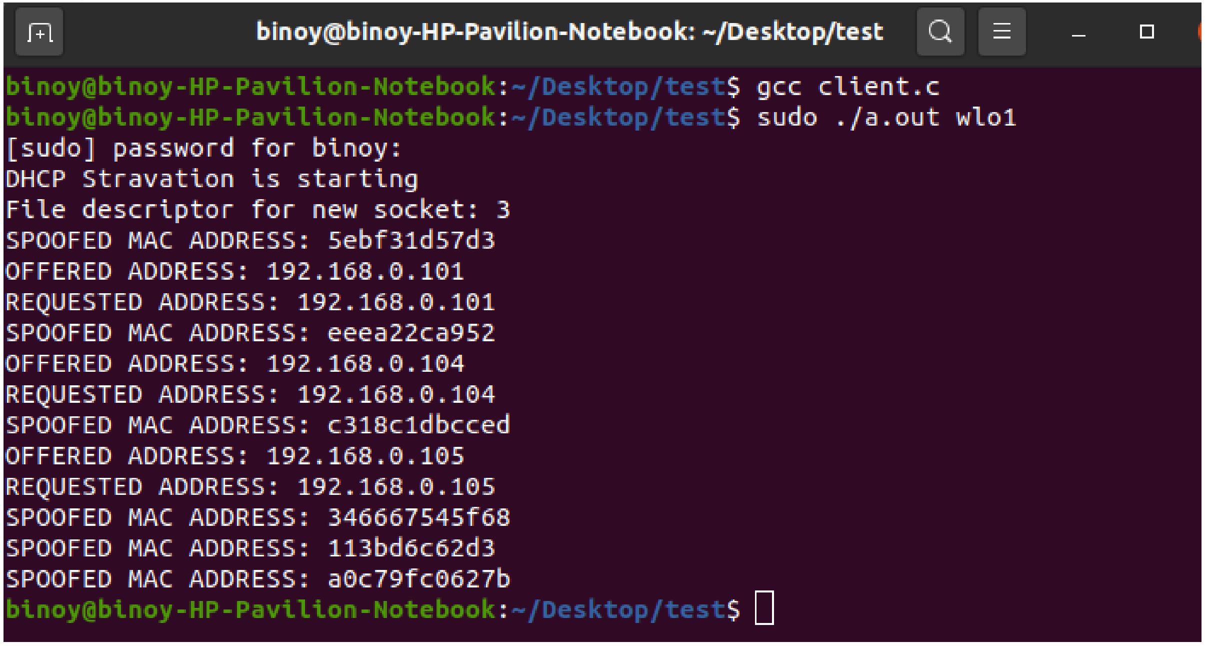
DHCP Settings Help

The Router is set up by default as a DHCP (Dynamic Host Configuration Protocol) server, which provides the TCP/IP configuration for all the PCs that are connected to the Router in the LAN.

- **DHCP Server - Enable** or **Disable** the server. If you disable the Server, you must have another DHCP server within your network or else you must configure the IP address of the computer manually.
- **Start IP Address -** This field specifies the first address in the IP Address pool. 192.168.0.100 is the default start IP address.
- **End IP Address** - This field specifies the last address in the IP Address pool. 192.168.0.199 is the default end IP address.
- **Address Lease Time** - The **Address Lease Time** is the length of time a network user will be allowed to keep connecting to the Router with the current DHCP Address. Enter the amount of time, in minutes, that the DHCP address will be "leased". The time range is 1~2880 minutes. The default value is 120 minutes.
- **Default Gateway -** (Optional) Suggest to input the IP Address of the LAN port of the Router, default value is 0.0.0.0.
- **Default Domain** - (Optional) Input the domain name of your network.
- **Primary DNS -** (Optional) Input the DNS IP address provided by your ISP. Or consult your ISP.
- **Secondary DNS** - (Optional) You can input the IP Address of another DNS server if your ISP provides two DNS servers.

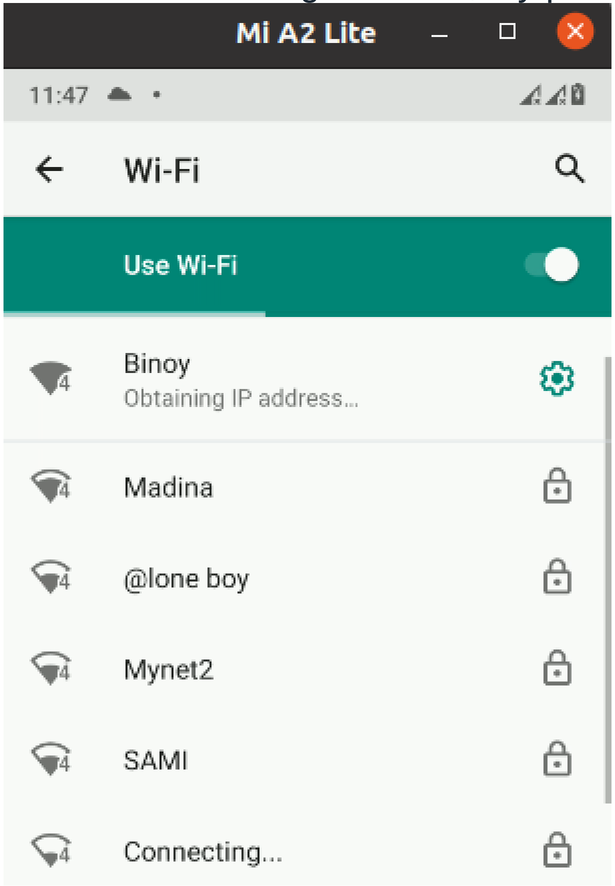
Note: To use the DHCP server function of the Router, you should configure all computers in the LAN as "Obtain an IP Address automatically" mode. This function will take effect until the Router

2. After manipulating the router, I run my code file from terminal and this was the output.
   Here, “wlo1” is the interface name.



```
binoy@binoy-HP-Pavilion-Notebook:~/Desktop/test$ gcc client.c
binoy@binoy-HP-Pavilion-Notebook:~/Desktop/test$ sudo ./a.out wlo1
[sudo] password for binoy:
DHCP Stravation is starting
File descriptor for new socket: 3
SPOOFED MAC ADDRESS: 5ebf31d57d3
OFFERED ADDRESS: 192.168.0.101
REQUESTED ADDRESS: 192.168.0.101
SPOOFED MAC ADDRESS: eeea22ca952
OFFERED ADDRESS: 192.168.0.104
REQUESTED ADDRESS: 192.168.0.104
SPOOFED MAC ADDRESS: c318c1dbcced
OFFERED ADDRESS: 192.168.0.105
REQUESTED ADDRESS: 192.168.0.105
SPOOFED MAC ADDRESS: 346667545f68
SPOOFED MAC ADDRESS: 113bd6c62d3
SPOOFED MAC ADDRESS: a0c79fc0627b
binoy@binoy-HP-Pavilion-Notebook:~/Desktop/test$
```

3. I disconnected my phone from the router before I manipulated the router. After running the code, my phone couldn’t connect to the router.



Binoy
Obtaining IP address...

Madina

@lone boy

Mynet2

SAMI

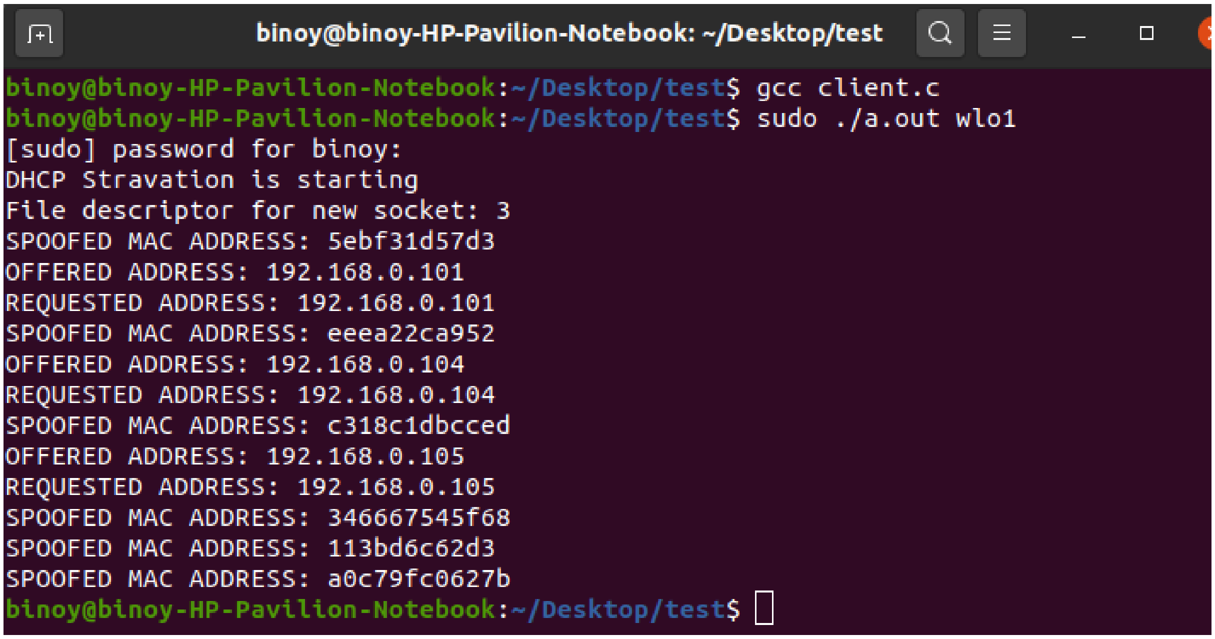
Connecting...

Implementation Details:

1. At first, I created a socket using using system call
   - socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP)
   - AF_INET for IPV4 protocol
   - SOCK_DGRAM for connectionless messages of a fixed maximum length whose reliability is not guaranteed.
   - IPPROTO_UDP as UDP is used by DHCP server in transport layer.

2. Then I performed the following steps for the attack to take place.
   I. I generated a fake MAC address.
   II. Using that fake MAC address as the Client Hardware Address, I made a DISCOVER packet and broadcasted on the network using sendto () system call.

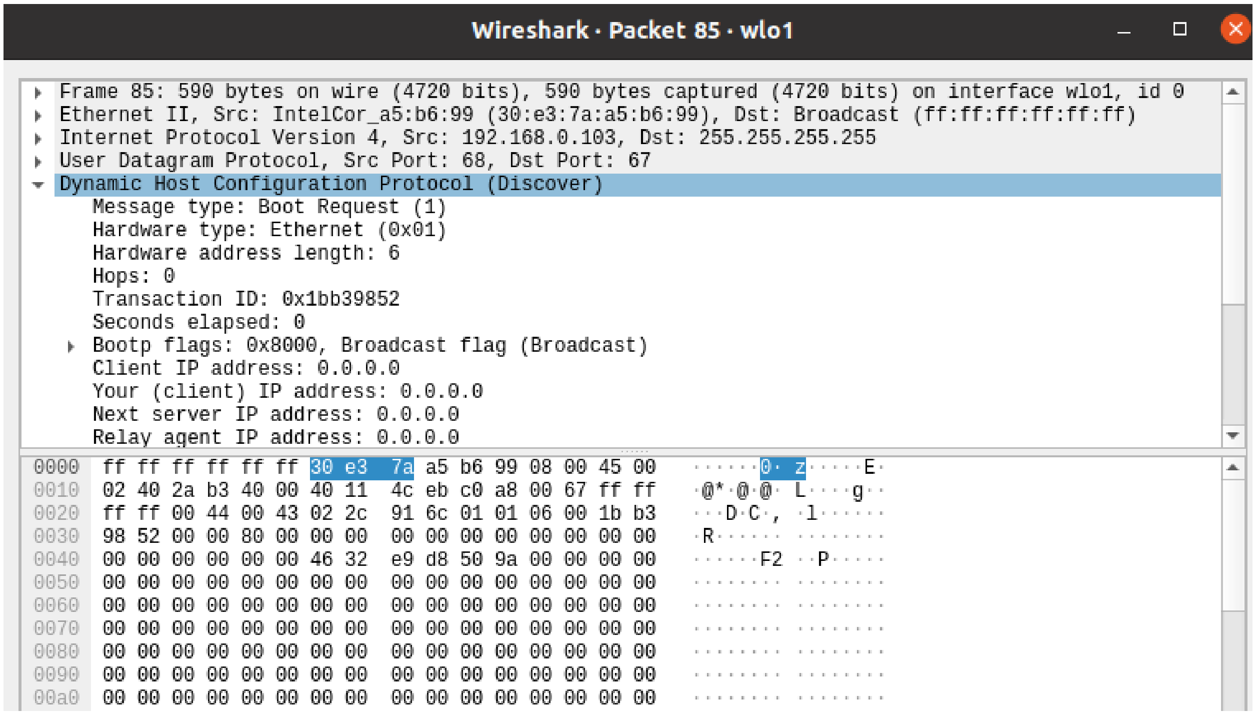
3. Step 2 is performed in the code until DHCP server runs out of IP addresses and will fail to provide IP address for any valid client in future. In my example, there were already 3 devices connected to the router including my laptop. As the size of address pool was reduced to 6, the router could only provide IP address for 3 more devices.
   So, when I performed the Step 2 six times, DHCP server ran out of IP addresses after providing 3 IP address. The other 3 DISCOVER packets didn't receive any response. That's what we could see in the terminal.



```
binoy@binoy-HP-Pavilion-Notebook:~/Desktop/test$ gcc client.c
binoy@binoy-HP-Pavilion-Notebook:~/Desktop/test$ sudo ./a.out wlo1
[sudo] password for binoy:
DHCP Stravation is starting
File descriptor for new socket: 3
SPOOFED MAC ADDRESS: 5ebf31d57d3
OFFERED ADDRESS: 192.168.0.101
REQUESTED ADDRESS: 192.168.0.101
SPOOFED MAC ADDRESS: eeea22ca952
OFFERED ADDRESS: 192.168.0.104
REQUESTED ADDRESS: 192.168.0.104
SPOOFED MAC ADDRESS: c318c1dbcced
OFFERED ADDRESS: 192.168.0.105
REQUESTED ADDRESS: 192.168.0.105
SPOOFED MAC ADDRESS: 346667545f68
SPOOFED MAC ADDRESS: 113bd6c62d3
SPOOFED MAC ADDRESS: a0c79fc0627b
binoy@binoy-HP-Pavilion-Notebook:~/Desktop/test$ 
```

Packets During Attack:

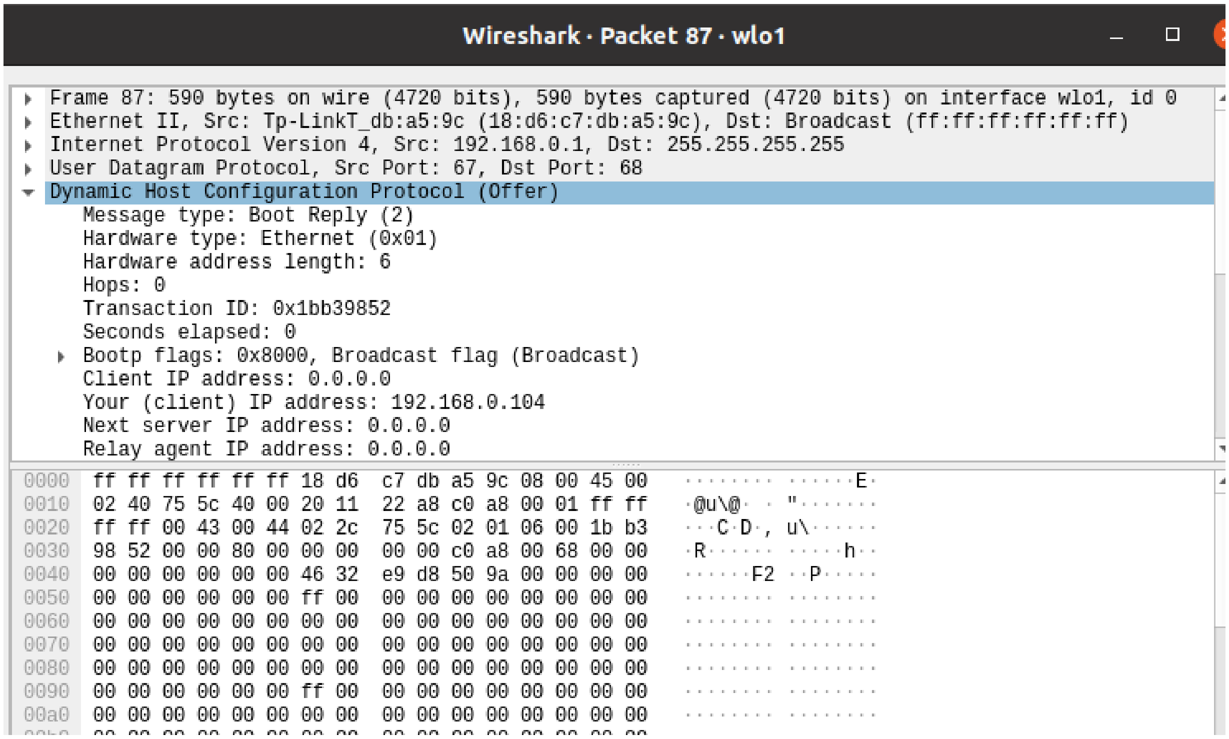
DISCOVER PACKET-



```
▸ Frame 85: 590 bytes on wire (4720 bits), 590 bytes captured (4720 bits) on interface wlo1, id 0
▸ Ethernet II, Src: IntelCor_a5:b6:99 (30:e3:7a:a5:b6:99), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
▸ Internet Protocol Version 4, Src: 192.168.0.103, Dst: 255.255.255.255
▸ User Datagram Protocol, Src Port: 68, Dst Port: 67
▾ Dynamic Host Configuration Protocol (Discover)
     Message type: Boot Request (1)
     Hardware type: Ethernet (0x01)
     Hardware address length: 6
     Hops: 0
     Transaction ID: 0x1bb39852
     Seconds elapsed: 0
   ▸ Bootp flags: 0x8000, Broadcast flag (Broadcast)
     Client IP address: 0.0.0.0
     Your (client) IP address: 0.0.0.0
     Next server IP address: 0.0.0.0
     Relay agent IP address: 0.0.0.0

0000  ff ff ff ff ff ff 30 e3  7a a5 b6 99 08 00 45 00   ······0· z·····E·
0010  02 40 2a b3 40 00 40 11  4c eb c0 a8 00 67 ff ff   ·@*·@·@· L····g··
0020  ff ff 00 44 00 43 02 2c  91 6c 01 01 06 00 1b b3   ···D·C·, ·l······
0030  98 52 00 00 80 00 00 00  00 00 00 00 00 00 00 00   ·R······ ········
0040  00 00 00 00 00 00 46 32  e9 d8 50 9a 00 00 00 00   ······F2 ··P·····
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
00a0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
```

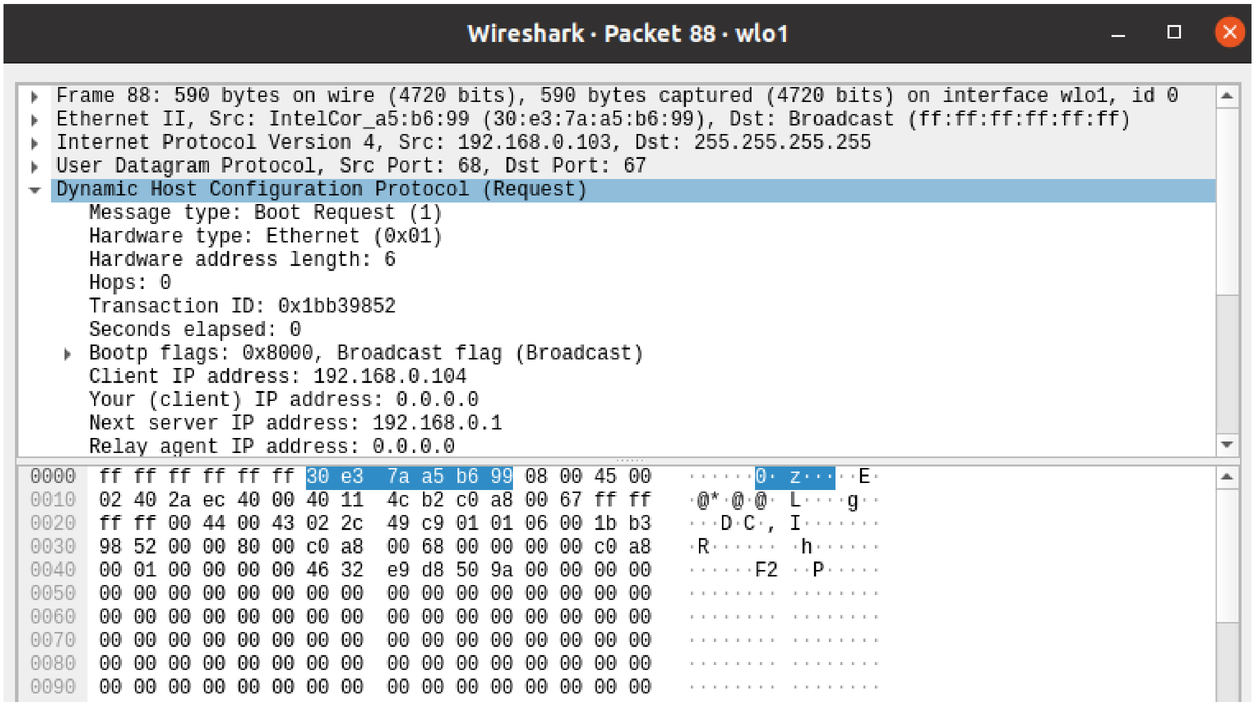
OFFER PACKET-



```
▸ Frame 87: 590 bytes on wire (4720 bits), 590 bytes captured (4720 bits) on interface wlo1, id 0
▸ Ethernet II, Src: Tp-LinkT_db:a5:9c (18:d6:c7:db:a5:9c), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
▸ Internet Protocol Version 4, Src: 192.168.0.1, Dst: 255.255.255.255
▸ User Datagram Protocol, Src Port: 67, Dst Port: 68
▾ Dynamic Host Configuration Protocol (Offer)
     Message type: Boot Reply (2)
     Hardware type: Ethernet (0x01)
     Hardware address length: 6
     Hops: 0
     Transaction ID: 0x1bb39852
     Seconds elapsed: 0
   ▸ Bootp flags: 0x8000, Broadcast flag (Broadcast)
     Client IP address: 0.0.0.0
     Your (client) IP address: 192.168.0.104
     Next server IP address: 0.0.0.0
     Relay agent IP address: 0.0.0.0

0000  ff ff ff ff ff ff 18 d6  c7 db a5 9c 08 00 45 00   ········ ······E·
0010  02 40 75 5c 40 00 20 11  22 a8 c0 a8 00 01 ff ff   ·@u\@· · "·······
0020  ff ff 00 43 00 44 02 2c  75 5c 02 01 06 00 1b b3   ···C·D·, u\······
0030  98 52 00 00 80 00 00 00  00 00 c0 a8 00 68 00 00   ·R······ ·····h··
0040  00 00 00 00 00 00 46 32  e9 d8 50 9a 00 00 00 00   ······F2 ··P·····
0050  00 00 00 00 00 00 ff 00  00 00 00 00 00 00 00 00   ········ ········
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0090  00 00 00 00 00 00 ff 00  00 00 00 00 00 00 00 00   ········ ········
00a0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
```

REQUEST PACKET-



```
▸ Frame 88: 590 bytes on wire (4720 bits), 590 bytes captured (4720 bits) on interface wlo1, id 0
▸ Ethernet II, Src: IntelCor_a5:b6:99 (30:e3:7a:a5:b6:99), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
▸ Internet Protocol Version 4, Src: 192.168.0.103, Dst: 255.255.255.255
▸ User Datagram Protocol, Src Port: 68, Dst Port: 67
▾ Dynamic Host Configuration Protocol (Request)
     Message type: Boot Request (1)
     Hardware type: Ethernet (0x01)
     Hardware address length: 6
     Hops: 0
     Transaction ID: 0x1bb39852
     Seconds elapsed: 0
   ▸ Bootp flags: 0x8000, Broadcast flag (Broadcast)
     Client IP address: 192.168.0.104
     Your (client) IP address: 0.0.0.0
     Next server IP address: 192.168.0.1
     Relay agent IP address: 0.0.0.0

0000  ff ff ff ff ff ff 30 e3  7a a5 b6 99 08 00 45 00   ······0· z·····E·
0010  02 40 2a ec 40 00 40 11  4c b2 c0 a8 00 67 ff ff   ·@*·@·@· L····g··
0020  ff ff 00 44 00 43 02 2c  49 c9 01 01 06 00 1b b3   ···D·C·, I·······
0030  98 52 00 00 80 00 c0 a8  00 68 00 00 00 00 c0 a8   ·R······ ·h······
0040  00 01 00 00 00 00 46 32  e9 d8 50 9a 00 00 00 00   ······F2 ··P·····
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
```

Justification:

I created 6 fake MAC addresses. Using those fake MAC addresses as source addresses, I created malicious DHCP DISCOVER packets.

The router was manipulated to provide only 6 IP addresses. 3 devices were already connected to the router. So, the router could provide only rest of the 3 IP addresses.

When I sent these 6 DISCOER packets to server, DHCP server responded to 3 malicious packets and assigned each of those fake MAC addresses with a valid IP addresses. It didn’t respond to rest of the packets as it didn’t had any IP address left to assign. As a result, when I tried to connect my phone to the router which was previously disconnected, it couldn’t connect even though the password was correct.

Thus the entire IP address pool of the DHCP server was depleted.

So, my attack was successful.

Countermeasure for DHCP Starvation Attack:

In case of wired connection, putting MAC address limit in Server Machine Port can prevent attack. MAC address limiting is enabled on interface (ports). MAC address limiting sets a limit on the number of MAC addresses that can be learned dynamically on a single Layer 2 access interface. When the limit is exceeded, incoming packets with new MAC addresses are treated as specified by the configuration.

Thus DHCP Starvation Attack can be prevented.