Project report

Final Project-
DHCP Server & Client

Course Title: *Internet Applications*

Name: *TANG Muyang (2014212929)*

OU Yihang (2014212948)

Date: *June 17th,2017*

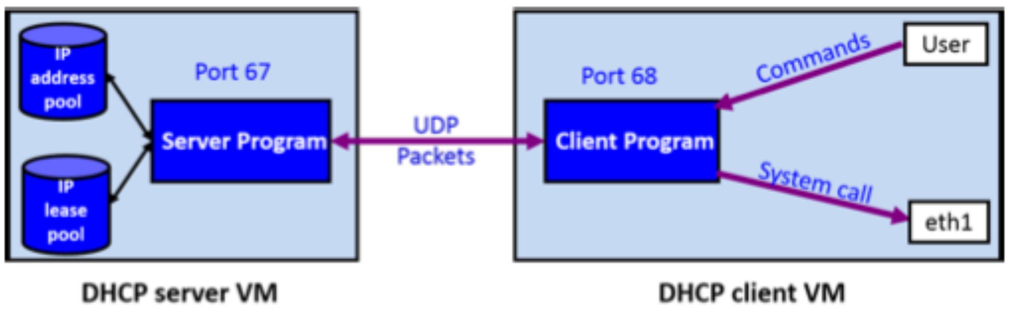
1. Overview

1.1 Overview requirements

This project aim to implements a DHCP process which includes client and server. Through this project, we should understand the DHCP (Dynamic Host Configuration Protocol) comprehensively. All details of DHCP include protocol and different packet information and attain IP from Ubuntu virtual machine.

1.2 Goals

- Deeply understand the details of DHCP (Dynamic Host Configuration Protocol)
- Complete a DHCP server program and run it in one Ubuntu virtual machine
- Complete a DHCP client program and run it in another Ubuntu virtual machine



(DHCP Server & Client Program Model)

2. Requirements Analysis

2.1 Development environment

- IDE: Sublime
- Language: C
- Operation system: Ubuntu
- Compiler: gcc

2.2 Special function requirement

2.2.1 Client

a) Listen to UDP port 68, attain or set the IP address, MAC address and netmask for the client virtual machine.

b) Obtain IP address, netmask, gateway, dns server address, dhcp server ID and IP address lease time from DHCP server and configure IP address.

c) Support DHCP messages: **DISCOVERY/OFFER/REQUEST/ACK, RELEASE, REQUEST/NAK, REQUEST/ACK, INFORM/ACK**

d) Support DHCP options:

1 (Subnet Mask Value)

3 (Router addresses)

6 (DNS Server addresses)

51 (IP Address Lease Time)

53 (DHCP Message Type)

54 (DHCP Server Identification)

55 (Parameter request list)

58 (DHCP Renewal Time T1)

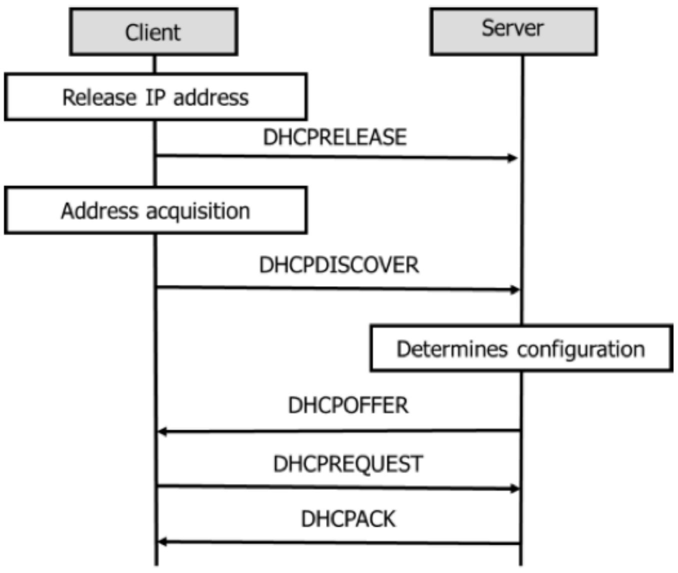
59 (DHCP Rebinding Time T2)

60 (Class Identifier, set as our student number)

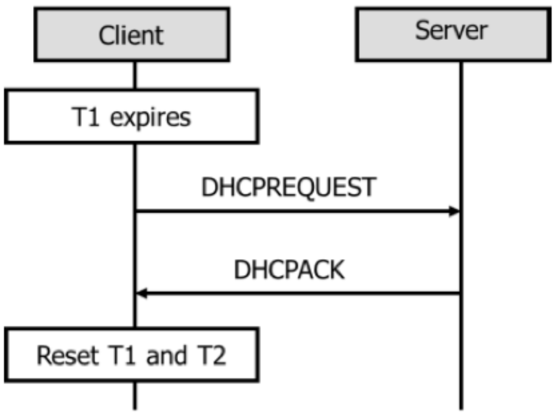
255 (END)

e) Four messages during address acquisition can be delivered on broadcast packets: DISCOVEY/OFFER/REQUEST/ACK.

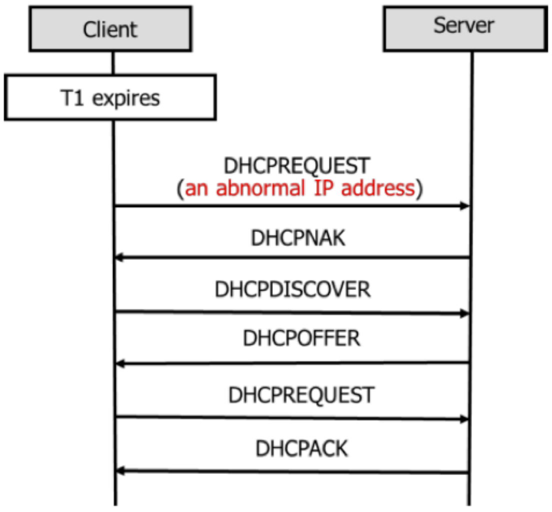
f) Finish the following DHCP procedures



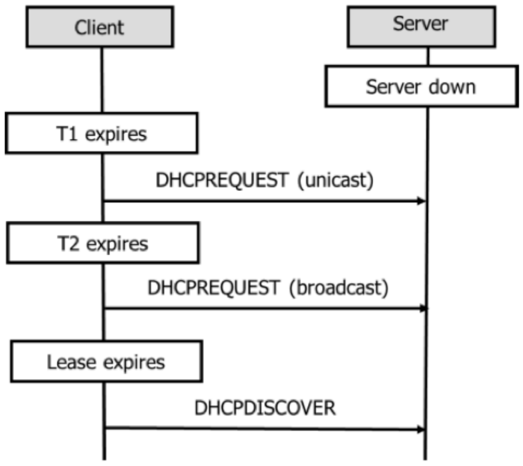
(release + address acquisition)



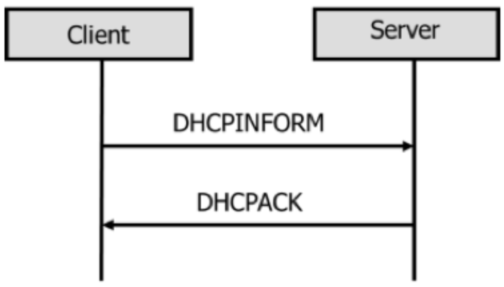
(successful lease renew)



(Fig.4 failed lease renew (with an abnormal IP address) + address acquisition again)



(failed lease renew (server down))



(inform)

2.2.2 Server

a) The main requirement of server is to response to the different requirements from the client. It mainly responses 3 kinds of packets: DHCP Offer, DHCP ACK and DHCP NAK, and some operation when it receives DHCP Release.

b) When the server receives DHCP Discover, it needs to broadcast or unicast DHCP Offer according to DHCP Discover’s broadcast flag, containing IP address offered from the dhcp.config file, server identifier (option 54), lease time, (option 51) from the dhcp.config file, subnet mask (option 1) from the dhcp.config file, router IP (option 3) from the dhcp.config file, domain name server IP (option 6) from the dhcp.config file, renewal time value (option 58), rebinding time value (option 59).

c) When the server receives DHCP Request for applying IP address, it needs to broadcast or unicast DHCP ACK according to DHCP Request’s broadcast flag, containing IP address offered, server identifier (option 54), lease time, (option 51), subnet mask (option 1), router IP (option 3), domain name server IP (option 6), renewal time value (option 58), rebinding time value (option 59).

d) When the server receives DHCP Request for applying IP address, it needs to broadcast or unicast DHCP ACK according to DHCP Request’s broadcast flag, containing IP address offered, server identifier (option 54), lease time, (option 51), subnet mask (option 1), router IP (option 3), domain name server IP (option 6), renewal time value (option 58), rebinding time value (option 59). And then modify the dhcp.config file to delete the offered IP and store the timestamp, mac address of client, and offered IP in the dhcp.lease.

e) When the server receives DHCP Request for renew a IP address, it needs to check in the dhcp.lease to see whether it is a legal IP. If it is a legal IP, it should unicast DHCP ACK, containing IP address offered, server identifier (option 54), lease time, (option 51), subnet mask (option 1), router IP (option 3), domain name server IP (option 6), renewal time value (option 58), rebinding time value (option 59). If it is not legal, the server will unicast DHCP NAK containing server identifier (option 54).

f) When the server receives DHCP Inform, it needs to unicast DHCP ACK containing IP address offered, server identifier (option 54), subnet mask (option 1), router IP (option 3), domain name server IP (option 6), renewal time value (option 58), rebinding time value (option 59).

g) When the server receives DHCP Release, it needs to delete the related part of the client in dhcp.lease and add the client IP to the IP address pool.

3. Preliminary Design

3.1 Client

3.1.1 Data structure design

- Struct

  sockaddr_in sendAddr //Client address struct

  sockaddr_in serverAddr //Server address struct

  sockaddr_in selfAddr //Client self-address struct

- Variable

  char sendBuf[] //Send data buffer

  char recvBuf[] //receive data buffer

  int sendSize=312 //Send packet size=312

  int recvSize //Receive packet size

  int svrAddrLen //Server Address Length

  int counter //Padding counter

  unsigned int DHCPServerIPAddress =0xffffffff //Broadcast Server IP address

  unsigned int LeaseTime //Lease time

  unsigned int T1Time //T1 expires time

  unsigned int T2Time //T2 expires time

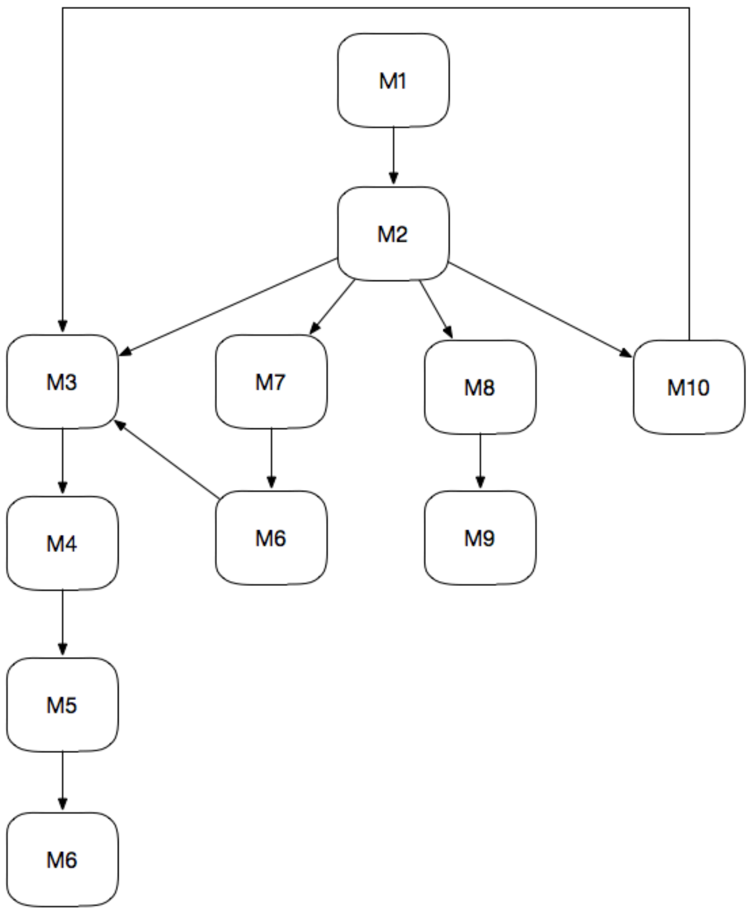
  int i=0 //timeclock

  pthread_t thd1 //thread id using for control T1, T2 and Lease time

```
int thd=0            //To judge thd1
int type =-1         //Set Message Type
int subtype =-1      //Set ACK message Type
```

3.1.2 Modules

a) C.M1: Set the Client Self Address and full-in DHCP packet
b) C.M2: Interaction command line
c) C.M3: Send Discover
d) C.M4: Receive Offer
e) C.M5: Send Request (Include T1 expire (Unicast) and T2 expire (Broadcast))
f) C.M6: Receive ACK and NAK (when receive Request)
g) C.M7: Send Release (Set abnormal IP)
h) C.M8: Send Inform
i) C.M9: Receive ACK (when receive Inform)
j) C.M10: T1, T2, Lease time interaction



(Client Module relationship)

3.2 Server

3.2.1 Data structure design

- Struct

  struct sockaddr_in svrAddr; /* Local address */

  struct sockaddr_in cltAddr; /* Client address */

  struct sockaddr_in broadcastAddr; /* Broadcast address */

- Variable

  unsigned int cliAddrLen; /* Length of client address */

  int sock; /* Socket */

  char recvBuf[MAX_SIZE]; /* Buffer for receive */

  char sendBuf[MAX_SIZE]; /* Buffer for send */

  int recvMsgSize; /* Size of received message */

  int i;   /* Counter */

  int LEASETIME;   /* Lease Time*/

3.2.2 Modules

a) S.M1: Receive Discover

b) S.M2: Send Offer

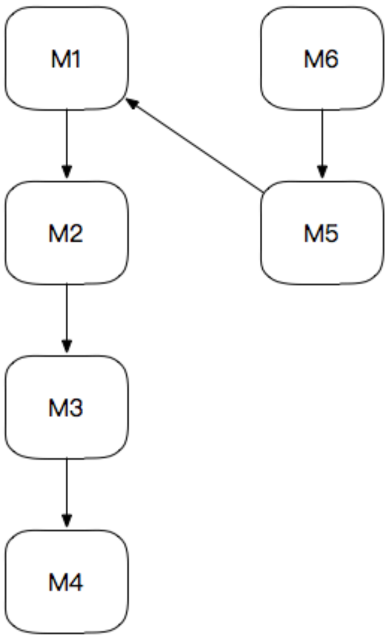
c) S.M3: Receive Request

d) S.M4: Send ACK

e) S.M5: Send ACK (renew IP)

f) S.M6: Receive Release (Release IP)

g) S.M7: Receive Inform



(Server Module relationship)

4. Detailed Design

4.1 Client

4.1.1 Main part

In main part, Client as the interaction role of this system, we should input command in command line. Using 7 different command to achieve different functions. Besides we should full in the content of DHCP packet information. When received packets and sending packets. Then modify the different parts and rewrite the options. When rewiring the different parts, the server will write down the content in the appropriate positions.

```
-------------------------------------------------------------------
Sock()
Bind to eth1()
Bind()
Full in DHCP Packet Basic information
Interact()
        For(;;)
                receive()
        sendBuf<-recBuf
                get mask, router, DNS, lease time, prepared IP
        if(option53=discover)/*broadcast offer*/
                        if(available IP=0)
                        warn
                        else
                        discoverflag<-1
                        fill in boot reply
                        fill in prepared IP
                        fill in option53
                        fill in option 54
                        fill in option 51
                        fill in option 1
                        fill in option 3
                        fill in option 6
                        fill in option 58
                        fill in option 59
                        fill in option 255
                        padding
```

```
else if(option53=Request)
        if(client IP=0.0.0.0)/*broadcast ACK*/
        fill in boot reply
        fill in prepared IP
        fill in option53
        fill in option 54
        fill in option 51
        fill in option 1
        fill in option 3
        fill in option 6
        fill in option 58
        fill in option 59
        fill in option 255
        padding
        else
        discoverflag<-1
        fill in boot reply
        if(legal)/*response ACK*/
            fill in client IP
            fill in option53
            fill in option 54
            fill in option 51
            fill in option 1
            fill in option 3
            fill in option 6
            fill in option 58
            fill in option 59
            fill in option 255
            padding
        else/*response NAK*/
            fill in option53
            fill in option 54
            fill in option 255
            padding
else if(option53=Release)
    fill in option53
    fill in option 54
```

```
                        fill in option 1
                        fill in option 3
                        fill in option 6
                        fill in option 58
                        fill in option 59
                        fill in option 255
                        padding
                    else if(option53=Inform) /*response ACK*/
                        fill in option53
                        fill in option 54
                        fill in option 1
                        fill in option 3
                        fill in option 6
                        fill in option 58
                        fill in option 59
                        fill in option 255
                        padding
                if(receive DHCP packet)
                    if(flag=unicast)
                        send unicast
                    else if (flag=broadcast)
                        send broadcast
                ------------------------------------------------------------------------
```

4.1.2 Get IP

Using ioctl to get IP from eth1.

4.1.3 T1, T2 and LeaseTime

```
------------------------------------------------------------------------
Set thread thd1
If (thd1)
    Int t=0
    t=t+1
    if(t=T1)
        send renew1
    if (t=T2)
        send renew2
        if (t=LeaseTime)
                send Discover
-----------------------------------------------------------------
```

4.2 Server

4.2.1 Main part

In main parts, firstly the server will copy all the content from the received packet to send packet since there are many parts are the same in the received packets and sending packets. Then modify the different parts and rewrite the options. When rewiring the different parts, the server will write down the content in the appropriate positions.

Pseudocode:

```
-----------------------------------------------------------------
Sock()
Bind to eth1()
Bind()
        For(;;)
                receive()
        sendBuf<-recBuf
                get mask, router, DNS, lease time, prepared IP
        if(option53=discover)/*broadcast offer*/
                        if(available IP=0)
                        warn
                        else
                        discoverflag<-1
                        fill in boot reply
                        fill in prepared IP
                        fill in option53
                        fill in option 54
                        fill in option 51
                        fill in option 1
                        fill in option 3
                        fill in option 6
                        fill in option 58
                        fill in option 59
                        fill in option 255
```

```
            padding
    else if(option53=Request)
            if(client IP=0.0.0.0)/*broadcast ACK*/
            fill in boot reply
            fill in prepared IP
            fill in option53
            fill in option 54
            fill in option 51
            fill in option 1
            fill in option 3
            fill in option 6
            fill in option 58
            fill in option 59
            fill in option 255
            padding
            else
            discoverflag<-1
            fill in boot reply
            check legal
            if(legal)/*response ACK*/
                renew dhcp.lease
                fill in client IP
                fill in option53
                fill in option 54
                fill in option 51
                fill in option 1
                fill in option 3
                fill in option 6
                fill in option 58
                fill in option 59
                fill in option 255
                padding
            else/*response NAK*/
                fill in option53
                fill in option 54
                fill in option 255
                padding
```

```
          else if(option53=Release)
               modify file
          else if(option53=Inform) /*response ACK*/
               fill in option53
               fill in option 54
               fill in option 1
               fill in option 3
               fill in option 6
               fill in option 58
               fill in option 59
               fill in option 255
               padding
  if(receive DHCP packet)
         if(flag=unicast)
              send unicast
         else if (flag=broadcast)
              send broadcast
  ------------------------------------------------------------------------
```

4.2.2 Get IP

Using ioctl to get IP from eth1.

4.2.3 Offer IP

Remove IP from dhcp.config and add the related information to the dhcp.lease. Get all the information except offered IP and write them in a new file. Then delete the former file and modify the file name.

Pseudocode:

```
--------------------------------------------------------------------
Input:MAC
Open dhcp.config
Create dhcp.config2
Get mask and write in dhcp.config2
Get router and write in dhcp.config2
Get DNS and write in dhcp.config2
Get lease time and write in dhcp.config2
Get prepared IP /*Don't write here*/
While (!EOF)
```

```
        Get free IP and write in dhcp.config2
Close files
dhcp.config<-dhcp.config2
get timestamp
open dhcp.lease to append
append timestamp MAC IP
close files
------------------------------------------------------------------
```

4.2.4 Get IP address number

Get the number of lines of dhcp.config, and the number-5 is the available number of IPs.

Pseudocode:

```
------------------------------------------------------------------
Input:file
Output:number of available IPs
Open dhcp.config
While (!EOF)
        number +1
Close files
Return number -5
------------------------------------------------------------------
```

4.2.5 Check IP

Since for each line in dhcp.lease, it will store the timestamp, MAC and client IP. Find the line contains IP and read line. Split it into 3 parts, check the IP and MAC, then compare the time now and lease time.

```
------------------------------------------------------------------
Input:IP,MAC
Output:have or not
Open dhcp.lease
While (!EOF)
        Get timestamp
        Timestamp<-timestamp+leasetime
        Get MAC
        Get IP
        Get now
        If IP=IP, MAC=MAC, timestamp>now
        Return 1
Init string
Close files
-----------------------------------------------------------------
```

4.2.6 Release IP

Since for each line in dhcp.lease, it will store the timestamp, MAC and client IP. Find the line contains IP and delete the line. Then append IP in the dhcp.config

```
-----------------------------------------------------------------
Input:IP
Flag<-0
Open dhcp.lease
Create dhcp.lease2
While (!EOF)
      If (there is a IP in lease pool)
            Get a line from dhcp.lease
            If (the IP is not a part of the line)
                Write it in the dhcp.lease2
                Init string
            Else
                Flag<-1
                Init string
Close files
Dhcp.lease<-dhcp.lease2
If (flag=1)
     Open dhcp.config
     Append IP to dhcp.config
Close files
-----------------------------------------------------------------
```

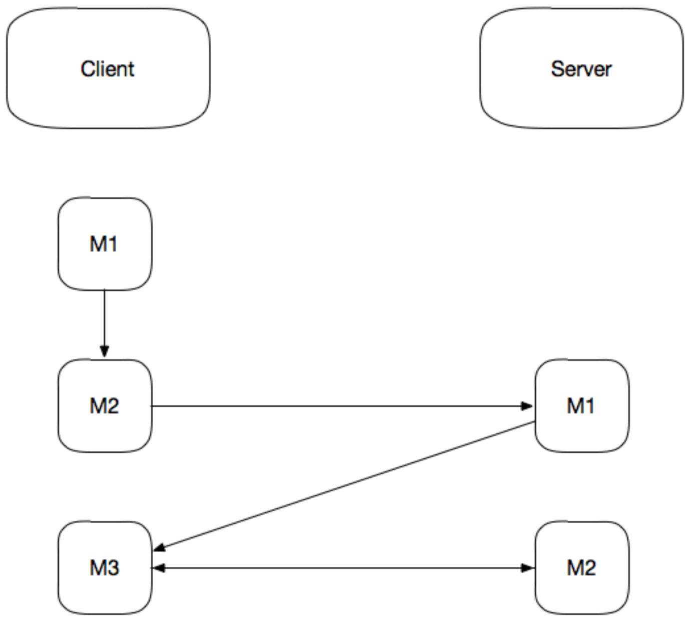
4.2.7 Renew IP

Similar as check. But write the time now in as new timestamp

```
-----------------------------------------------------------------
Input:IP,MAC
Output:have or not
Open dhcp.lease
Open dhcp.lease2
While (!EOF)
        Get timestamp
        Timestamp<-timestamp+leasetime
        Get MAC
        Get IP
        Copy string
        If IP=IP
           Write timestamp MAC IP in the dhcp.lease2
        Else
          Write string in the dhcp.lease
Init string
Close files
dhcp.lease<-dhcp.lease2
--------------------------------------------------------------------
```

4.3 Interaction between Client and Server

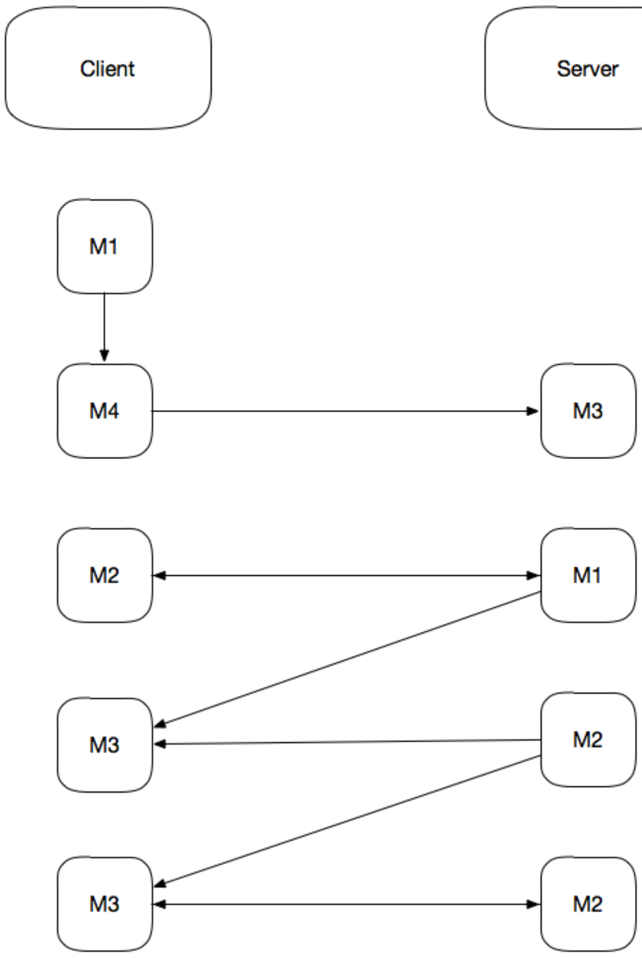
4.3.1 Discover-Offer-Request-ACK



C.M1: main()   C.M2: Send Discover

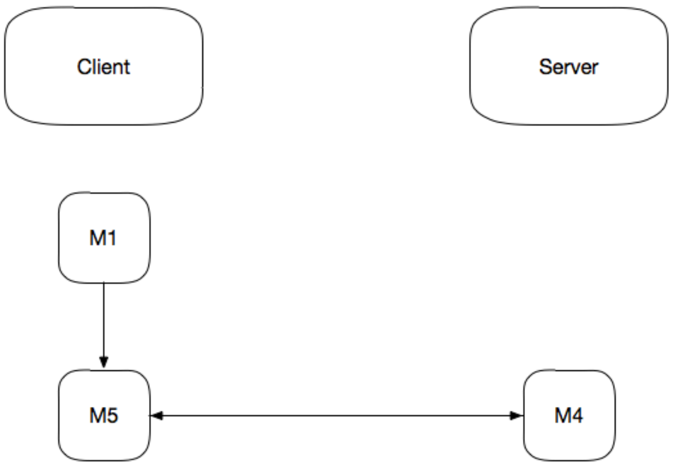
C.M3: Send Request S.M1: Send Offer   S.M2: Send ACK

4.3.2 Release-Renew



C.M1: main() C.M2: Send Discover C.M3: Send Request
C.M4: Send Release S.M1: Send Offer S.M2: Send ACK
S.M3: Release IP (Send NAK)

4.3.3 Inform

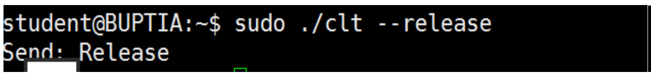


C.M1: main() C.M5: Send Inform S.M4: Send ACK

5. Results

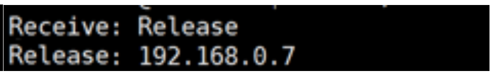
At the beginning, the client releases the IP. Client sends release and the server operates the files.

Client:



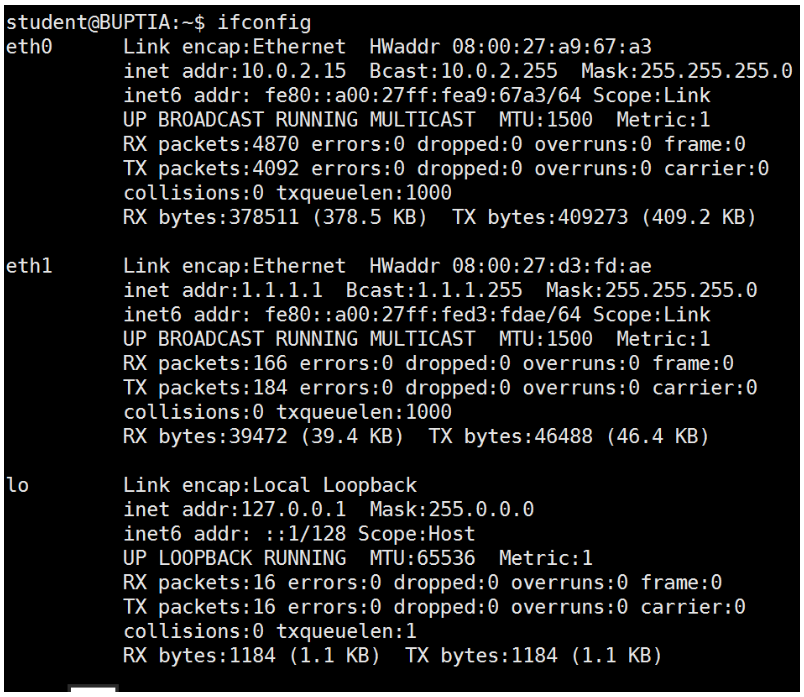
```
student@BUPTIA:~$ sudo ./clt --release
Send: Release
```

Server:



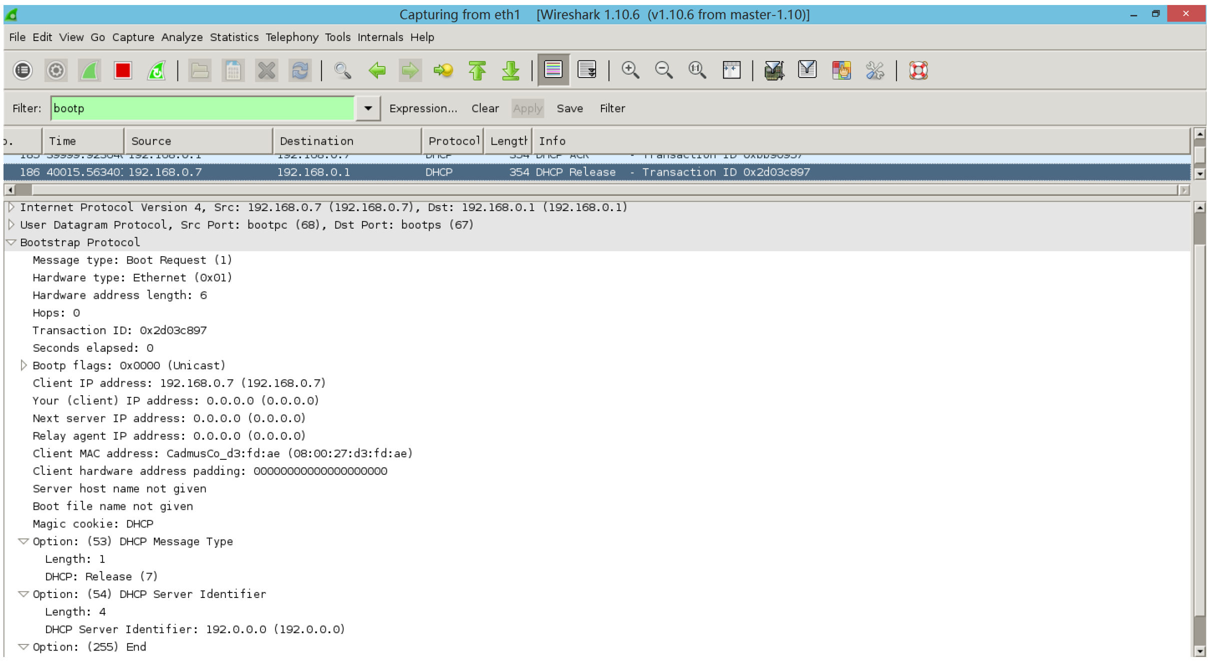
```
Receive: Release
Release: 192.168.0.7
```

Use ifconfig to see whether it is released:



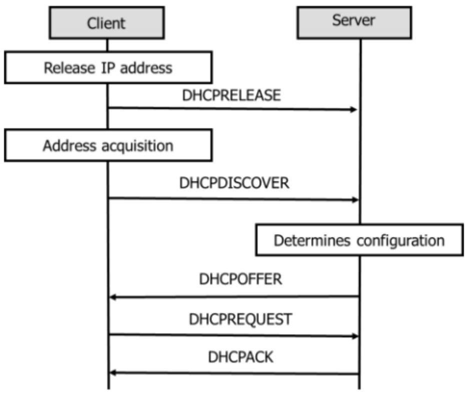
```
student@BUPTIA:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:a9:67:a3
          inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fea9:67a3/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:4870 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4092 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:378511 (378.5 KB)  TX bytes:409273 (409.2 KB)

eth1      Link encap:Ethernet  HWaddr 08:00:27:d3:fd:ae
          inet addr:1.1.1.1  Bcast:1.1.1.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fed3:fdae/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:166 errors:0 dropped:0 overruns:0 frame:0
          TX packets:184 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:39472 (39.4 KB)  TX bytes:46488 (46.4 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:16 errors:0 dropped:0 overruns:0 frame:0
          TX packets:16 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:1184 (1.1 KB)  TX bytes:1184 (1.1 KB)
```

Release packet:

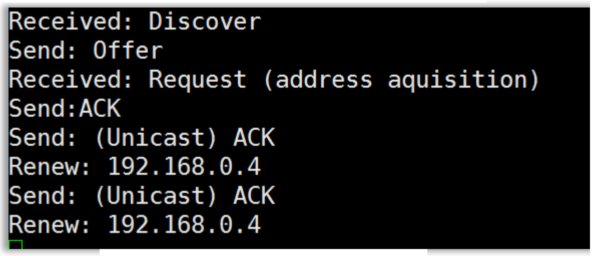

Capturing from eth1   [Wireshark 1.10.6 (v1.10.6 from master-1.10)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: bootp
Expression... Clear Apply Save Filter

```
Time                Source          Destination     Protocol  Length  Info
186  40015.56340    192.168.0.7     192.168.0.1     DHCP      354 DHCP Release  - Transaction ID 0x2d03c897

Internet Protocol Version 4, Src: 192.168.0.7 (192.168.0.7), Dst: 192.168.0.1 (192.168.0.1)
User Datagram Protocol, Src Port: bootpc (68), Dst Port: bootps (67)
Bootstrap Protocol
  Message type: Boot Request (1)
  Hardware type: Ethernet (0x01)
  Hardware address length: 6
  Hops: 0
  Transaction ID: 0x2d03c897
  Seconds elapsed: 0
  Bootp flags: 0x0000 (Unicast)
  Client IP address: 192.168.0.7 (192.168.0.7)
  Your (client) IP address: 0.0.0.0 (0.0.0.0)
  Next server IP address: 0.0.0.0 (0.0.0.0)
  Relay agent IP address: 0.0.0.0 (0.0.0.0)
  Client MAC address: CadmusCo_d3:fd:ae (08:00:27:d3:fd:ae)
  Client hardware address padding: 00000000000000000000
  Server host name not given
  Boot file name not given
  Magic cookie: DHCP
  Option: (53) DHCP Message Type
    Length: 1
    DHCP: Release (7)
  Option: (54) DHCP Server Identifier
    Length: 4
    DHCP Server Identifier: 192.0.0.0 (192.0.0.0)
  Option: (255) End
```

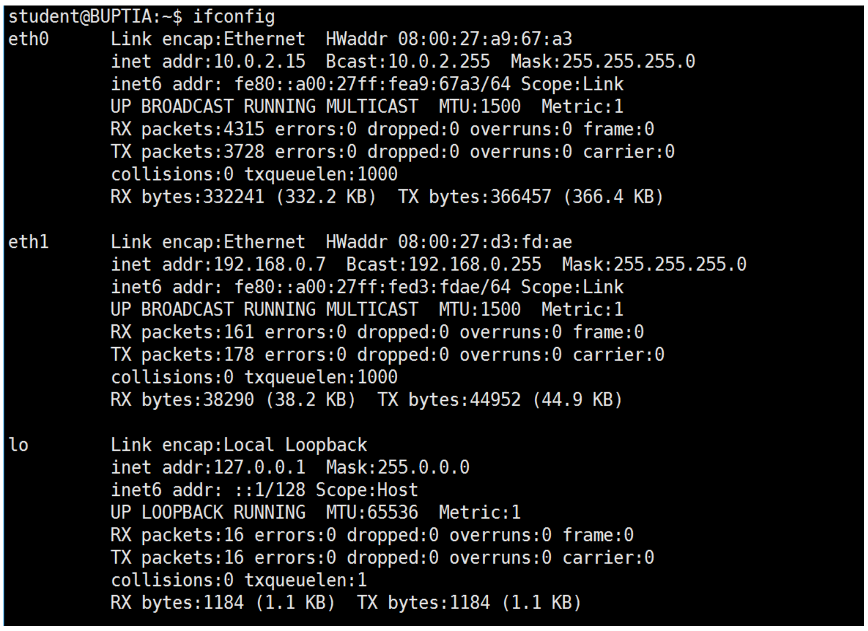
After we run the client with command, we can see the whole process of getting an IP address. After the discover-offer-request-ack process, the client get the information of IP, next server IP, DHCP server IP, lease time, mask, router, DNS and T1, T2.



Server:

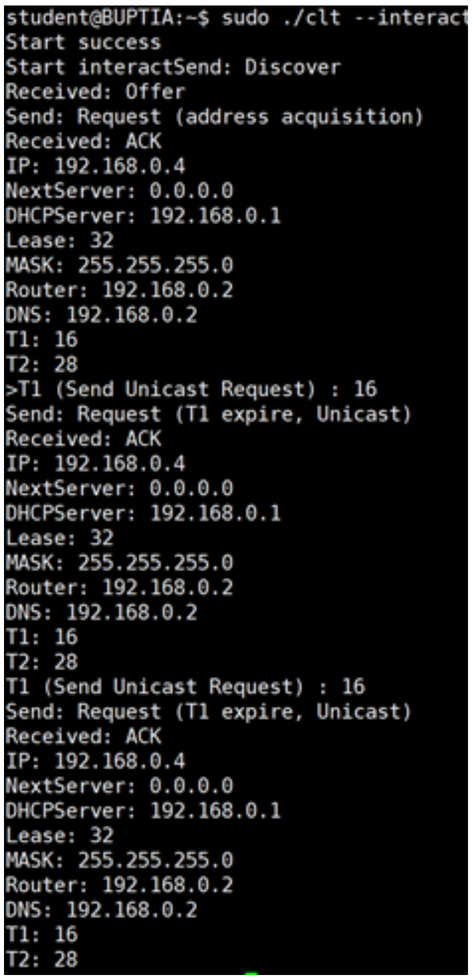


```
Received: Discover
Send: Offer
Received: Request (address aquisition)
Send:ACK
Send: (Unicast) ACK
Renew: 192.168.0.4
Send: (Unicast) ACK
Renew: 192.168.0.4
```



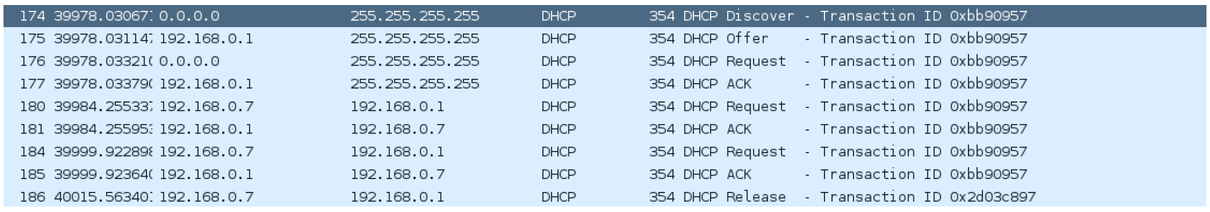
```
student@BUPTIA:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:a9:67:a3
          inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fea9:67a3/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:4315 errors:0 dropped:0 overruns:0 frame:0
          TX packets:3728 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:332241 (332.2 KB)  TX bytes:366457 (366.4 KB)

eth1      Link encap:Ethernet  HWaddr 08:00:27:d3:fd:ae
          inet addr:192.168.0.7  Bcast:192.168.0.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fed3:fdae/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:161 errors:0 dropped:0 overruns:0 frame:0
          TX packets:178 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:38290 (38.2 KB)  TX bytes:44952 (44.9 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:16 errors:0 dropped:0 overruns:0 frame:0
          TX packets:16 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:1184 (1.1 KB)  TX bytes:1184 (1.1 KB)
```

Client:



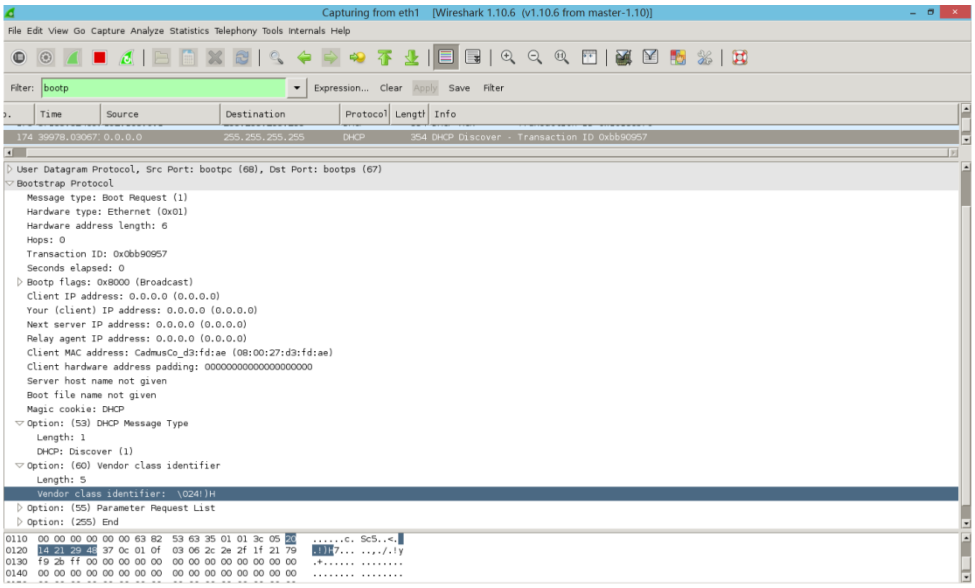
```
student@BUPTIA:~$ sudo ./clt --interact
Start success
Start interactSend: Discover
Received: Offer
Send: Request (address acquisition)
Received: ACK
IP: 192.168.0.4
NextServer: 0.0.0.0
DHCPServer: 192.168.0.1
Lease: 32
MASK: 255.255.255.0
Router: 192.168.0.2
DNS: 192.168.0.2
T1: 16
T2: 28
>T1 (Send Unicast Request) : 16
Send: Request (T1 expire, Unicast)
Received: ACK
IP: 192.168.0.4
NextServer: 0.0.0.0
DHCPServer: 192.168.0.1
Lease: 32
MASK: 255.255.255.0
Router: 192.168.0.2
DNS: 192.168.0.2
T1: 16
T2: 28
T1 (Send Unicast Request) : 16
Send: Request (T1 expire, Unicast)
Received: ACK
IP: 192.168.0.4
NextServer: 0.0.0.0
DHCPServer: 192.168.0.1
Lease: 32
MASK: 255.255.255.0
Router: 192.168.0.2
DNS: 192.168.0.2
T1: 16
T2: 28
```

Then after the T1, the client will ask for renew, unicast a request and get an ACK from the server. It will reset the timer and wait for the next T1.



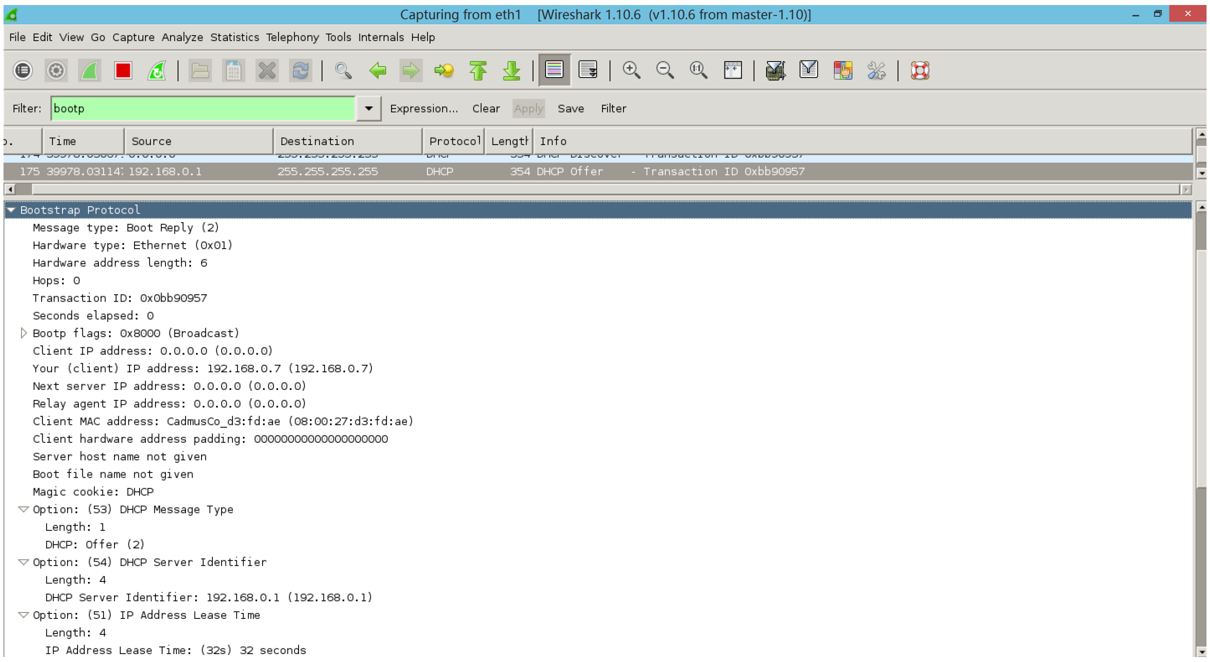
```
174  39978.03067: 0.0.0.0          255.255.255.255    DHCP       354 DHCP Discover  - Transaction ID 0xbb90957
175  39978.03114: 192.168.0.1      255.255.255.255    DHCP       354 DHCP Offer     - Transaction ID 0xbb90957
176  39978.03321( 0.0.0.0          255.255.255.255    DHCP       354 DHCP Request   - Transaction ID 0xbb90957
177  39978.03379( 192.168.0.1      255.255.255.255    DHCP       354 DHCP ACK       - Transaction ID 0xbb90957
180  39984.25533: 192.168.0.7      192.168.0.1        DHCP       354 DHCP Request   - Transaction ID 0xbb90957
181  39984.25595: 192.168.0.1      192.168.0.7        DHCP       354 DHCP ACK       - Transaction ID 0xbb90957
184  39999.92289( 192.168.0.7      192.168.0.1        DHCP       354 DHCP Request   - Transaction ID 0xbb90957
185  39999.92364( 192.168.0.1      192.168.0.7        DHCP       354 DHCP ACK       - Transaction ID 0xbb90957
186  40015.56340: 192.168.0.7      192.168.0.1        DHCP       354 DHCP Release   - Transaction ID 0x2d03c897
```

Discover packet:

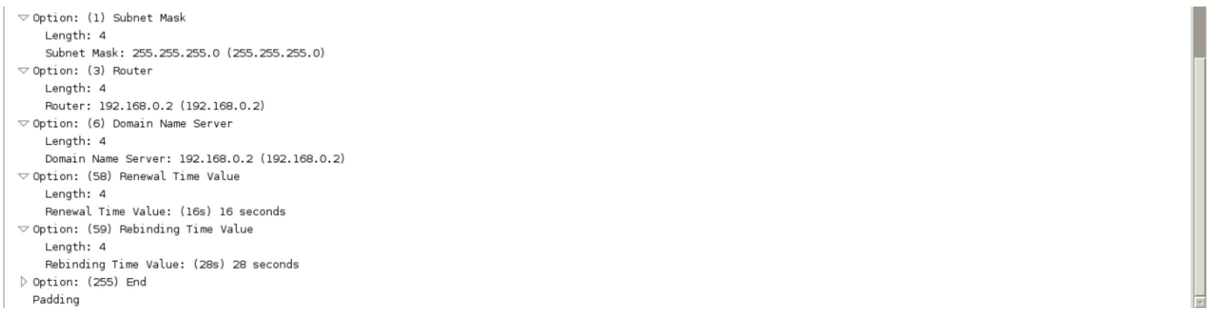

Capturing from eth1   [Wireshark 1.10.6 (v1.10.6 from master-1.10)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
bootp
Expression... Clear Apply Save Filter

```
 o.  Time          Source          Destination       Protocol  Length  Info
174  39978.03067:  0.0.0.0         255.255.255.255   DHCP       354 DHCP Discover  - Transaction ID 0xbb90957

▷ User Datagram Protocol, Src Port: bootpc (68), Dst Port: bootps (67)
▽ Bootstrap Protocol
    Message type: Boot Request (1)
    Hardware type: Ethernet (0x01)
    Hardware address length: 6
    Hops: 0
    Transaction ID: 0x0bb90957
    Seconds elapsed: 0
  ▷ Bootp flags: 0x8000 (Broadcast)
    Client IP address: 0.0.0.0 (0.0.0.0)
    Your (client) IP address: 0.0.0.0 (0.0.0.0)
    Next server IP address: 0.0.0.0 (0.0.0.0)
    Relay agent IP address: 0.0.0.0 (0.0.0.0)
    Client MAC address: CadmusCo_d3:fd:ae (08:00:27:d3:fd:ae)
    Client hardware address padding: 00000000000000000000
    Server host name not given
    Boot file name not given
    Magic cookie: DHCP
  ▽ Option: (53) DHCP Message Type
      Length: 1
      DHCP: Discover (1)
  ▽ Option: (60) Vendor class identifier
      Length: 5
      Vendor class identifier:  \024!)H
  ▷ Option: (55) Parameter Request List
  ▷ Option: (255) End

0110  00 00 00 00 00 00 63 82  53 63 35 01 01 3c 05 20   ......c. Sc5..<. 
0120  14 21 29 48 37 0c 01 0f  03 06 2c 2e 2f 1f 21 79   .!)H7... ..,./.!y
0130  f9 2b ff 00 00 00 00 00  00 00 00 00 00 00 00 00   .+...... ........
0140  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
```

Offer:

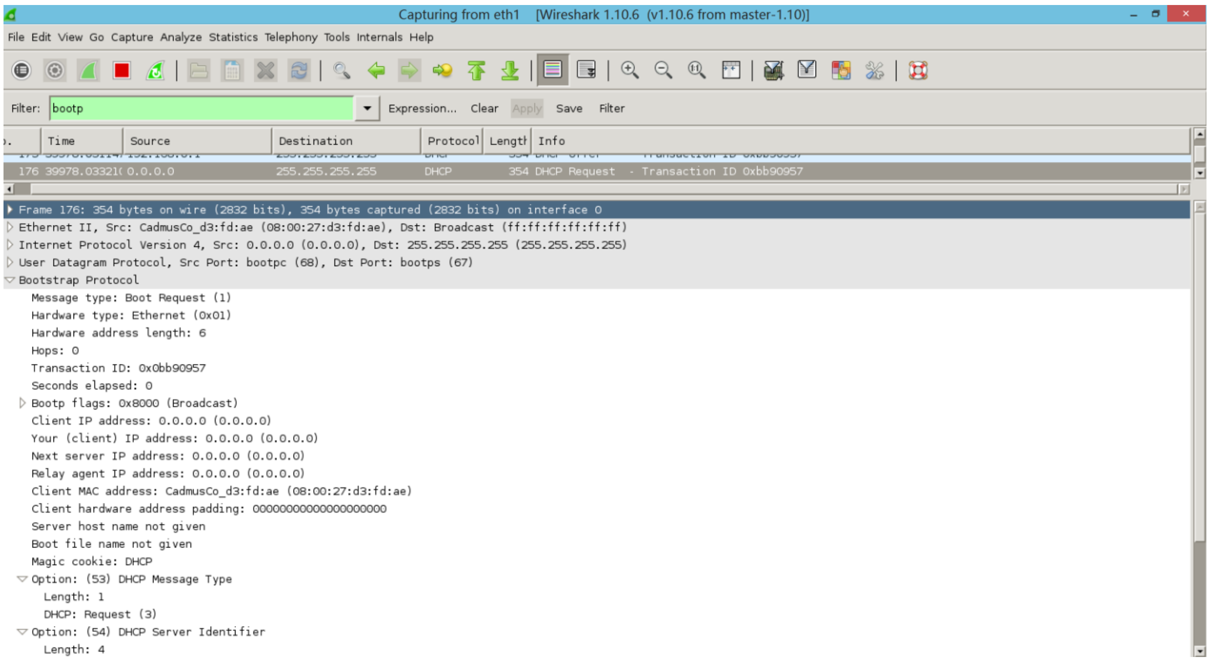

Capturing from eth1   [Wireshark 1.10.6 (v1.10.6 from master-1.10)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
bootp
Expression... Clear Apply Save Filter

```
 o.  Time          Source          Destination       Protocol  Length  Info
175  39978.03114:  192.168.0.1     255.255.255.255   DHCP       354 DHCP Offer     - Transaction ID 0xbb90957

▼ Bootstrap Protocol
    Message type: Boot Reply (2)
    Hardware type: Ethernet (0x01)
    Hardware address length: 6
    Hops: 0
    Transaction ID: 0x0bb90957
    Seconds elapsed: 0
  ▷ Bootp flags: 0x8000 (Broadcast)
    Client IP address: 0.0.0.0 (0.0.0.0)
    Your (client) IP address: 192.168.0.7 (192.168.0.7)
    Next server IP address: 0.0.0.0 (0.0.0.0)
    Relay agent IP address: 0.0.0.0 (0.0.0.0)
    Client MAC address: CadmusCo_d3:fd:ae (08:00:27:d3:fd:ae)
    Client hardware address padding: 00000000000000000000
    Server host name not given
    Boot file name not given
    Magic cookie: DHCP
  ▽ Option: (53) DHCP Message Type
      Length: 1
      DHCP: Offer (2)
  ▽ Option: (54) DHCP Server Identifier
      Length: 4
      DHCP Server Identifier: 192.168.0.1 (192.168.0.1)
  ▽ Option: (51) IP Address Lease Time
      Length: 4
      IP Address Lease Time: (32s) 32 seconds
```



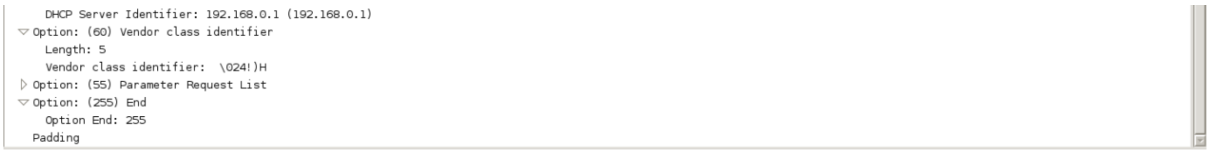
```
  Option: (1) Subnet Mask
    Length: 4
    Subnet Mask: 255.255.255.0 (255.255.255.0)
  Option: (3) Router
    Length: 4
    Router: 192.168.0.2 (192.168.0.2)
  Option: (6) Domain Name Server
    Length: 4
    Domain Name Server: 192.168.0.2 (192.168.0.2)
  Option: (58) Renewal Time Value
    Length: 4
    Renewal Time Value: (16s) 16 seconds
  Option: (59) Rebinding Time Value
    Length: 4
    Rebinding Time Value: (28s) 28 seconds
  Option: (255) End
  Padding
```

Request(d-o-r-a):



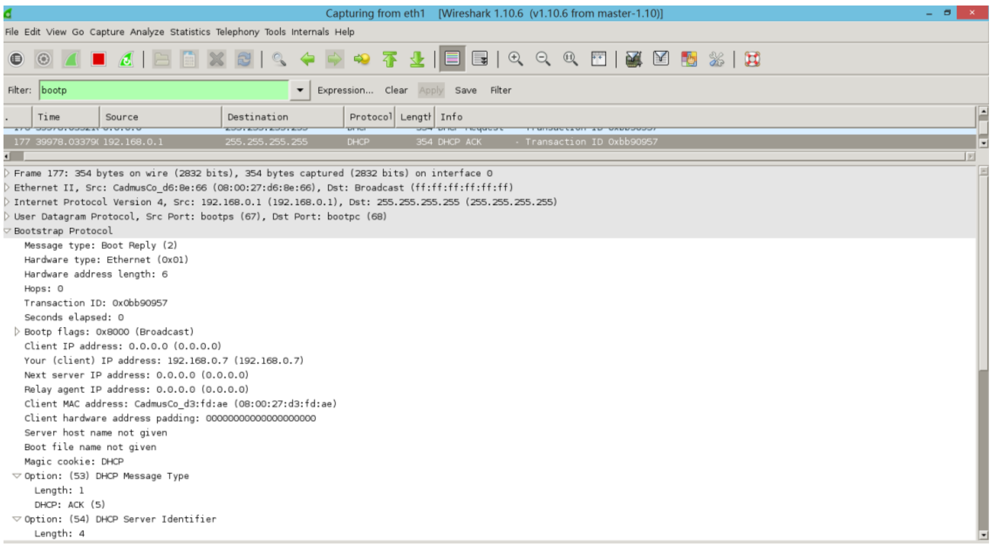
Capturing from eth1   [Wireshark 1.10.6 (v1.10.6 from master-1.10)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: bootp
Expression... Clear Apply Save Filter

```
Time          Source     Destination       Protocol Length Info
176 39978.03321( 0.0.0.0  255.255.255.255  DHCP     354 DHCP Request  - Transaction ID 0xbb90957

Frame 176: 354 bytes on wire (2832 bits), 354 bytes captured (2832 bits) on interface 0
Ethernet II, Src: CadmusCo_d3:fd:ae (08:00:27:d3:fd:ae), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
Internet Protocol Version 4, Src: 0.0.0.0 (0.0.0.0), Dst: 255.255.255.255 (255.255.255.255)
User Datagram Protocol, Src Port: bootpc (68), Dst Port: bootps (67)
Bootstrap Protocol
  Message type: Boot Request (1)
  Hardware type: Ethernet (0x01)
  Hardware address length: 6
  Hops: 0
  Transaction ID: 0x0bb90957
  Seconds elapsed: 0
  Bootp flags: 0x8000 (Broadcast)
  Client IP address: 0.0.0.0 (0.0.0.0)
  Your (client) IP address: 0.0.0.0 (0.0.0.0)
  Next server IP address: 0.0.0.0 (0.0.0.0)
  Relay agent IP address: 0.0.0.0 (0.0.0.0)
  Client MAC address: CadmusCo_d3:fd:ae (08:00:27:d3:fd:ae)
  Client hardware address padding: 00000000000000000000
  Server host name not given
  Boot file name not given
  Magic cookie: DHCP
  Option: (53) DHCP Message Type
    Length: 1
    DHCP: Request (3)
  Option: (54) DHCP Server Identifier
    Length: 4
```



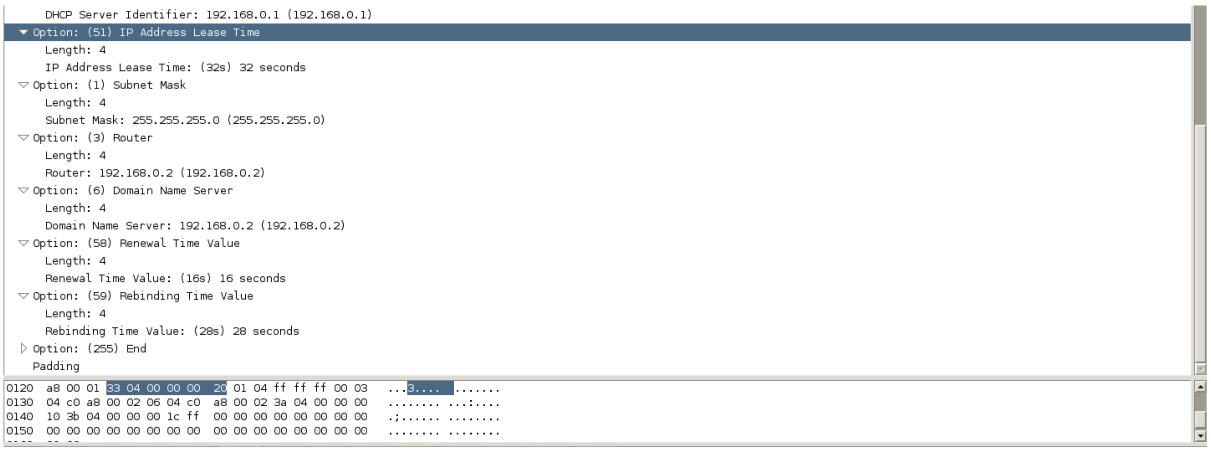
```
    DHCP Server Identifier: 192.168.0.1 (192.168.0.1)
  Option: (60) Vendor class identifier
    Length: 5
    Vendor class identifier:  \024!)H
  Option: (55) Parameter Request List
  Option: (255) End
    Option End: 255
  Padding
```

ACK:



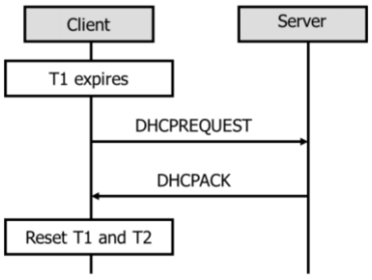
Capturing from eth1   [Wireshark 1.10.6 (v1.10.6 from master-1.10)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: bootp
Expression... Clear Apply Save Filter

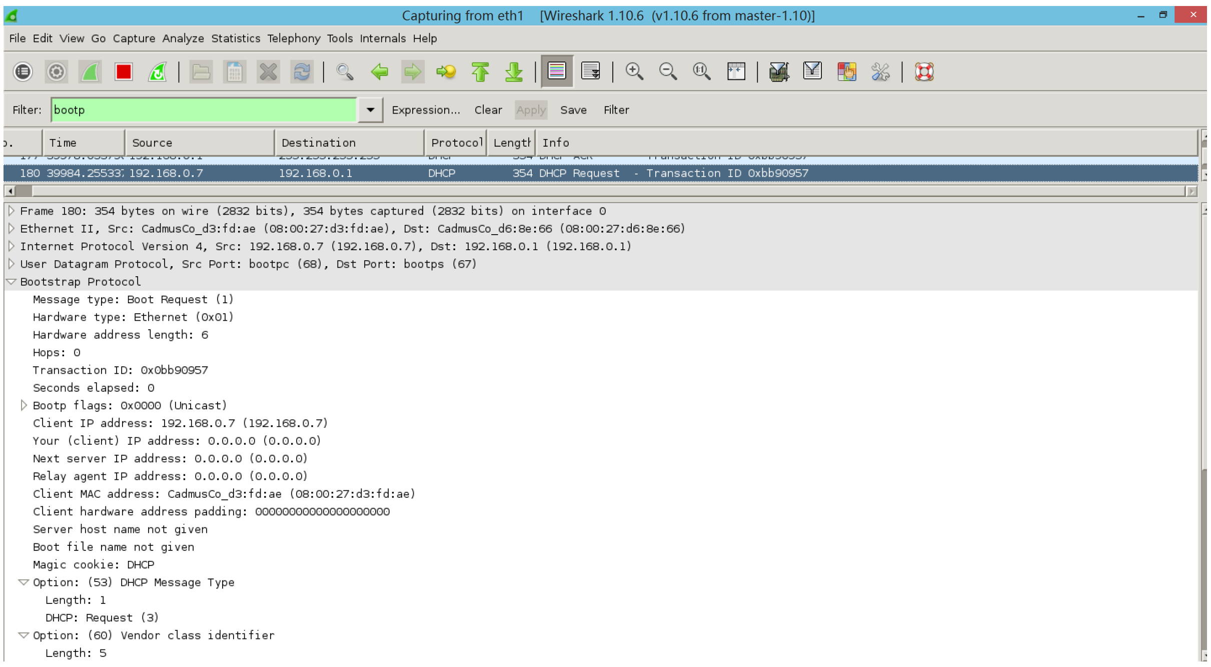
```
Time          Source       Destination       Protocol Length Info
177 39978.03379( 192.168.0.1  255.255.255.255  DHCP     354 DHCP ACK      - Transaction ID 0xbb90957

Frame 177: 354 bytes on wire (2832 bits), 354 bytes captured (2832 bits) on interface 0
Ethernet II, Src: CadmusCo_d6:8e:66 (08:00:27:d6:8e:66), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
Internet Protocol Version 4, Src: 192.168.0.1 (192.168.0.1), Dst: 255.255.255.255 (255.255.255.255)
User Datagram Protocol, Src Port: bootps (67), Dst Port: bootpc (68)
Bootstrap Protocol
  Message type: Boot Reply (2)
  Hardware type: Ethernet (0x01)
  Hardware address length: 6
  Hops: 0
  Transaction ID: 0x0bb90957
  Seconds elapsed: 0
  Bootp flags: 0x8000 (Broadcast)
  Client IP address: 0.0.0.0 (0.0.0.0)
  Your (client) IP address: 192.168.0.7 (192.168.0.7)
  Next server IP address: 0.0.0.0 (0.0.0.0)
  Relay agent IP address: 0.0.0.0 (0.0.0.0)
  Client MAC address: CadmusCo_d3:fd:ae (08:00:27:d3:fd:ae)
  Client hardware address padding: 00000000000000000000
  Server host name not given
  Boot file name not given
  Magic cookie: DHCP
  Option: (53) DHCP Message Type
    Length: 1
    DHCP: ACK (5)
  Option: (54) DHCP Server Identifier
    Length: 4
```



```
      DHCP Server Identifier: 192.168.0.1 (192.168.0.1)
  ▼ Option: (51) IP Address Lease Time
      Length: 4
      IP Address Lease Time: (32s) 32 seconds
  ▽ Option: (1) Subnet Mask
      Length: 4
      Subnet Mask: 255.255.255.0 (255.255.255.0)
  ▽ Option: (3) Router
      Length: 4
      Router: 192.168.0.2 (192.168.0.2)
  ▽ Option: (6) Domain Name Server
      Length: 4
      Domain Name Server: 192.168.0.2 (192.168.0.2)
  ▽ Option: (58) Renewal Time Value
      Length: 4
      Renewal Time Value: (16s) 16 seconds
  ▽ Option: (59) Rebinding Time Value
      Length: 4
      Rebinding Time Value: (28s) 28 seconds
  ▷ Option: (255) End
    Padding

0120  a8 00 01 33 04 00 00 00  20 01 04 ff ff ff 00 03   ...3.... ........
0130  04 c0 a8 00 02 06 04 c0  a8 00 02 3a 04 00 00 00   ........ ...:....
0140  10 3b 04 00 00 00 1c ff  00 00 00 00 00 00 00 00   .;...... ........
0150  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
```

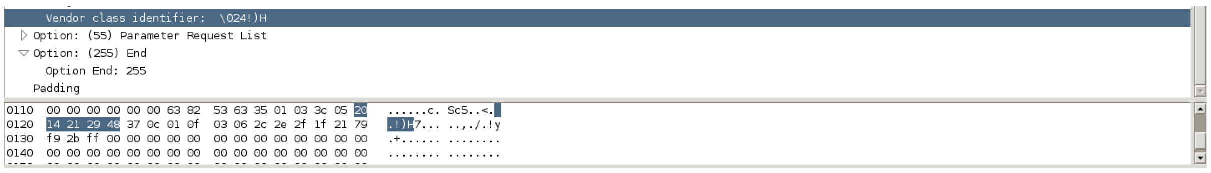
Request(T1):




Capturing from eth1   [Wireshark 1.10.6 (v1.10.6 from master-1.10)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Expression... Clear Apply Save Filter

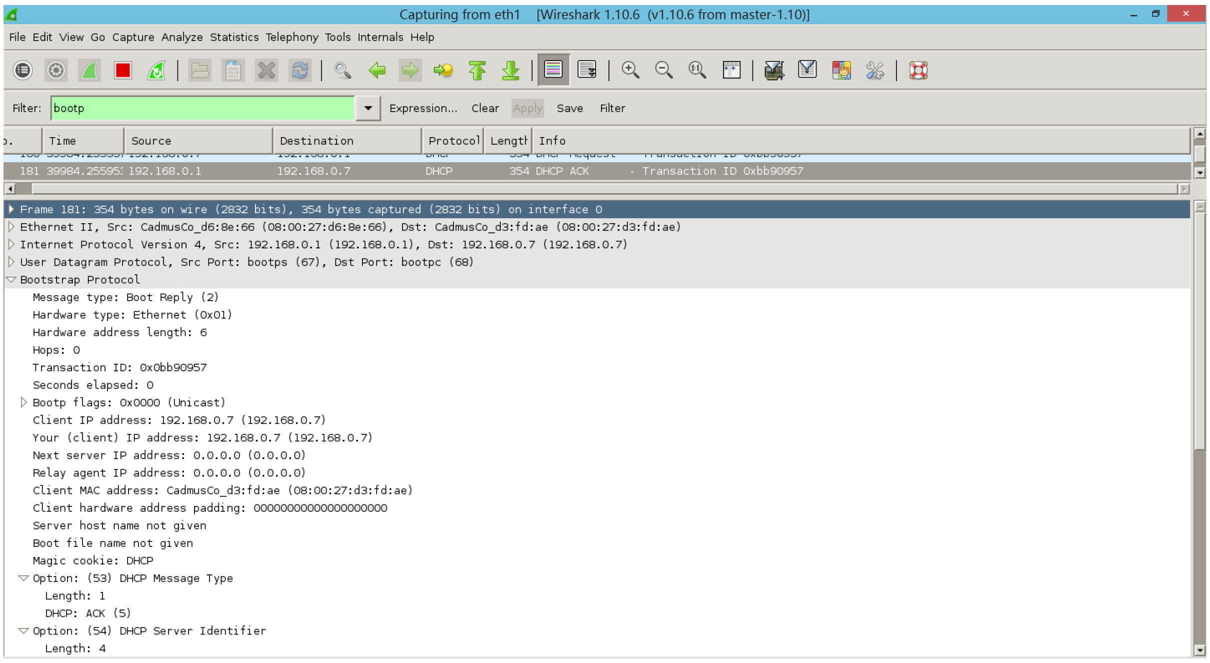
Filter: bootp

```
No.   Time           Source        Destination   Protocol Length Info
  180 39984.25533:   192.168.0.7   192.168.0.1   DHCP     354    DHCP Request  - Transaction ID 0xbb90957

▷ Frame 180: 354 bytes on wire (2832 bits), 354 bytes captured (2832 bits) on interface 0
▷ Ethernet II, Src: CadmusCo_d3:fd:ae (08:00:27:d3:fd:ae), Dst: CadmusCo_d6:8e:66 (08:00:27:d6:8e:66)
▷ Internet Protocol Version 4, Src: 192.168.0.7 (192.168.0.7), Dst: 192.168.0.1 (192.168.0.1)
▷ User Datagram Protocol, Src Port: bootpc (68), Dst Port: bootps (67)
▽ Bootstrap Protocol
    Message type: Boot Request (1)
    Hardware type: Ethernet (0x01)
    Hardware address length: 6
    Hops: 0
    Transaction ID: 0x0bb90957
    Seconds elapsed: 0
  ▷ Bootp flags: 0x0000 (Unicast)
    Client IP address: 192.168.0.7 (192.168.0.7)
    Your (client) IP address: 0.0.0.0 (0.0.0.0)
    Next server IP address: 0.0.0.0 (0.0.0.0)
    Relay agent IP address: 0.0.0.0 (0.0.0.0)
    Client MAC address: CadmusCo_d3:fd:ae (08:00:27:d3:fd:ae)
    Client hardware address padding: 00000000000000000000
    Server host name not given
    Boot file name not given
    Magic cookie: DHCP
  ▽ Option: (53) DHCP Message Type
      Length: 1
      DHCP: Request (3)
  ▽ Option: (60) Vendor class identifier
      Length: 5
```



```
      Vendor class identifier:  \024!)H
  ▷ Option: (55) Parameter Request List
  ▽ Option: (255) End
      Option End: 255
    Padding

0110  00 00 00 00 00 00 63 82  53 63 35 01 03 3c 05 20   ......c. Sc5..<.
0120  14 21 29 48 37 0c 01 0f  03 06 2c 2e 2f 1f 21 79   .!)H7... ..,./.!y
0130  f9 2b ff 00 00 00 00 00  00 00 00 00 00 00 00 00   .+...... ........
0140  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
```

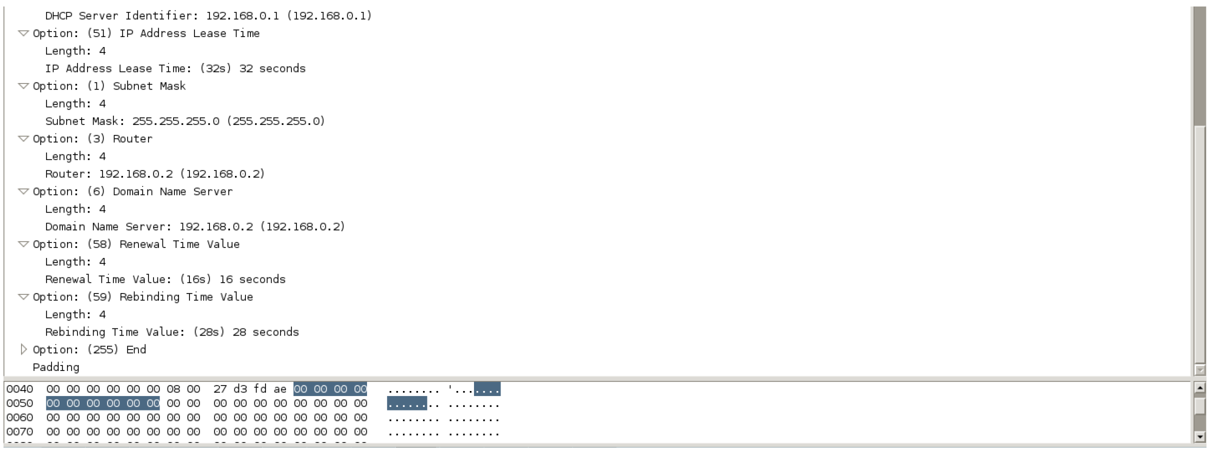
ACK(Unicast):



Capturing from eth1 [Wireshark 1.10.6 (v1.10.6 from master-1.10)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: bootp
Expression... Clear Apply Save Filter

```
o.  Time           Source        Destination   Protocol Length Info
181 39984.25595: 192.168.0.1   192.168.0.7   DHCP     354 DHCP ACK      - Transaction ID 0xbb90957
```

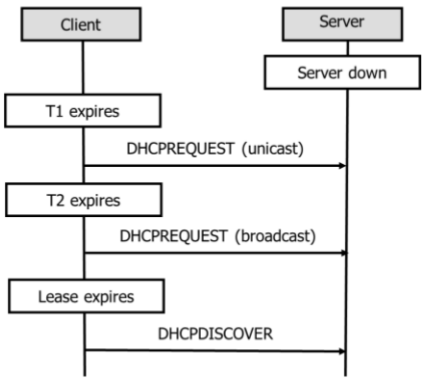
```
▸ Frame 181: 354 bytes on wire (2832 bits), 354 bytes captured (2832 bits) on interface 0
▷ Ethernet II, Src: CadmusCo_d6:8e:66 (08:00:27:d6:8e:66), Dst: CadmusCo_d3:fd:ae (08:00:27:d3:fd:ae)
▷ Internet Protocol Version 4, Src: 192.168.0.1 (192.168.0.1), Dst: 192.168.0.7 (192.168.0.7)
▷ User Datagram Protocol, Src Port: bootps (67), Dst Port: bootpc (68)
▽ Bootstrap Protocol
    Message type: Boot Reply (2)
    Hardware type: Ethernet (0x01)
    Hardware address length: 6
    Hops: 0
    Transaction ID: 0x0bb90957
    Seconds elapsed: 0
  ▷ Bootp flags: 0x0000 (Unicast)
    Client IP address: 192.168.0.7 (192.168.0.7)
    Your (client) IP address: 192.168.0.7 (192.168.0.7)
    Next server IP address: 0.0.0.0 (0.0.0.0)
    Relay agent IP address: 0.0.0.0 (0.0.0.0)
    Client MAC address: CadmusCo_d3:fd:ae (08:00:27:d3:fd:ae)
    Client hardware address padding: 00000000000000000000
    Server host name not given
    Boot file name not given
    Magic cookie: DHCP
  ▽ Option: (53) DHCP Message Type
      Length: 1
      DHCP: ACK (5)
  ▽ Option: (54) DHCP Server Identifier
      Length: 4
```



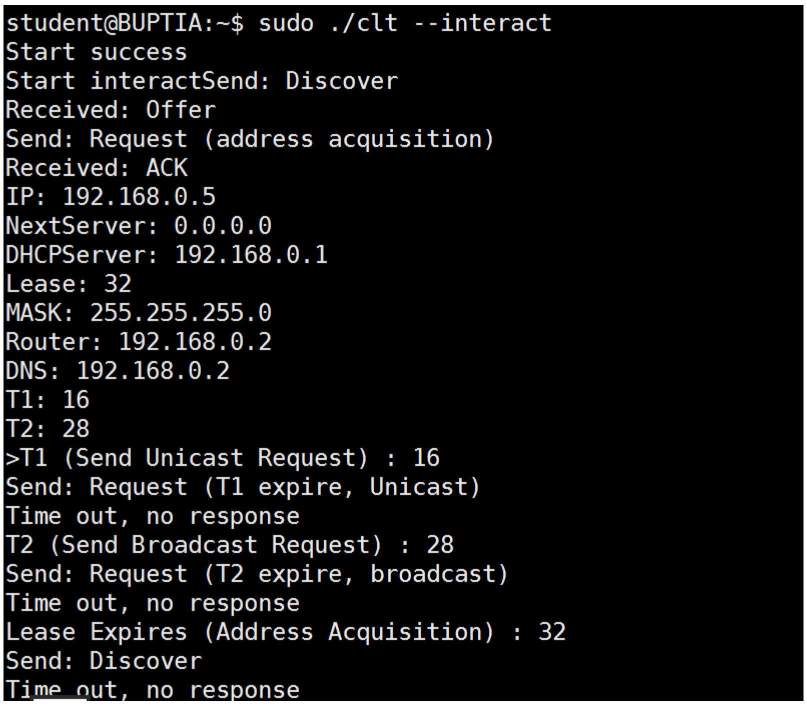
```
      DHCP Server Identifier: 192.168.0.1 (192.168.0.1)
  ▽ Option: (51) IP Address Lease Time
      Length: 4
      IP Address Lease Time: (32s) 32 seconds
  ▽ Option: (1) Subnet Mask
      Length: 4
      Subnet Mask: 255.255.255.0 (255.255.255.0)
  ▽ Option: (3) Router
      Length: 4
      Router: 192.168.0.2 (192.168.0.2)
  ▽ Option: (6) Domain Name Server
      Length: 4
      Domain Name Server: 192.168.0.2 (192.168.0.2)
  ▽ Option: (58) Renewal Time Value
      Length: 4
      Renewal Time Value: (16s) 16 seconds
  ▽ Option: (59) Rebinding Time Value
      Length: 4
      Rebinding Time Value: (28s) 28 seconds
  ▷ Option: (255) End
    Padding
```

```
0040  00 00 00 00 00 00 08 00  27 d3 fd ae 00 00 00 00   ........ '.......
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   ........ ........
```

If the client get the IP but server shuts down, then in the T1 expire it will unicast request, then T2 expire it will broadcast request, after lease time it will broadcast discover.



Client:

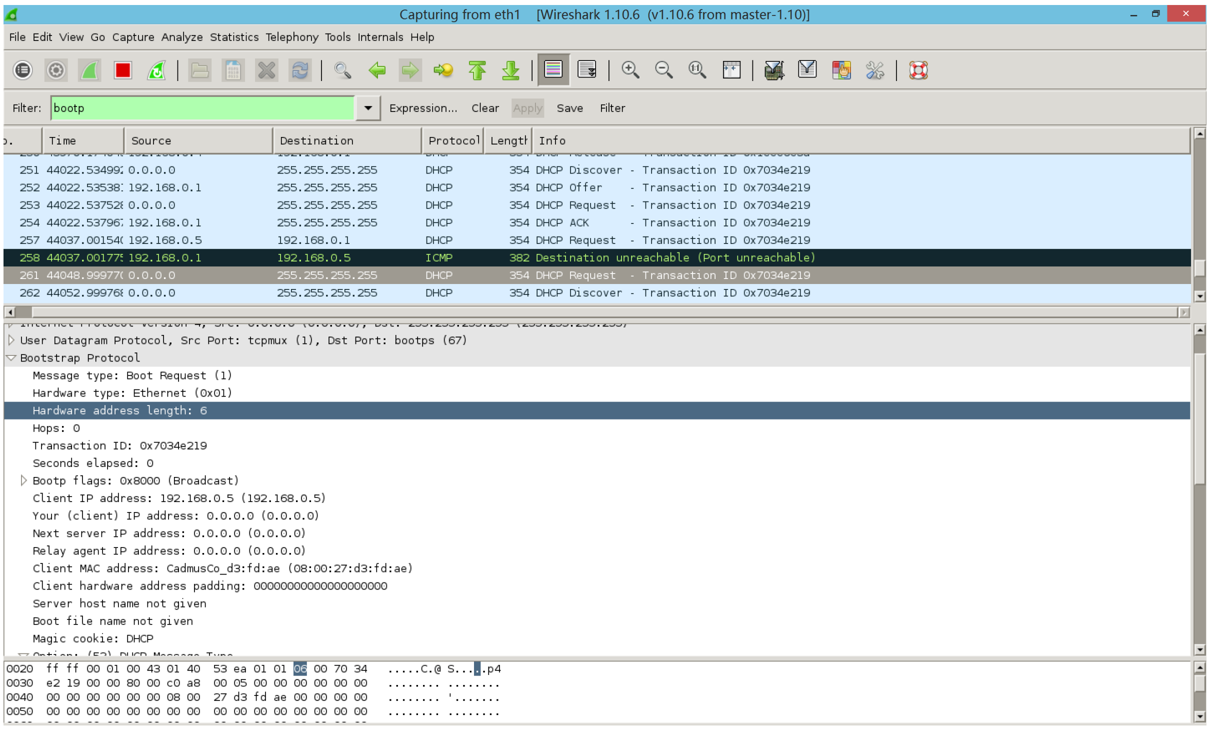


```
student@BUPTIA:~$ sudo ./clt --interact
Start success
Start interactSend: Discover
Received: Offer
Send: Request (address acquisition)
Received: ACK
IP: 192.168.0.5
NextServer: 0.0.0.0
DHCPServer: 192.168.0.1
Lease: 32
MASK: 255.255.255.0
Router: 192.168.0.2
DNS: 192.168.0.2
T1: 16
T2: 28
>T1 (Send Unicast Request) : 16
Send: Request (T1 expire, Unicast)
Time out, no response
T2 (Send Broadcast Request) : 28
Send: Request (T2 expire, broadcast)
Time out, no response
Lease Expires (Address Acquisition) : 32
Send: Discover
Time out, no response
```

Packet process:

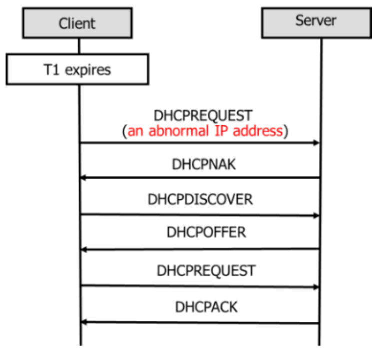
Discover->Offer->Request->ACK->Request(unicast)->

Request(broadcast)->discover.

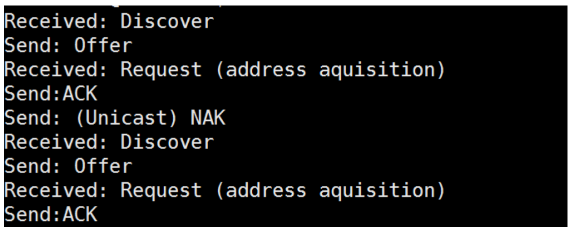

Capturing from eth1 [Wireshark 1.10.6 (v1.10.6 from master-1.10)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Expression... Clear Apply Save Filter

```
Filter: bootp

   Time          Source        Destination      Protocol Length Info
251 44022.53499; 0.0.0.0       255.255.255.255  DHCP     354 DHCP Discover - Transaction ID 0x7034e219
252 44022.53538: 192.168.0.1   255.255.255.255  DHCP     354 DHCP Offer    - Transaction ID 0x7034e219
253 44022.53752( 0.0.0.0       255.255.255.255  DHCP     354 DHCP Request  - Transaction ID 0x7034e219
254 44022.53796: 192.168.0.1   255.255.255.255  DHCP     354 DHCP ACK      - Transaction ID 0x7034e219
257 44037.00154( 192.168.0.5   192.168.0.1      DHCP     354 DHCP Request  - Transaction ID 0x7034e219
258 44037.00177! 192.168.0.1   192.168.0.5      ICMP     382 Destination unreachable (Port unreachable)
261 44048.99977( 0.0.0.0       255.255.255.255  DHCP     354 DHCP Request  - Transaction ID 0x7034e219
262 44052.99976( 0.0.0.0       255.255.255.255  DHCP     354 DHCP Discover - Transaction ID 0x7034e219

▷ User Datagram Protocol, Src Port: tcpmux (1), Dst Port: bootps (67)
▽ Bootstrap Protocol
    Message type: Boot Request (1)
    Hardware type: Ethernet (0x01)
    Hardware address length: 6
    Hops: 0
    Transaction ID: 0x7034e219
    Seconds elapsed: 0
  ▷ Bootp flags: 0x8000 (Broadcast)
    Client IP address: 192.168.0.5 (192.168.0.5)
    Your (client) IP address: 0.0.0.0 (0.0.0.0)
    Next server IP address: 0.0.0.0 (0.0.0.0)
    Relay agent IP address: 0.0.0.0 (0.0.0.0)
    Client MAC address: CadmusCo_d3:fd:ae (08:00:27:d3:fd:ae)
    Client hardware address padding: 00000000000000000000
    Server host name not given
    Boot file name not given
    Magic cookie: DHCP

0020  ff ff 00 01 00 43 01 40  53 ea 01 01 06 00 70 34   .....C.@ S.....p4
0030  e2 19 00 00 80 00 c0 a8  00 05 00 00 00 00 00 00   ........ ........
0040  00 00 00 00 00 00 08 00  27 d3 fd ae 00 00 00 00   ........ '.......
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
```

When lease time passed, the client wants to renew for an illegal IP, it will receive NAK, and the client will go to the discover process.

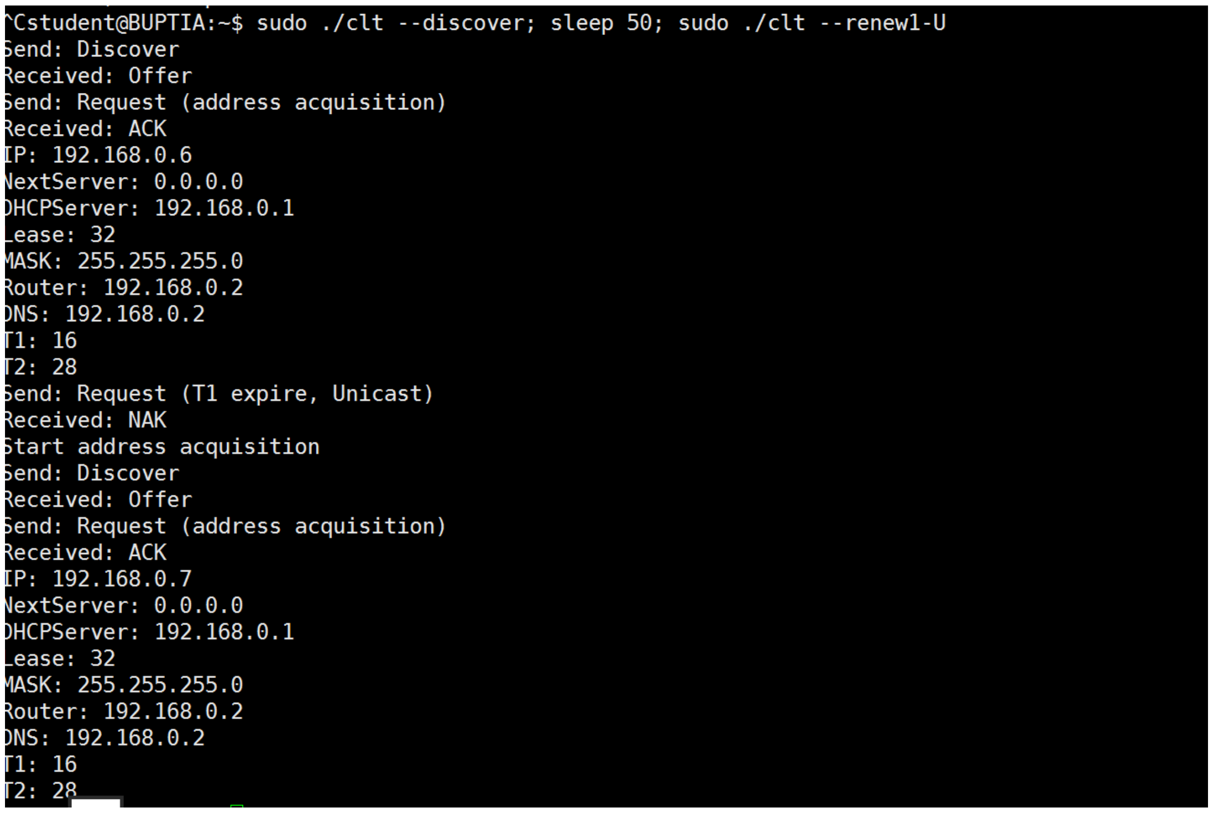


Server:



```
Received: Discover
Send: Offer
Received: Request (address aquisition)
Send:ACK
Send: (Unicast) NAK
Received: Discover
Send: Offer
Received: Request (address aquisition)
Send:ACK
```

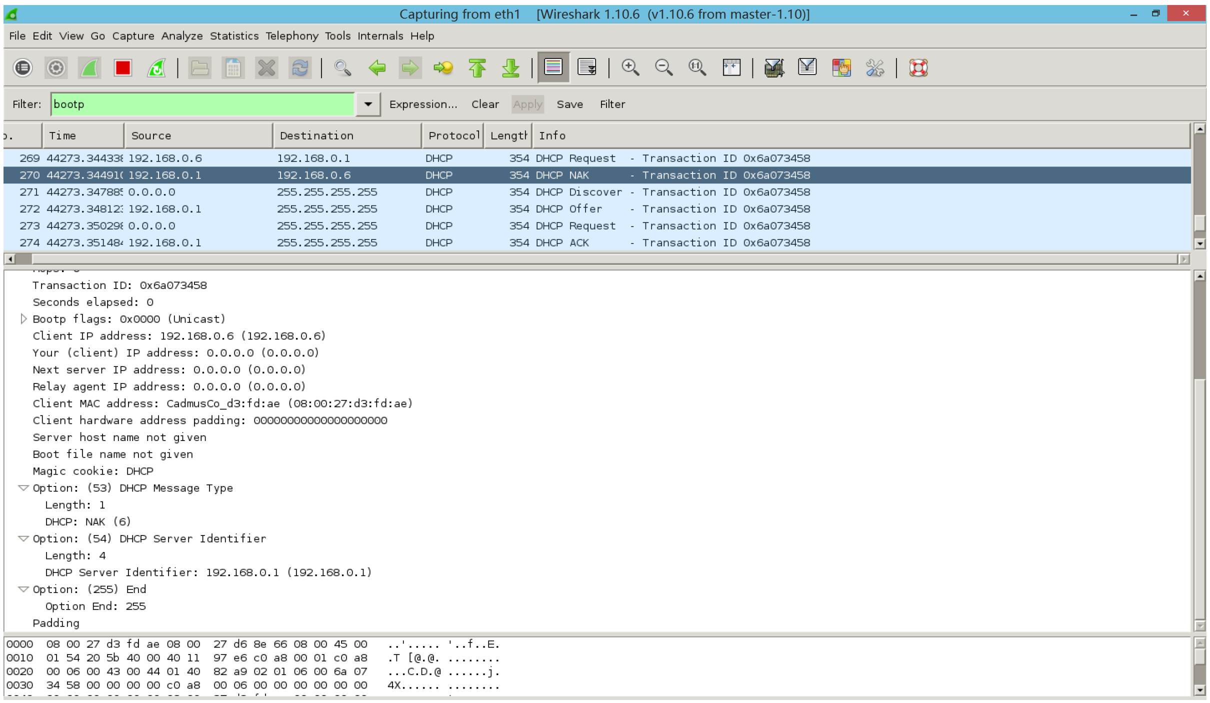
Client:



```
^Cstudent@BUPTIA:~$ sudo ./clt --discover; sleep 50; sudo ./clt --renew1-U
Send: Discover
Received: Offer
Send: Request (address acquisition)
Received: ACK
IP: 192.168.0.6
NextServer: 0.0.0.0
DHCPServer: 192.168.0.1
Lease: 32
MASK: 255.255.255.0
Router: 192.168.0.2
DNS: 192.168.0.2
T1: 16
T2: 28
Send: Request (T1 expire, Unicast)
Received: NAK
Start address acquisition
Send: Discover
Received: Offer
Send: Request (address acquisition)
Received: ACK
IP: 192.168.0.7
NextServer: 0.0.0.0
DHCPServer: 192.168.0.1
Lease: 32
MASK: 255.255.255.0
Router: 192.168.0.2
DNS: 192.168.0.2
T1: 16
T2: 28
```

Packet process:

Request(unicast)->NAK->Discover->Offer->Request->ACK. NAK packet:



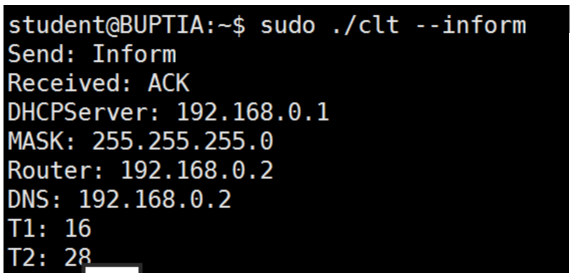
```
Capturing from eth1    [Wireshark 1.10.6  (v1.10.6 from master-1.10)]
Filter: bootp

o.    Time           Source         Destination       Protocol Length Info
  269 44273.34433{   192.168.0.6    192.168.0.1       DHCP        354 DHCP Request  - Transaction ID 0x6a073458
  270 44273.34491(   192.168.0.1    192.168.0.6       DHCP        354 DHCP NAK      - Transaction ID 0x6a073458
  271 44273.34788!   0.0.0.0        255.255.255.255   DHCP        354 DHCP Discover - Transaction ID 0x6a073458
  272 44273.34812:   192.168.0.1    255.255.255.255   DHCP        354 DHCP Offer    - Transaction ID 0x6a073458
  273 44273.35029{   0.0.0.0        255.255.255.255   DHCP        354 DHCP Request  - Transaction ID 0x6a073458
  274 44273.35148·   192.168.0.1    255.255.255.255   DHCP        354 DHCP ACK      - Transaction ID 0x6a073458

    Transaction ID: 0x6a073458
    Seconds elapsed: 0
  ▷ Bootp flags: 0x0000 (Unicast)
    Client IP address: 192.168.0.6 (192.168.0.6)
    Your (client) IP address: 0.0.0.0 (0.0.0.0)
    Next server IP address: 0.0.0.0 (0.0.0.0)
    Relay agent IP address: 0.0.0.0 (0.0.0.0)
    Client MAC address: CadmusCo_d3:fd:ae (08:00:27:d3:fd:ae)
    Client hardware address padding: 00000000000000000000
    Server host name not given
    Boot file name not given
    Magic cookie: DHCP
  ▽ Option: (53) DHCP Message Type
      Length: 1
      DHCP: NAK (6)
  ▽ Option: (54) DHCP Server Identifier
      Length: 4
      DHCP Server Identifier: 192.168.0.1 (192.168.0.1)
  ▽ Option: (255) End
      Option End: 255
    Padding

0000  08 00 27 d3 fd ae 08 00  27 d6 8e 66 08 00 45 00   ..'..... '..f..E.
0010  01 54 20 5b 40 00 40 11  97 e6 c0 a8 00 01 c0 a8   .T [@.@. ........
0020  00 06 00 43 00 44 01 40  82 a9 02 01 06 00 6a 07   ...C.D.@ ......j.
0030  34 58 00 00 00 00 c0 a8  00 06 00 00 00 00 00 00   4X...... ........
```

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Expression...
Clear
Apply
Save
Filter

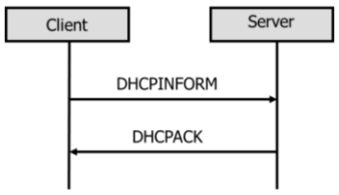
Inform:

After get an IP, client unicast inform and server sends ACK without option 51.

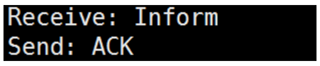
Client:



```
student@BUPTIA:~$ sudo ./clt --inform
Send: Inform
Received: ACK
DHCPServer: 192.168.0.1
MASK: 255.255.255.0
Router: 192.168.0.2
DNS: 192.168.0.2
T1: 16
T2: 28
```

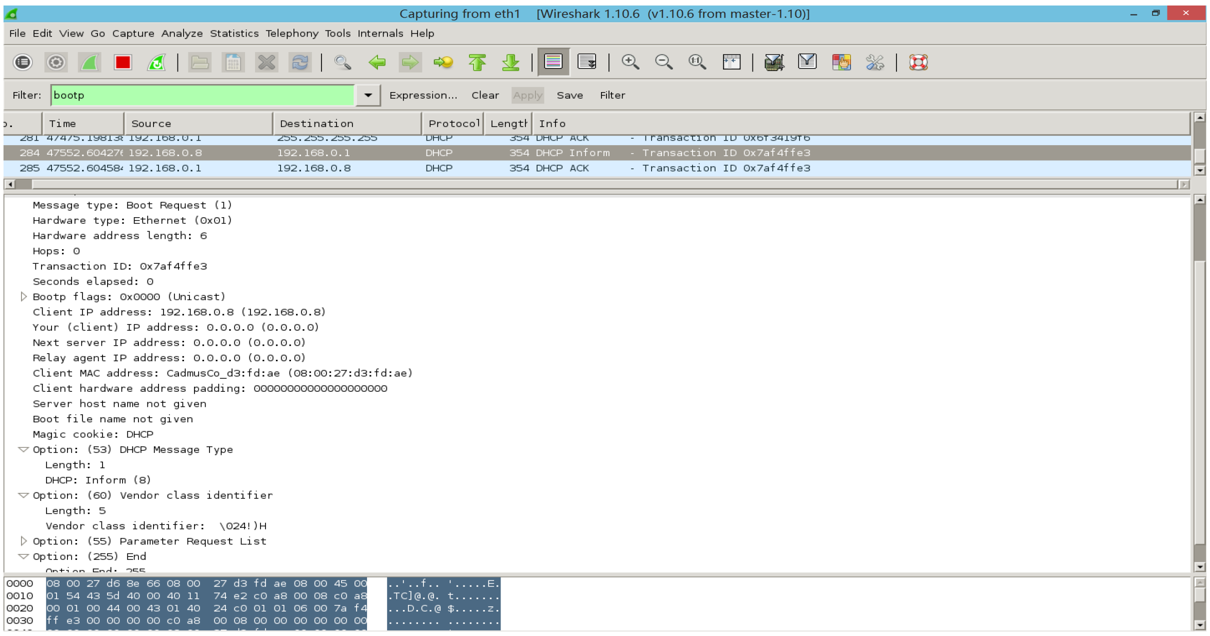


Server:



```
Receive: Inform
Send: ACK
```

Inform:

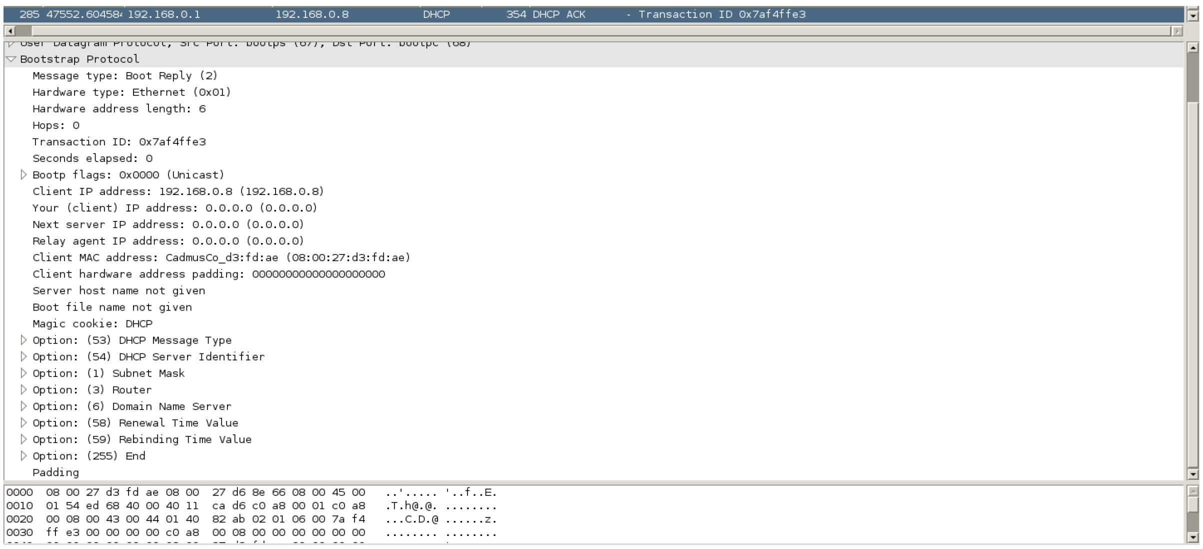

Capturing from eth1 [Wireshark 1.10.6 (v1.10.6 from master-1.10)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: bootp
Expression... Clear Apply Save Filter

```
o.   Time          Source        Destination       Protocol Length Info
 281 47475.19813   192.168.0.1   255.255.255.255   DHCP     354 DHCP ACK     - Transaction ID 0x6f3419f6
 284 47552.60427   192.168.0.8   192.168.0.1       DHCP     354 DHCP Inform  - Transaction ID 0x7af4ffe3
 285 47552.60458   192.168.0.1   192.168.0.8       DHCP     354 DHCP ACK     - Transaction ID 0x7af4ffe3
```

```
    Message type: Boot Request (1)
    Hardware type: Ethernet (0x01)
    Hardware address length: 6
    Hops: 0
    Transaction ID: 0x7af4ffe3
    Seconds elapsed: 0
  ▷ Bootp flags: 0x0000 (Unicast)
    Client IP address: 192.168.0.8 (192.168.0.8)
    Your (client) IP address: 0.0.0.0 (0.0.0.0)
    Next server IP address: 0.0.0.0 (0.0.0.0)
    Relay agent IP address: 0.0.0.0 (0.0.0.0)
    Client MAC address: CadmusCo_d3:fd:ae (08:00:27:d3:fd:ae)
    Client hardware address padding: 00000000000000000000
    Server host name not given
    Boot file name not given
    Magic cookie: DHCP
  ▽ Option: (53) DHCP Message Type
      Length: 1
      DHCP: Inform (8)
  ▽ Option: (60) Vendor class identifier
      Length: 5
      Vendor class identifier:  \024!)H
  ▷ Option: (55) Parameter Request List
  ▽ Option: (255) End
      Option End: 255
```

```
0000  08 00 27 d6 8e 66 08 00  27 d3 fd ae 08 00 45 00   ..'..f.. '.....E.
0010  01 54 43 5d 40 00 40 11  74 e2 c0 a8 00 08 c0 a8   .TC]@.@. t.......
0020  00 01 00 44 00 43 01 40  24 c0 01 01 06 00 7a f4   ...D.C.@ $.....z.
0030  ff e3 00 00 00 00 c0 a8  00 08 00 00 00 00 00 00   ........ ........
```

ACK(no option 51):

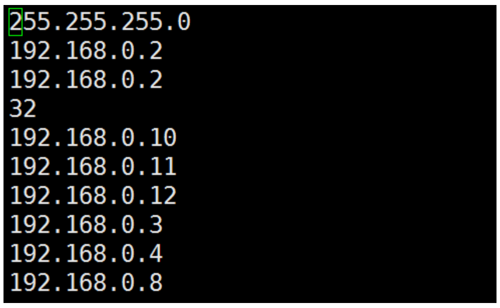


```
 285 47552.60458   192.168.0.1   192.168.0.8       DHCP     354 DHCP ACK     - Transaction ID 0x7af4ffe3
```

```
▷ User Datagram Protocol, Src Port: bootps (67), Dst Port: bootpc (68)
▽ Bootstrap Protocol
    Message type: Boot Reply (2)
    Hardware type: Ethernet (0x01)
    Hardware address length: 6
    Hops: 0
    Transaction ID: 0x7af4ffe3
    Seconds elapsed: 0
  ▷ Bootp flags: 0x0000 (Unicast)
    Client IP address: 192.168.0.8 (192.168.0.8)
    Your (client) IP address: 0.0.0.0 (0.0.0.0)
    Next server IP address: 0.0.0.0 (0.0.0.0)
    Relay agent IP address: 0.0.0.0 (0.0.0.0)
    Client MAC address: CadmusCo_d3:fd:ae (08:00:27:d3:fd:ae)
    Client hardware address padding: 00000000000000000000
    Server host name not given
    Boot file name not given
    Magic cookie: DHCP
  ▷ Option: (53) DHCP Message Type
  ▷ Option: (54) DHCP Server Identifier
  ▷ Option: (1) Subnet Mask
  ▷ Option: (3) Router
  ▷ Option: (6) Domain Name Server
  ▷ Option: (58) Renewal Time Value
  ▷ Option: (59) Rebinding Time Value
  ▷ Option: (255) End
    Padding
```

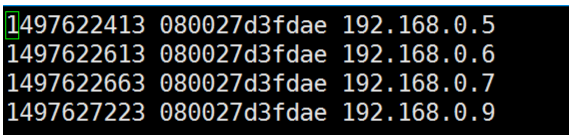
```
0000  08 00 27 d3 fd ae 08 00  27 d6 8e 66 08 00 45 00   ..'..... '..f..E.
0010  01 54 ed 68 40 00 40 11  ca d6 c0 a8 00 01 c0 a8   .T.h@.@. ........
0020  00 08 00 43 00 44 01 40  82 ab 02 01 06 00 7a f4   ...C.D.@ ......z.
0030  ff e3 00 00 00 00 c0 a8  00 08 00 00 00 00 00 00   ........ ........
```

dhcp.config



```
255.255.255.0
192.168.0.2
192.168.0.2
32
192.168.0.10
192.168.0.11
192.168.0.12
192.168.0.3
192.168.0.4
192.168.0.8
```

dhcp.lease



```
1497622413 080027d3fdae 192.168.0.5
1497622613 080027d3fdae 192.168.0.6
1497622663 080027d3fdae 192.168.0.7
1497627223 080027d3fdae 192.168.0.9
```

6. User Manual

Please configure the dhcp.config file in the format of:

Submask

Router IP

DNS IP

Lease time

IPs

Use “sudo ./client --interact” to run the whole process and input renew to manually renew.

Use “sudo ./client --discover” to get new IP.

Use “sudo ./client --inform” to inform.

Use “sudo ./client –renew1-U” to manually renew.

Use “sudo ./client --release” to release.