AJAYKRISHNAN JAYAGOPAL
CS14B033

Lab3: Configuring Interfaces and Wireshark usage

24th January 2017

QUESTION 2.2

NAT stands for Network Address Translation. It is a method used to remap one IP address space into another by directly modifying the network address information in the IP datagram before transmission.
The two machines used are: A and B.

For A, the gateway, the following commands were used:

```
$ sudo ifconfig eth0 192.168.144.100 netmask 255.255.255.0
```

This command sets the ip address associated with the ethernet link as 192.168.144.100. Since the netmask is 255.255.255.0, the last 8 bits of the IP address serve to identify the host whereas the first 24 bits identify the private subnet.

```
$ sudo echo 1 > /proc/sys/net/ipv4/ip_forward
```

This is used to enable ip forwarding on the machine A. This is done so that IP requests received by A as a gateway, from B are forwarded to the external network.

```
$ sudo /sbin/iptables -t nat -A POSTROUTING -o wlan0 -j
MASQUERADE
```

The -t option allows us to choose a table, here "nat". The "nat" table is consulted as soon a packet that creates a new connection is encountered.

The -A option allows us to "append" a rule to the "nat" table. The rule specified is "POSTROUTING", which is for altering packets as they are about to go out.

The -o option allows to specify the out-interface, ie., the interface via which we forward the packet. We choose wlan0 as the out-interface since it is the interface via which A is connected to the external network.

The -j option (jump) specifies the action that must be performed on the matching packets. Here the action specified is MASQUERADE. This means that all packets forwarded to the external

network through this gateway will appear to be coming from the gateway itself (hence, masquerading like the gateway).

```
$ sudo /sbin/iptables -A FORWARD -i wlan0 -o eth0 -m state
--state RELATED,ESTABLISHED -j ACCEPT
```

This command sets up the iptables such that all packets received from the external network on wlan0 are also forwarded to eth0 which is where B is connected.

Since no -t option is specified, the table selected is the "filter" table that has the built-in chains INPUT, FORWARD and OUTPUT. Among these, the chain FORWARD is appended with the -A option.

-i allows to specify the in-interface, ie., the interface from which we receive the packet. We choose wlan0 as the in-interface since it is the interface via which A is connected to the external network.

-o allows to specify the out-interface, ie., the interface via which we forward the packet. We choose wlan0 as the out-interface since it is the interface via which A is physically connected to B.

-m (match) allows to specify an extension module to use, in this case "state"

-j specifies the action, in this case, ACCEPT the packet

```
$ sudo /sbin/iptables -A FORWARD -i eth0 -o wlan0 -j ACCEPT
```

This command sets up the iptables to forward all packets it receives from eth0 interface(private subnet) to the wlan0 interface(external network).

For B, the node in the private network, the following commands were used:

```
$ sudo ifconfig eth0 192.168.144.200 netmask 255.255.255.0
```

This command sets the ip address associated with the ethernet link as 192.168.144.200. Since the netmask is 255.255.255.0, the last 8 bits of the IP address serve to identify the host whereas the first 24 bits identify the private subnet.

```
$ sudo route add default gw 192.168.144.100
```

This command sets the default gateway for internet access to 192.168.144.100, which is the ethernet ip address of A.

```
$ The file /etc/resolv.conf was modified to look like the
following:
nameserver 10.6.0.11
```

This sets up B to use 10.6.0.11 as the DNS server.

QUESTION 3.1

1. The transport layer protocol used for the DNS(Domain Name System) query is **User Datagram Protocol (UDP)**.
The DNS response for the ip address of the domain www.cse.iitm.ac.in yields the IP address as 10.6.8.2

2. ARP - Address Resolution Protocol
My mac address: 4c:bb:58:13:73:f0
ARP request for 10.6.8.2 was not showing up on wireshark. A sample ARP response looks like the following:

```
ARP response: Source : "ArubaNet_c8:a9:65"  Data : "172.31.98.1 is at 9c:1c:12:c8:a9:65"
```

3. ICMP echo and reply packets

Request packet

```
Type: 8 (Echo (ping) request)
Data (48 bytes) :
35:07:02:00:00:00:00:00:10:11:12:13:14:15:16:17:18:19:1a:1b:1c:1d:1e:1f:20:21:22:23:24:25:26:2
7:28:29:2a:2b:2c:2d:2e:2f:30:31:32:33:34:35:36:37
```

Response packet

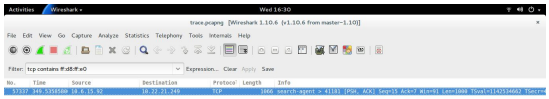
```
Type: 0 (Echo (ping) reply)
Data (48 bytes)
35:07:02:00:00:00:00:00:10:11:12:13:14:15:16:17:18:19:1a:1b:1c:1d:1e:1f:20:21:22:23:24:25:26:2
7:28:29:2a:2b:2c:2d:2e:2f:30:31:32:33:34:35:36:37
```

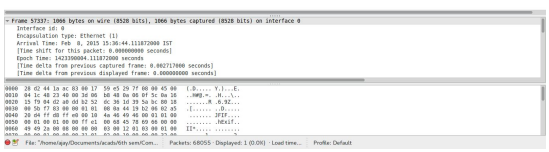
4. The website www.google.com/loon/ uses HTTPS to secure communications. Hence, all packets sent between the client and the server are encrypted. Hence, the packets captured by wireshark cannot be filtered to view only the GET requests for images since the all that data is encrypted. However, since the browser(which makes the request in the first place) has the private key, we can view the the response in the browser console itself. The results as obtained from the browser console are as follows:
The data in the user agent field of the HTTP GET request for images was:

```
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:50.0) Gecko/20100101 Firefox/50.0\r\n
```

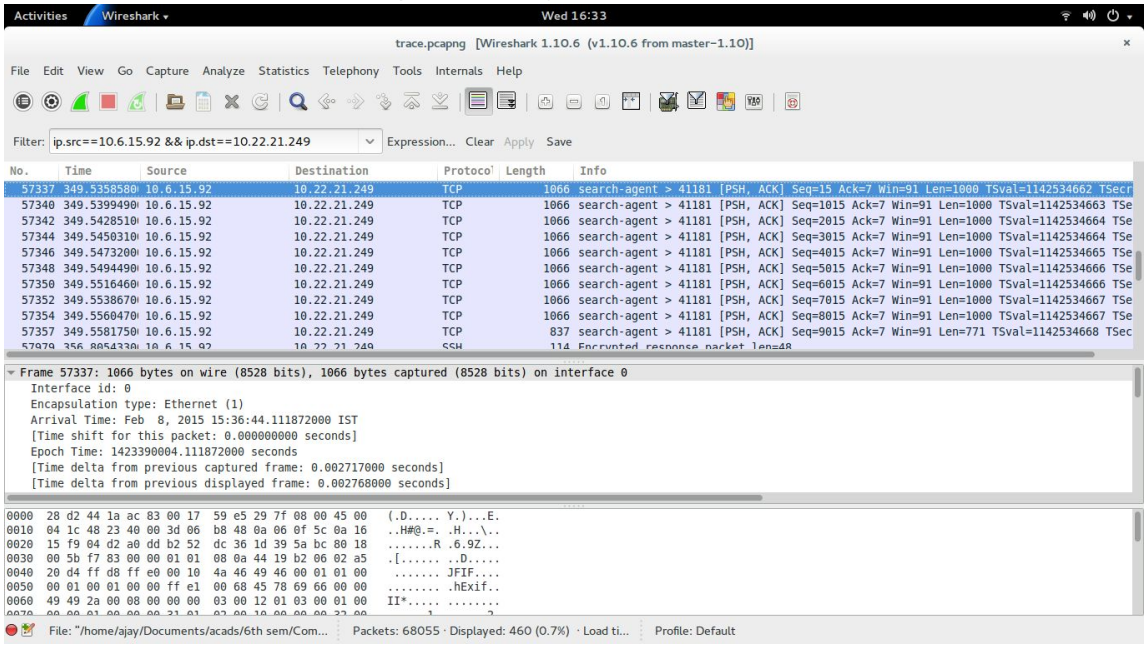
5. The other filters tried out:
   - contains
     - eg. udp contains “Bob”, tcp contains ff:f8:ff:e0

     This helped to locate the conversation and the start of the file respectively in question 3.2





- ip.src, ip.dst, ip.addr
    - Used for filtering packets by source(ip.src), destination (ip.dst) or either(ip.addr)



QUESTION 3.2

1. X : Bob (IP = 10.22.21.249)
   Y : Abhik (IP = 10.6.15.92)

   First message:
   Src = X
   Dst = Y
   Data: "Hi Abhik!"

   Last message:
   Src = X
   Dst = Y
   Data: ":)"

The full conversation was as follows:

```
Bob: Hi Abhik!
Abhik: Hello Bob. What's up?
Bob: Nothing much. Doing some research. What about u?
Abhik: Same here. I'm doing some TA work da. These juniors are N painful.
Bob: Hahahahaha, give them some hard assignments and pain them :P
Abhik: Poor guys da. I'm not like Piney!
Or maybe I am! :P
Bob: Oh, and btw, I've been playing this new awesome game. It's
brilliant!
Abhik: Oh nice. What is it?
Bob: And I heard these juniors implemented an FTP Se
rver and client
```

```
I'm curious to try it out!
Abhik: Hmmm, okay
Bob: I'm sending you a file
Abhik: Okay
Got the file!
Oh! I see! So that's the game you were talking about!
Yep, it's a pretty awesome. Should play it soon.
Abhik: :)
```

2. The image file was split into 20 packets. The file format was JPEG File Interchange Format (JFIF).
This was the image:

3. The game was Watch Dogs.