Computer Networks – Exercise 6

1 Traceroute: Theory

Consult the man page for traceroute (man traceroute from the Linux command line or on the web at http://linux.die.net/man/8/traceroute). Consider the following traceroute output:

```
[daves@pyxis]$ traceroute -n ethz.ch
traceroute to ethz.ch (129.132.128.139), 30 hops max, 60 byte packets
  1  192.33.93.1  0.822 ms  0.856 ms  0.893 ms
  2  18.10.1.81  1.222 ms  1.267 ms  1.265 ms
  3  18.1.17.242  1.252 ms  0.988 ms  1.117 ms
  4  192.33.92.185  1.615 ms  1.891 ms  1.778 ms
  5  * * *
  6  * * *
```

Figure 1: Traceroute output

1. What does each of the three measurements next to each IP address on lines 1–4 mean?
2. Give one possible reason why hops 5 and 6 show “* * *”

2 Traceroute: Practice

In the last exercise, you used traceroute to identify routers on the path to a given destination. In this exercise, you will use traceroute as a starting point to familiarize yourself with another useful tool: Wireshark, which is an open-source packet analyzer that enables you to inspect incoming and outgoing packets from any network interface of your machine. It can parse multiple networking protocols and display the relevant fields in its graphical user interface. As a starting point you will have to download and install Wireshark. We suggest to run it on Linux (possibly on a virtual machine).

After successfully starting Wireshark, you will have to start capturing traffic from your machine. You can select from which interfaces to capture traffic through the capture options. To ensure that Wireshark will display traffic from all interfaces, you can choose the “any” option. For more details on Wireshark usage and features please consult online tutorials, which are widely available. While capturing traffic, run the following command:

```
traceroute -M icmp mit.edu
```

Ideally, Wireshark has captured—among other traffic—packets that are related to your traceroute command.\(^1\)

---

\(^1\)We refer to the Linux traceroute utility. Utilities on other platforms may use different parameters to set the protocol.

\(^2\)If you run many applications that heavily exchange traffic, you may find it difficult to identify the relevant packets in the capture. To help in this direction, you can try to apply display filters.
1. Locate the “Echo (ping) request” packets. What are the source and destination IP addresses and to which entities do they correspond? For these packets, locate the TTL value in the IP header. What do you observe?

2. Locate the corresponding ICMP replies to the ping requests and compare the source IP addresses with the traceroute results. What do you notice? Why does each packet appear multiple times?

3. Capture again the traffic with the following traceroute command: `traceroute mit.edu`. Do you observe any difference in the traceroute results? What is the difference when you look at the captured packets? Is there a conceptual difference between the two traceroute commands? Try to briefly explain.

4. For both incoming and outgoing packets, try to locate MAC addresses. What addresses do you observe and to which entities do they correspond?

3 Packet Switching

In packet-switching networks, the source host segments long application-layer messages (for example images or music files) into smaller packets and sends the packets into the network. The receiver reassembles the packets back into the original message. Figure 2 illustrates the end-to-end transmission of a message with and without segmentation. Consider a $7.5 \times 10^6$ bits long message that is to be sent from the source to the destination as shown in the figure 2. Suppose that each link in the figure is 1.5 Mbps. Ignore propagation, queuing and processing delays.

Figure 2: Transmission (a) without message segmentation, and (b) with message segmentation.

1. Consider sending the message from source to destination without message segmentation. How long does it take to move the message from the source host to the first packet switch? Keep in mind that each packet switch uses a store and forward packet switching. What is the total time to move the message from the source host to the destination host?

2. Now suppose that the message is segmented into 5,000 packets, with each packet being 1500 bits long. How long does it take to move the first packet from the source host to the first packet switch?

3. How long does it take to move the file from the source host to the destination host when message segmentation is used? Compare and comment.

4. What are the drawbacks of message segmentation?

4 Network Address Translation (NAT)

Unlike the traditional client-server model, in peer-to-peer (P2P) applications, peers communicate with each other directly. In this question, we explore how network address translation (NAT) complicates P2P communication. When two peers both have public IP addresses, P2P communication works without a
hitch, as both peers are directly reachable. But in reality, most Internet users are behind a NAT device, and these devices can disrupt P2P communication. Despite NAT devices, we can enjoy P2P applications, such as BitTorrent, Skype, etc. We will explore here how peers behind NAT devices can still connect to each other for P2P applications.

For the following questions, assume that all communication is taking place over UDP. Also, assume that the NAT devices behave identically for UDP and TCP, i.e., they perform the same address and port translation:

1. Revisiting the traditional client-server model, Figure 3 illustrates a typical communication model that we see in the Internet, where a client is behind a NAT and a server has a public IP address. How can the client access the server? How can the server respond back to the client? Assume that the server listens for UDP packets over port 10,000. Please illustrate the information the NAT needs to maintain and how this information is used by the NAT.

![Figure 3: Client/Server Model](image)

2. Now let’s shift our discussion to P2P communication. As shown in Figure 4, both Peer A and B are behind NAT devices. Can peers communicate with each other? Explain. Provide a brief justification. Assume that peers know each other’s external IP addresses a priori.

![Figure 4: P2P Model](image)

3. Most P2P systems (including Skype) have special servers called rendezvous servers to assist Peer A to talk with Peer B as shown in Figure 5. Let’s consider a Skype call between the two peers, where Peer A (whose Skype ID is “OS”) calls Peer B (whose Skype ID is “NET”). How can this call be made? Please explain the steps that are necessary for Peer A, Peer B and the rendezvous server to make this call work. Furthermore, please explain the information that may be necessary at the two NATs and the rendezvous server. Assume that the rendezvous server listens for UDP packets over port 10,000. Hint: You may want to define application layer messages between the peers and the rendezvous server.

![Figure 5: P2P Model with Rendezvous Server](image)
5 Dijkstra’s Algorithm

A network between A and B is depicted in Figure 6. The numbers on the links correspond to the probability (as a percentage) that the link may fail. Link failures are independent from each other. How could you find the most reliable path from A to B? **Hint:** Use Dijkstra’s algorithm.

![Figure 6: Network Topology](image)

6 Distance-Vector Routing

In Figure 7, after the network stabilizes, the link connecting nodes C and D becomes broken. Show how nodes A, B and C might experience the count-to-infinity phenomenon for their distances to node D. In demonstrating the count-to-infinity problem, show at least two updates containing distance-to-D, for each of the nodes A, B and C. For each of these updates, specify the distance-to-D advertised to the neighbors. Will the problem be resolved after some iterations?

![Figure 7: Network Topology](image)