COMPSCI 314 S1T Assignment 1
2008

Department of Computer Science
The University of Auckland

Carefully review the tutorial document before starting the assignment. This assignment contributes to ***5%*** *of your overall course mark. Submit your assignment in* ***PDF*** *format to* ***Assignment Drop Box****. Include all* ***workings*** *and* ***explanations****. Marks will be deducted for ambiguous solutions. Zero marks are awarded if the answers contain no explanation. Also, refer to the Departmental Policy on Cheating on Assignments.*

Assignment Drop Box *(https://adb.ec.auckland.ac.nz/adb/).*
Departmental Policy on Cheating on Assignments *(http://www.cs.auckland.ac.nz/CheatingPolicy.php)*

Important: For the questions **Q1(b), Q2, Q3 and Q4(b)**, you must also attach one or two pages of your actual capture files for each question, as proof that you did the work, e.g., output screen-shots. You can use File/Export/File in Wireshark to save a text file.

[Total: 50 marks]

Q1. Packet capturing [10 marks]

[Mark allocation: 2, 3, 3, 2]

Go to Capture Options in Wireshark, and check '*Limit each packet to…*'.

a) What is the smallest and largest size limit you can set?
Smallest: 68 bytes
Largest: 65535 bytes

b) Explain why Wireshark imposes a lower and higher limit.
Low limit: capturing fewer than 68 bytes per packet could lead to a loss of information, e.g., TCP optional headers. Because most packets contain TCP segments, often the optimistic minimum size could only be 54 bytes, i.e., frame header (14 bytes) + IP header (20 bytes) + TCP header (20 bytes) = 54 bytes. However, both IP and TCP headers are then the minimum size; they can both have optional fields associated. So Wireshark moderately assumes 68 bytes is a reasonable length. *(In windump, you can even set lower to 1byte*!)
High limit: IP packets or IP routers cannot handle any packet size more than 64kB. So it is unnecessary to assume the packets are larger than 64kB. (Also, IP packets are usually limited by the most widely used Ethernet frame size, i.e., 1500 payload bytes)

c) Explain one advantage and one disadvantage of setting a small size limit.
Advantage: System workload is reduced; it only requires reading a given number of bytes for each packet before dumping into memory/trace file. Also, the trace file size is significantly smaller than if we captured every byte.

Disadvantage: we cannot understand the application data; since they are truncated, there is no way to reconstruct them. The minimum 68 bytes may only capture the first few bytes of application layer data.

d) Start capturing packets and stay *idle* for a few minutes. Then stop the capture. What kind of protocols do you see? List and explain at least three of them.
(Hint: See list of protocol abbreviations in Wireshark Help/Supported Protocols)
- OSPFv2 - routing protocol
- NBSS (NET-BIOS) - Microsoft's local network protocol
- ARP - How to find the Ethernet address for an IP address

Q2. IP packet size distributions [10 marks]

[Mark allocation: 5, 5]

Use a web browser to visit a few HTTP web pages, e.g., www.cs.auckland.ac.nz. Capture the full packets that are travelling between you and the web page. You only need to visit once to capture all the packets. Make sure to avoid HTTP status code *304*; this is most likely to happen if you are refreshing the web page.

a) List at least two frequently observed packet sizes, e.g. a small size and a large size.
~60 bytes
~1514 bytes
(note: any approx. sizes are okay, Q3a is more detailed)

b) Explain why these two different sizes exist. What is the difference in contents of the two sizes of packet?
Small packet sizes (~60 bytes) show that they contain no application data; indicating that these are most likely acknowledgement packets.
Bigger packet sizes (~1514 bytes) most likely contain the application data, i.e., the actual user-data traffic.

Q3. Packet trace file [20 marks]

[Mark allocation: 5, 10, 2, 3]

Use a web browser to download two files `BigFile1` and `BigFile2` from the 314 webpage *Assignment* section. Make sure to capture only packets belonging to the file download (i.e. set a host IP filter). Save the packets separately into a capture file for each download.
(Hint: See list of Packet length in Wireshark Statistic Menu)

a) Assuming the two files are the *user-data*, and observing from the switch level, list and explain the overheads (extra bytes per **user-datagram**) involved in layer-4, layer-3 and layer-2 (TCP, IP and Ethernet).
Assuming the optimistic case, we can see that each user-data packet is encapsulated by three lower layers: TCP (L4), IP (L3) and Ethernet (L2).
L4: requires adding 20 bytes per user-data
L3: requires adding 20 bytes per TCP segment

L2: requires adding 18 bytes per IP packet (including 4 bytes of Frame Checksum Sequence that are truncated before pcap captures them)
Thus, we can say that the overhead = 58 bytes per user-datagram

b) Based on your answer (a) and the packet size seen by Wireshark, calculate the efficiency *ratio* of your user-data to data transmitted by the switch. In other words, approximately what percentage of the total transmission should be user-data?
Efficiency ratio per packet = user-data (user-datagram size) / total-data (frame size)
Based on answer (a), we find that the network switch would require an additional 58 bytes per user-data for the transmission. Furthermore,

- For a simple ACK packet, they do not have user-data, overhead ratio = 100%
- For a user-datagram, overhead ratio = (1518-58) / 1518 = 96.2%

Total overhead seen by the network switch (**BigFile1**):
Overhead for the number of ACK packets = 1580 x 58 = 91640 bytes
Overhead for the number of user-data packets = 3523 x 58 = 204334 bytes
= 91640 + 204334 = 295974 bytes
User-data = 3523 x 1460 = 5143580 bytes
Thus, 5143580 / (295974+5143580) = 94.56%

Total overhead seen by the network switch (**BigFile2**):
Overhead for the number of ACK packets = 5696 x 58 = 330368 bytes
Overhead for the number of user-data packets = 14224 x 58 = 824992 bytes
= 330368 + 824992 = 1155360 bytes
User-data = 14224 x 1460 = 20767040 bytes
Thus, 20767040 / (1155360 +20767040) = 94.73%

Further remark:

- Above example simply used small and large packet sizes. There are however, a few 'medium' user-data packets (e.g., ~600 bytes), especially when an application data stream ends.
- We used minimum 58 bytes, but often each start of TCP stream contains optional header specifying MSS.
- The network switch transmitted approximately 5.4% more data, e.g., if we are using a switch rated at 100Mbp/s, can we transmit 12.5MB of a file in 1s?
- It is possible to simply obtain Wireshark's Summary (total bytes): for example, BigFile1 / (Summary_Bytes + (#FCS_Bytes)). Wireshark's 'saved-file' should not be used here as it contains its own overheads (see Q3d) that are not part of the transmission.

c) Based on your answer (b), comment on the overhead effects for transmitting small versus large packet size. Which is more efficient?

- The *larger* the packet size, the *smaller* the overhead (or *smaller* the packet size, the *larger* the overhead)
- Further remark – 'tradeoff': if the large packet size is lost, then we are 'wasting' that amount of the available bandwidth. In such cases, small packet size is better. To work

out the best possible packet size, you need to know the probabilities of lost/corrupted packets too.

d) Comment on the size of the actual downloaded files (BigFile1 and BigFile2) compared with the total sizes indicated by your captured files. Why are they different?
   - (transmission overheads/etc are explained already in (a), (b))
   - The trace file itself contains pcap header information as well as the actual packets.
   - Per-packet timestamp used by pcap: each captured packet is marked with pcap specific timestamp (e.g., 32bit timestamp per packet).
   - The packets transmitted in either direction that may have been lost on the LAN are still recorded, i.e., they do not contribute to the downloaded file size, but only occur in Wireshark's record.

Q4. Transport layer [10 marks]
[Mark allocation: 5, 5]
TCP and UDP are the most widely used protocols in the transport layer. However, they are very different. For example, TCP provides end-to-end reliable transmission, while UDP does not. Often, TCP carries HTTP traffic and together they often contribute more than 90% of total volume in our network.

a) Assuming that all of your HTTP traffic is carried by TCP; would you be able to visit web pages such as www.google.com if a network administrator blocks all UDP traffic? Justify your answer.
No: you are unlikely to reach the web sites, since the web browser would request DNS name-to-IP lookup (which is queried using UDP packets). But, yes: If you visited a web site shortly before the UDP blockage, the browser would likely have access to a local DNS cache (along with an appropriate TTL record), so a repeat DNS lookup would not be needed. If you happened to know the IP address of the web site, you could also enter that directly in the browser.

b) Often we use the term *TCP stream*, or simply a *connection* to represent data transmission between two end points. Technically, there can be many connections at the same time. List the fields/attributes used to identify a *single* stream. In other words, what makes a series of packet transmissions belong to the same connection?
(Hint: In the Wireshark capture window, right-click on a packet and use the 'Follow TCP stream' option.)
   - Source IP address
   - Destination IP address
   - Source Port number
   - Destination Port number
   - Protocol number

If any of the five attributes are different, then the packet belonged to a different stream/connection. Note that these are bi-directional, i.e., source and destination fields can be 'swapped' and still be regarded as the same stream/connection.