COMP90007 Internet Technology

Week2

Yiran (Scott) Ruan

Email: yrrua@unimelb.edu.au

*Copy from Yang Lu

Welcome to COMP90007

- A little bit about myself…
  - Master of Information Technology (Computing)
  - Second Year
  - Casual teaching: Internet; Algorithm
  - Timetable:

Tue	10.00AM-11.00AM	PAR-207-221 Bouverie St-B132
Tue	11.00AM-12.00AM	PAR-207-221 Bouverie St-B116
Thu	1.00PM - 2.00PM	PAR-207-221 Bouverie St-B113
Thu	3.15PM - 4.15AM	PAR-207-221 Bouverie St-B113

  - Use the Discussion Board on the LMS

    Also welcome to send Email: yrrua@unimelb.edu.au

Lab - Wireshark

- To examine the concept of **encapsulation** in networking and the overheads associated with the layered networking model.
- To learn how **protocols and layering** are represented in packets through the use of Wireshark.
- To **examine IP** (Internet Protocol) in detail. IP is the main network layer protocol used throughout the Internet.
- To **examine TCP** (Transmission Control Protocol) in detail. TCP is the main transport layer protocol used in the Internet.
- To **examine HTTP** (HyperText Transfer Protocol) in detail. HTTP is the main application layer protocol underlying the Web.

Requirements

- Two software requirements for this lab
  - ➢Wireshark
  - – A sniffing tool to examine a packet trace.
  - – Download Link: http://www.wireshark.org/download. html.

  - ➢Wget
  - – goal: Fetch web resources
  - – Install: Windows: https://eternallybored.org/misc/wget/
    Linux or OS X: Terminal -> ‘curl’ / ‘wget’ preinstalled

Procedures

- 1. CLOSE all applications may interfere the test (browsers)
- 2. configuring Wireshark
  - Capture options
    - Untick ‘use promiscuous mode’
    - Fill ‘tcp port http’
    - Resolving name
    - Choose interface
- 3. Go to wget and type command: wget www.wikidot.com
- 4. start wireshark
- 5. enter the command
- 6. stop wireshark
- Hint: if the trace list starts with 3-way hand shaking and 4. ‘Http Get Request’, then the result is fine for further study

Question 1

- What is the source IP address of your computer? What is the destination IP address of the web page you requested?

Solution 1

- Several ways to know this, for instance, you can use *ipconfig* on the command line (on Windows) to get your IP address;
  - In terms of Linux or OS X: Terminal —> ifconfig
  - For OS X: "Network" in the System Preferences
- Alternatively, from Wireshark, determine that it is your computer sending the HTTP GET request to request a web page from the remote server.
  - The source IP address will depend on your computer.
  - The destination IP address for the unimelb.edu.au web server is 172.25.128.1

Question 2

- How much traffic in bytes was received and transmitted in your request for the Unimelb web page?
- What is the percentage of traffic that originated from your computer, and what is the percentage of traffic that was sent by the remote host?

Solution 2

- The amount of trace captured will again depend on the network conditions, whether some packets were lost and needed to be retransmitted, whether some background network trace was also captured, etc.
- A handy tip is to use Excel to filter the rows by the Source IP address, so summing the length column is much easier. In general, the receiving trace may vary between 90% to 97% of the total trace, the remaining is the trace to request the web page.

Question 3

- How do these sizes (trace length) compare with the file size of the web page you downloaded? Estimate the download protocol overhead, or percentage of the download bytes taken up by protocol overhead.

Solution 3

- Results may vary, but the overhead including the headers from each of the packets and also the trace involved with establishing and ending the connection (3-way handshaking).
- It can be obtained by subtracting the file size of the retrieved HTML file from the *total trace* in bytes received from the previous question.

Question 4

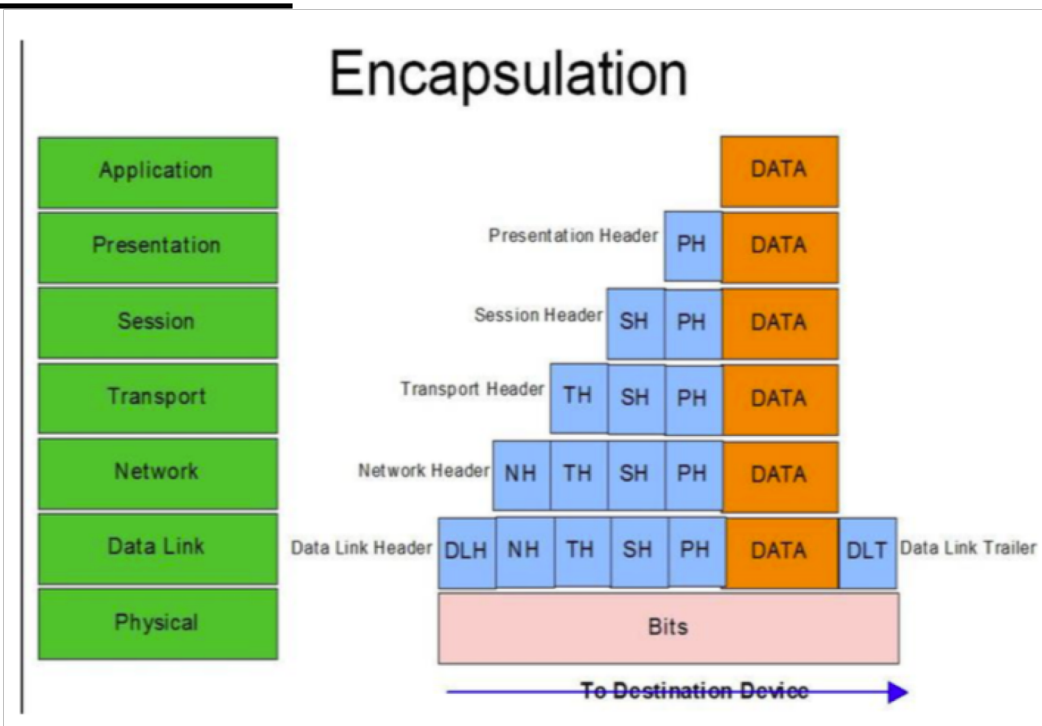
- What are the protocol overheads used for? Why are they useful? Aren't they a waste of space?

Solution 4

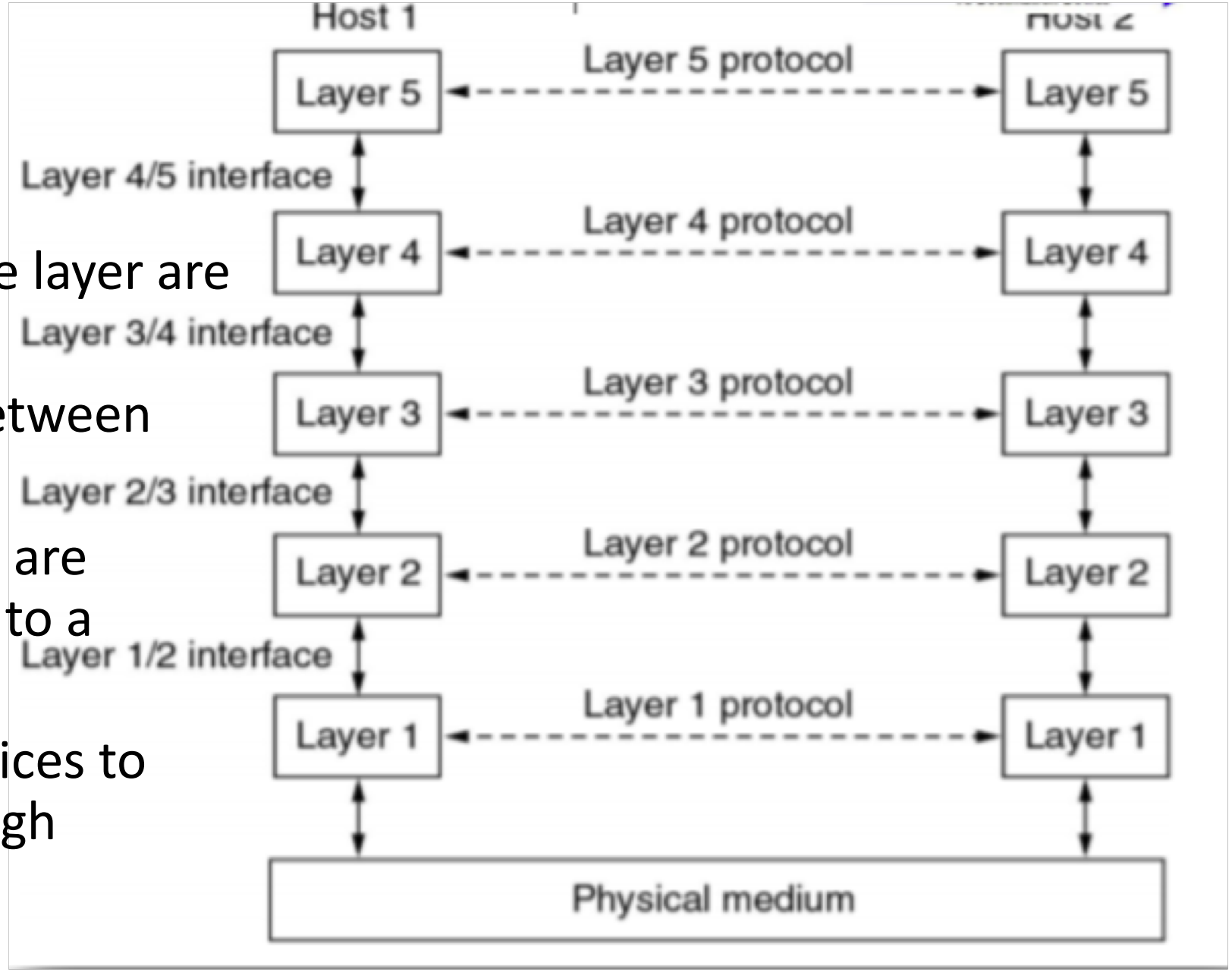
- The overheads/headers are used to allow each layer in the networking architecture perform their particular service.
  – For example, the networking layer headers will have the source IP and destination IP addresses for that packet, which tells the intermediate hosts where the final destination of the packet is intended.
- These headers are part of how data is transferred between the networking layers. Adding headers is called as *encapsulation*, which keeps networking functions separate and modular between the layers.

Encapsulation

- When a protocol on the sending host adds data to the packet header, the process is called **encapsulation**
  - During encapsulation, each layer builds a protocol data unit (PDU) by adding a header containing control information to the PDU from the layer above.



- Machines at the same layer are called as peers
- Virtual connection between peers
- Inter-layer exchanges are conducted according to a protocol
- Each layer offers services to layers above it (through interface)



Question 5

- Are you able to identify other packets apart from the HTTP GET request and response packets in your capture? What might these packets be used for?

Solution 5

- Apart from HTTP GET and HTTP 200 OK request and response packets, other packets of interest include the SYN and SYN ACK packets at the beginning which establishes a reliable connection between two hosts (more on that when we get to TCP), and also the RST or FIN packet at the end when the connection is closed.
- The packets in the middle are the transferring of the actual web page. It can also be observed that between every two or three packets that are received, our computer sends an ACK or acknowledgement packet to communicate to the remote host that the packets sent were successfully received. If the remote host fails to receive these packets, then it can be assumed that they were lost in transmission and need to be resent. (can you find those packets in different font/background colours?)

Question 6

- How do the protocol overheads relate to the networking layering architecture you have learned in class?
- Is this layered architecture efficient? What are the advantages/disadvantages of this layered architecture?

Solution 6

- A non-layered architecture may be more efficient (less or even zero protocol overhead), but at the cost of flexibility and modularity.
- Advantages of layered architecture include
  - 1. information hiding (i.e. if you were a network engineer, you would not need to know detailed information on the physical layer or link layer – remember the interface?)
  - 2. flexibility (the ability to change protocols in a certain layer, **without affecting** the operation of other layers).

Question 7 Bonus

- Calculate the inter-arrival times (IAT) of the packets, i.e. the time between each packet arrival.
- Plot the number of packets against the IAT. What sort of distribution do you observe? You may need to capture many more HTTP packets (this time using a web browser, for example) for a proper distribution to be seen.

Arrival Time vs. Inter-Arrival Time

- Arrival time – Clock time of arrival (wireshark)
- Inter-Arrival Time: time between successive arrivals
- Example: Initialize: Clock = 0

Difference (Inter-arrival times)	Arrival Time (Clock)
3	3
7	10
2	12
……	……

Solution 7

- For this question, you would have needed to plot a histogram, i.e. on the x-axis is a "bin" for a certain range of IATs, and on the y-axis the number of packets which fall into each particular bin. You should observe an exponential distribution. Feel free to use your favourite scientific computing/plotting package (Excel, MATLAB, Python, Mathematica, pen and paper, etc.).

Inter-arrival Time	Number of packets
0-1	2000
2-3	1800
3-4	500