WEEK 4 – TASK 4.1P

Pass Task.
Release Date: 12 April, Due Date: 26 April, End Date: 3 May.

Learning Outcomes

In this task, you will learn about network traffic sniffing and Wireshark. This task complements the theoretical concepts covered in Module 1 of Week 4.

Instructions

An **answer sheet template** is available on OnTrack as a `**Resources**'. Please download the answer sheet and fill it with your answers. To upload on OnTrack, you need to convert the answer sheet template document to **PDF**. MS Word includes built-in PDF conversation capability.

All 4 questions and their sub-questions of this task must be attempted. If screenshots are required, please ensure that text in screenshots is readable.

Remember that troubleshooting technical problems is part of learning in this field. You must patiently work through issues and solve these. Tasks are not step-by-step guide. You need to be in the driver seat and learn concepts by doing – as you would when you start your future job (many times even your future supervisor doesn't know the answer to problems you face). After patent troubleshooting and research, if you need help:

Help is always available in SIT182. **On-campus pracs** is available to those who have enrolled for these sessions. **Online Pracs** are available throughout the week (except Friday) for all students. Just join the session that best fits in your schedule and seek help from tutors. Alternatively, you can also use **Discussions** on the unit site and ask your questions about this task in **Task 4.1P**. All students are encouraged to participate and help peers with their questions. Helping others is a great way to learn and think about aspects you may have overlooked. You can also seek help from tutors during online and face-to-face pracs. Please do not raise your questions through Teams, OnTrack, or Email.

References In cyber security, our preferred referencing style is **IEEE** – however, you are allowed to use any Deakin approved referencing style in this unit. Please refer to unit site > Content > Referencing - Hints & Tips for more information.

Let's start with some background information[1].

Please note that concepts covered in Module 1 of Week 4 are not repeated here and you are assumed to be familiar with them in this task.

One's understanding of network protocols can often be greatly deepened by "seeing protocols in action" and by "playing around with protocols" – observing the sequence of messages exchanged between two protocol entities, delving down into the details of protocol operation, and causing protocols to perform certain actions and then observing these actions and their consequences.

The basic tool for observing the messages exchanged between executing protocol entities is called a **packet sniffer**. As the name suggests, a packet sniffer captures ("sniffs") messages being sent/received from/by your computer; it will also typically store and/or display the contents of the various protocol fields in these captured messages. A packet sniffer itself is passive. It observes messages being sent and received by applications and protocols running on your computer, but never sends packets itself. Similarly, received packets are never explicitly addressed to the packet sniffer. Instead, a packet sniffer receives a *copy* of packets that are sent/received from/by application and protocols executing on your machine.

Figure 1 shows the structure of a packet sniffer. At the right of Figure 1 are the protocols (in this case, Internet protocols) and applications (such as a web browser or email client) that normally run on your computer. The packet sniffer, shown within the dashed rectangle in Figure 1 is an addition to the usual software in your computer, and consists of two parts. The **packet capture library** receives a copy of every link-layer frame that is sent from or received by your computer over a given interface (link layer, such as Ethernet or WiFi). Recall from Week4 Module 1 that messages exchanged by higher layer protocols such as HTTP, FTP, TCP, UDP, DNS, or IP all are eventually encapsulated in link-layer frames that are transmitted over physical media such as an Ethernet cable or an 802.11 WiFi radio. Capturing all link-layer frames thus gives you all messages sent/received across the monitored link from/by all protocols and applications executing in your computer.

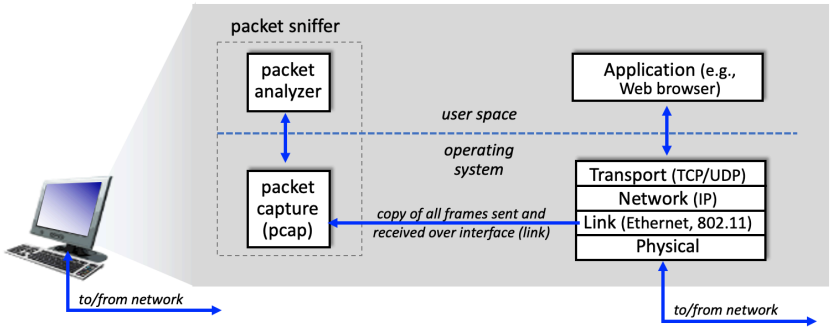


Figure 1: packet sniffer structure

The second component of a packet sniffer is the **packet analyser**, which displays the contents of all fields within a protocol message. In order to do so, the packet analyser must "understand" the structure of all messages exchanged by protocols. For example, suppose we are interested in displaying the various fields in messages exchanged by the HTTP protocol in Figure 1. The packet

[1] From [3] with amendments, adjustments, and additions.

analyser understands the format of Ethernet frames, and so can identify the IP datagram within an Ethernet frame. It also understands the IP datagram format, so that it can extract the TCP segment within the IP datagram. Finally, it understands the TCP segment structure, so it can extract the HTTP message contained in the TCP segment. Finally, it understands the HTTP protocol and so, for example, knows that the first bytes of an HTTP message will contain the string "GET," "POST," or "HEAD".

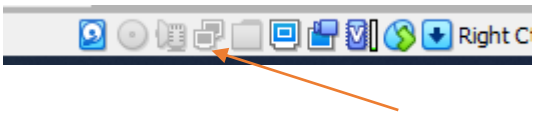
We will be using the Wireshark packet sniffer [http://www.wireshark.org/] for this task, allowing us to display the contents of messages being sent/received from/by protocols at different levels of the protocol stack. Technically speaking, Wireshark is a packet analyzer that uses a packet capture library in your computer. Also, technically speaking, Wireshark captures link-layer frames as shown in Figure 1, but uses the generic term "packet" to refer to link-layer frames, network-layer datagrams, transport-layer segments, and application-layer messages, so we'll use the less-precise "packet" term here to go along with Wireshark convention.

Wireshark is a free open-source network protocol analyser. It is used for network troubleshooting and communication protocol analysis. Wireshark captures network packets in real time and display them in human-readable format. It provides many advanced features including live capture and offline analysis, three-pane packet browser, colouring rules for analysis. Wireshark has a reputation for its stability, large user base and well-documented support that includes a user-guide (http://www.wireshark.org/docs/wsug_html_chunked/), man pages (http://www.wireshark.org/docs/man-pages/), and a detailed FAQ (http://www.wireshark.org/faq.html), rich functionality that includes the capability to analyze hundreds of protocols, and a well-designed user interface. It operates in computers using Ethernet, serial (PPP), 802.11 (WiFi) wireless LANs, and many other link-layer technologies.

Let's get started.

Wireshark is included in Kali Linux. We will be using the Kali Linux VM that you have already used in previous weeks.

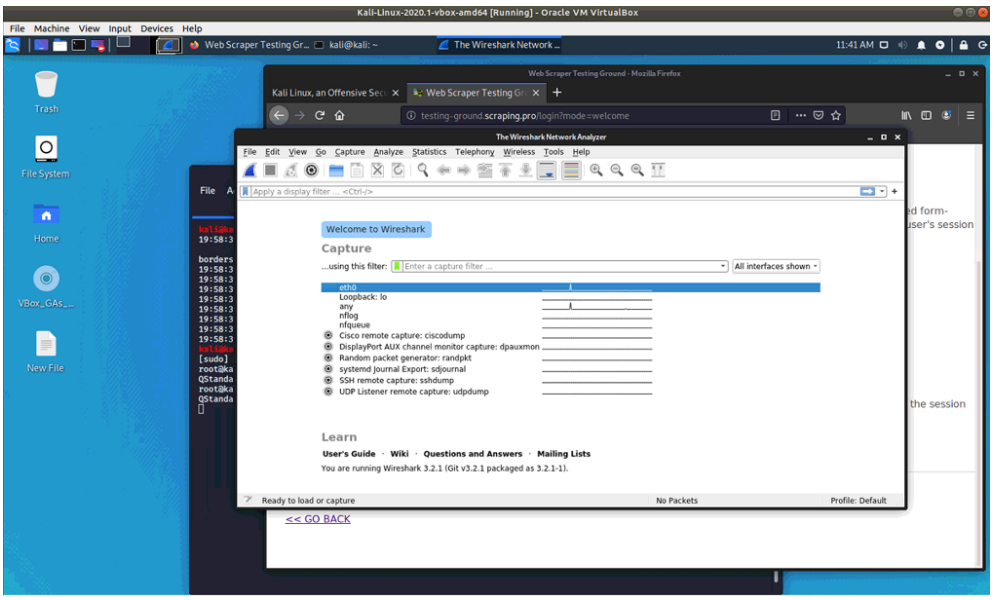
- If you are in the cyber lab, ensure that you import Kali VM from My Computer > Drive D > SIT182 > Kali Linux Docker Enabled. Remember that the cyber lab VMs are not persistent.
- If you have uninstalled Kali VM from your own machine, refer to Task 1.4P and follow the installation guidelines.
- Recall that username and password to login into Kali VM are both "kali" – without quotes.
- Ensure that Kali VM is connected to Internet. For this check network settings and ensure network is set to NAT or Bridged Adapter.
- If your VM was not connected to Internet (despite correct network settings), right-click on network setting in the bottom bar of VirtualBox window and click on "Connect Network Adapter"



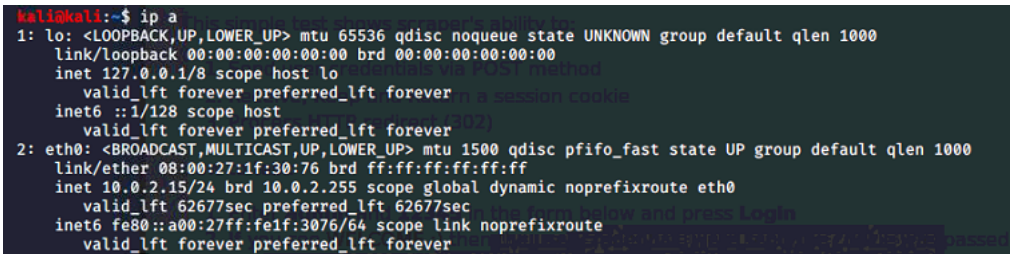
Next, we need to run Wireshark. Open Terminal and run:

sudo wireshark

If the command is successful, you should see Wireshark's GUI.



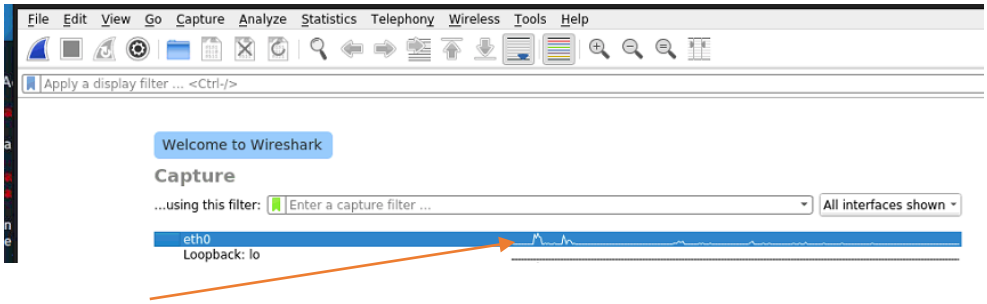
You will need to identify the interface that Wireshark should be capturing packets from. This is the interface that has valid IP, and you use that to connect to Internet. If you run `ip a' in Terminal, the output is something like the following:



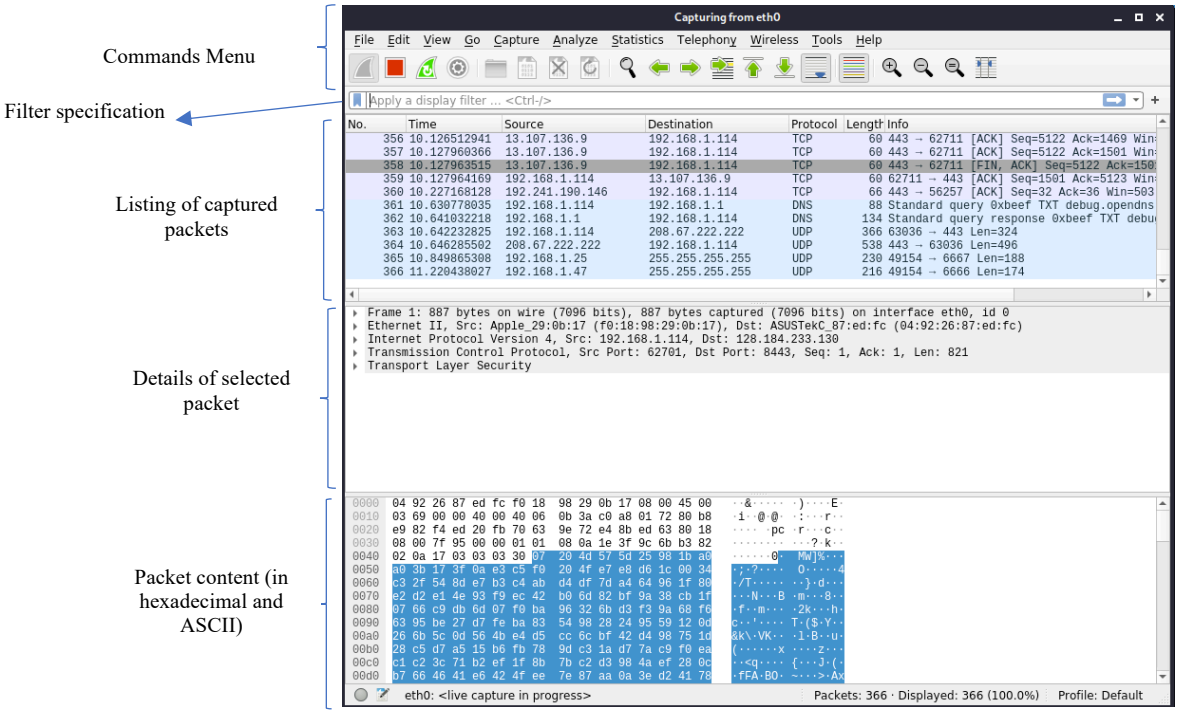
In this case, looking at the output, `eth0' is the interface that has external IP (10.0.2.15) and `lo' has local IP of `127.0.0.1'. So, we pick `eth0' to capture traffic when browsing online using Firefox.

- Please note that your interface name may be different, and you will need to interpret the output shown after running "ip a" and identify the correct interface.

Hint: an alternative way of deciding what interface to choose is to look at the traffic activity as shown on Wireshark interface. Discard 'any' as this will pick traffic from any of your device interfaces.



If you click on one of these interfaces to start packet capture (i.e., for Wireshark to begin capturing all packets being sent to/from that interface), a screen like the one below will be displayed, showing information about the packets being captured. Once you start packet capture, you can stop it by using the Capture pull down menu and selecting Stop (or by clicking on the red square button).



The Wireshark interface has five major components:

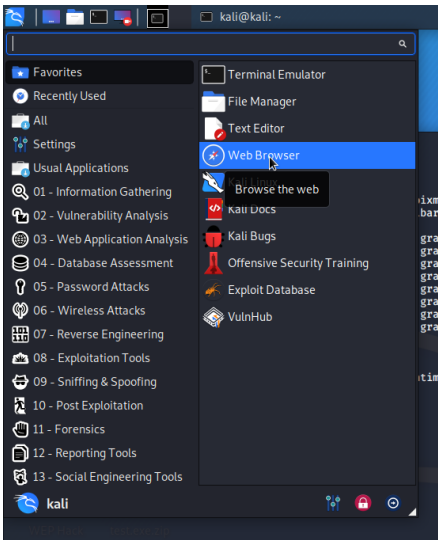
- The **command menus** are standard pulldown menus located at the top of the Wireshark window. Of interest to us now are the File and Capture menus. The File menu allows you to save captured packet data or open a file containing previously captured packet data and exit the Wireshark application. The Capture menu allows you to begin packet capture.
- The **packet-listing window** displays a one-line summary for each packet captured, including the packet number (assigned by Wireshark; note that this is *not* a packet number contained in any protocol's header), the time at which the packet was captured, the packet's source and destination addresses, the protocol type, and protocol-specific information contained in the packet. The packet listing can be sorted according to any of these categories by clicking

on a column name. The protocol type field lists the highest-level protocol that sent or received this packet, i.e., the protocol that is the source or ultimate sink for this packet.

- The **packet-header details window** provides details about the packet selected (highlighted) in the packet-listing window. (To select a packet in the packet-listing window, place the cursor over the packet's one-line summary in the packet-listing window and click with the left mouse button.). These details include information about the Ethernet frame (assuming the packet was sent/received over an Ethernet interface) and IP datagram that contains this packet. The amount of Ethernet and IP-layer detail displayed can be expanded or minimized by clicking on the plus/minus boxes or right/downward-pointing triangles to the left of the Ethernet frame or IP datagram line in the packet details window. If the packet has been carried over TCP or UDP, TCP or UDP details will also be displayed, which can similarly be expanded or minimized. Finally, details about the highest-level protocol that sent or received this packet are also provided.
- The **packet-contents window** displays the entire contents of the captured frame, in both ASCII and hexadecimal format.
- Towards the top of the Wireshark graphical user interface, is the **packet display filter field,** into which a protocol name or other information can be entered in order to filter the information displayed in the packet-listing window (and hence the packet-header and packet-contents windows). In the example below, we'll use the packet-display filter field to have Wireshark hide (not display) packets except those that correspond to HTTP messages.

Stop Capture and close Wireshark without saving.

Open Web Browser in Kali VM (not your host OS). Leave it running.



Start Wireshark, pick the interface that your VM is using to connect to Internet (as discussed), and you will notice that traffic capture is started.

(see next page)

While Wireshark is running, access the following URL in the Web Browser:

http://gaia.cs.umass.edu/wireshark-labs/INTRO-wireshark-file1.html

(copy/paste the link to your browser if clicking on the link didn't work)

and have that page displayed in your browser. In order to display this page, your browser will contact the HTTP server at gaia.cs.umass.edu and exchange HTTP messages with the server in order to download this page. The Ethernet or WiFi frames containing these HTTP messages (as well as all other frames passing through your Ethernet or WiFi adapter) will be captured by Wireshark.

After your browser has displayed the INTRO-wireshark-file1.html page (it is a simple one line of congratulations), **stop** Wireshark packet capture by selecting stop in the Wireshark capture window. You now have live packet data that contains all protocol messages exchanged between your computer and other network entities! The HTTP message exchanges with the gaia.cs.umass.edu web server should appear somewhere in the listing of packets captured. But there will be many other types of packets displayed as well (see, e.g., the many different protocol types shown in the *Protocol* column). Even though the only action you took was to download a web page, there were evidently many other protocols running on your computer that are unseen by the user. You'll learn much more about these protocols in future networking units!

Type in "http" (without the quotes, and *in lower case* – all protocol names are in lower case in Wireshark, and make sure to press enter/return key) into the Filter specification section at the top of the main Wireshark window. Then select *Apply* (to the right of where you entered "http") or just hit return. This will cause only HTTP message to be displayed in the packet-listing window.

Question 1:

A. Find the HTTP GET message that was sent from your computer to the gaia.cs.umass.edu HTTP server. (Look for an HTTP GET message in the "listing of captured packets" portion of the Wireshark window that shows "GET" followed by the gaia.cs.umass.edu URL that you entered. When you select the HTTP GET message, the Ethernet frame, IP datagram, TCP segment, and HTTP message header information will be displayed in the packet-header window[2]. By clicking on '+' and '-' and right-pointing and down-pointing arrowheads to the left side of the packet details window, you can *minimize/Maximise* the amount of Frame, Ethernet, Internet Protocol, and Transmission Control Protocol information displayed.

    1. Include a screenshot of your Kali VM that has the Wireshark window running. Ensure that your screenshot shows that have selected the packet with HTTP GET message and details of the packet are visible either as minimised or maximised.

    2. What is the Internet address of the gaia.cs.umass.edu (also known as www-net.cs.umass.edu)? What is the Internet address of your computer?

[2] Recall the concept of encapsulation as covered in Module 1 of Week 4: HTTP GET message that is sent to the gaia.cs.umass.edu web server is contained within a TCP segment, which is contained (encapsulated) in an IP datagram, which is encapsulated in an Ethernet frame. If this process of encapsulation isn't quite clear yet, review Module 1 of Week 4.

3. Check the packet details for HTTP Get message (refer to `Details of the selected packet' section of Wireshark window). What type of Web browser issued the HTTP request? Hint: "User-Agent:" field in the expanded HTTP message display. This field value in the HTTP message is how a web server learns what type of browser you are using.

4. What is the destination port number (Hint: the number following "Dest Port:" for the TCP segment containing the HTTP request) to which this HTTP request is being sent? What is the source port number?

B. HTTP was also referred to in Module 1 of Week 4.

1. What is HTTP used for in the world wide web?
2. At what network-layer is HTTP located?
3. What is an HTTP request? What is included in a typical HTTP request?
4. What is an HTTP method? how do Get and Post methods differ.
5. What is an HTTP response? What is included in a typical HTTP response?
6. Is HTTP a stateless or a stateful protocol?

C. You were introduced to Denial-of-Service Attacks in Module 2 of Week 4.
1. Can HTTP be used to execute a DoS attack?

When answering Questions B and C, ensure that you paraphrase the information you find online and use references. As minimum, you are expected to have 2 references when answering Questions C and C. C is a common interview question.

Stop Wireshark, close without saving, and then close your web browser in Kali VM.

Read about HTTP authentication from the following URL:

http://frontier.userland.com/stories/storyReader$2159

Now, let's try visiting a web site that is password-protected and uses HTTP. Start Wireshark again and initiate capture on the correct interface. Run the Web Browser in Kali VM.

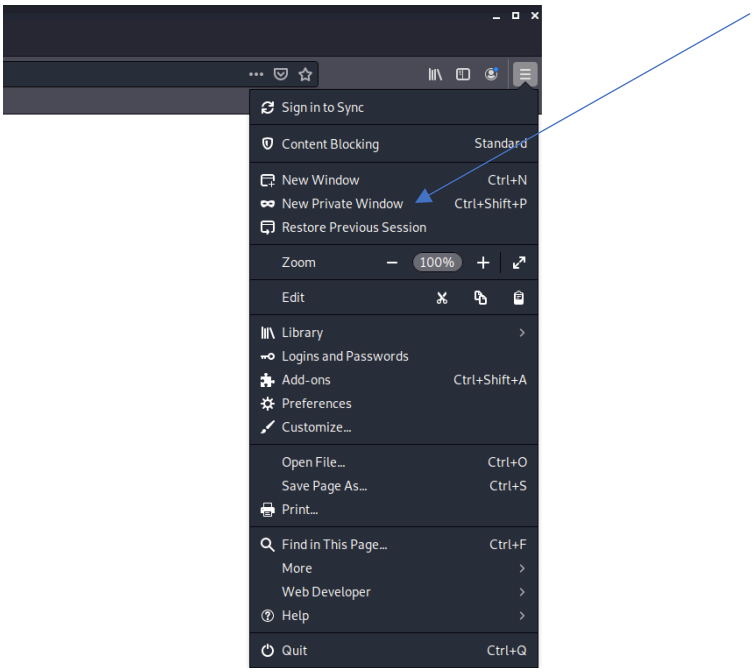
Visit the following `secured' URL:

http://gaia.cs.umass.edu/wireshark-labs/protected_pages/HTTP-wireshark-file5.html

The username is "wireshark-students" (without the quotes), and the password is "network" (again, without the quotes).

Stop capture in Wireshark. Enter `http' in the filter specification section.

Note: if you enter credentials correctly and access the secured URL then you will stay logged in (it will not require you to provide username and password) until you close your browser and access the URL again. If you were still not prompted for credentials after opening your browser again, access the link using New Private Windows of your browser in Kali VM.



Question 2: based on information in Wireshark windows answer the following questions.

A. What is the server's response (status code and phrase) in response to the initial HTTP GET message from your browser? (i.e., before you provided the credentials) Include a screenshot of the Wireshark window showing the relevant packet and its details.
B. Was does "Connection: keep-alive" mean in Hypertext Transfer Protocol of HTTP GET message?
C. The packet length for HTTP GET message is showing as 506 bytes (if yours is different assume it was meant to be 506 bytes). How many bits is 506 bytes? What is the difference between bit and byte?

The username (wireshark-students) and password (network) that you entered are encoded in the string of characters (d2lyZXNoYXJrLXN0dWRlbnRzOm5ldHdvcms=) following the "Authorization: Basic" header in the client's HTTP GET message. While it may appear that your username and password are encrypted, they are simply encoded in a format known as Base64 format. The username and password are *not* encrypted!

To see this, go to http://www.motobit.com/util/base64-decoder-encoder.asp and enter the base64-encoded string d2lyZXNoYXJrLXN0dWRlbnRz and decode. *Voila!* You have translated from Base64 encoding to ASCII encoding, and thus should see your username! To view the password, enter the remainder of the string Om5ldHdvcms= and press decode.

Since anyone can download a tool like Wireshark and sniff packets (not just their own) passing by their network adaptor, and anyone can translate from Base64 to ASCII (you just did it!), it should be clear to you that simple passwords on WWW sites are not secure unless additional measures are taken.

Stop Wireshark, close without saving, and then close your web browser in Kali VM.

You may have noticed that if you expand Authorisation in Hypertext Transfer Protocol in packet detail section of Wireshark, the username and password are shown in clear text and you don't need to covert manually. This is just Wireshark trying to make things even easier for you!

Now, let's try another HTTP website.

Start Wireshark again and initiate capture on the correct interface. Run the Web Browser in Kali VM. Visit the following `secured' URL:

http://www.techpanda.org

As for Email, enter your Deakin student ID **excluding the last digit** followed by @myemail.com. For instance, if your student ID is 12345 then your Email for this form is 1234@myemail.com. Enter `sit1822021' as password without quote.

Please note that with the above credentials you will not be able to login to the website, and you will receive an error. However, successful login is not what we worry about. Let's see if someone was intercepting your traffic how easy/difficult would have been for them to retrieve your credentials.

Stop Wireshark. Enter `http' in the filter specification section. Find the HTTP message that includes the email and password you used.

Question 3:

A. Was techpanda.org using Base64 encoding?
B. Did you find the password in an HTTP GET or POST message? Why?
C. Include a screenshot of Wireshark window that shows the packet with Email and Password you used (i.e., the email and password should be clearly visible in packet details section of Wireshark.)

To increase the security of WWW, HTTPS was introduced. HTTPS is HTTP with encryption. The only difference between the two protocols is that HTTPS uses TLS (or SSL) to encrypt normal HTTP requests and responses. As a result, HTTPS is far more secure than HTTP. A website that uses HTTP has http:// in its URL, while a website that uses HTTPS has https://.

With the addition of encryption in HTTPS, the attacker would see a bunch of seemingly random characters when sniffing traffic.

In other words, Instead of:

GET /hello.txt HTTP/1.1
User-Agent: curl/7.63.0 libcurl/7.63.0 OpenSSL/1.1.l zlib/1.2.11
Host: www.example.com
Accept-Language: en

The attacker sees something like - We will study cryptography in future weeks, which helps understand this change better!

t8Fw6T8UV81pQfyhDkhebbz7+oiwldr1j2gHBB3L3RFTRsQCpaSnSBZ78Vme+DpDVJPvZdZUZHpzbb
cqmSW1+3xXGsERHg9YDmpYk0VVDiRvw1H5miNieJeJ/FNUjgH0BmVRWII6+T4MnDwmCMZUI/orx
P3HGwYCSIvyzS3MpmmSe4iaWKCOHQ==

If you were keen to see this in action, try to sniff the password you provided for Task 3.1P when accessing http://cyber-challenges.com.

A good resource to learn more about Wireshark is [2], which includes hands-on practices as well.

Question 4:

Reflection point – What did you learn that was new to you? How interesting you find network security compared to other topics covered? How did this task complement the theoretical concepts covered in Week 4 lecture?

References and Further Reading:

[1]: "*Computer Networking: A Top-Down Approach", 8th ed.*, J.F. Kurose and K.W. Ross, 2021.

https://ezproxy.deakin.edu.au/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=cat00097a&AN=deakin.b4166302&authtype=sso&custid=deakin&site=eds-live&scope=site

[2]: "Wireshark for security professionals: using Wireshark and the Metasploit Framework" by Jessey Bullock, Jeff T. Parke, 2017.

https://ezproxy.deakin.edu.au/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=cat00097a&AN=deakin.b3809734&authtype=sso&custid=deakin&site=eds-live&scope=site

[3]: *Lab 1 and 2 of https://gaia.cs.umass.edu/kurose_ross/wireshark.htm provided as supplementary material for [1]. (last access: 8 April 2021)*