

Part 4A: Capture and Analyze Packet with Wireshark

This part A of the lab is not graded.

Required Hardware: You can do this project exercise on any computer that is connected to the Internet and has WireShark software installed. Download and install WireShark from www.wireshark.org and install it on your local PC. You can also install it on your ubuntu VM (Installed in Lab 1)

Wireshark is a network packet analyzer. A network packet analyzer presents captured packet data in as much detail as possible. You could think of a network packet analyzer as a measuring device for examining what's happening inside a network cable, just like an electrician uses a voltmeter for examining what's happening inside an electric cable (but at a higher level, of course).

In the past, such tools were either very expensive, proprietary, or both. However, with the advent of Wireshark, that has changed. Wireshark is available for free, is open source, and is one of the best packet analyzers available today.

Here are some reasons people use Wireshark:

- Network administrators use it to troubleshoot network problems
- Network security engineers use it to examine security problems
- QA engineers use it to verify network applications
- Developers use it to debug protocol implementations
- People use it to learn network protocol internals

In this lab, you will:

1. Execute Wireshark and practice capturing data packets
2. Analyze the results from capturing packets for a file download from a web server
3. Analyze pcap file provided and find security flaws

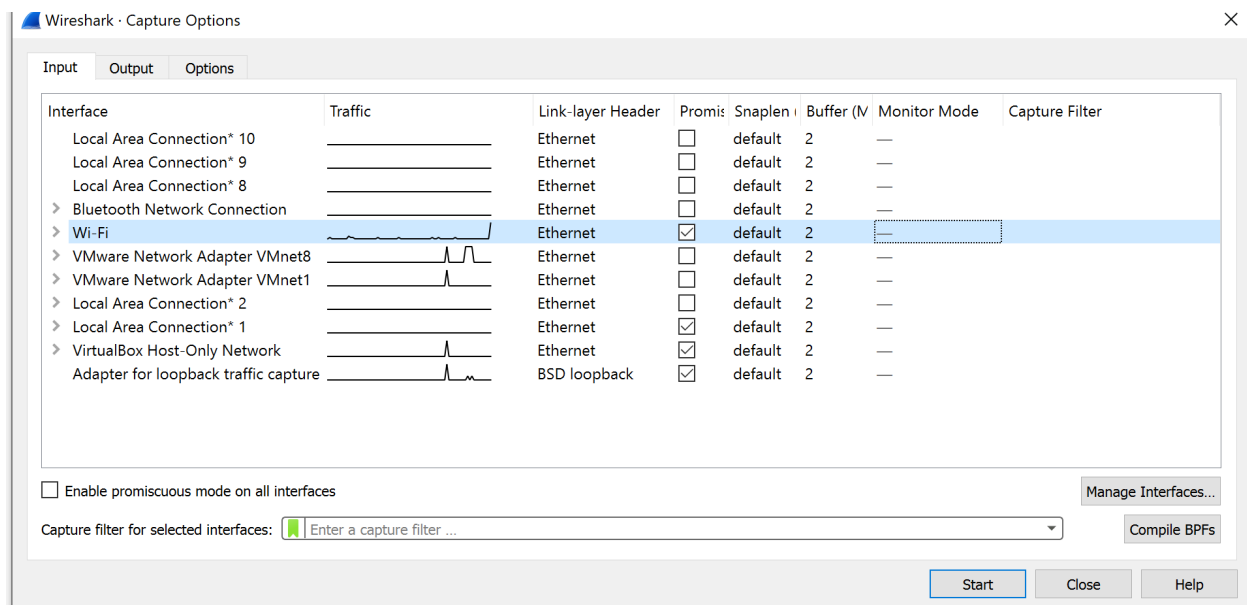
Lab 2:

Part 1: Check computer addresses and start-up Wireshark Capture

Determine and **note the Physical MAC address and the IP** address of the device you are using. If needed, follow the tutorial uploaded on Blackboard to find your physical MAC address and the IP address.

Close your Browser -> Startup Wireshark, click **Capture → Options → Choose the input** → Click the **Start/Capture** button corresponding to your active interface.

Wireshark will begin running in Capture Mode and will open up a Wireshark Capture window showing you how many packets have been captured in real time.



Part 2: Download Web Page

1. With Wireshark still running in Capture Mode, start up your favorite browser (Chrome, Internet Explorer, Mozilla, Firefox or whatever).
2. Enter the following address or any address of your choice:
<http://info.cern.ch/hypertext/WWW/TheProject.html>
3. A web page should appear in your browser from my home page. Wait for a few seconds for a page to load
1. Close your browser window.
2. Go back to the Wireshark Capture window and click the **Stop** button to stop the packet capture.

Part 3: Verify that the Web Page Download Has Been Captured

1. Back in the Wireshark window, you should now see lots of packets in the top summary pane. You can filter out all packets except HTTP packets by typing the word "http" into the Filter box (click View→Filter Toolbar if you don't see a Filter box at the top). This will make things much easier to read.
2. You should see a packet containing something like **GET /
<http://info.cern.ch/hypertext/WWW/TheProject.html>** sent by your PC to request the web page download.

No.	Time	Source	Destination	Protocol	Length	Info
659	8.224459	10.65.60.10	188.184.21.108	HTTP	513	GET /hypertext/WWW/TheProject.html HTTP/1.1
683	8.336891	188.184.21.108	10.65.60.10	HTTP	1188	HTTP/1.1 200 OK (text/html)
787	8.557345	10.65.60.10	188.184.21.108	HTTP	454	GET /favicon.ico HTTP/1.1
710	8.655626	188.184.21.108	10.65.60.10	HTTP	312	HTTP/1.1 200 OK (image/vnd.microsoft.icon)
814	14.825574	10.65.60.10	188.184.21.108	HTTP	569	GET /hypertext/WWW/WhatIs.html HTTP/1.1
819	14.936397	188.184.21.108	10.65.60.10	HTTP	1436	HTTP/1.1 200 OK (text/html)
846	17.338960	10.65.60.10	188.184.21.108	HTTP	564	GET /hypertext/WWW/Terms.html HTTP/1.1
870	17.424482	188.184.21.108	10.65.60.10	HTTP	254	HTTP/1.1 200 OK (text/html)

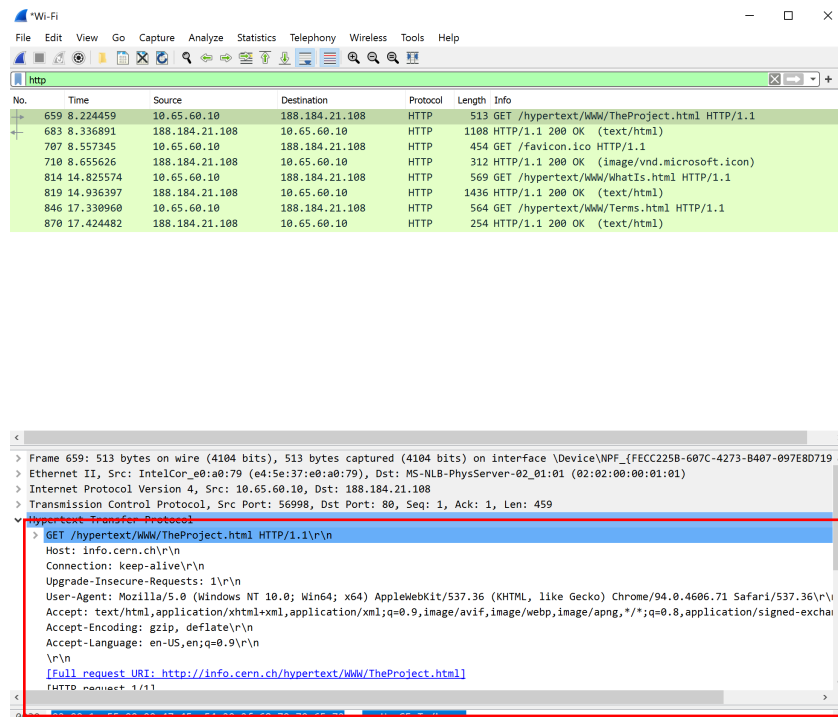
```

Frame 659: 513 bytes on wire (4104 bits), 513 bytes captured (4104 bits) on interface \Device\NPF_{FECC2258-607C-4273-B407-097E8D719...}
Ethernet II, Src: IntelCor_e0:a0:79:(e4:5e:37:e0:a0:79), Dst: MS-NL8-PhysServer-02_01:01 (02:02:00:00:01:01)
Internet Protocol Version 4, Src: 10.65.60.10, Dst: 188.184.21.108
Transmission Control Protocol, Src Port: 56998, Dst Port: 80, Seq: 1, Ack: 1, Len: 459
Hypertext Transfer Protocol
  GET /hypertext/WWW/TheProject.html HTTP/1.1\r\n
    Host: info.cern.ch\r\n
    Connection: keep-alive\r\n
    Upgrade-Insecure-Requests: 1\r\n
    User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/94.0.4606.71 Safari/537.36\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9\r\n
    Accept-Encoding: gzip, deflate\r\n
    Accept-Language: en-US,en;q=0.9\r\n
    \r\n
    [Full request URI: http://info.cern.ch/hypertext/WWW/TheProject.html]
  (HTTP request 1/1)
  
```

- HTTP is a Hypertext Transfer Protocol. HTTP is an application layer protocol in the OSI or TCP/IP model. HTTP is used by the World Wide Web (w.w.w) and it defines how messages are formatted and transmitted by browser. So HTTP define rules for what action should be taken when a browser receives an HTTP command. And also, HTTP defines rules for transmitting HTTP commands to get data from server.
- There are some set of methods for HTTP/1.1 (This is HTTP version)
 - GET, HEAD, POST, PUT, DELETE, CONNECT, OPTION and TRACE.
 - We will not go in details of each method instead we will get to know about the methods which are seen quite often. Such as
- GET: GET request asks data from web server. This is a main method used document retrieval. We will see one practical example of this method.

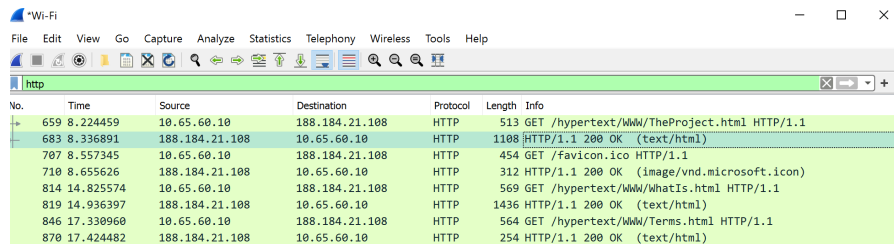
HTTP GET:

After TCP 3-way handshake [SYN, SYN+ACK and ACK packets] is done HTTP GET request is sent to the server and here are the important fields in the packet. Click on the first packet to see the details (GET /teaching.html HTTP/1.1).

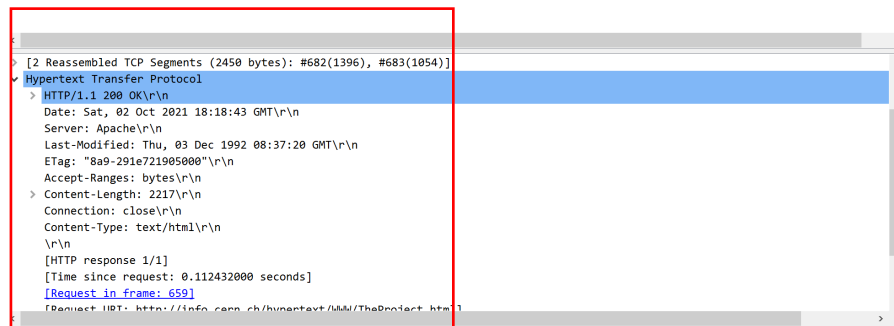


- I. Request Method: GET ==> The packet is a HTTP GET .
- II. Request URI: /TheProject.html ==> The client is asking for contents of TheProject.html
- III. Request version: HTTP/1.1 ==> It's HTTP version 1.1
- IV. Accept: text/html, application/xhtml+xml, image/jxr, /* ==> Tells server about the type of file it [client side browser] can accept. Here the client is expecting teaching.html which is html type.
- V. Accept-Language: en-US ==> Accepted language standard.
- VI. User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko ==> Client side browser type. Even if we used internet explorer but we see it always/maximum time says Mozilla
- VII. Accept-Encoding: gzip, deflate ==> Accepted encoding in client side.
- VIII. Host: www.info.cern.ch ==> This is the web server name where client is sending HTTP GET request.
- IX. Connection: Keep-Alive ==> Connection controls whether the network connection stays open after the current transaction finishes. Connection type is keep alive.

3. If the next packet listed (containing the reply from the web server to your PC) contains “HTTP/1.1 200 OK” and the next few packets are “Continuation” packets, then you have successfully captured the packets containing the html web page. Click on the HTTP/1.1 200 OK packet to see the details. You can now Skip step 4. (Note: Go to Step 4 if you see 304 Not modified)



No.	Time	Source	Destination	Protocol	Length	Info
659	8.224459	10.65.60.10	188.184.21.108	HTTP	513	GET /hypertext/WWW/TheProject.html HTTP/1.1
683	8.336891	188.184.21.108	10.65.60.10	HTTP	1108	HTTP/1.1 200 OK (text/html)
707	8.557345	10.65.60.10	188.184.21.108	HTTP	454	GET /favicon.ico HTTP/1.1
710	8.655626	188.184.21.108	10.65.60.10	HTTP	312	HTTP/1.1 200 OK (image/vnd.microsoft.icon)
814	14.825574	10.65.60.10	188.184.21.108	HTTP	569	GET /hypertext/WWW/WhatIs.html HTTP/1.1
819	14.936397	188.184.21.108	10.65.60.10	HTTP	1436	HTTP/1.1 200 OK (text/html)
846	17.330960	10.65.60.10	188.184.21.108	HTTP	564	GET /hypertext/WWW/Terms.html HTTP/1.1
870	17.424482	188.184.21.108	10.65.60.10	HTTP	254	HTTP/1.1 200 OK (text/html)



HTTP OK:

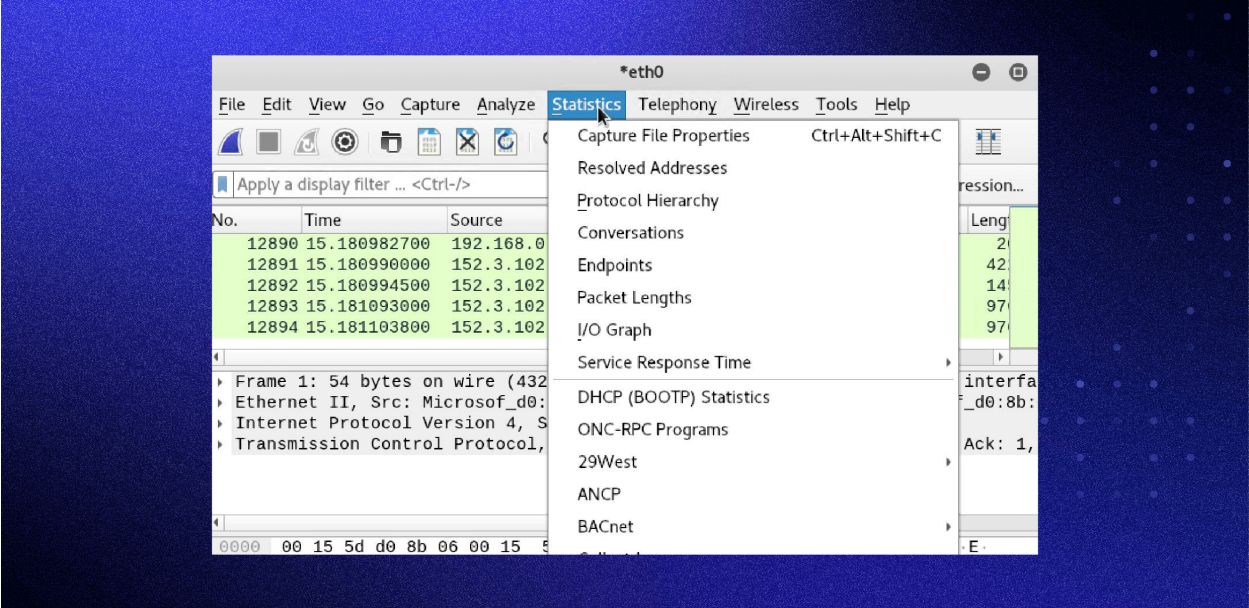
After TCP data [content of teaching.html] is sent successfully HTTP OK is sent to the client and here are the important fields in the packet.

- I. Response Version: HTTP/1.1 ==> Here server also in HTTP version 1.1
- II. Status Code: 200 ==> Status code sent by server.
- III. Response Phrase: OK ==> Response phrase sent by server.
- IV. So the from 2 and 3 we get 200 OK which means the request [HTTP GET] has succeeded.
- V. Server: Apache ==> Server details and configurations versions.

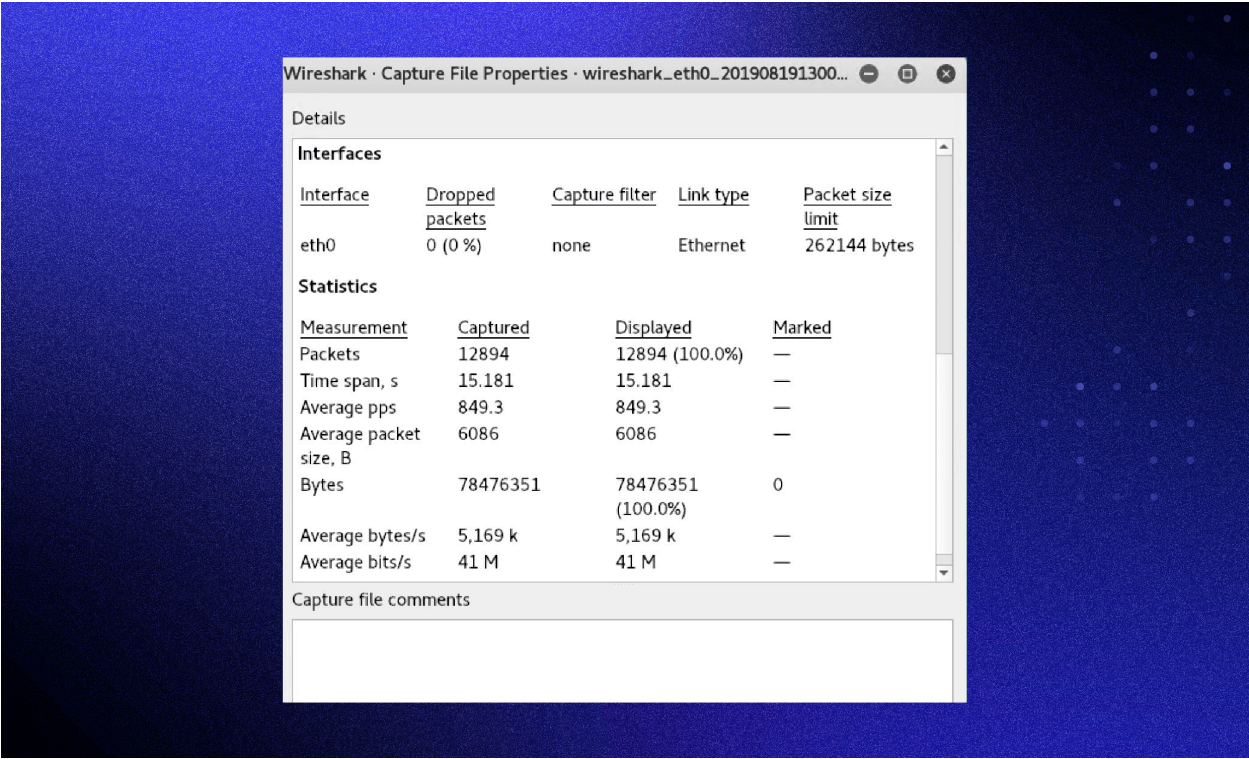
- VI. Last-Modified: Thu, 03 Dec 1992 08:37:20 GMT ==> Last modified date and time for the file
 - VII. ETag: "8a9-291e721905000" ==> The ETag indicates the content is not changed to assist caching and improve performance. Or if the content has changed, etags are useful to help prevent simultaneous updates of a resource from overwriting each other.
 - VIII. Accept-Ranges: bytes ==> Byte is the unit used in server for content.
 - IX. Content-Length: 2217 ==> This is the total length of the teaching.html in bytes.
 - X. Content-Type: text/html; charset=utf-8 ==> The content type is html and charset standard is UTF-8.
-
- 4. On the other hand, if the reply from the web server contains “HTTP/1.1 304 Not Modified”, then this means Wireshark **did not capture** the packets from the web site because the web page was already stored (cached) in your browser. In this case you must clear your browser cache and then go back and re-do the capture as follows:
 - a. First, you must clear the web cache in your browser.
 - i. For Internet Explorer, click **Tools → Internet Options**, then, under the General tab, click the **Delete Files** button within the **Temporary Internet Files** box area.
 - ii. For Firefox, click **Tools → Options**, then **Privacy, Cache** and click **Clear**.
 - iii. For Chrome, click **Settings → Privacy and Security**, then **Clear browsing data, check cookies and other site data, cached imaged and files** and click **Clear data**
 - iv. For other browsers, you’re on your own.
 - b. Now in your Wireshark window, again select **Capture → Interfaces** and click the **Capture** button corresponding to the Ethernet interface. When prompted whether to save the previous capture, click **Continue without saving**.
 - c. Go back to Part 2, step 1 above to download the web page again while Wireshark is capturing packets.

Metrics and statistics

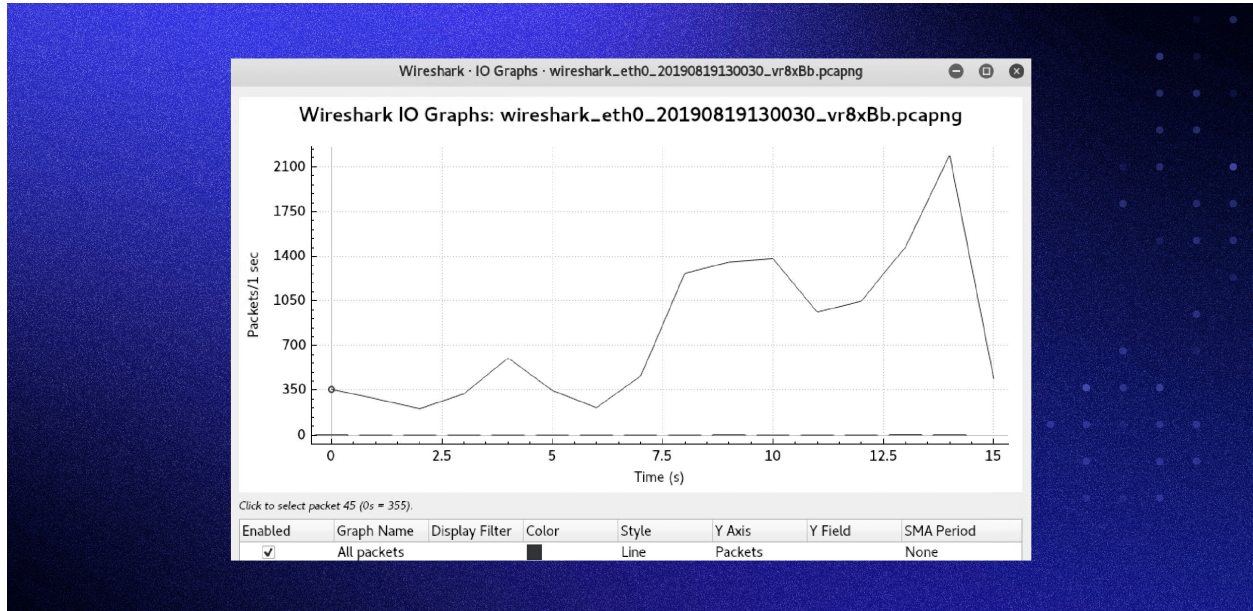
Under the Statistics menu, you’ll find a plethora of options to view details about your capture.



Capture File Properties:



Wireshark I/O Graph:



Additional Wireshark resources and tutorials

There are many tutorials and videos that show you how to use Wireshark for specific purposes. You should begin your search on the main [Wireshark website](#) and move forward from there. You can find the official documentation and [Wiki](#) on that site as well. Wireshark is a great network sniffer and analysis tool — however, it's best used once you know what you're looking for.

Lab 5a is now complete.