CSE 3501

INFORMATION SECURITY ANALYSIS & AUDIT

Lab Assessment – 5

L9+L10 | PLBG04
Dr. Vimala Devi K

FALL SEMESTER 2021-22

by

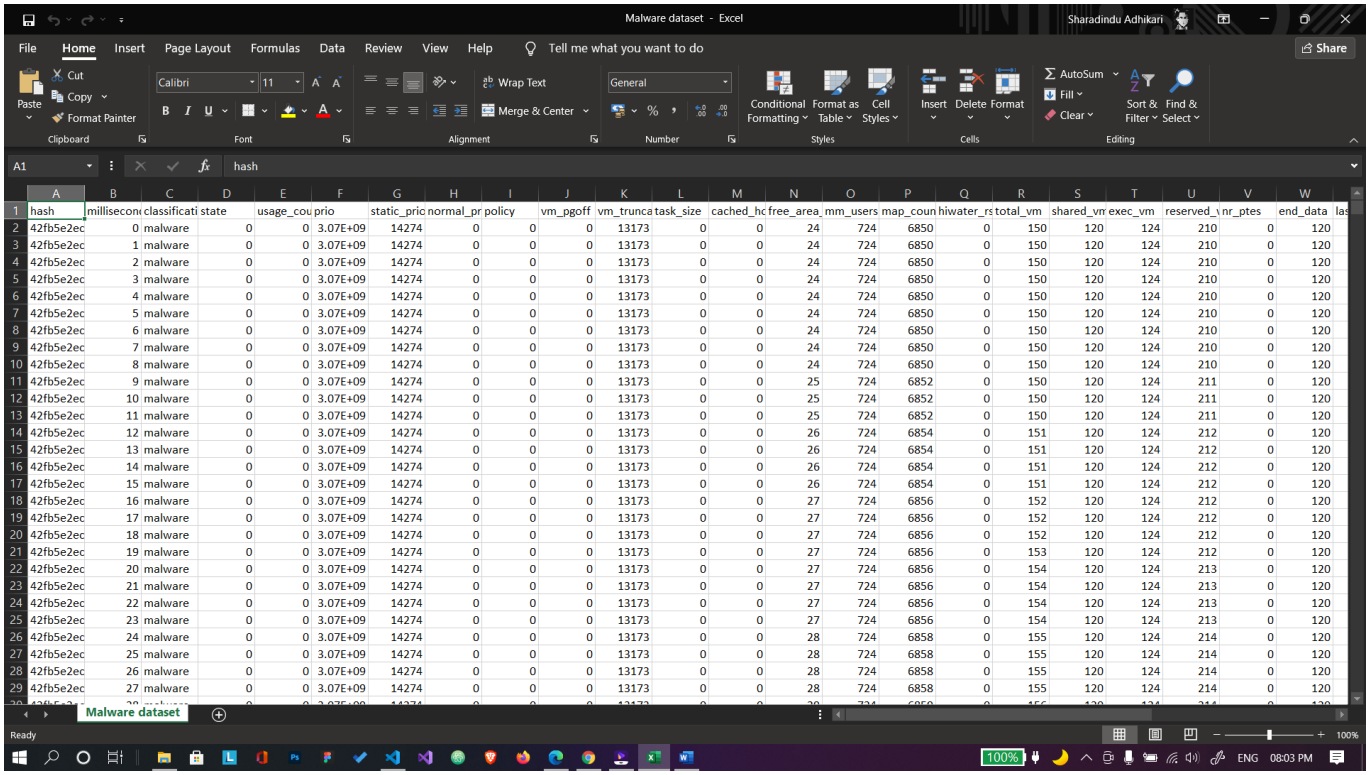
SHARADINDU ADHIKARI
19BCE2105

Index

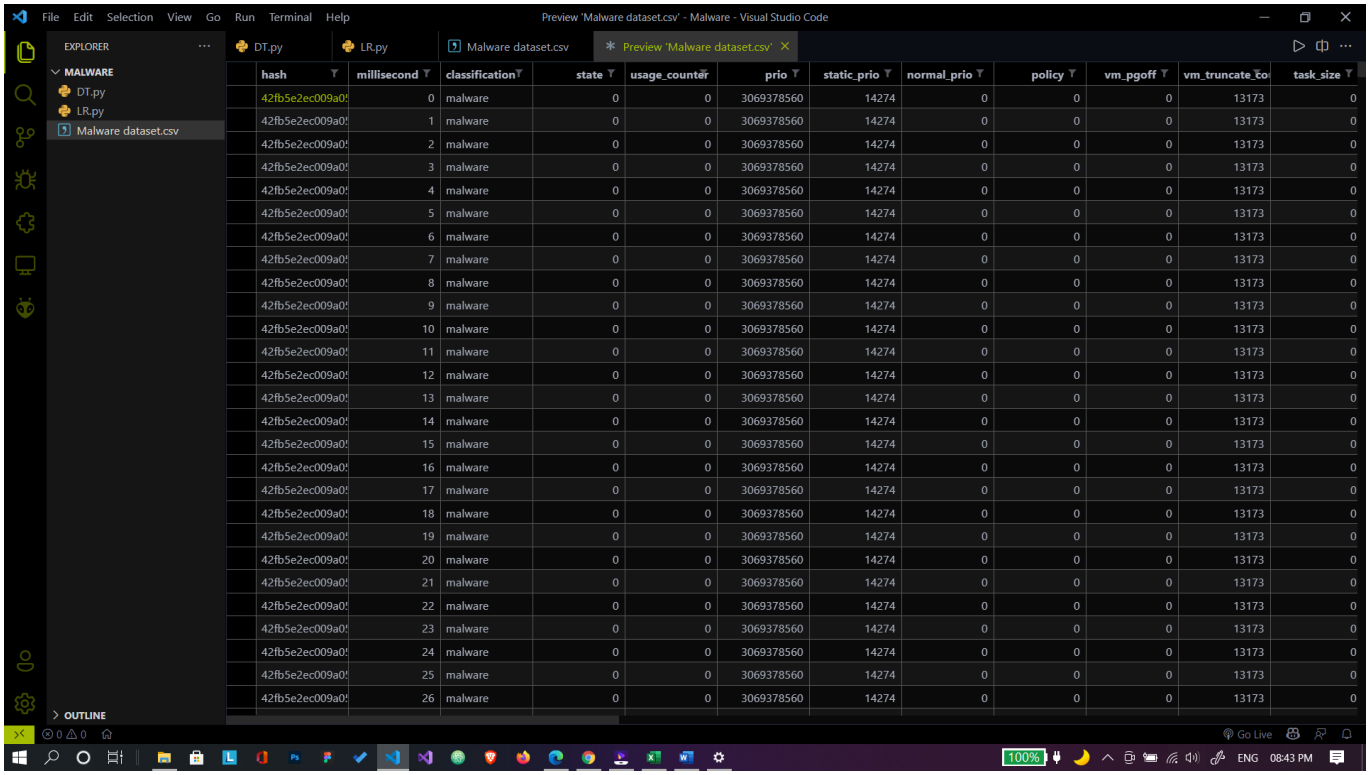
sharadindu.adhikari2019@vitstudent.ac.in

Exp 0: Malware Detection using ML

Dataset:

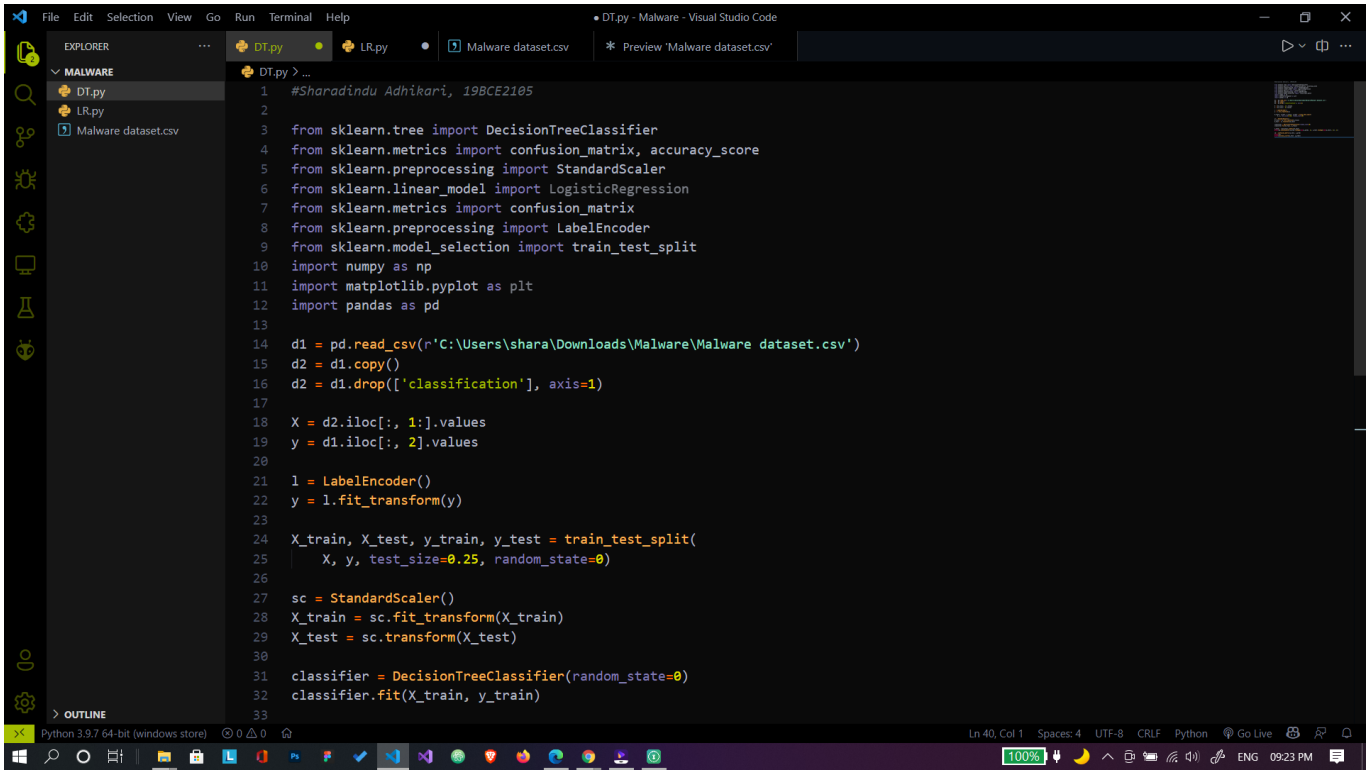


hash	millisecond	classification	state	usage_counter	prio	static_prio	normal_prio	policy	vm_pgoff	vm_truncate_counter	task_size
42fb5e2ec009a05	0	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	1	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	2	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	3	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	4	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	5	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	6	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	7	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	8	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	9	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	10	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	11	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	12	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	13	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	14	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	15	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	16	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	17	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	18	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	19	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	20	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	21	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	22	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	23	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	24	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	25	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a05	26	malware	0	0	3069378560	14274	0	0	0	13173	0



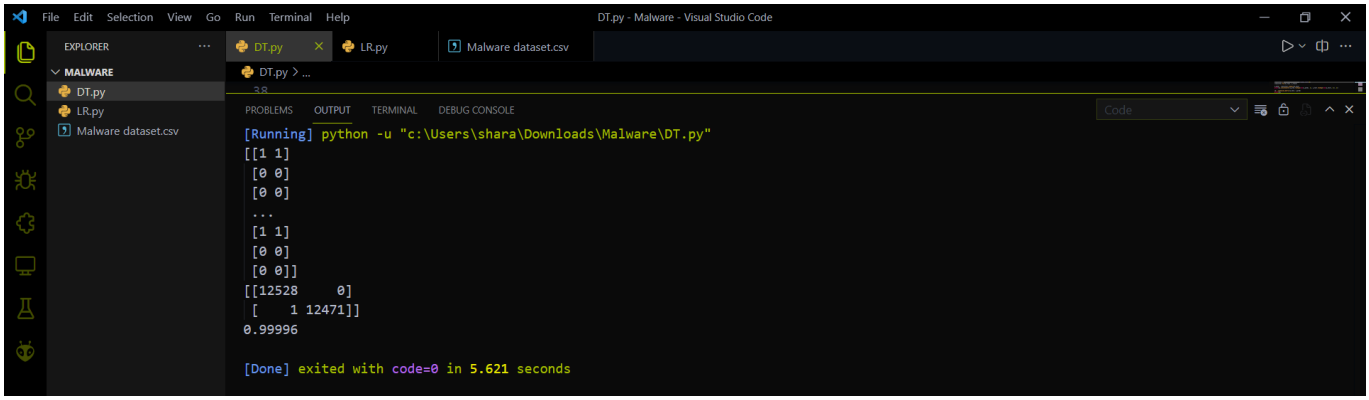
Decision tree:

```
#Sharadindu Adhikari, 19BCE2105

from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

d1 = pd.read_csv(r'C:\Users\shara\Downloads\Malware\Malware dataset.csv')
d2 = d1.copy()
d2 = d1.drop(['classification'], axis=1)

X = d2.iloc[:, 1:].values
y = d1.iloc[:, 2].values

l = LabelEncoder()
y = l.fit_transform(y)

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.25, random_state=0)

sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

classifier = DecisionTreeClassifier(random_state=0)
classifier.fit(X_train, y_train)

y_pred = classifier.predict(X_test)
print(np.concatenate((y_pred.reshape(len(y_pred), 1), y_test.reshape(len(y_test),
1)), 1))

cm = confusion_matrix(y_test, y_pred)
print(cm)
print(accuracy_score(y_test, y_pred))
```



```python
#Sharadindu Adhikari, 19BCE2105

from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

d1 = pd.read_csv(r'C:\Users\shara\Downloads\Malware\Malware dataset.csv')
d2 = d1.copy()
d2 = d1.drop(['classification'], axis=1)

X = d2.iloc[:, 1:].values
y = d1.iloc[:, 2].values

l = LabelEncoder()
y = l.fit_transform(y)

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.25, random_state=0)

sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

classifier = DecisionTreeClassifier(random_state=0)
classifier.fit(X_train, y_train)
```

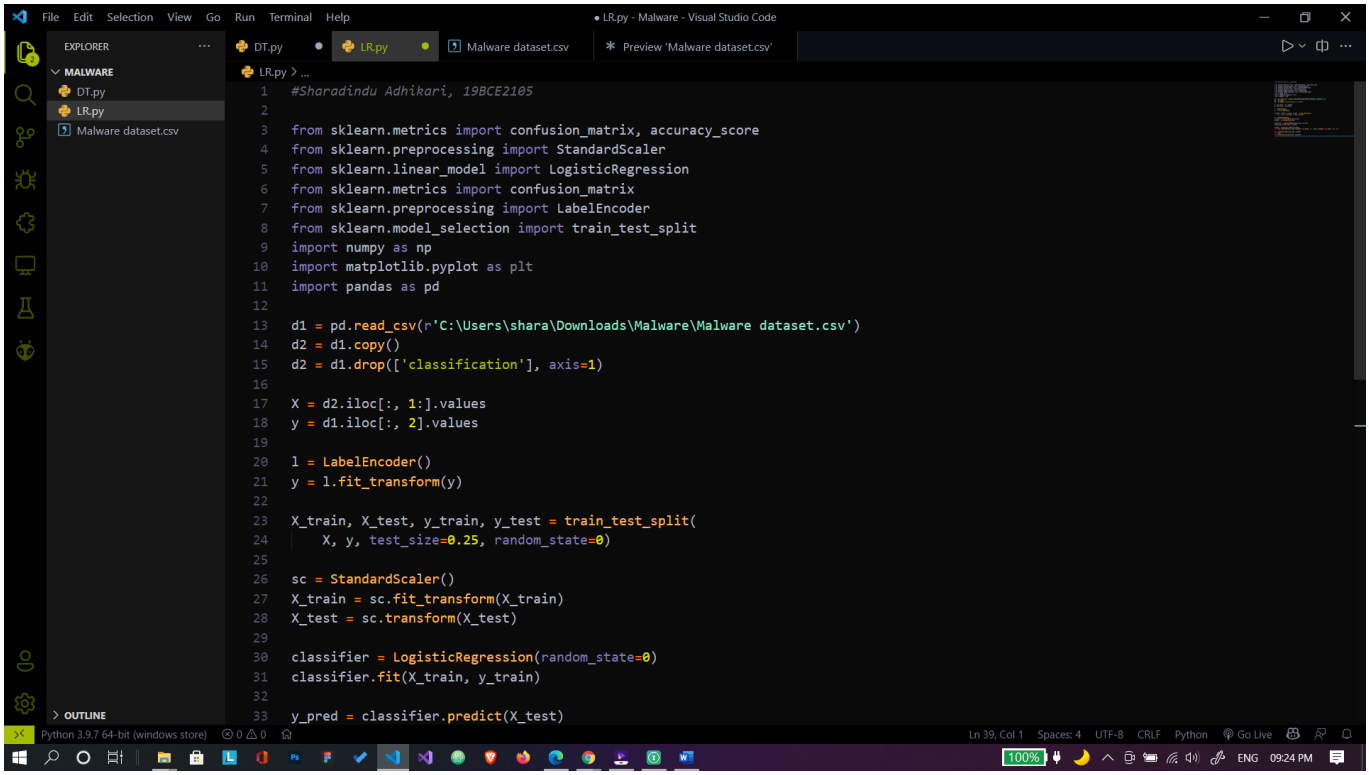
Predicting the Test set results, Confusion Matrix and Accuracy:



```
[Running] python -u "c:\Users\shara\Downloads\Malware\DT.py"
[[1 1]
 [0 0]
 [0 0]
 ...
 [1 1]
 [0 0]
 [0 0]]
[[12528     0]
 [    1 12471]]
0.99996

[Done] exited with code=0 in 5.621 seconds
```

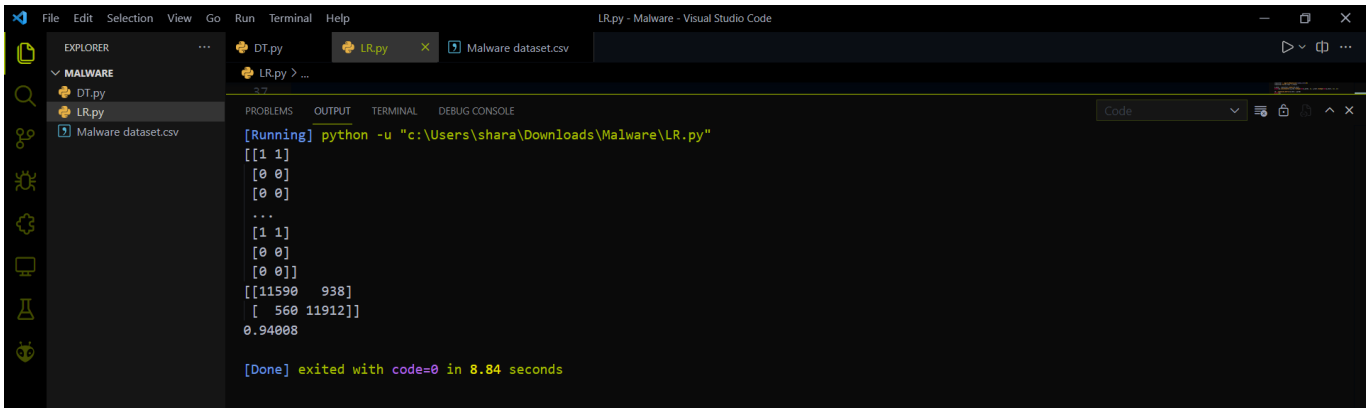
Logistic Regression:

```
#Sharadindu Adhikari, 19BCE2105

from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

d1 = pd.read_csv(r'C:\Users\shara\Downloads\Malware\Malware dataset.csv')
d2 = d1.copy()
d2 = d1.drop(['classification'], axis=1)

X = d2.iloc[:, 1:].values
y = d1.iloc[:, 2].values

l = LabelEncoder()
y = l.fit_transform(y)

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.25, random_state=0)

sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

classifier = LogisticRegression(random_state=0)
classifier.fit(X_train, y_train)

y_pred = classifier.predict(X_test)
print(np.concatenate((y_pred.reshape(len(y_pred), 1), y_test.reshape(len(y_test),
1)), 1))


cm = confusion_matrix(y_test, y_pred)
print(cm)
print(accuracy_score(y_test, y_pred))
```



```python
#Sharadindu Adhikari, 19BCE2105

from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

d1 = pd.read_csv(r'C:\Users\shara\Downloads\Malware\Malware dataset.csv')
d2 = d1.copy()
d2 = d1.drop(['classification'], axis=1)

X = d2.iloc[:, 1:].values
y = d1.iloc[:, 2].values

l = LabelEncoder()
y = l.fit_transform(y)

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.25, random_state=0)

sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

classifier = LogisticRegression(random_state=0)
classifier.fit(X_train, y_train)

y_pred = classifier.predict(X_test)
```

Predicting the Test set results, Confusion Matrix and Accuracy:



```
[Running] python -u "c:\Users\shara\Downloads\Malware\LR.py"
[[1 1]
 [0 0]
 [0 0]
 ...
 [1 1]
 [0 0]
 [0 0]]
[[11590   938]
 [  560 11912]]
0.94008

[Done] exited with code=0 in 8.84 seconds
```

sharadindu.adhikari2019@vitstudent.ac.in

Comparison of Decision Tree and Logistic Regression:

	DT	LR
Confusion matrix	$\begin{Vmatrix} 12528 & 0 \\ 1 & 12471 \end{Vmatrix}$	$\begin{Vmatrix} 11590 & 938 \\ 560 & 11912 \end{Vmatrix}$
Accuracy (in percentage)	**99.996**	**94.008**
Avg Precision recall score (range: [0,1])	**1.00**	**0.91**

Experiment 1: Wireshark DNS

Part 1: nslookup

1. **Run *nslookup* to obtain the IP address of a Web server in Asia. What is the IP address of that server?**

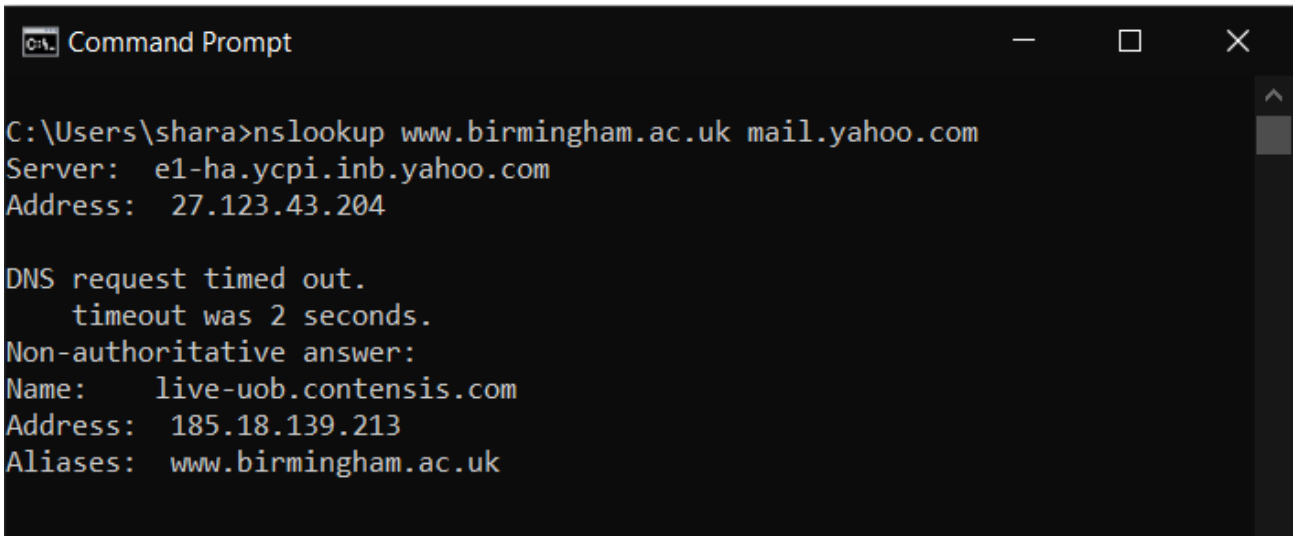
   For the question, I queried the webpage of Vellore Institute of Technology (https://www.vit.ac.in). The IP address of that server was 136.233.9.13

2. **Run *nslookup* to determine the authoritative DNS servers for a university in Europe.**

   For this question, I used the webpage for University of Birmingham in England. The url is: https://www.birmingham.ac.uk

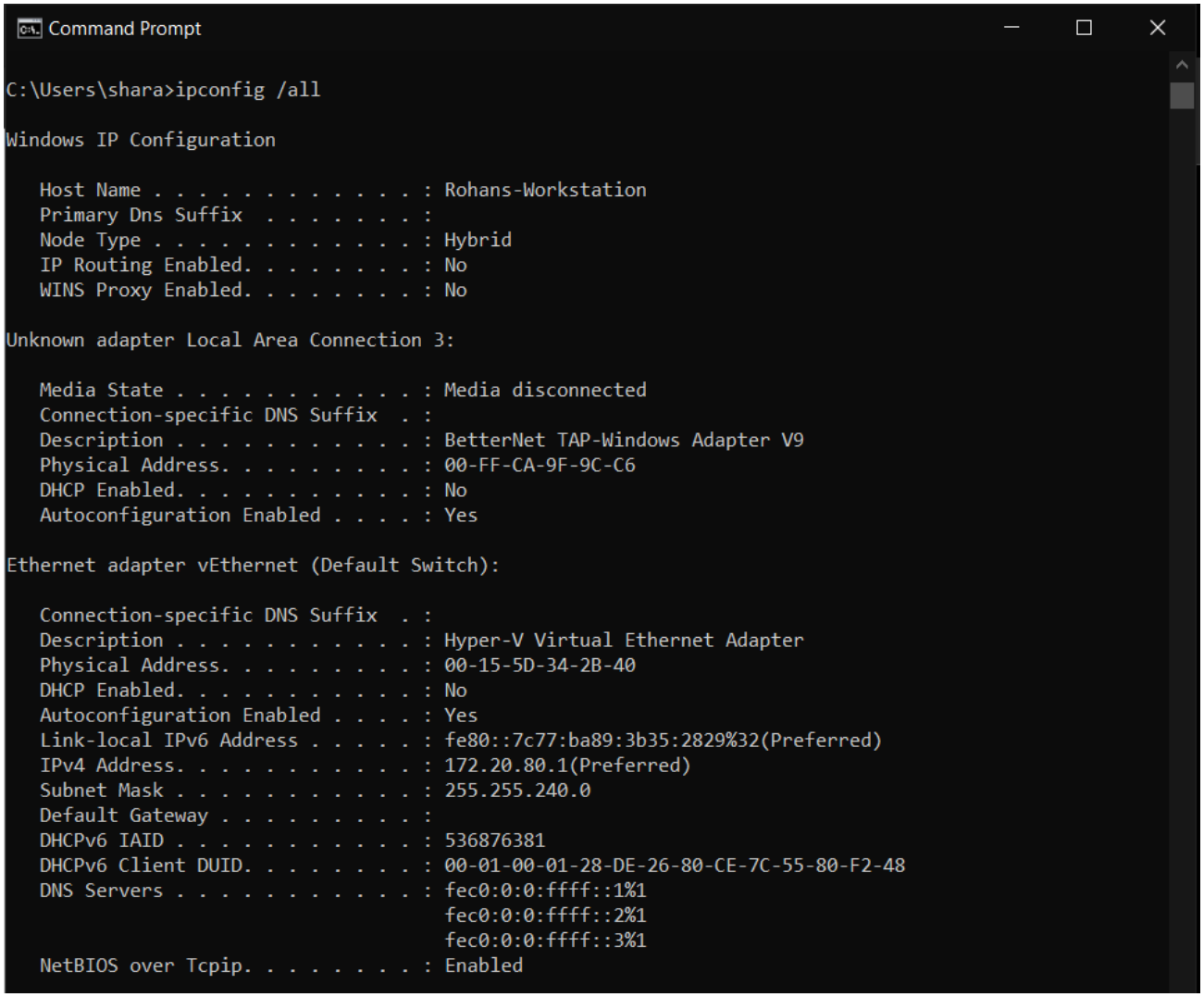
3. **Run *nslookup* so that one of the DNS servers obtained in Question 2 is queried for the mail servers for Yahoo! mail. What is its IP address?**

   The IP address for the DNS server if queried for the Yahoo! mail server is 27.123.43.204



Part 2: ipconfig

There are no questions for part two of this lab. All it requires is that we run *ipconfig /all* on our current machine. This will display my machines current TCP/IP information, including my IP address, DNS server address and other additional information.

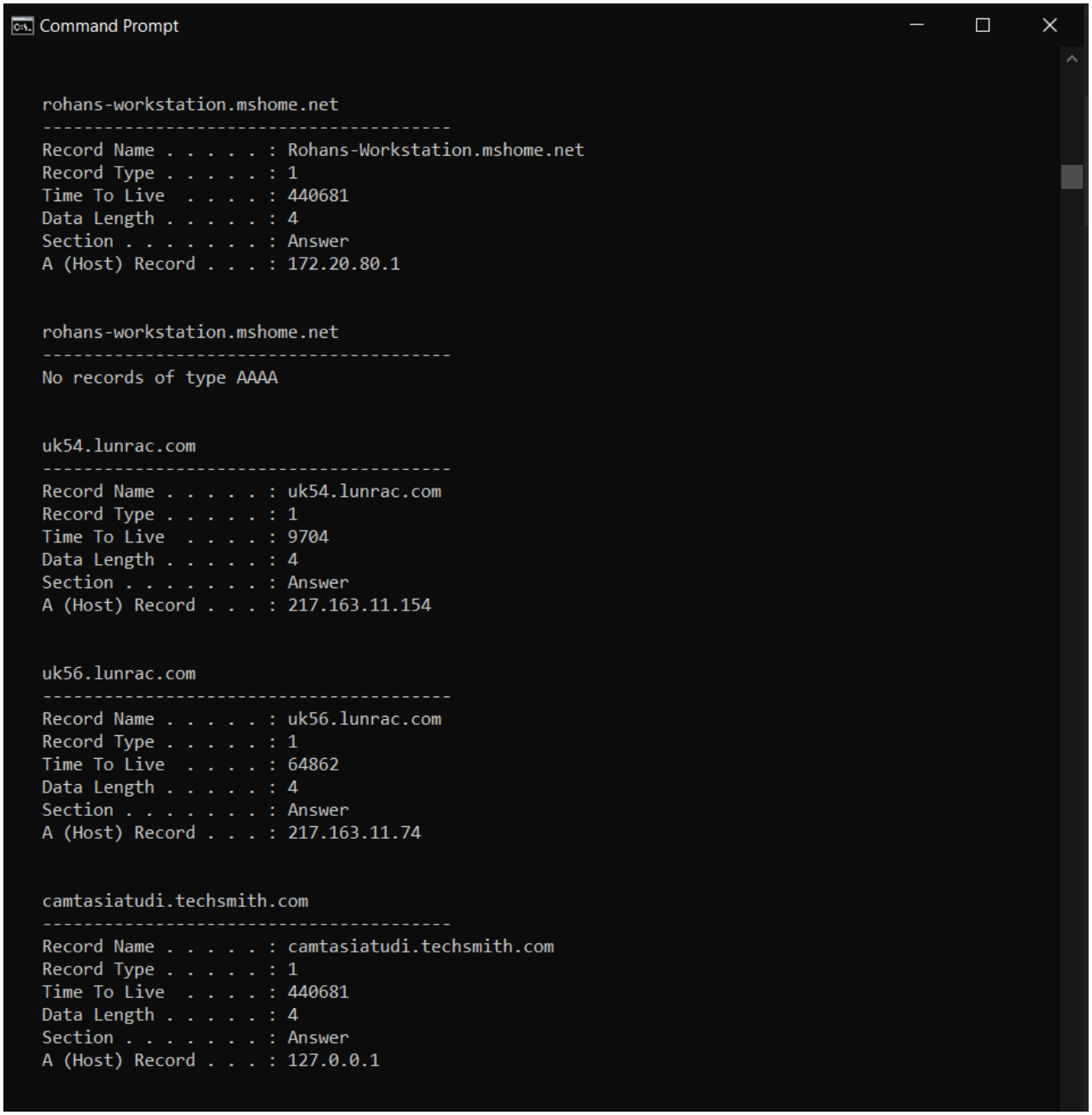


```
Wireless LAN adapter Local Area Connection* 9:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Microsoft Wi-Fi Direct Virtual Adapter
   Physical Address. . . . . . . . . : 28-3A-4D-36-BE-07
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes

Wireless LAN adapter Local Area Connection* 10:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Microsoft Wi-Fi Direct Virtual Adapter #2
   Physical Address. . . . . . . . . : AA-3A-4D-36-BE-07
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes

Ethernet adapter VMware Network Adapter VMnet1:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : VMware Virtual Ethernet Adapter for VMnet1
   Physical Address. . . . . . . . . : 00-50-56-C0-00-01
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::946c:32d8:c77e:b1e9%13(Preferred)
   IPv4 Address. . . . . . . . . . . : 192.168.254.1(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Lease Obtained. . . . . . . . . . : 12 October 2021 9.13.32 AM
   Lease Expires . . . . . . . . . . : 12 October 2021 11.28.32 AM
   Default Gateway . . . . . . . . . :
   DHCP Server . . . . . . . . . . . : 192.168.254.254
   DHCPv6 IAID . . . . . . . . . . . : 620777558
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-28-DE-26-80-CE-7C-55-80-F2-48
   DNS Servers . . . . . . . . . . . : fec0:0:0:ffff::1%1
                                       fec0:0:0:ffff::2%1
                                       fec0:0:0:ffff::3%1
   NetBIOS over Tcpip. . . . . . . . : Enabled

Ethernet adapter VMware Network Adapter VMnet8:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : VMware Virtual Ethernet Adapter for VMnet8
   Physical Address. . . . . . . . . : 00-50-56-C0-00-08
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::acad:81d2:f4da:2392%12(Preferred)
   IPv4 Address. . . . . . . . . . . : 192.168.109.1(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Lease Obtained. . . . . . . . . . : 12 October 2021 9.13.32 AM
   Lease Expires . . . . . . . . . . : 12 October 2021 11.28.32 AM
   Default Gateway . . . . . . . . . :
   DHCP Server . . . . . . . . . . . : 192.168.109.254
   DHCPv6 IAID . . . . . . . . . . . : 637554774
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-28-DE-26-80-CE-7C-55-80-F2-48
   DNS Servers . . . . . . . . . . . : fec0:0:0:ffff::1%1
                                       fec0:0:0:ffff::2%1
                                       fec0:0:0:ffff::3%1
   Primary WINS Server . . . . . . . : 192.168.109.2
   NetBIOS over Tcpip. . . . . . . . : Enabled

Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Realtek 8822BE Wireless LAN 802.11ac PCI-E NIC
   Physical Address. . . . . . . . . : 28-3A-4D-36-BE-07
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
```

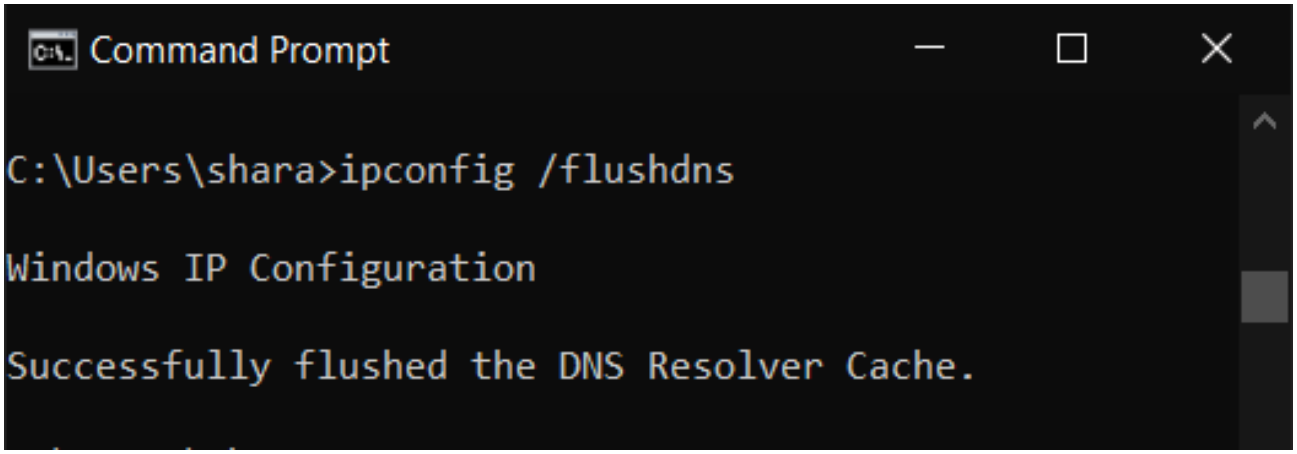
```
   Link-local IPv6 Address . . . . . : fe80::b99f:f0cd:bc55:18ac%22(Preferred)
   IPv4 Address. . . . . . . . . . . : 172.22.54.131(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Lease Obtained. . . . . . . . . . : 12 October 2021 9.13.46 AM
   Lease Expires . . . . . . . . . . : 12 October 2021 12.13.44 PM
   Default Gateway . . . . . . . . . : 172.22.54.1
   DHCP Server . . . . . . . . . . . : 172.22.54.1
   DHCPv6 IAID . . . . . . . . . . . : 438843981
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-28-DE-26-80-CE-7C-55-80-F2-48
   DNS Servers . . . . . . . . . . . : 172.22.54.1
   NetBIOS over Tcpip. . . . . . . . : Enabled

Ethernet adapter Bluetooth Network Connection:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Bluetooth Device (Personal Area Network)
   Physical Address. . . . . . . . . : 28-3A-4D-36-BE-08
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
```

It then asks that we display our recent cached memory by using the command *ipconfig /displaydns*

```
Command Prompt

C:\Users\shara>ipconfig /displaydns

Windows IP Configuration

    safebrowsing.googleapis.com
    ----------------------------------------
    Record Name . . . . . : safebrowsing.googleapis.com
    Record Type . . . . . : 1
    Time To Live  . . . . : 113
    Data Length . . . . . : 4
    Section . . . . . . . : Answer
    A (Host) Record . . . : 142.250.67.202


    be7.lunrac.com
    ----------------------------------------
    Record Name . . . . . : be7.lunrac.com
    Record Type . . . . . : 1
    Time To Live  . . . . : 61897
    Data Length . . . . . : 4
    Section . . . . . . . : Answer
    A (Host) Record . . . : 185.213.22.12


    edge.microsoft.com
    ----------------------------------------
    Record Name . . . . . : edge.microsoft.com
    Record Type . . . . . : 5
    Time To Live  . . . . : 79
    Data Length . . . . . : 8
    Section . . . . . . . : Answer
    CNAME Record  . . . . : edge-microsoft-com.a-0016.a-msedge.net


    Record Name . . . . . : edge-microsoft-com.a-0016.a-msedge.net
    Record Type . . . . . : 5
    Time To Live  . . . . : 79
    Data Length . . . . . : 8
    Section . . . . . . . : Answer
    CNAME Record  . . . . : a-0016.a-msedge.net
```



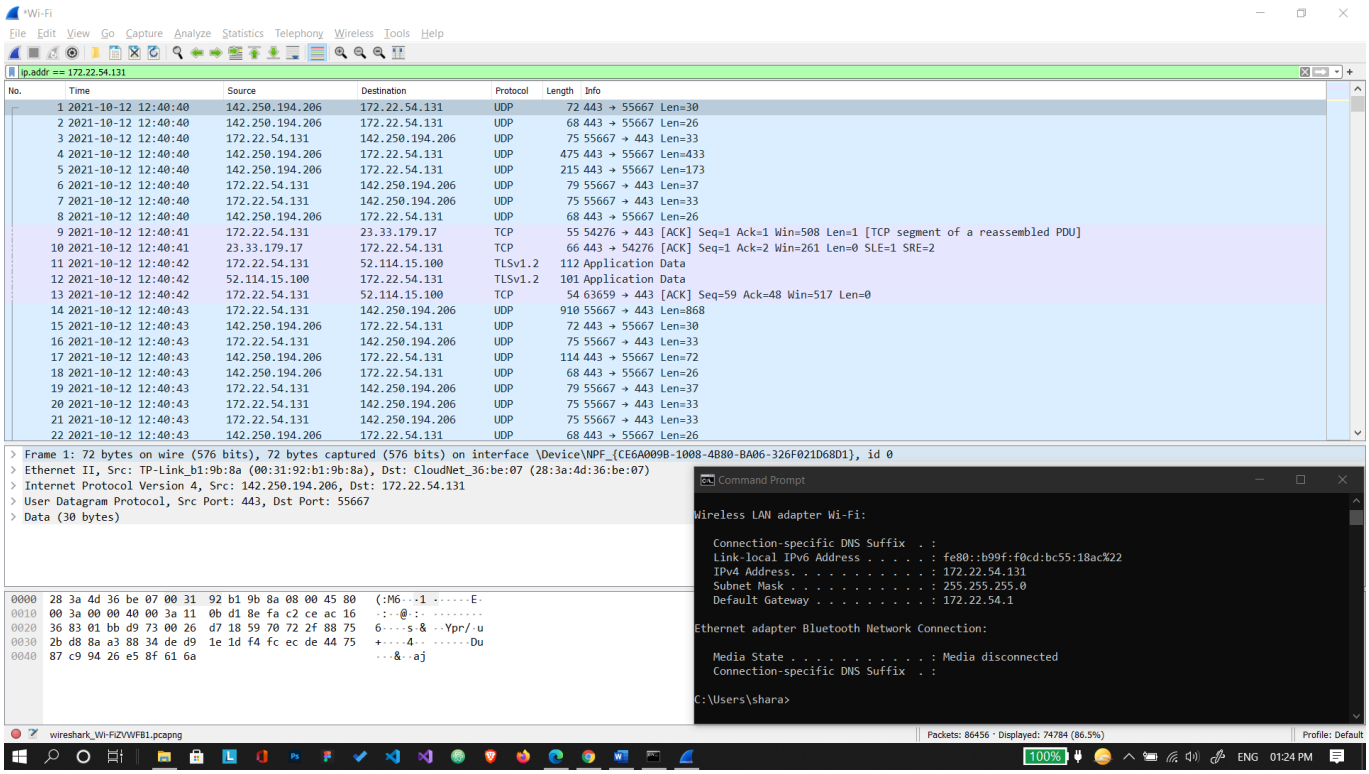
Finally, we are told to clear the above cache by entering *ipconfig /flushdns*



Part 3: Tracing DNS with Wireshark

Section 1 – Steps: *ipconfig*

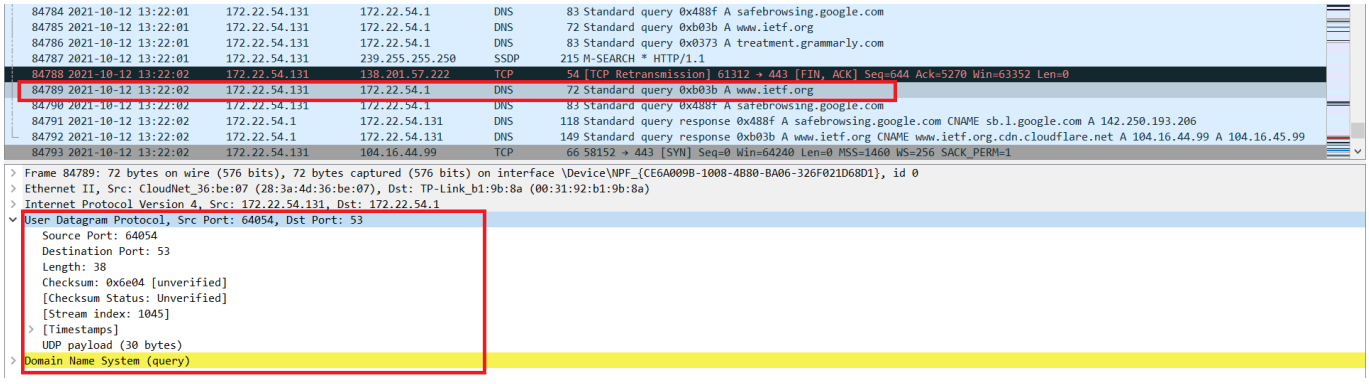
Step 1: Use ipconfig to empty the DNS cache in our host.
Step 2: Open the default browser and empty the browser cache
Step 3: Open Wireshark and enter "*ip.addr == 172.22.54.131*" into the filter, where I've obtained my default ipv4 address using *ipconfig* from cmd. This filter removes all packets that neither originate nor are destined to my host.
Step 4: Start packet capture in Wireshark.
Step 5: With browser, visit the web page: http://www.ietf.org
Step 6: Stop packet capture.
Step 7: Take a screenshot.



Back to questions:

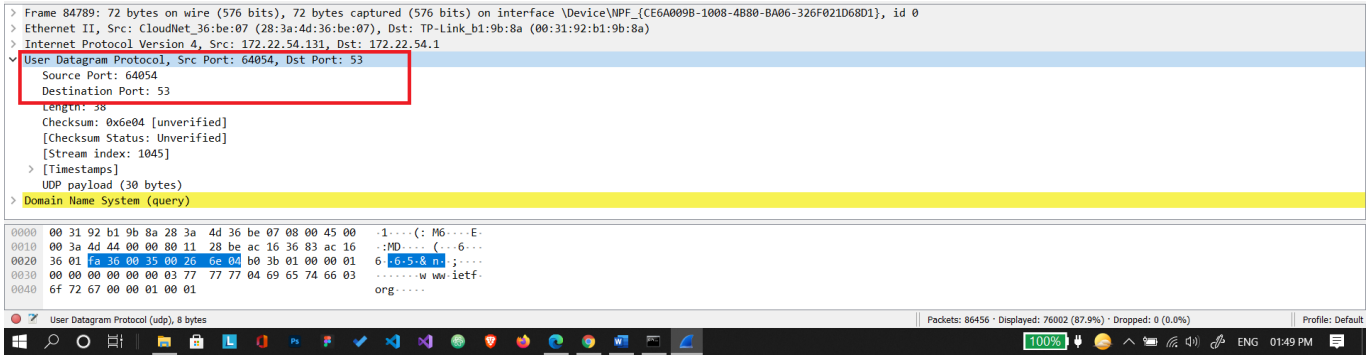
4. **Locate the DNS query and response messages. Are then sent over UDP or TCP?**

   The DNS query and response messages are sent over UDP.



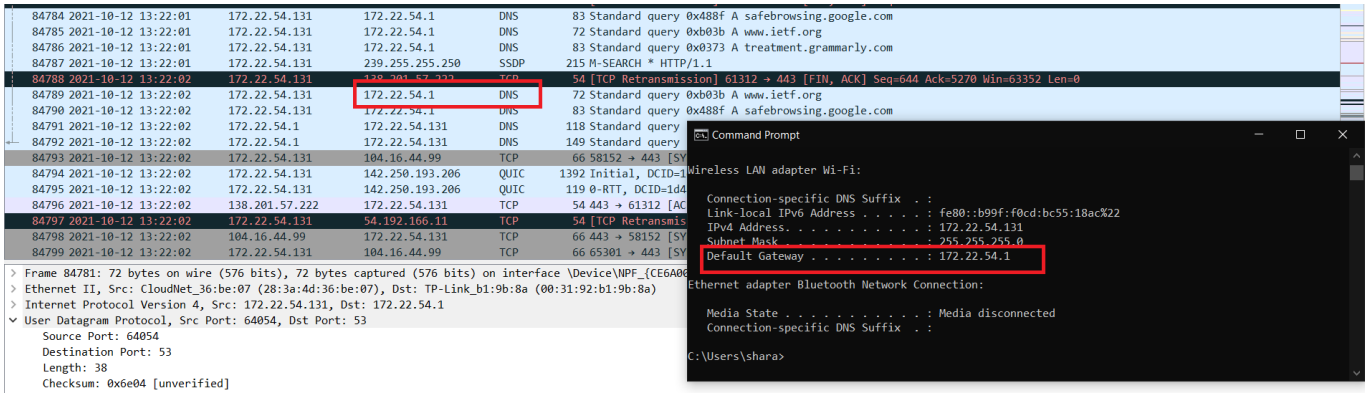
5. **What is the destination port for the DNS query message? What is the source port of DNS response message?**

   The destination port is 53
   The source port is 64054



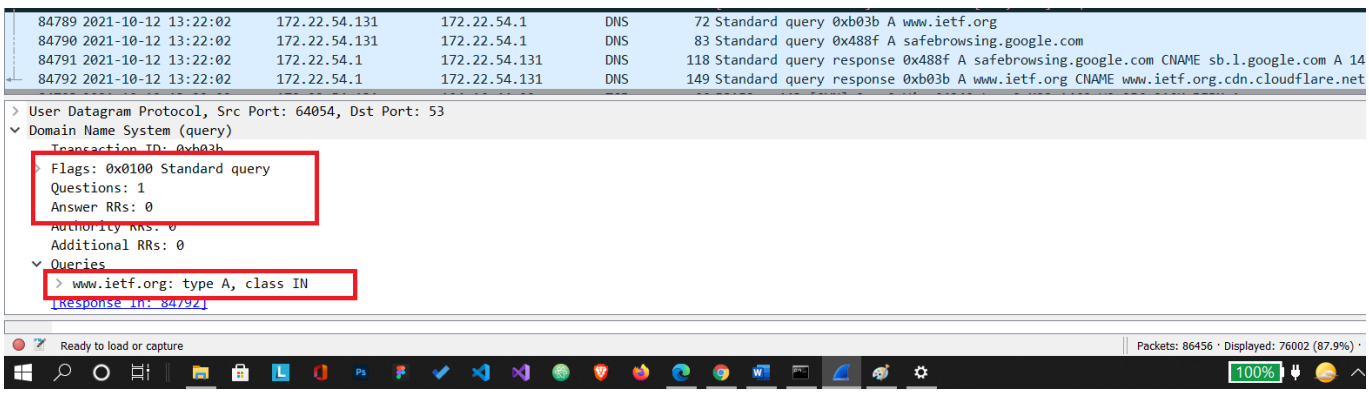
6. **To what IP address is the DNS query message sent? Use *ipconfig* to determine the IP address of your local DNS server. Are these two IP addresses the same?**

   The DNS query was sent to the address 172.22.54.1. Yes, it is the same IP address as that of my local DNS server (aka the Default Gateway).



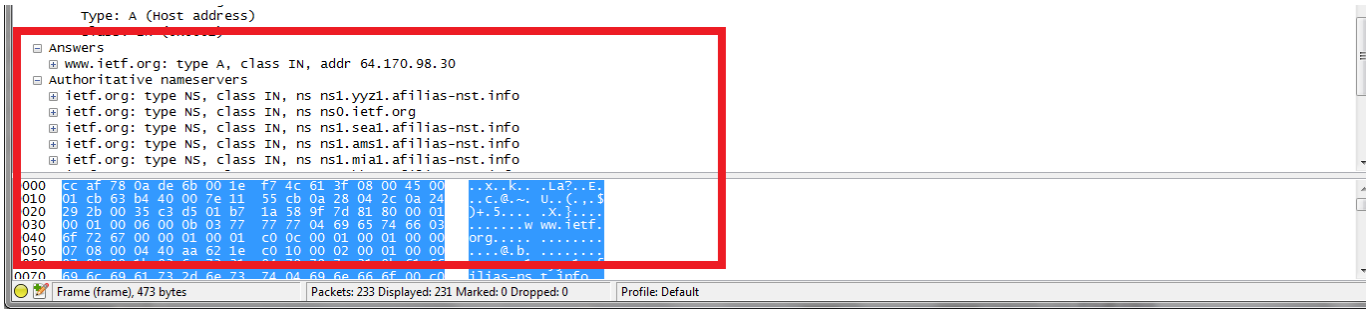
7. **Examine the DNS query message. What "Type" of DNS query is it? Does the query message contain any "answers"?**

   The query message was a type "A" query, but the message did not contain any "answers".



8. **Examine the DNS response message. How many "answers" are provided? What do each of these answers contain?**

   The response message contained one answer to the query which was the sites address [64.170.98.30]. Although it also provided 6 authoritative nameservers, and 11 other responses containing additional information.

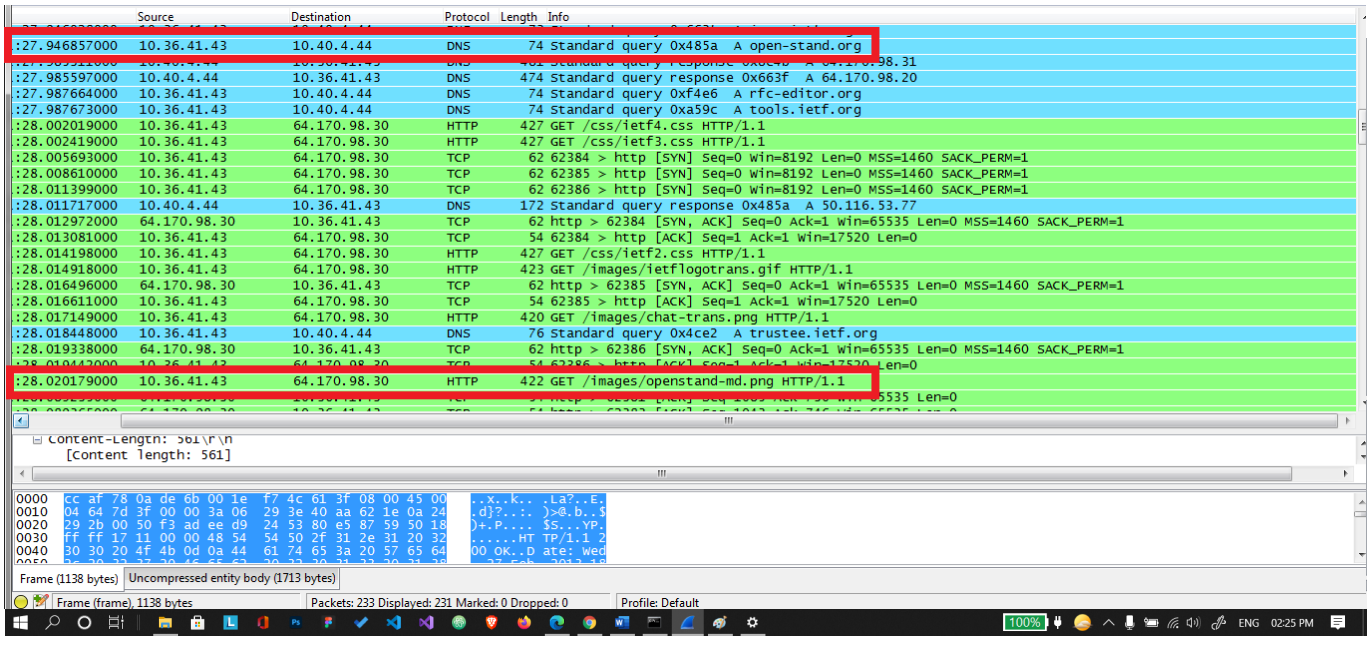


9. **Consider the subsequent TCP SYN packet sent by your host. Does the destination IP address of the SYN packet correspond to any of the IP addresses provided in the DNS response message?**

   The destination of the SYN packet is 64.170.98.30, the same address that was provided in the DNS response message as the type "A" address of the webpage.

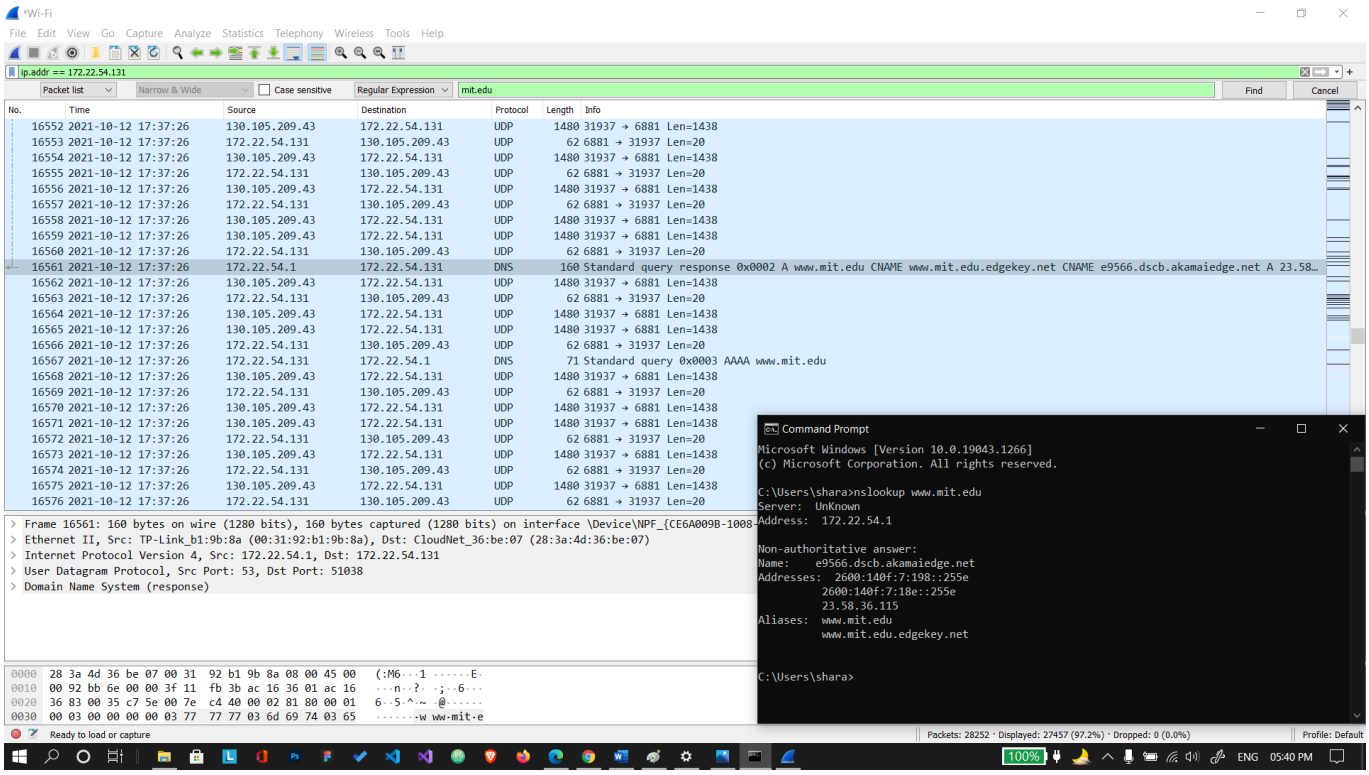
10. **This web page contains images. Before retrieving each image, does your host issue new DNS queries?**

    Yes, my host did issue new DNS queries before the images were retrieved. For example, one such query was for an image from [open-stand.org](). The image corresponding to the page had not returned until this query was made.



Section 2 – Steps: *nslookup*

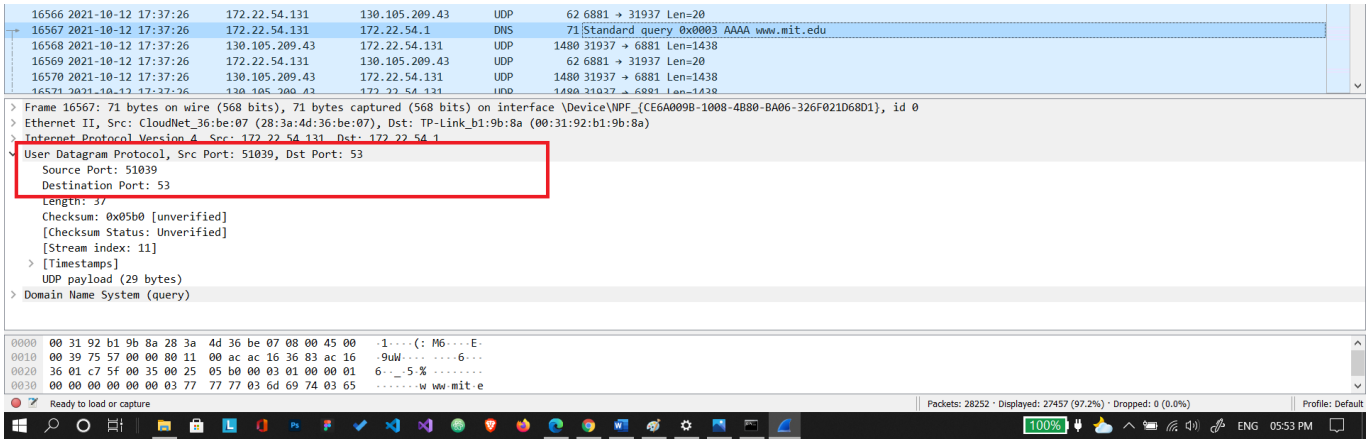
Step 1: Start packet capture.
Step 2: Do an *nslookup* on http://www.mit.edu
Step 3: Stop packet capture.
Step 4: Take a screenshot.



Back to questions:

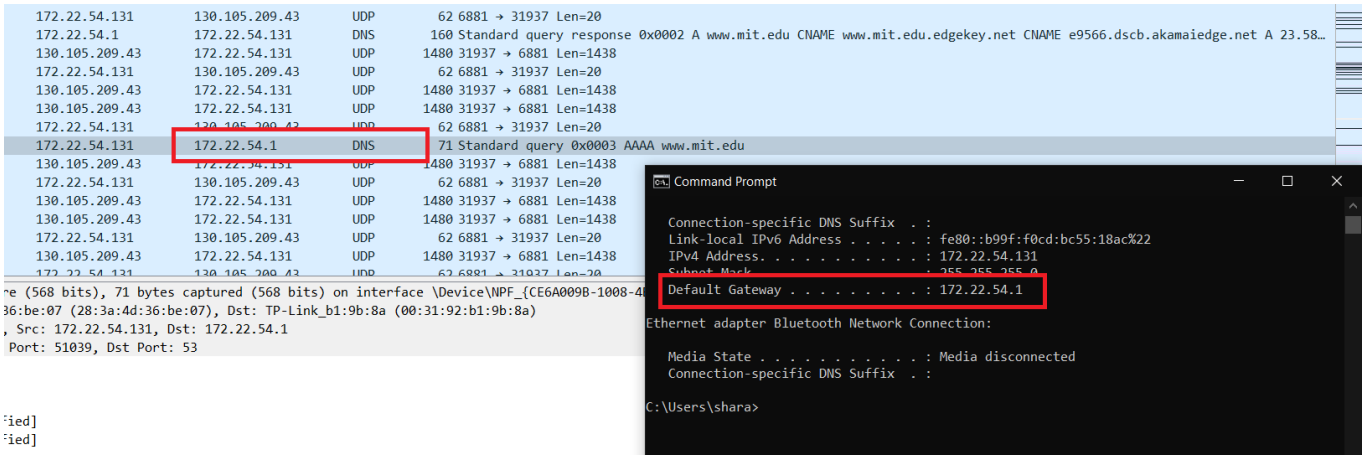
11. **What is the destination port for the DNS query message? What is the source port of DNS response message?**

    Destination Port: 53
    Source Port: 51039



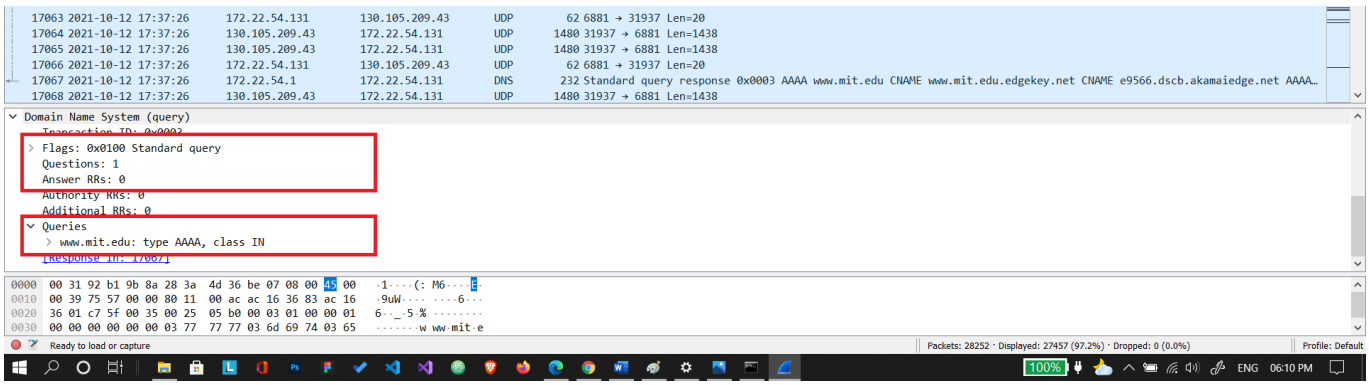
12. **To what IP address is the DNS query message sent? Is this the IP address of your default local DNS server?**

    The DNS query message is sent to IP address 172.22.54.1, the same address as my default local DNS server (aka the Default Gateway).



13. **Examine the DNS query message. What "Type" of DNS query is it? Does the query message contain any "answers"?**

    The DNS query message is a type "A" query, containing only one question and not containing any answers.



14. **Examine the DNS response message. How many "answers" are provided? What do each of these answers contain?**

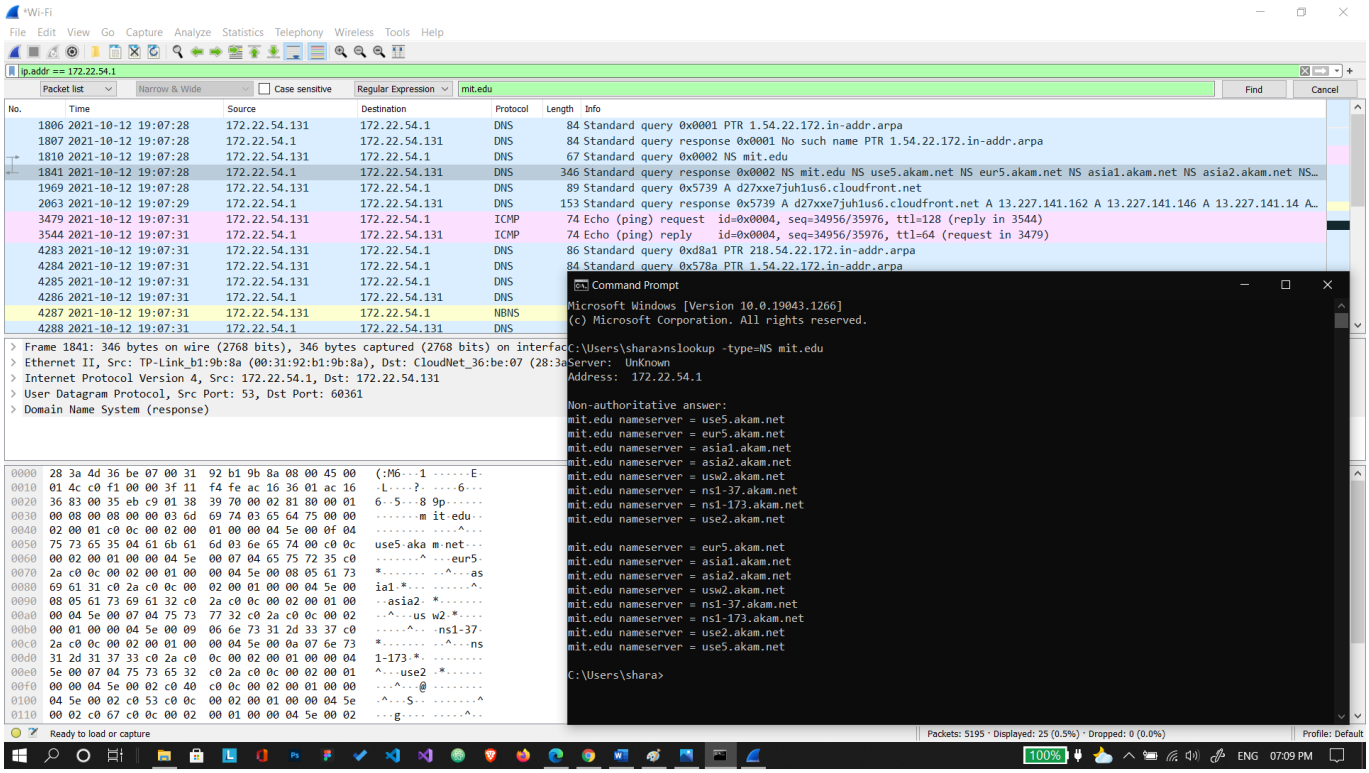
    The response message contains one answer to the aforementioned query which is the type "A" address of http://www.mit.edu or 18.9.22.169. It also contained information on 3 authoritative nameservers and 3 additional records.

15. **Provide a screenshot.**

    Provided every step of the way.

Section 2.1 – Steps:

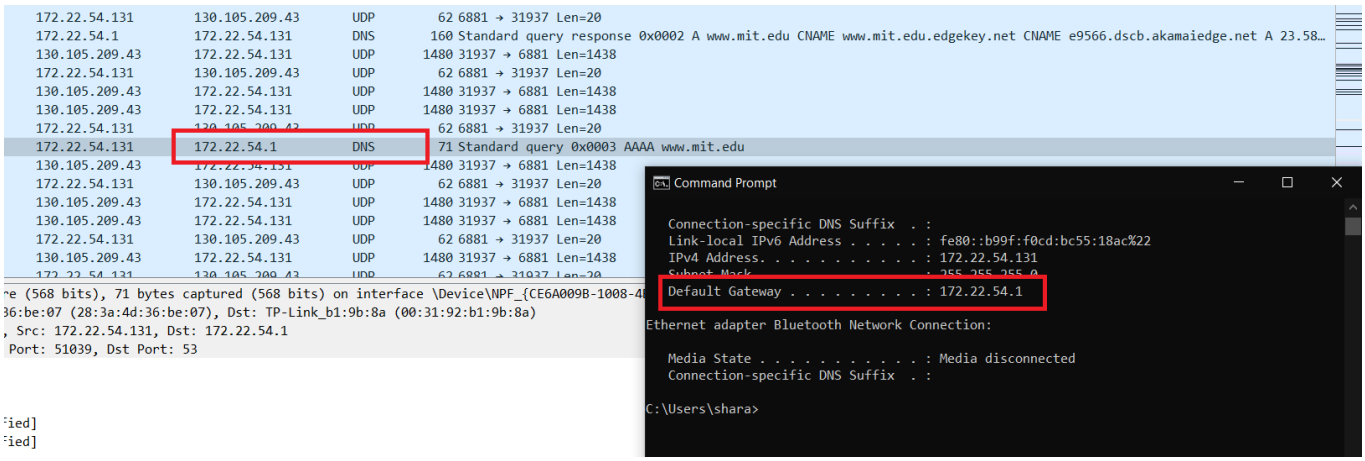
Step 1: Repeat the previous experiment.
Step 2: But instead issue the command: *nslookup –type=NS mit.edu*
Step 3: Take a screenshot.



Back to questions:

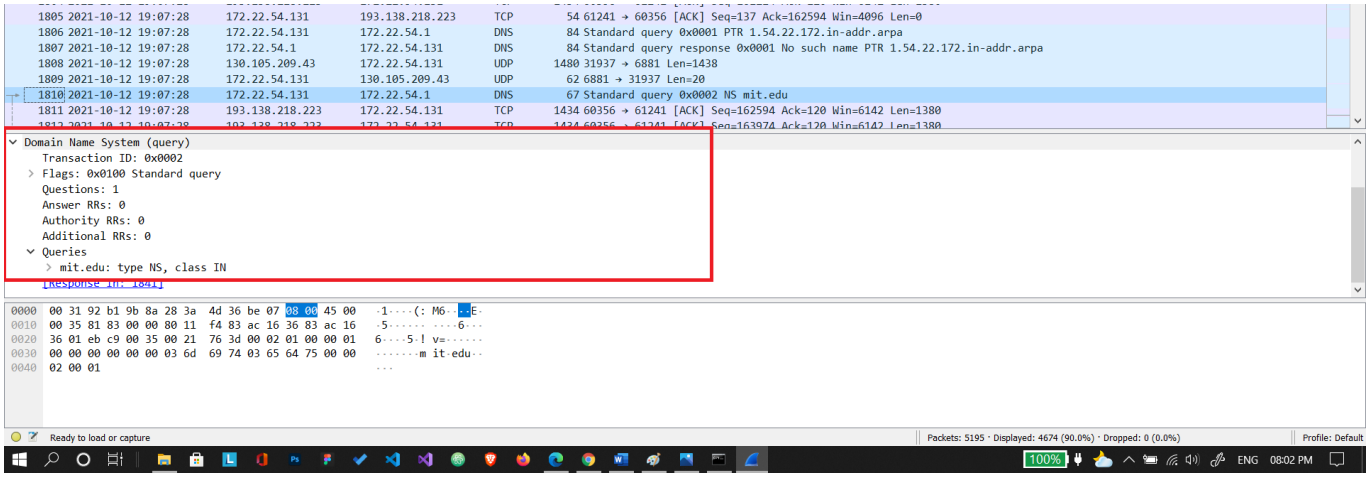
16. **To what IP address is the DNS query message sent? Is this the IP address of your default local DNS server?**

    The DNS query message is sent to IP address 172.22.54.1. And yes, it is the same address as my default local DNS server (aka the Default Gateway).



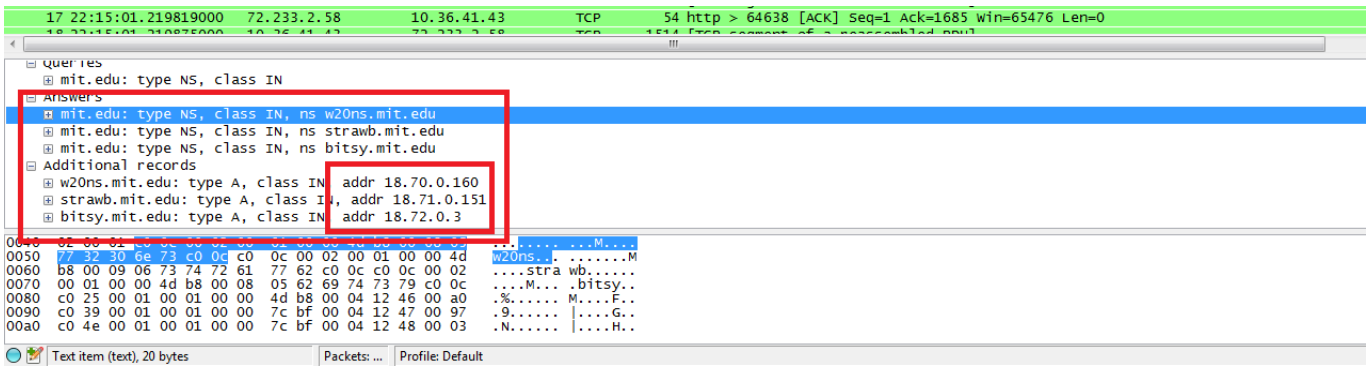
17. **Examine the DNS query message. What "Type" of DNS query is it? Does the query message contain any "answers"?**

    The DNS query is a type "NS" message including one question. The query message did not contain any answers.



18. **Examine the DNS response message. What MIT nameservers does the response message provide? Does this response message also provide the IP addresses of the MIT nameservers?**

    The response message provides 3 MIT nameservers: w20ns.mit.edu [18.70.0.160], strawb.mit.edu [18.71.0.150], and bitsy.mit.edu [18.72.0.3]. The IP addresses for the nameservers were included under the additional records category sent back as part of the response message.

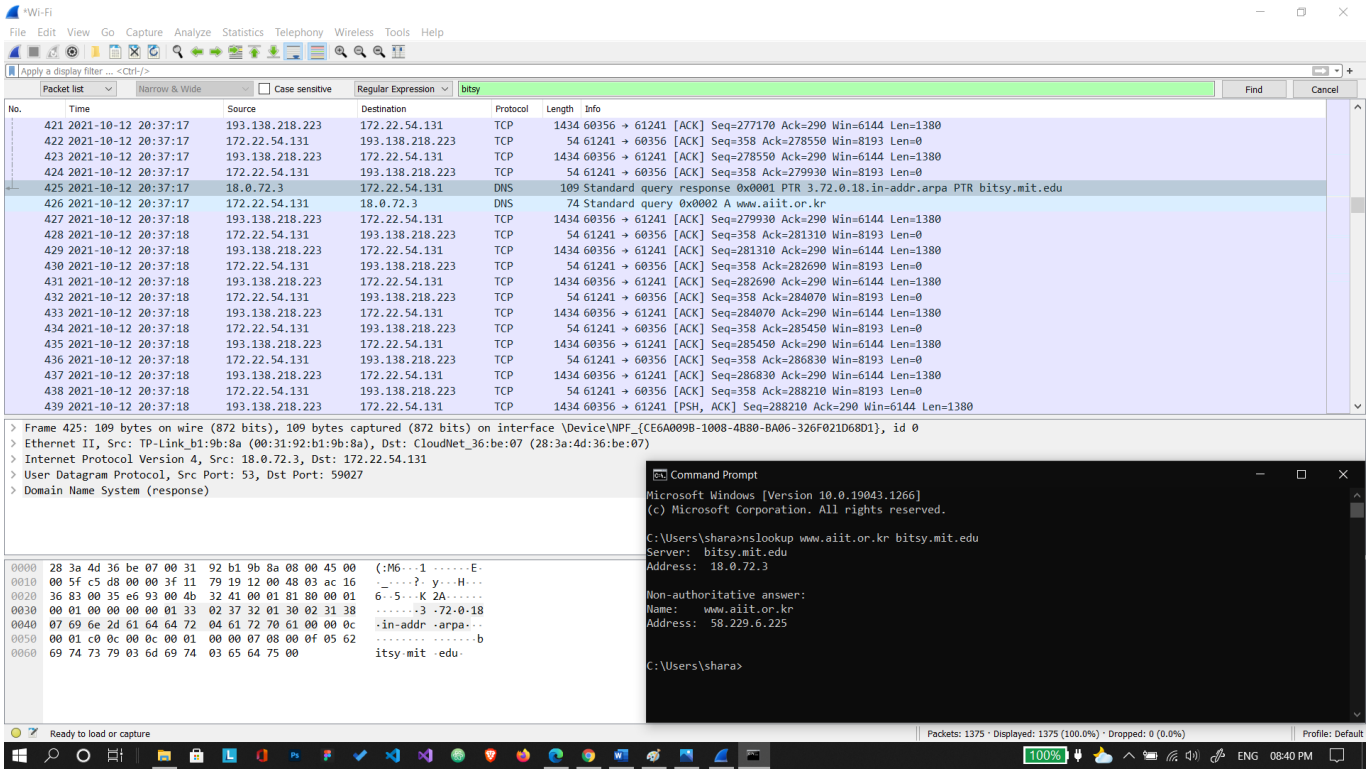


19. **Provide a screenshot.**

    Provided every step of the way.

Section 2.2 – Steps:

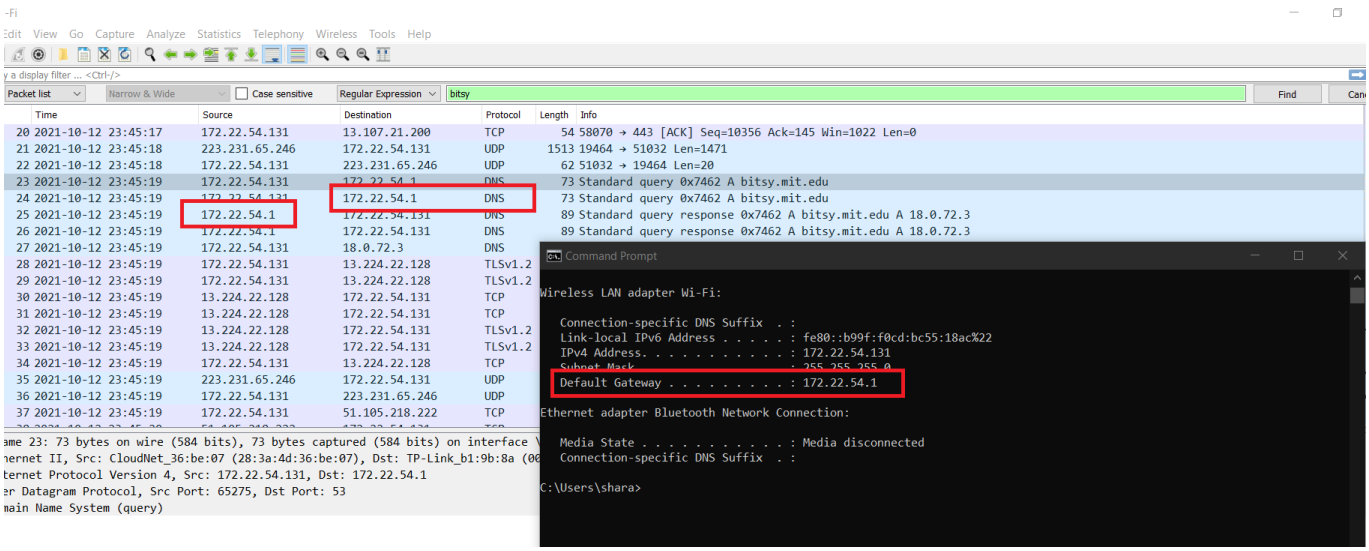
Step 1: Repeat the previous experiment.
Step 2: But instead issue the command: *nslookup www.aiit.or.kr bitsy.mit.edu*
Step 3: Take a screenshot.



Back to questions:

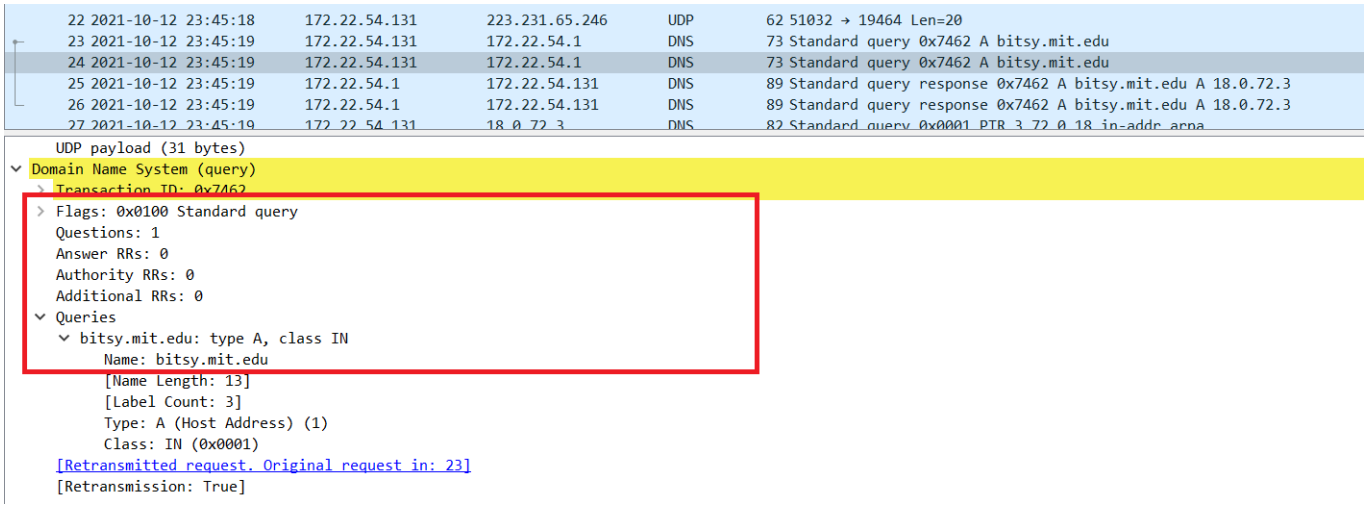
20. **To what IP address is the DNS query message sent? Is this the IP address of your default local DNS server? If not, what does the IP address correspond to?**

    The DNS query message is sent to IP address 172.22.54.1, the same address as my default local DNS server (aka the Default Gateway).



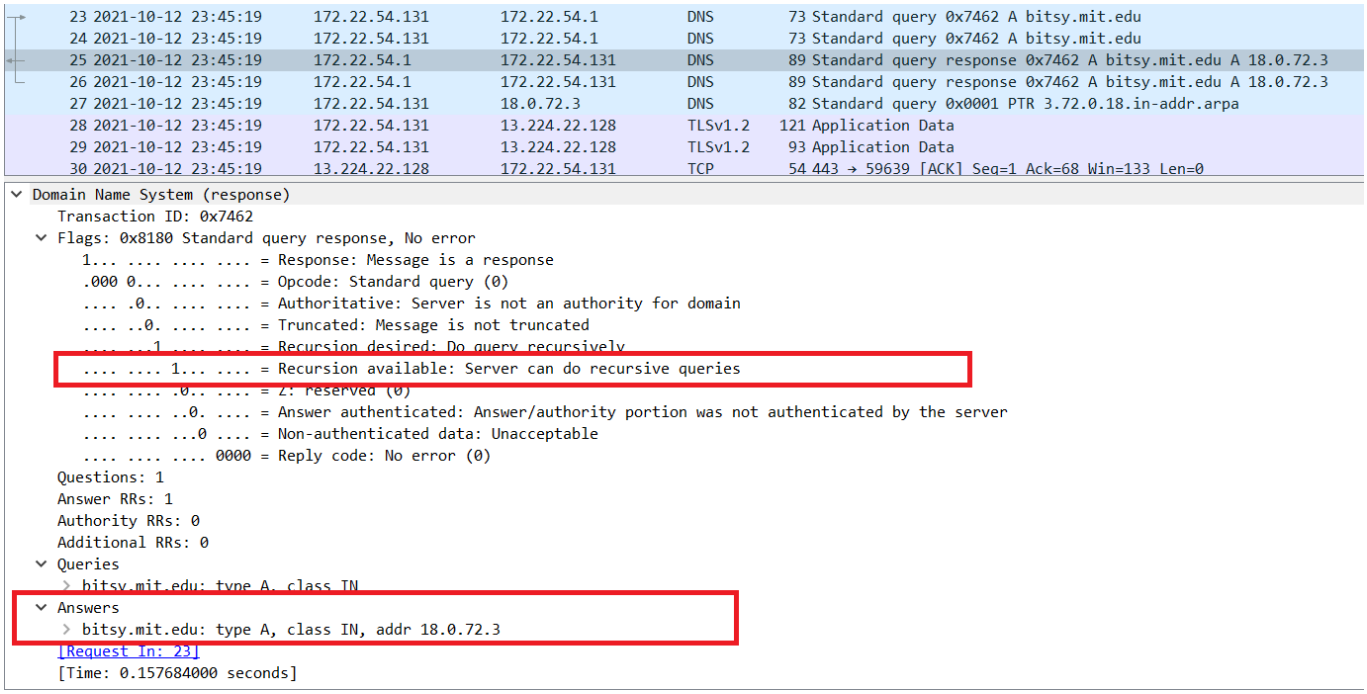
21. **Examine the DNS query message. What "Type" of DNS query is it? Does the query message contain any "answers"?**

    This DNS query is a type "A" query. The message does not contain any answers.



22. **Examine the DNS response message. How many "answers" are provided? What does each of these answers contain?**

    It only provided one "answer" containing the servers IP address, however, the server also returned a flag that stated that it could complete a recursive query.



23. **Provide a screenshot.**

    Provided every step of the way.

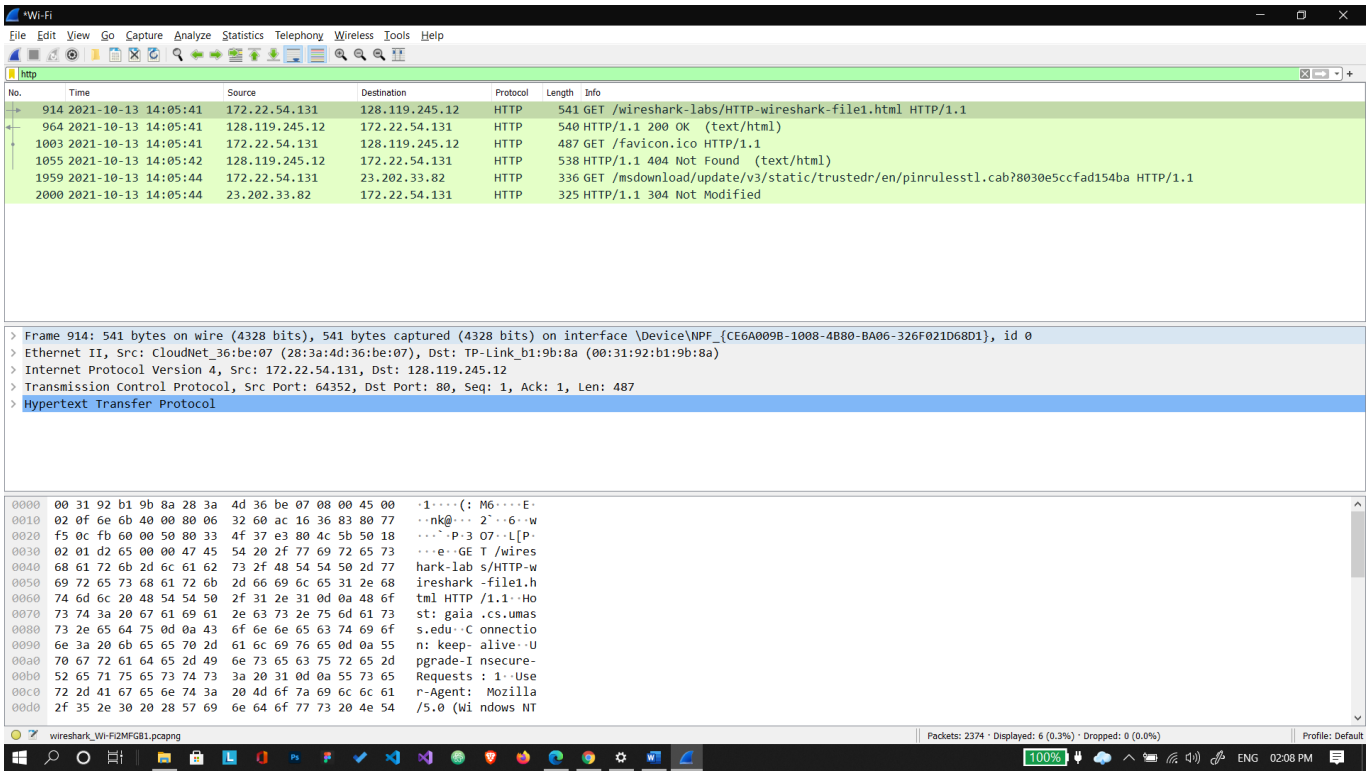
Experiment 2: Wireshark HTTP

Part 1: The Basic HTTP GET/response interaction

Steps:

Step 1: Start up the web browser.
Step 2: Start up the Wireshark packet sniffer. Enter “http” in the display-filter-specification window, so that only captured HTTP messages will be displayed later in the packet-listing window.
Step 3: Wait a bit more than one minute (we’ll see why shortly), and then begin Wireshark packet capture.
Step 4: Enter the following in the browser: http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file1.html
Step 5: Stop Wireshark packet capture.
Step 6: Take a screenshot.

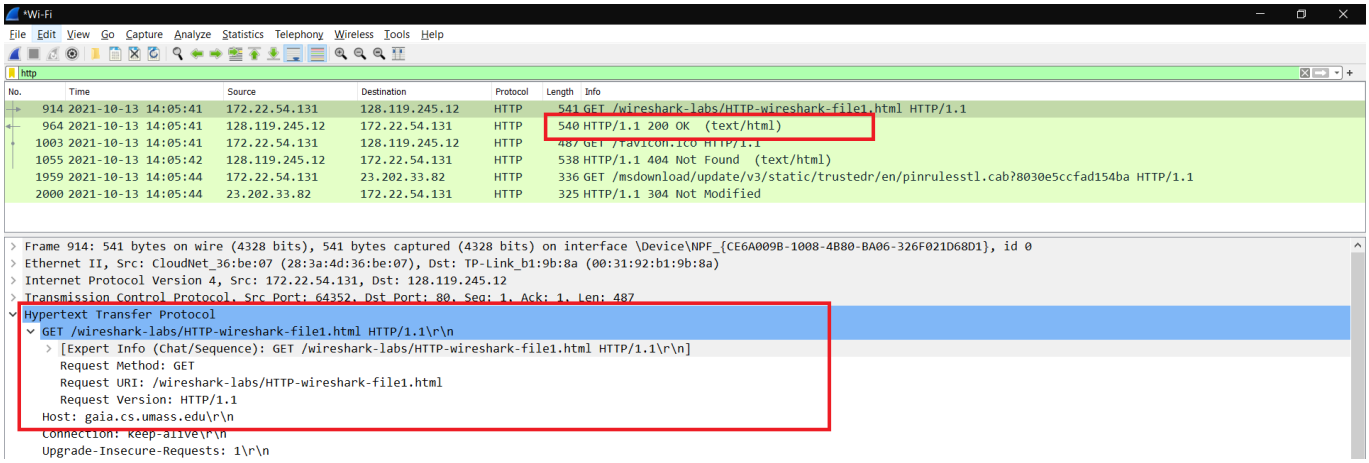
Congratulations. You've downloaded the file http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file1.html!



Back to questions:

1. **Is your browser running HTTP version 1.0 or 1.1? What version of HTTP is the server running?**

   My browser is running http version 1.1
   The server is also running http version 1.1

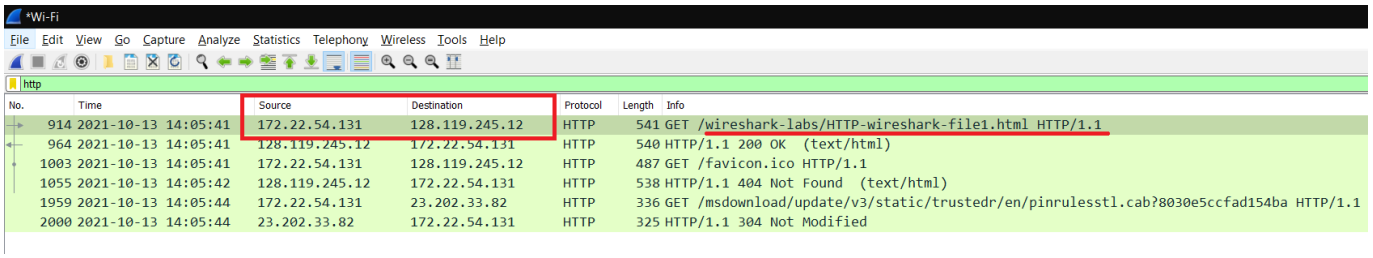


2. What languages (if any) does your browser indicate that it can accept to the server?

My browser indicates that it will accept English-US, English-UK, and other English dialects from the server.

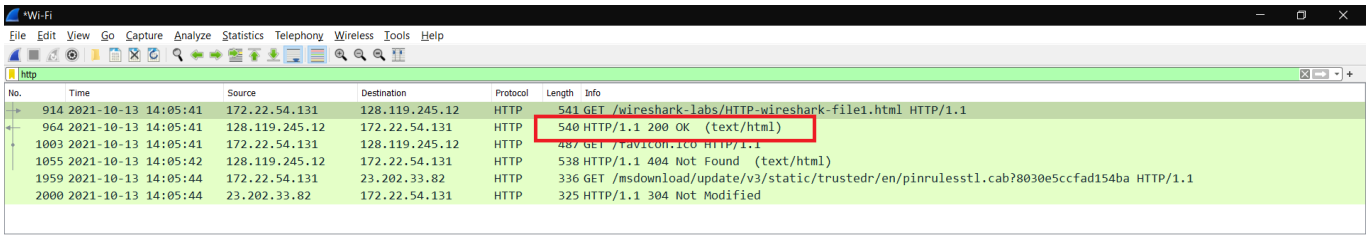
3. What is the IP address of your computer? Of the gaia.cs.umass.edu server?

The IP address of my computer is 172.22.54.131
The IP address of the server is 128.119.245.12



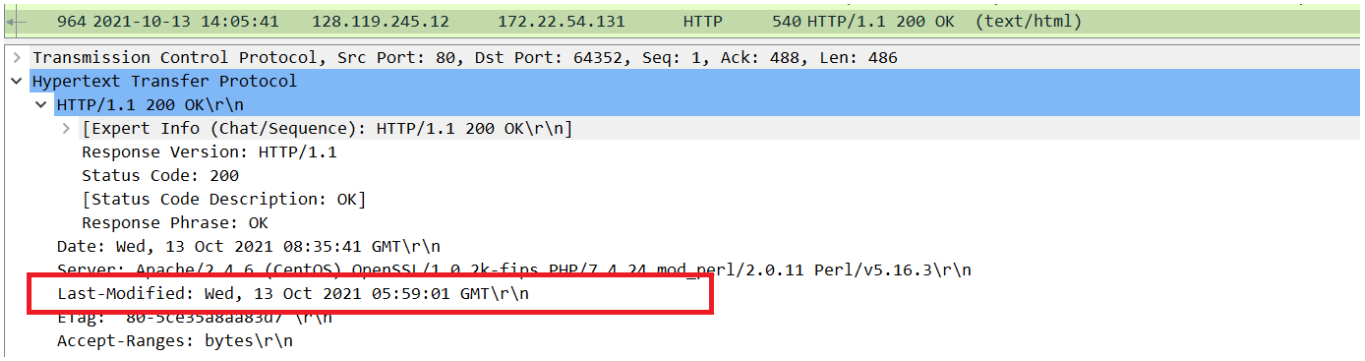
4. What is the status code returned from the server to your browser?

The status code returned was “200 OK”.



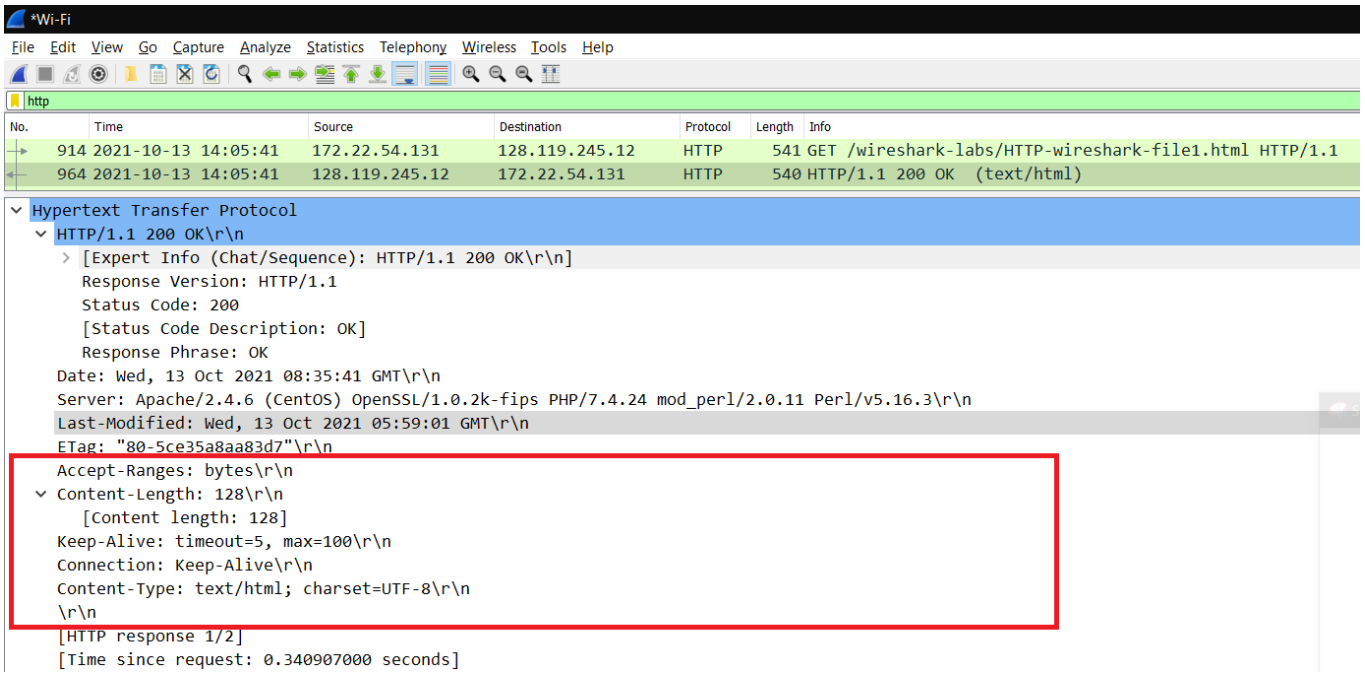
5. When was the HTML file that you are retrieving last modified at the server?

The file was last modified on Wednesday, October 13, 2021 at 05:59:01 GMT



6. How many bytes of content are being returned to your browser?

128 bytes of content are being returned.



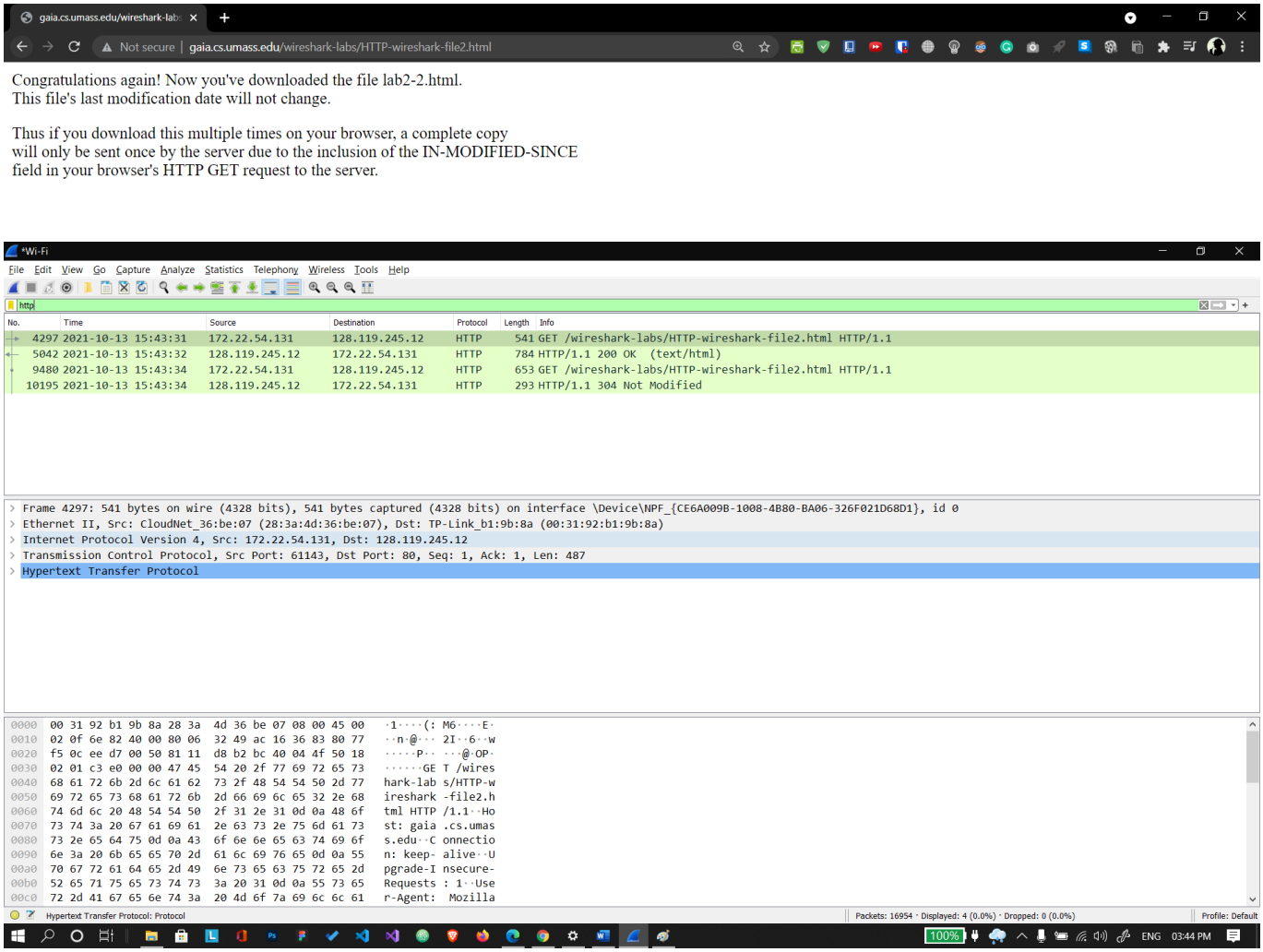
7. By inspecting the raw data in the packet content window, do you see any headers within the data that are not displayed in the packet-listing window? If so, name one.

I do not see any different headings between the two windows.

Part 2: The HTTP CONDITIONAL GET/response interaction

Steps:

Step 1: Start up the web browser and make sure the browser's cache is cleared.
Step 2: Start up the Wireshark packet sniffer.
Step 3: Enter the following URL in the browser: http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file2.html.The browser should display a very simple five-line HTML file.
Step 4: Quickly enter the same URL into the browser again (or simply select the refresh button on the browser).
Step 5: Stop Wireshark packet capture, and enter "http" in the display-filter-specification window, so that only captured HTTP messages will be displayed later in the packet-listing window.
Step 6: Take a screenshot.



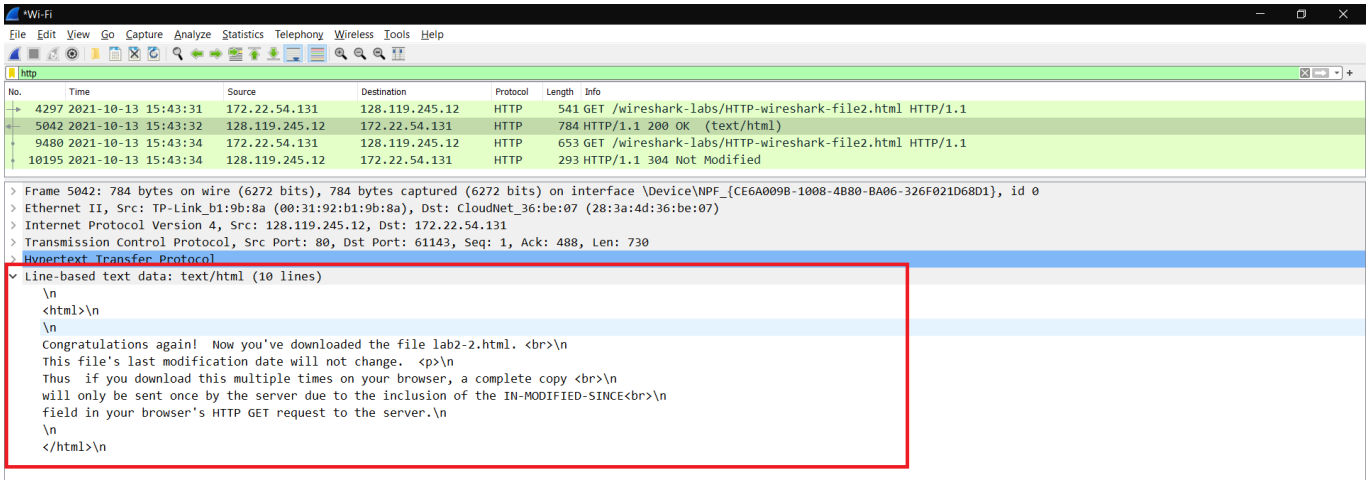
Back to questions:

8. **Inspect the contents of the first HTTP GET request from your browser to the server. Do you see an "IF-MODIFIED-SINCE" line in the HTTP GET?**

   No, there is no IF-MODIFIED-SINCE line in the GET message.

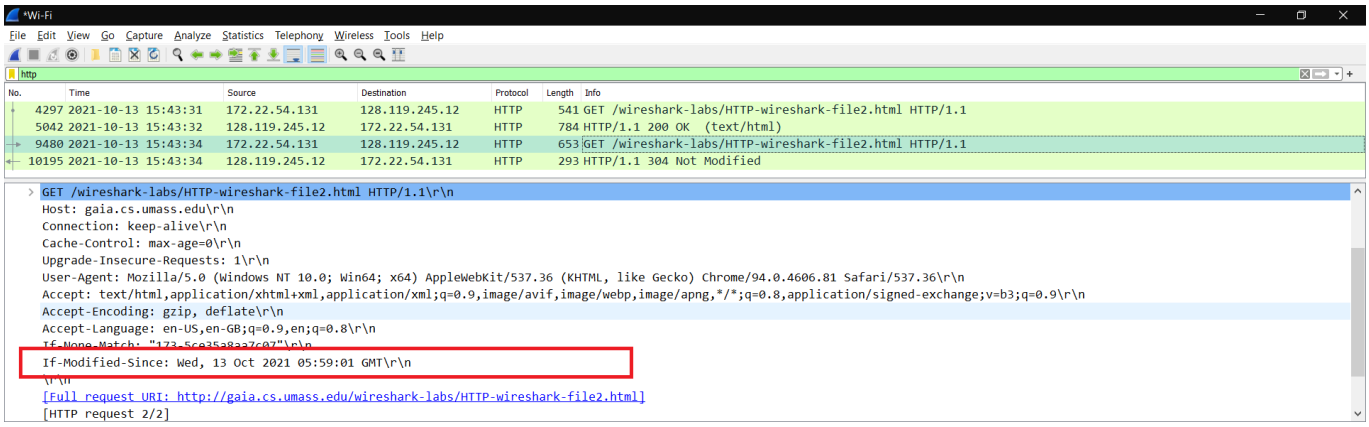
9. Inspect the contents of the server response. Did the server explicitly return the contents of the file? How can you tell?

The server did explicitly return the contents of the file. Wireshark includes a section titled “Line-Based Text Data” which shows what the server sent back to my browser which is specifically what the website showed when I brought it up on my browser.



10. Now inspect the contents of the second HTTP GET request from your browser to the server. Do you see an “IF-MODIFIED-SINCE:” line in the HTTP GET? If so, what information follows the “IF-MODIFIED-SINCE:” header?

Yes, in the second HTTP message an IF-MODIFIED-SINCE line is included. The information that follows is the date and time that I last accessed the webpage.



11. What is the HTTP status code and phrase returned from the server in response to this second HTTP GET? Did the server explicitly return the contents of the file? Explain.

The HTTP status code is “304: Not Modified”.

The server did not return the contents of the file because the browser simply retrieved the contents from its cache. Had the file been modified since it was last accessed, it would have returned the contents of the file; instead, it simply told my browser to retrieve the old file from its cached memory.

Part 3: Retrieving Long Documents

Steps:

Step 1: Start up the web browser and make sure the browser's cache is cleared.
Step 2: Start up the Wireshark packet sniffer.
Step 3: Enter the following URL in the browser: http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file3.html. The browser should display the rather lengthy US Bill of Rights.
Step 4: Stop Wireshark packet capture, and enter "http" in the display-filter-specification window, so that only captured HTTP messages will be displayed. Take a screenshot.

THE BILL OF RIGHTS
Amendments 1-10 of the Constitution

The Conventions of a number of the States having, at the time of adopting the Constitution, expressed a desire, in order to prevent misconstruction or abuse of its powers, that further declaratory and restrictive clauses should be added, and as extending the ground of public confidence in the Government will best insure the beneficent ends of its institution;

Resolved, by the Senate and House of Representatives of the United States of America, in Congress assembled, two-thirds of both Houses concurring, that the following articles be proposed to the Legislatures of the several States, as amendments to the Constitution of the United States; all or any of which articles, when ratified by three-fourths of the said Legislatures, to be valid to all intents and purposes as part of the said Constitution, namely:

Amendment I

Congress shall make no law respecting an establishment of religion, or prohibiting the free exercise thereof; or abridging the freedom of speech, or of the press; or the right of the people peaceably to assemble, and to petition the government for a redress of grievances.

Amendment II

A well regulated militia, being necessary to the security of a free state, the right of the people to keep and bear arms, shall not be infringed.

Amendment III

No soldier shall, in time of peace be quartered in any house, without the consent of the owner, nor in time of war, but in a manner to be prescribed by law.

Amendment IV

The right of the people to be secure in their persons, houses, papers, and effects, against unreasonable searches and seizures, shall not be violated, and no warrants shall issue, but upon probable cause, supported by oath or affirmation, and particularly describing the place to be searched, and the persons or things to be seized.

Amendment V

No person shall be held to answer for a capital, or otherwise infamous crime, unless on a presentment or indictment of a grand jury, except in cases arising in the land or naval forces, or in the militia, when in actual service in time of war or public danger; nor shall any person be subject for the same offense to be twice put in jeopardy of life or limb; nor shall be compelled in any criminal case to be a witness against himself, nor be deprived of life, liberty, or property, without due process of law; nor shall private property be taken for public use, without just compensation.

Amendment VI

In all criminal prosecutions, the accused shall enjoy the right to a speedy and public trial, by an impartial jury of the state and district wherein the crime shall have been committed, which district shall have been previously ascertained by law, and to be informed of the nature and cause of the accusation; to be confronted with the witnesses against him; to have compulsory process for obtaining witnesses in his favor, and to have the assistance of counsel for his defense.

Amendment VII

In suits at common law, where the value in controversy shall exceed twenty dollars, the right of trial by jury shall be preserved, and no fact tried by a jury, shall be otherwise reexamined in any court of the United States, than according to the rules of the common law.

Amendment VIII

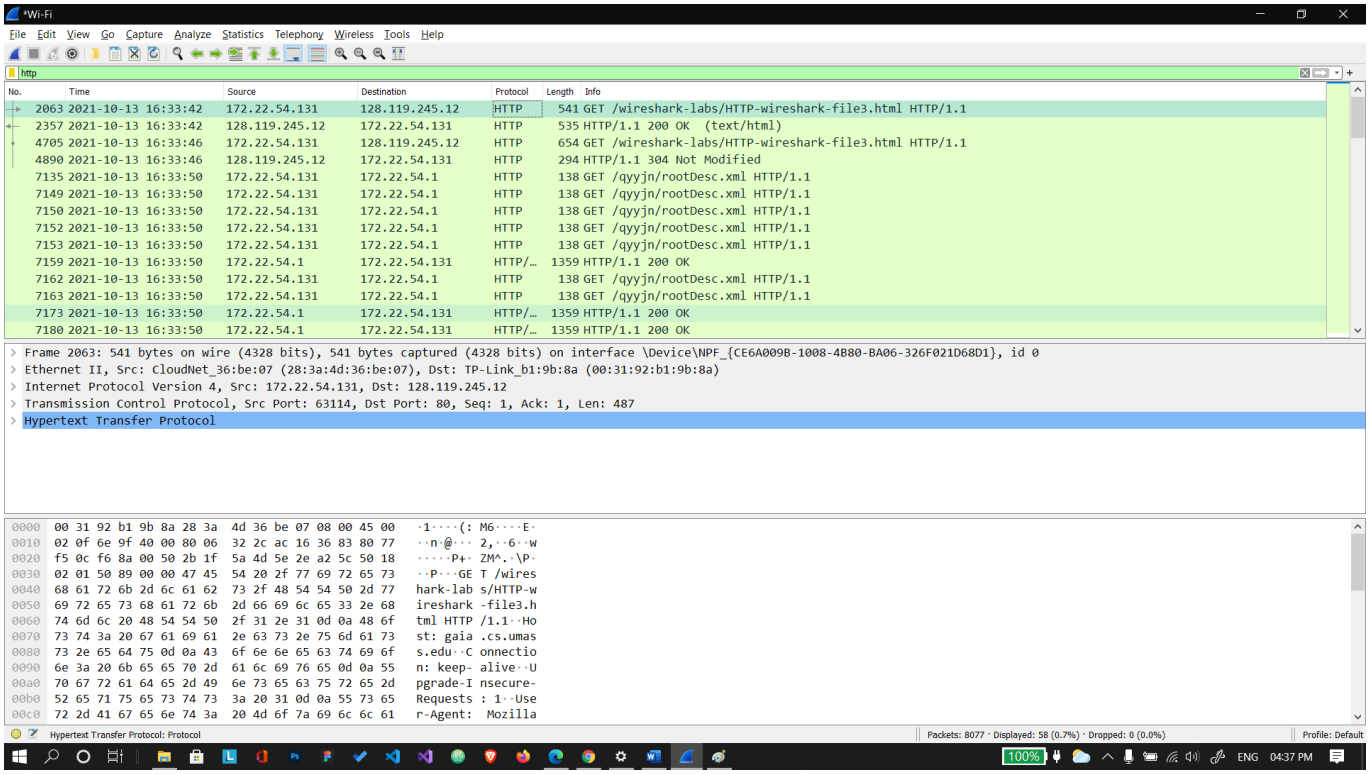
Excessive bail shall not be required, nor excessive fines imposed, nor cruel and unusual punishments inflicted.

Amendment IX

The enumeration in the Constitution, of certain rights, shall not be construed to deny or disparage others retained by the people.

Amendment X

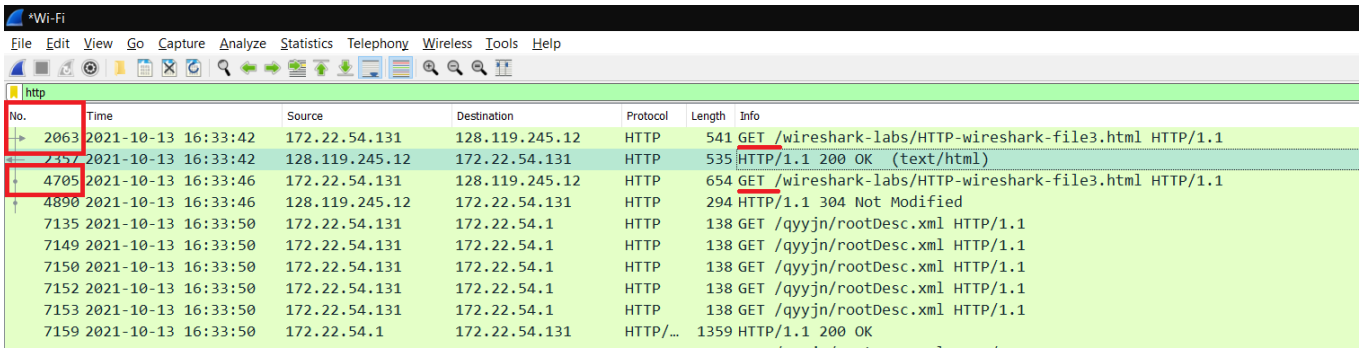
The powers not delegated to the United States by the Constitution, nor prohibited by it to the states, are reserved to the states respectively, or to the people.



Back to questions:

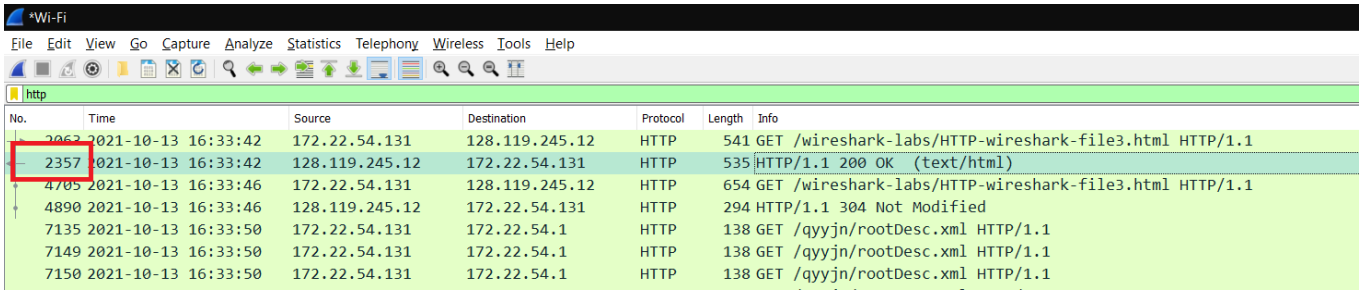
12. How many HTTP GET request messages did your browser send? Which packet number in the trace contains the GET message for the Bill of Rights?

My browser has sent 2 HTTP GET requests to the server (since I'd refreshed the page once). The Packets that contained the GET messages were packet numbers 2063 and 4705.



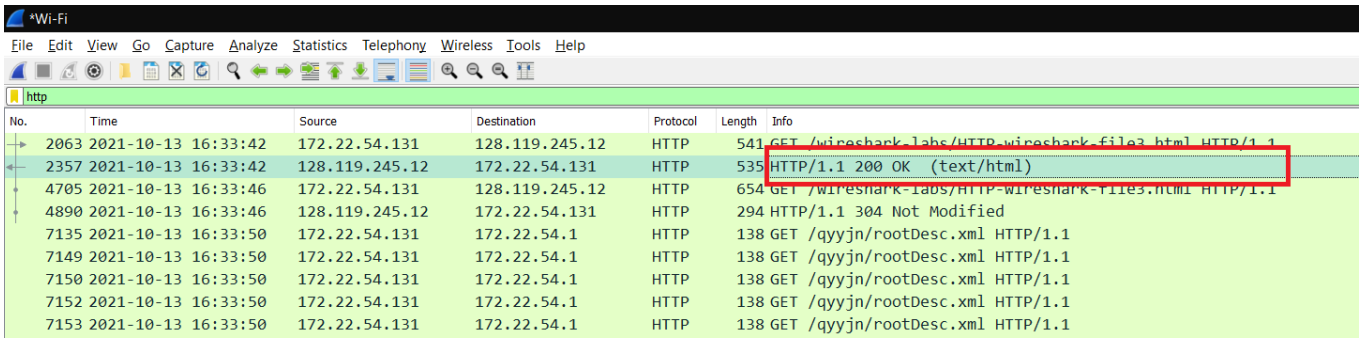
13. Which packet number in the trace contains the status code and phrase associated with the response to the HTTP GET request?

The packet that contains the status code and phrase which the server sent in response to the GET message was packet number 2357.



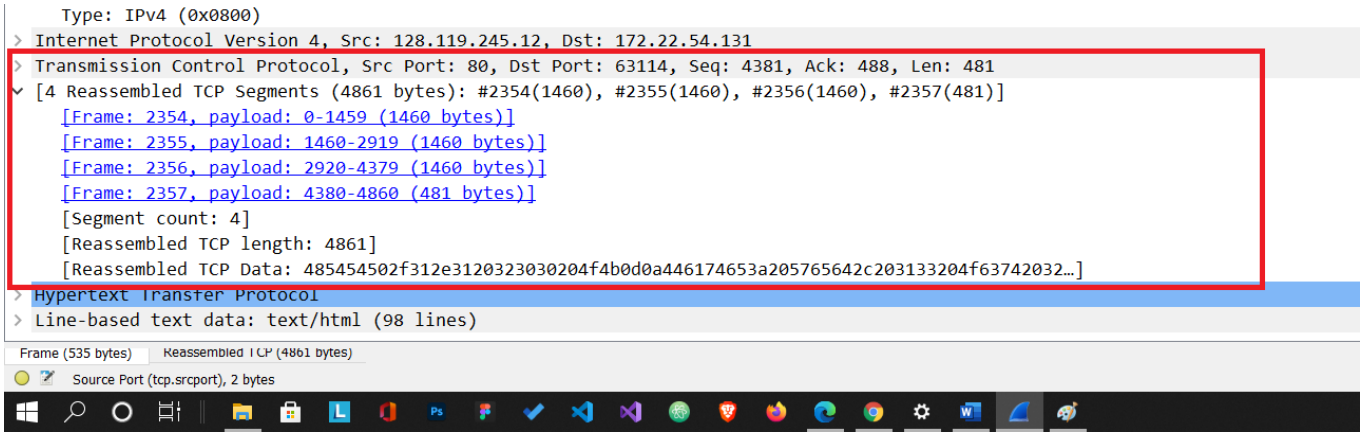
14. What is the status code and phrase in the response?

The code and phrase in the response was "200 OK".



15. How many data-containing TCP segments were needed to carry the single HTTP response and the text of the Bill of Rights?

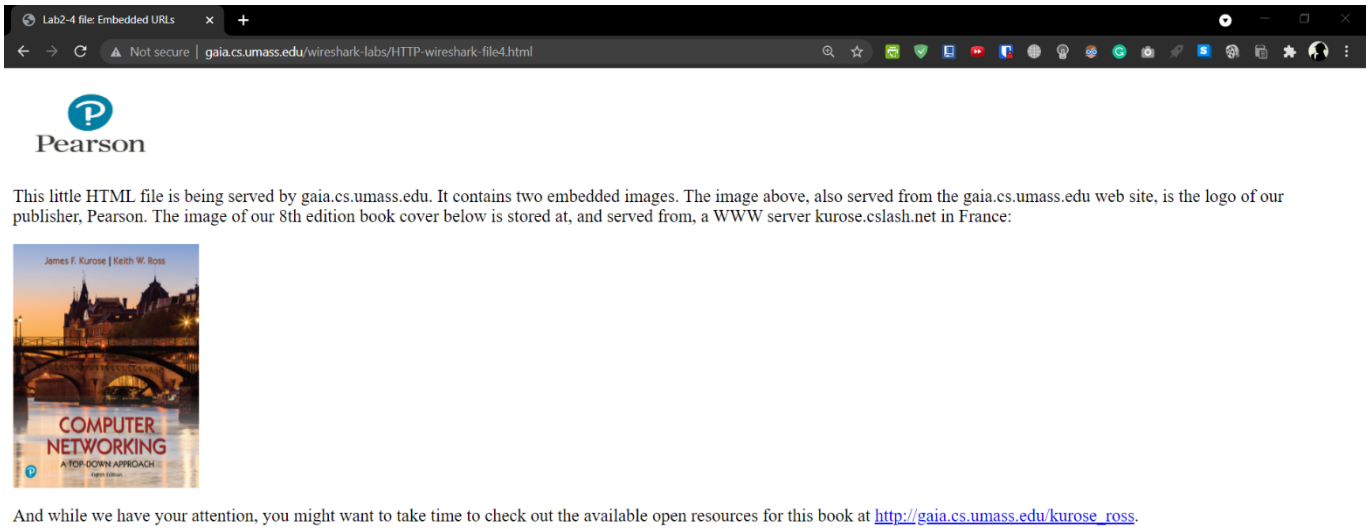
The data was sent in 4 TCP segments to the browser, then reassembled.

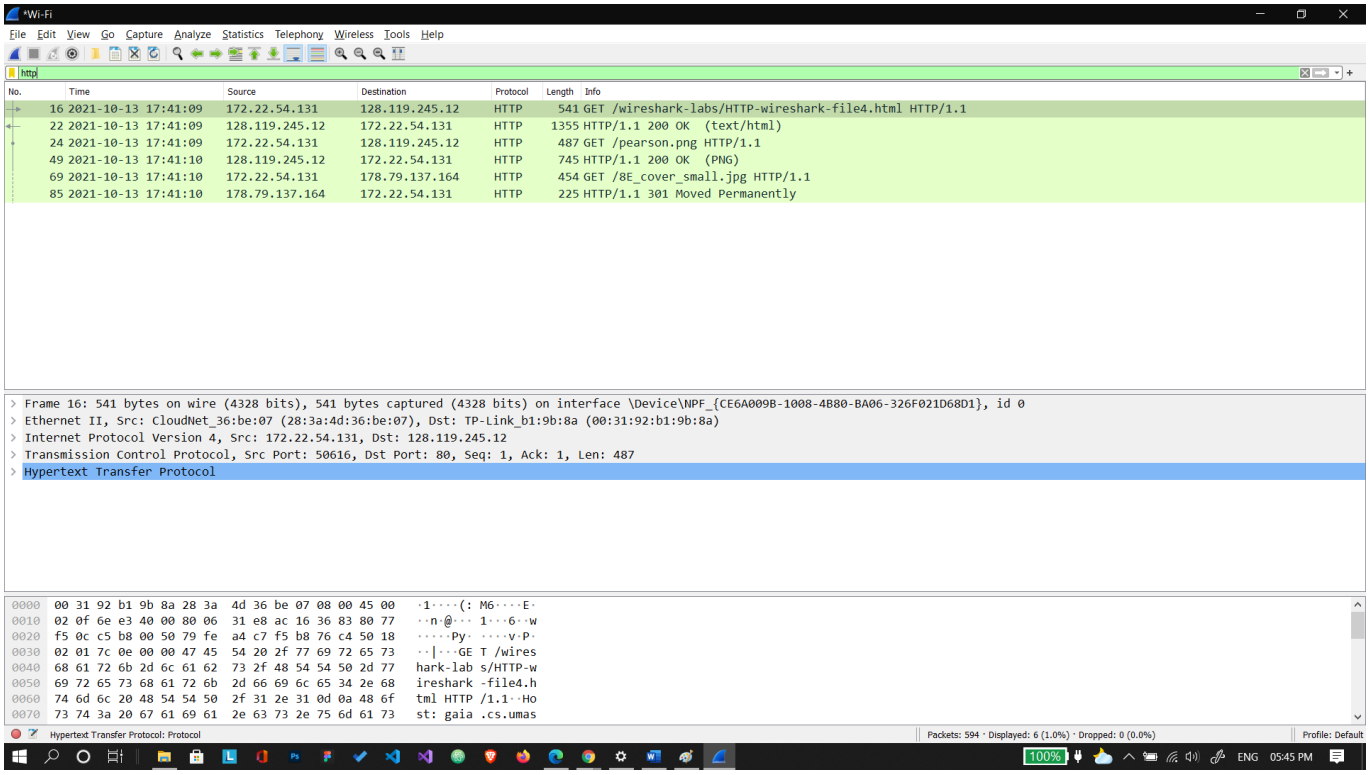


Part 4: HTML Documents with Embedded Objects

Steps:

Step 1: Start up the web browser and make sure the browser's cache is cleared.
Step 2: Start up the Wireshark packet sniffer.
Step 3: Enter the following URL in the browser: http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file4.html. The browser should display a short HTML file with two images. These two images are referenced in the base HTML file. That is, the images themselves are not contained in the HTML; instead, the URLs for the images are contained in the downloaded HTML file. The browser will have to retrieve these logos from the indicated web sites. The publisher's logo is retrieved from the gaia.cs.umass.edu website. The image of the cover for the 5th edition is stored at the caite.cs.umass.edu server. (These are two different web servers inside cs.umass.edu).
Step 4: Stop Wireshark packet capture, and enter "http" in the display-filter-specification window, so that only captured HTTP messages will be displayed.
Step 5: Take a screenshot.



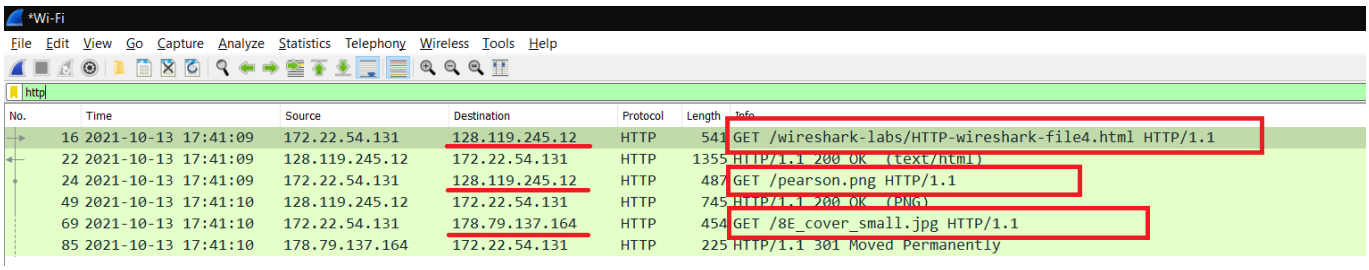


Back to questions:

16. How many HTTP GET request messages did your browser send? To which Internet addresses were these GET requests sent?

My browser sent 3 HTTP GET message requests. One to each of the following: The initial page, the Pearson logo, and the cover of the 8th Edition Pearson book.

Initial Page address: 128.119.245.12
Pearson logo: 128.119.245.12
Pearson book, 8th edition: 178.79.137.164



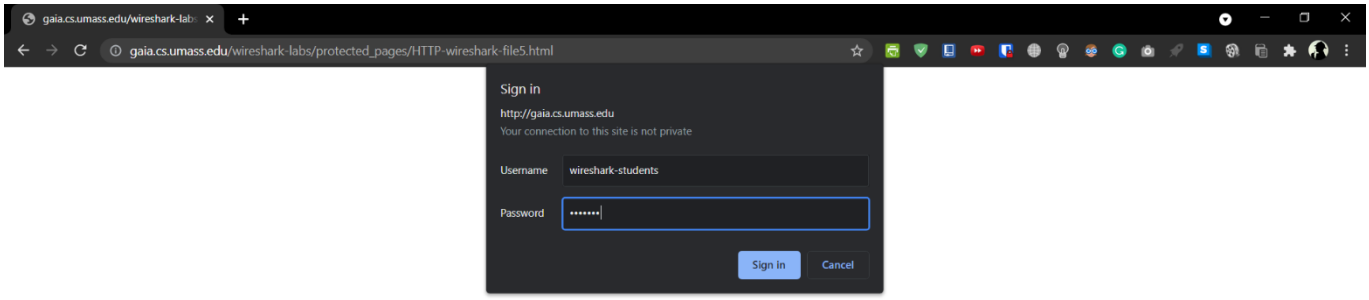
17. Can you tell whether your browser downloaded the two images serially, or whether they were downloaded from the two web sites in parallel? Explain.

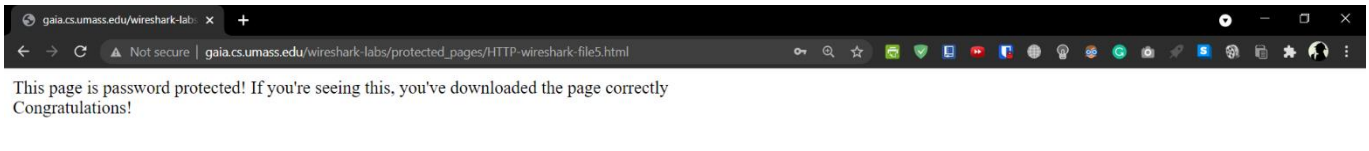
The browser downloaded the two images serially. I believe this to be the case because the first image was requested and sent before the second image was requested by the browser. Had they been running in parallel, both files would have been requested then would have returned in the same time period. In this case however, the second image was only requested after the first image came back.

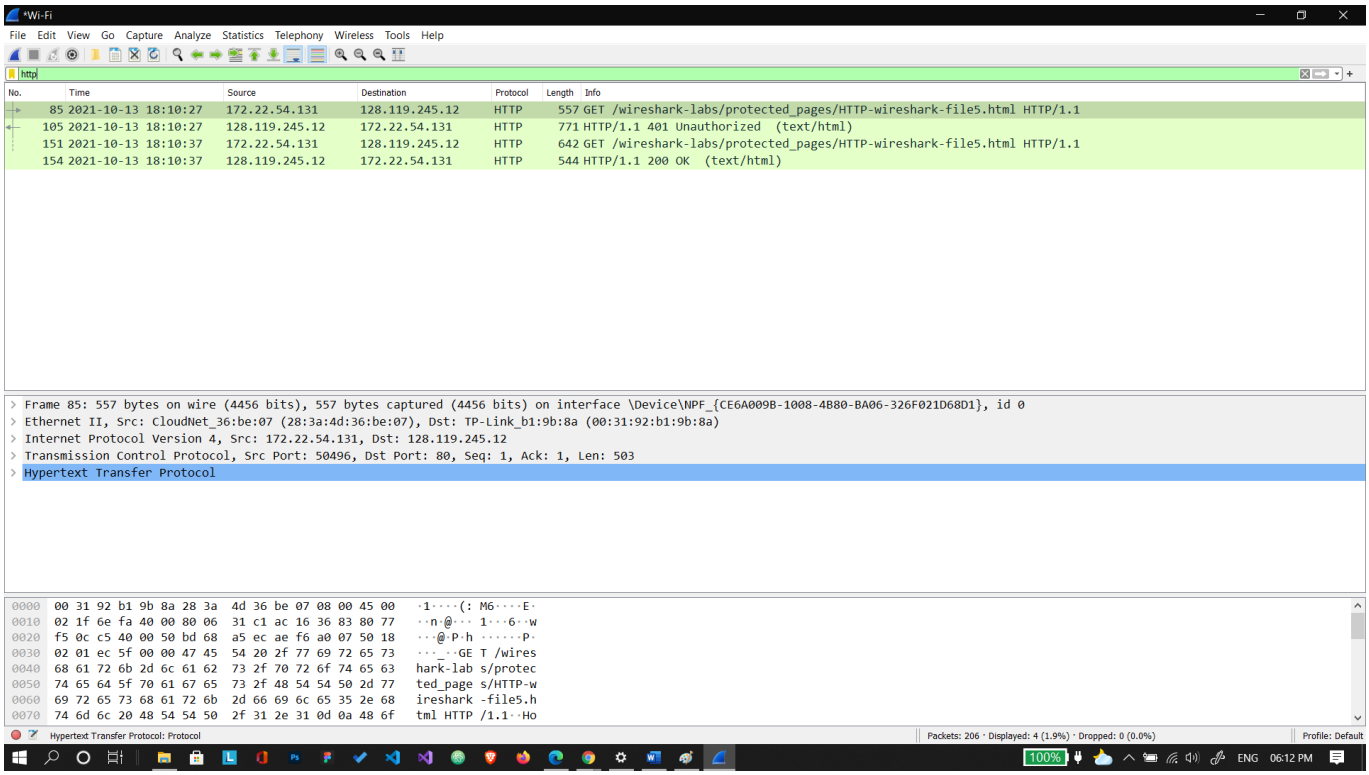
Part 5: HTTP Authentication

Steps:

Step 1: Make sure browser's cache is cleared, and close down the browser. Then restart it.
Step 2: Start up the Wireshark packet sniffer.
Step 3: Enter the following URL in the browser: http://gaia.cs.umass.edu/wireshark-labs/protected_pages/HTTP-wireshark-file5.html, which is password protected. The username is "wireshark-students" (without the quotes), and the password is "network" (again, without the quotes).
Step 4: Type the requested username and password into the pop-up box.
Step 5: Stop Wireshark packet capture, and enter "http" in the display-filter-specification window, so that only captured HTTP messages will be displayed later in the packet-listing window.
Step 6: Take a screenshot.



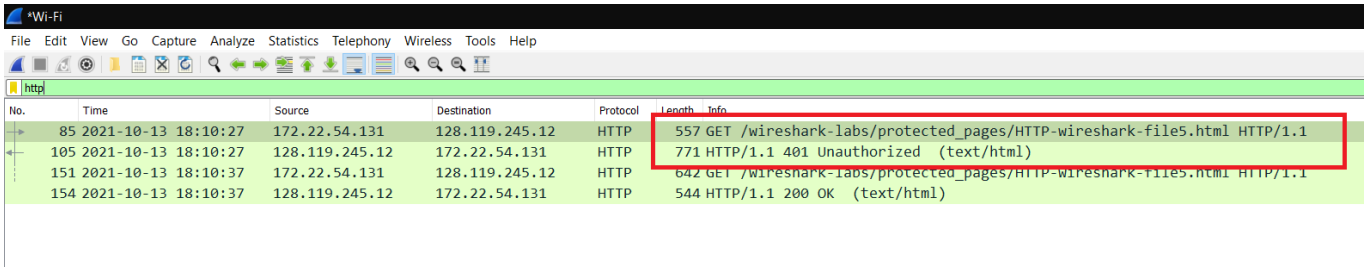




Back to questions:

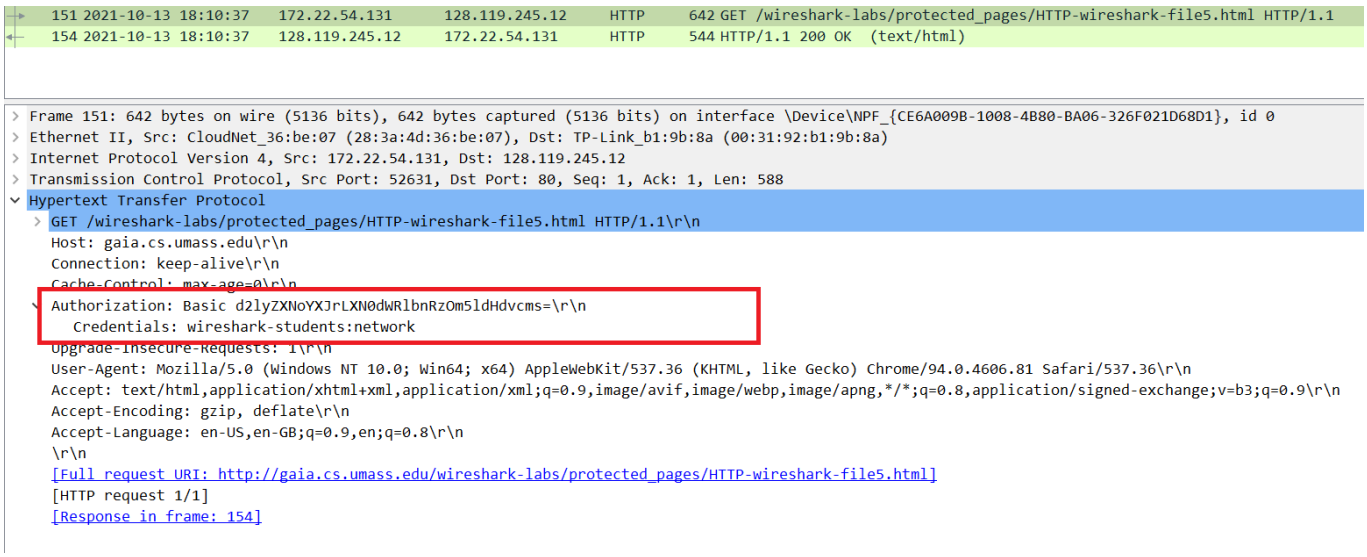
18. What is the server's response (status code and phrase) in response to the initial HTTP GET message from your browser?

The server's initial response was "401 Unauthorized".



19. When your browser sends the HTTP GET message for the second time, what new field is included in the HTTP GET message?

The new field that is now included is the authorization field. This is included because we sent the server a username and password along with our request stating that we were authorized to receive the page.

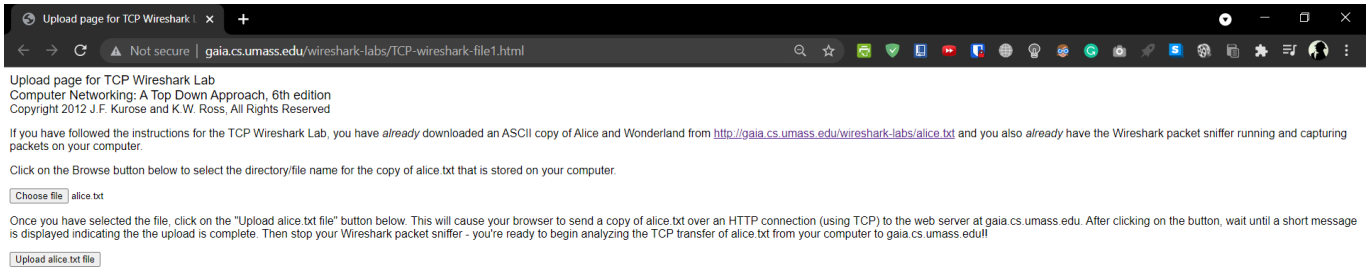


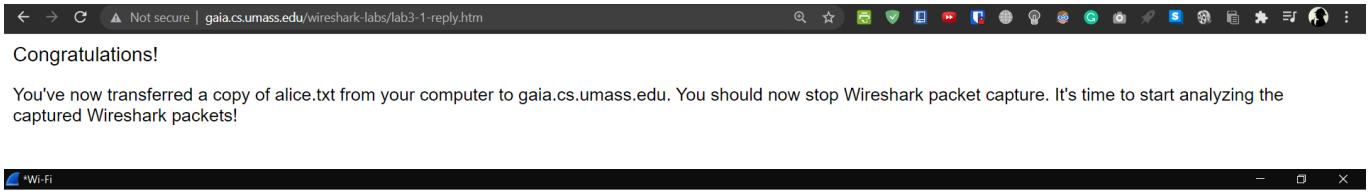
Experiment 3: Wireshark TCP

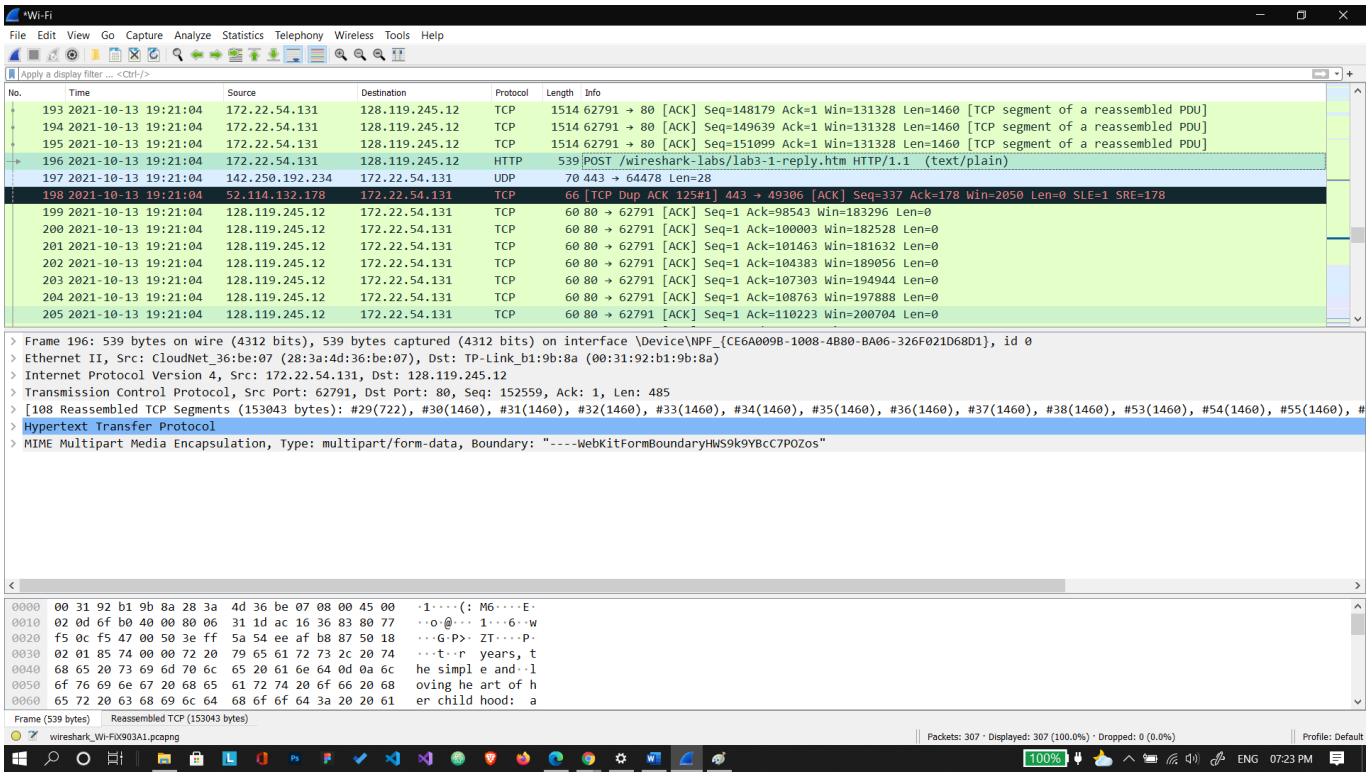
Part 1: Capturing a bulk TCP transfer from your computer to a remote server

Steps:

Step 1: Start up the web browser. Go to http://gaia.cs.umass.edu/wiresharklabs/alice.txt and retrieve an ASCII copy of *Alice in Wonderland*. Store this file somewhere on computer.
Step 2: Next go to http://gaia.cs.umass.edu/wireshark-labs/TCP-wireshark-file1.html.
Step 3: Use the Browse button in this form to enter the name of the file (full path name) on the computer containing *Alice in Wonderland* (or do so manually). Don't yet press the "*Upload alice.txt file*" button.
Step 4: Now start up Wireshark and begin packet capture (*Capture->Start*) and then press *OK* on the Wireshark Packet Capture Options screen (we'll not need to select any options here).
Step 5: Returning to the browser, press the "*Upload alice.txt file*" button to upload the file to the gaia.cs.umass.edu server. Once the file has been uploaded, a short congratulations message will be displayed in the browser window.
Step 6: Stop Wireshark packet capture. Take a screenshot.



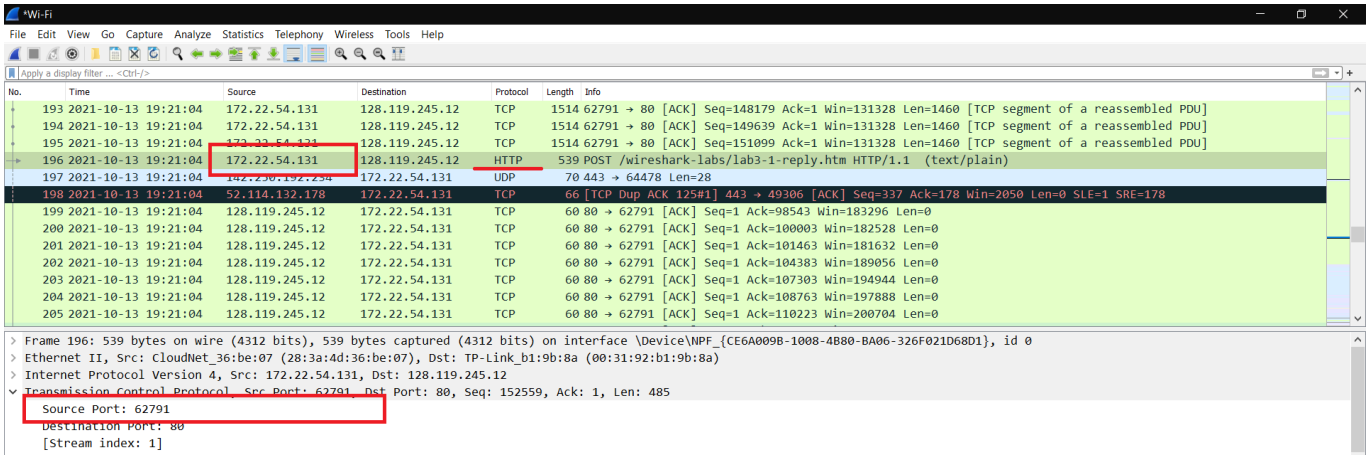




Part 2: A first look at the captured trace

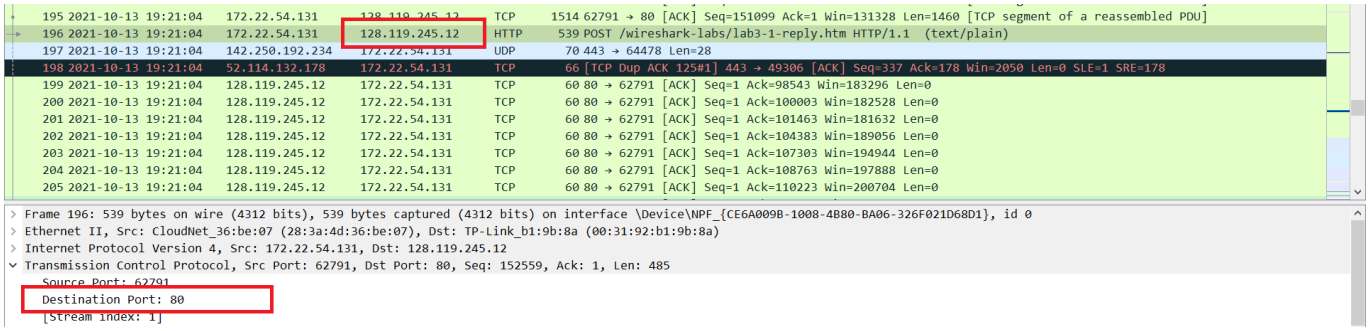
1. **What is the IP address and TCP port number used by the client computer (source) that is transferring the file to gaia.cs.umass.edu? To answer this question, it's probably easiest to select an HTTP message and explore the details of the TCP packet used to carry this HTTP message, using the "details of the selected packet header window".**

   The source IP address was 172.22.54.131 using source port 62791.



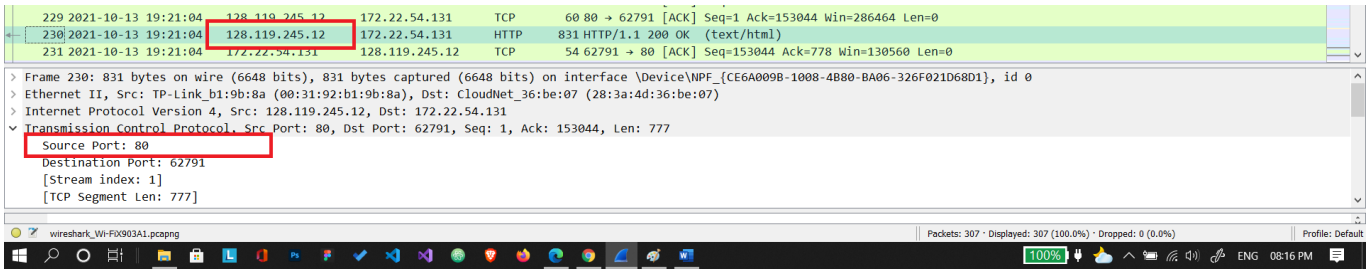
2. **What is the IP address of gaia.cs.umass.edu? On what port number is it sending and receiving TCP segments for this connection?**

   The destination IP address is 128.119.245.12 receiving on port 80.



3. **What is the IP address and TCP port number used by your client computer (source) to transfer the file to gaia.cs.umass.edu?**

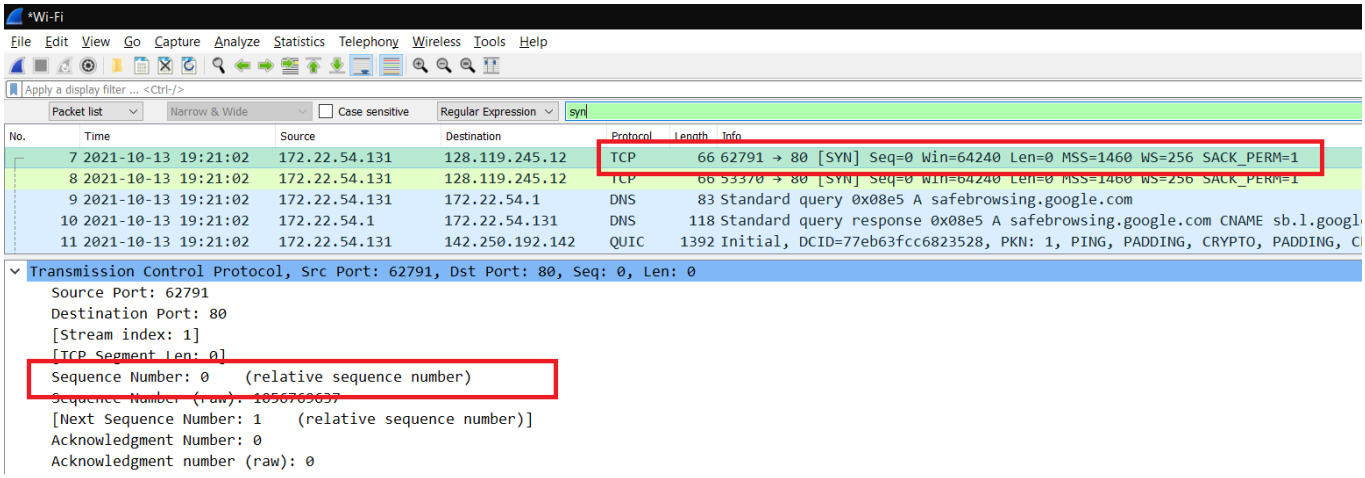
   My IP address source is 128.119.245.12 sending on port 80.



Part 3: TCP Basics

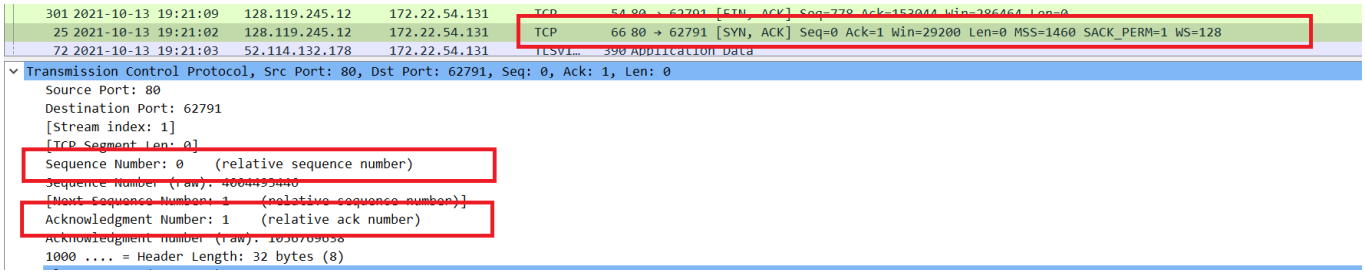
4. **What is the sequence number of the TCP SYN segment that is used to initiate the TCP connection between the client computer and gaia.cs.umass.edu? What is it in the segment that identifies the segment as a SYN segment?**

   The sequence number of the segment used to initiate the TCP connection is 0. We can see that the message contains a SYN flag indicating that it is a SYN segment.



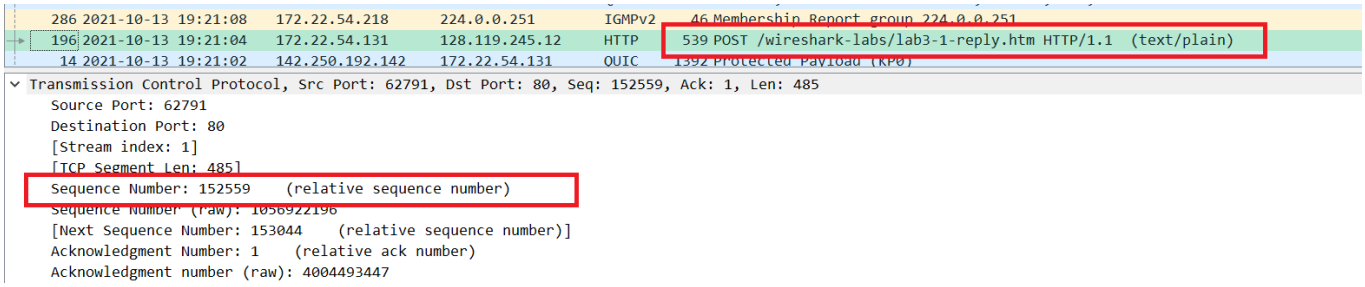
5. **What is the sequence number of the SYNACK segment sent by gaia.cs.umass.edu to the client computer in reply to the SYN? What is the value of the Acknowledgement field in the SYNACK segment? How did gaia.cs.umass.edu determine that value? What is it in the segment that identifies the segment as a SYNACK segment?**

   The sequence number of the SYNACK segment is 0.
   The value of the acknowledgement field is 1. This value is determined by the initial sequence number +1.
   The message carries flags that show it to be a SYN ACK message.



6. **What is the sequence number of the TCP segment containing the HTTP POST command? Note that in order to find the POST command, you'll need to dig into the packet content field at the bottom of the Wireshark window, looking for a segment with a "POST" within its DATA field.**

   The sequence number of the TCP segment containing the HTTP Post Command is 152559.



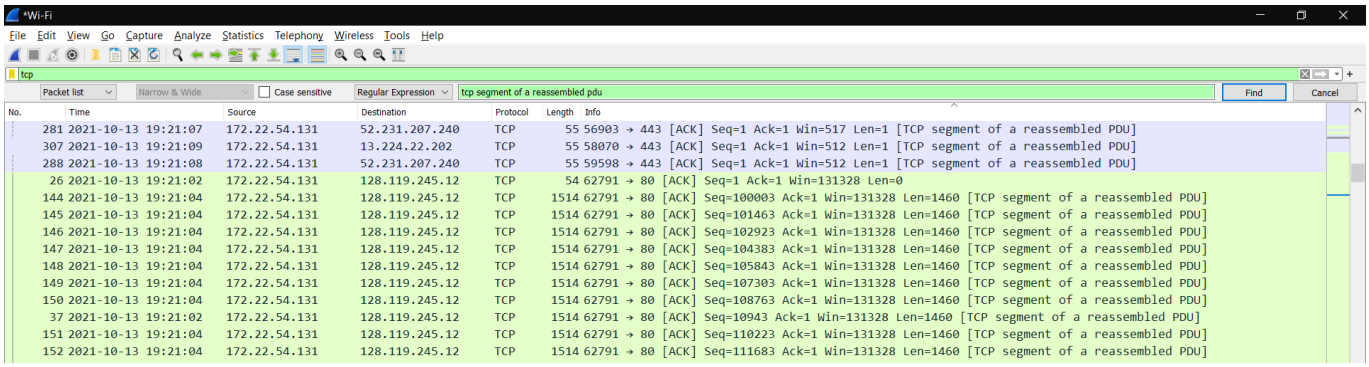
7. Consider the TCP segment containing the HTTP POST as the first segment in the TCP connection. What are the sequence numbers of the first six segments in the TCP connection (including the segment containing the HTTP POST)? At what time was each segment sent? When was the ACK for each segment received? Given the difference between when each TCP segment was sent, and when its acknowledgement was received, what is the RTT value for each of the six segments? What is the "EstimatedRTT" value after the receipt of each ACK? Assume that the value of the "EstimatedRTT" is equal to the measured RTT for the first segment, and then is computed using the "EstimatedRTT" equation for all subsequent segments.

Segment	Sequence number	Segment number	Time sent	Acknowledgement received	RTT	Estimated RTT
1	1	0dd601f	0.359375	0.703125	0.34375	0.34375
2	777	0dd6042	0.359375	0.734375	0.375	0.3475
3	2203	0dd609d	0.703125	1.062500	0.359375	0.3489
4	3629	0dd60f9	0.703125	1.093750	0.390625	0.3541
5	5055	0dd60f9	0.734375	1.109375	0.375	0.3567
6	6481	0dd61af	0.734375	1.140625	0.40625	0.3628

EstimatedRTT = 0.875 * EstimatedRTT + 0.125 * SampleRTT
EstimatedRTT of Segment 1 = 0.34375
EstimatedRTT of Segment 2 = 0.875 * 0.34375 + 0.125 * 0.375 = 0.3475
EstimatedRTT of Segment 3 = 0.875 * 0.3475 + 0.125 * 0.359375 = 0.3489
EstimatedRTT of Segment 4 = 0.875 * 0.3489 + 0.125 * 0.390625 = 0.3541
EstimatedRTT of Segment 5 = 0.875 * 0.3541 + 0.125 * 0.375 = 0.3567
EstimatedRTT of Segment 6 = 0.875 * 0.3567 + 0.125 * 0.40625 = 0.3628

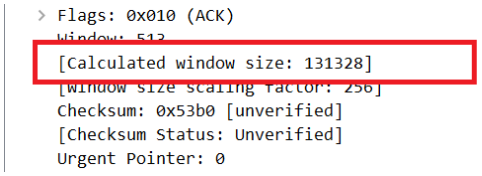
8. What is the length of each of the first six TCP segments?

The lengths of each of the first 3 TCP segments are 55. The following segments are all 1514.



9. What is the minimum amount of available buffer space advertised at the received for the entire trace? Does the lack of receiver buffer space ever throttle the sender?

The minimum amount of available buffer space (calculated window) is listed as 131328. The lack of receiver buffer space does not ever throttle the sender.



10. Are there any retransmitted segments in the trace file? What did you check for (in the trace) in order to answer this question?

No, no segments were ever retransmitted. To check this, I looked for any repeating segment numbers; no old acknowledgement number was ever resent in order to re-request former packets.

11. How much data does the receiver typically acknowledge in an ACK? Can you identify cases where the receiver is ACKing every other received segment?

The receiver is typically ACKing 1460 bytes. There are cases where the receiver acks every other segment (when more than one ACK occurs in a row).

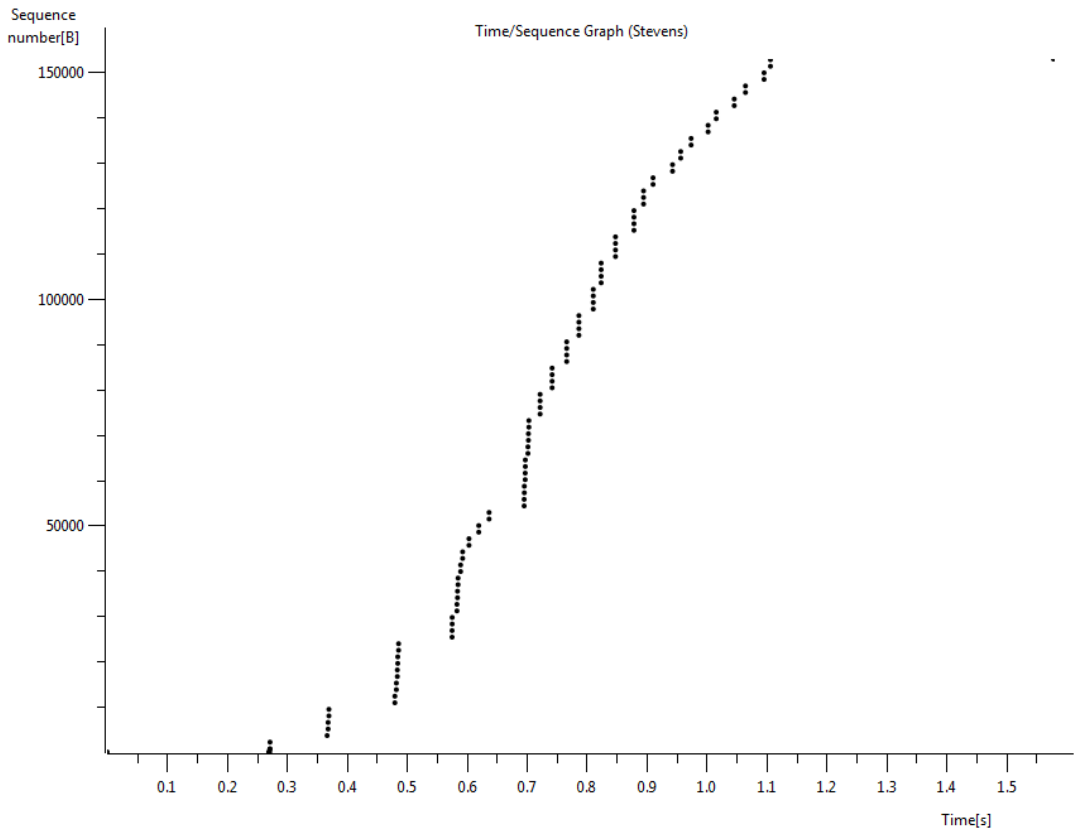
12. What is the throughput (bytes transferred per unit time) for the TCP connection? Explain how you calculated this value.

The throughput can be calculated by using: the value of the last ACK (149,629) − the first sequence number (1) divided by the time since first frame (1.6) = 93517.6 bytes per second.

Part 4: TCP congestion control in action

Step: Select a TCP segment in the Wireshark's "listing of captured-packets" window. Then select the menu: *Statistics->TCP Stream Graph-> Time-SequenceGraph(Stevens).*

13. Use the *Time-Sequence-Graph(Stevens)* plotting tool to view the sequence number versus time plot of segments being sent from the client to the gaia.cs.umass.edu server. Can you identify where TCP's slowstart phase begins and ends, and where congestion avoidance takes over?



The slow start of the TCP seems to begin at about 0.27 seconds and then ends at about 0.35 seconds. Congestion avoidance takes over at about 0.7 seconds because it cut down the amount being sent.

14. Answer each of two questions above for the trace that you have gathered when you transferred a file from your computer to gaia.cs.umass.edu.

This has already been answered. When we have a lot of traffic on network, TCP sender uses AIMD algorithm for the reduction of window size.