

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

Headings	Page numbers
0. Experiment 0: Malware Detection using ML	3
1. Experiment 1: Wireshark DNS	9
1.1. Part 1: NSLookup	9
1.2. Part 2: IPconfig	10
1.3. Part 3: Tracing DNS with Wireshark	14
1.3.1. Section 1	14
1.3.2. Section 2	17
1.3.2.1. Section 2.1	19
1.3.2.2. Section 2.2	21
2. Experiment 2: Wireshark HTTP	23
2.1. Part 1: The Basic HTTP GET/response interaction	23
2.2. Part 2: The HTTP CONDITIONAL GET/response interaction	26
2.3. Part 3: Retrieving Long Documents	28
2.4. Part 4: HTML Documents with Embedded Objects	30
2.5. Part 5: HTTP Authentication	32
3. Experiment 3: Wireshark TCP	34
3.1. Part 1: Capturing a bulk TCP transfer from your computer to a remote server	34
3.2. Part 2: A first look at the captured trace	35
3.3. Part 3: TCP Basics	36
3.4. Part 4: TCP congestion control in action	39

Exp 0: Malware Detection using ML

Dataset:

Malware dataset - Excel

Sharadindu Adhikari

File Home Insert Page Layout Formulas Data Review View Help Tell me what you want to do

Font Alignment Number Styles Cells Editing

hash

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	last						
1	hash	millisecond	classification	state	usage	cou	prio	static	prio	normal	pr	policy	vm_pgoff	vm_truncate	task_size	cached	hc_free	area	mm_users	map_coun	hiwater	rs_total	vm_shared	vr_exec	vm_reserved	nr_ptes	end	data	last
2	42fb5e2ec	0	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	24	724	6850	0	150	120	124	210	0	120						
3	42fb5e2ec	1	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	24	724	6850	0	150	120	124	210	0	120						
4	42fb5e2ec	2	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	24	724	6850	0	150	120	124	210	0	120						
5	42fb5e2ec	3	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	24	724	6850	0	150	120	124	210	0	120						
6	42fb5e2ec	4	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	24	724	6850	0	150	120	124	210	0	120						
7	42fb5e2ec	5	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	24	724	6850	0	150	120	124	210	0	120						
8	42fb5e2ec	6	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	24	724	6850	0	150	120	124	210	0	120						
9	42fb5e2ec	7	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	24	724	6850	0	150	120	124	210	0	120						
10	42fb5e2ec	8	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	24	724	6850	0	150	120	124	210	0	120						
11	42fb5e2ec	9	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	25	724	6852	0	150	120	124	211	0	120						
12	42fb5e2ec	10	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	25	724	6852	0	150	120	124	211	0	120						
13	42fb5e2ec	11	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	25	724	6852	0	150	120	124	211	0	120						
14	42fb5e2ec	12	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	26	724	6854	0	151	120	124	212	0	120						
15	42fb5e2ec	13	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	26	724	6854	0	151	120	124	212	0	120						
16	42fb5e2ec	14	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	26	724	6854	0	151	120	124	212	0	120						
17	42fb5e2ec	15	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	26	724	6854	0	151	120	124	212	0	120						
18	42fb5e2ec	16	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	27	724	6856	0	152	120	124	212	0	120						
19	42fb5e2ec	17	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	27	724	6856	0	152	120	124	212	0	120						
20	42fb5e2ec	18	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	27	724	6856	0	152	120	124	212	0	120						
21	42fb5e2ec	19	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	27	724	6856	0	153	120	124	212	0	120						
22	42fb5e2ec	20	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	27	724	6856	0	154	120	124	213	0	120						
23	42fb5e2ec	21	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	27	724	6856	0	154	120	124	213	0	120						
24	42fb5e2ec	22	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	27	724	6856	0	154	120	124	213	0	120						
25	42fb5e2ec	23	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	27	724	6856	0	154	120	124	213	0	120						
26	42fb5e2ec	24	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	28	724	6858	0	155	120	124	214	0	120						
27	42fb5e2ec	25	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	28	724	6858	0	155	120	124	214	0	120						
28	42fb5e2ec	26	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	28	724	6858	0	155	120	124	214	0	120						
29	42fb5e2ec	27	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	28	724	6858	0	155	120	124	214	0	120						
30	42fb5e2ec	28	malware	0	0	3.07E+09	14274	0	0	0	13173	0	0	29	724	6850	0	156	120	124	214	0	120						

Malware dataset

Ready

File Explorer Taskbar

Preview 'Malware dataset.csv' - Malware - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER DT.py LR.py Malware dataset.csv * Preview 'Malware dataset.csv' x

hash	millisecond	classification	state	usage_counter	prio	static_prio	normal_prio	policy	vm_pgoff	vm_truncate	task_size
42fb5e2ec009a0	0	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	1	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	2	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	3	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	4	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	5	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	6	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	7	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	8	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	9	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	10	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	11	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	12	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	13	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	14	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	15	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	16	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	17	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	18	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	19	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	20	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	21	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	22	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	23	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	24	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	25	malware	0	0	3069378560	14274	0	0	0	13173	0
42fb5e2ec009a0	26	malware	0	0	3069378560	14274	0	0	0	13173	0

File Explorer Taskbar

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 = d2.drop(['classification'], axis=1)

X = d2.iloc[:, 1:].values
y = d2.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))
```

```
#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 = d2.drop(['classification'], axis=1)
X = d2.iloc[:, 1:].values
y = d2.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 = d2.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))
```

```
#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 = d2.drop(['classification'], axis=1)
X = d2.iloc[:, 1:].values
y = d2.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
```

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



```
Command Prompt
Microsoft Windows [Version 10.0.19043.1266]
(c) Microsoft Corporation. All rights reserved.

C:\Users\shara>nslookup www.vit.ac.in
Server: UnKnown
Address: 172.22.54.1

Non-authoritative answer:
Name: vit.ac.in
Address: 136.233.9.13
Aliases: www.vit.ac.in
```

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>

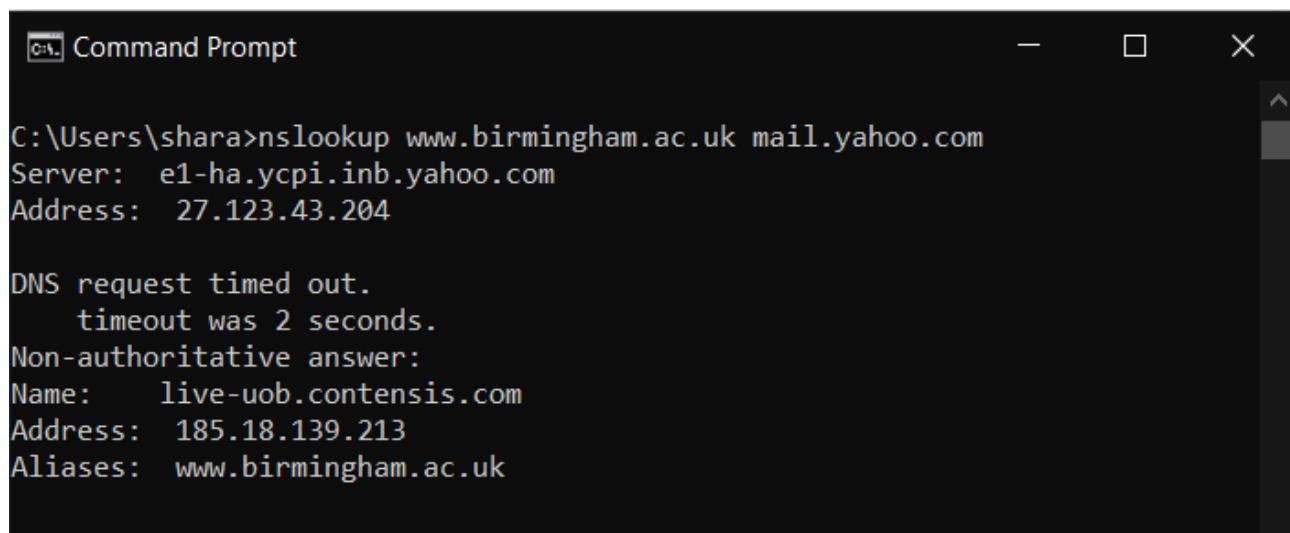


```
Command Prompt
C:\Users\shara>nslookup -type=NS www.birmingham.ac.uk
Server: UnKnown
Address: 172.22.54.1

Non-authoritative answer:
www.birmingham.ac.uk canonical name = live-uob.contensis.com
live-uob.contensis.com internet address = 185.18.139.213
```

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

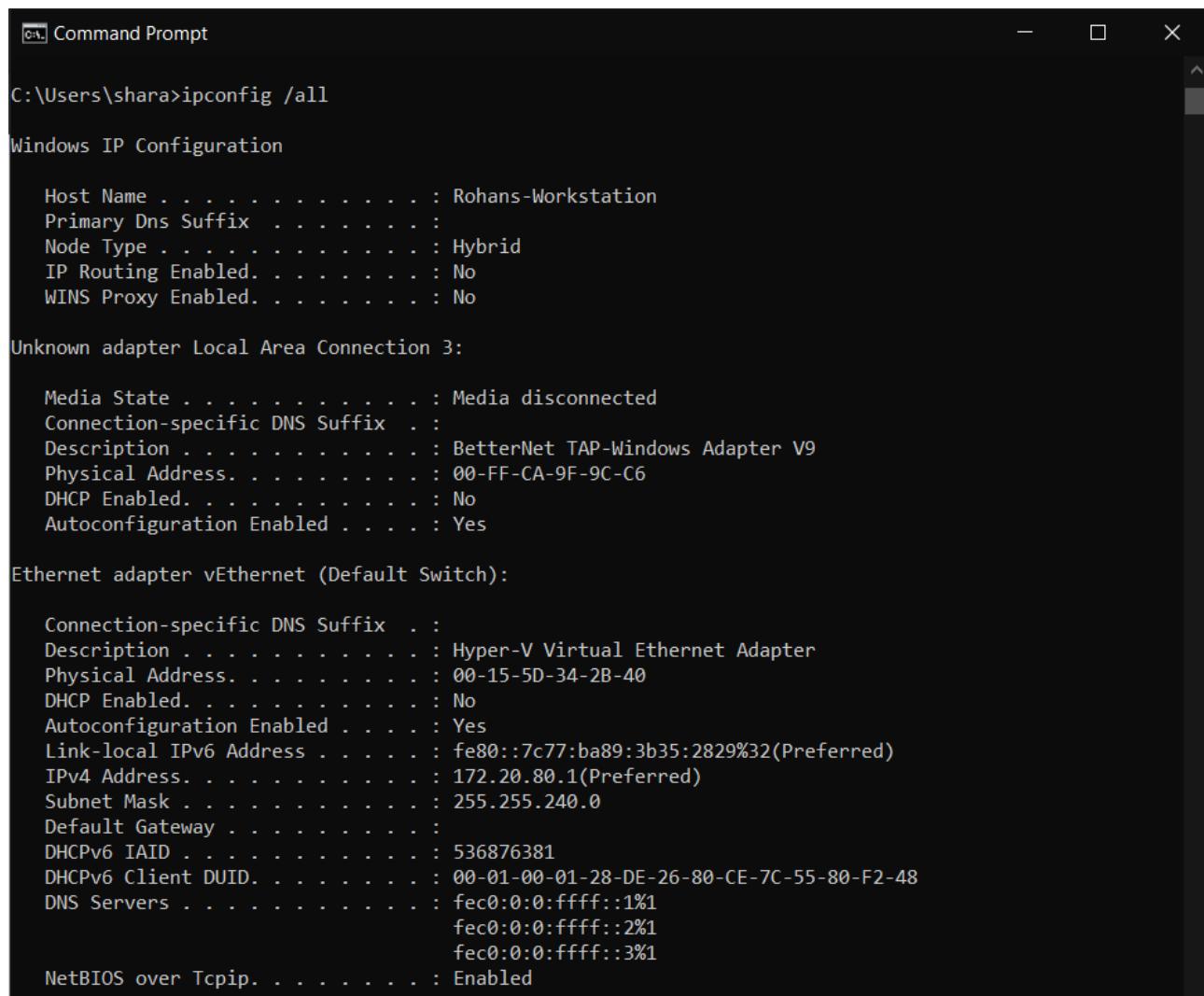


```
C:\Users\shara>nslookup www.birmingham.ac.uk mail.yahoo.com
Server: e1-ha.ycpi.inb.yahoo.com
Address: 27.123.43.204

DNS request timed out.
    timeout was 2 seconds.
Non-authoritative answer:
Name: live-uob.contensis.com
Address: 185.18.139.213
Aliases: www.birmingham.ac.uk
```

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.



```
C:\Users\shara>ipconfig /all

Windows IP Configuration

Host Name . . . . . : Rohans-Workstation
Primary Dns Suffix . . . . . :
Node Type . . . . . : Hybrid
IP Routing Enabled. . . . . : No
WINS Proxy Enabled. . . . . : No

Unknown adapter Local Area Connection 3:

Media State . . . . . : Media disconnected
Connection-specific DNS Suffix . . . . . :
Description . . . . . : BetterNet TAP-Windows Adapter V9
Physical Address. . . . . : 00-FF-CA-9F-9C-C6
DHCP Enabled. . . . . : No
Autoconfiguration Enabled . . . . . : Yes

Ethernet adapter vEthernet (Default Switch):

Connection-specific DNS Suffix . . . . . :
Description . . . . . : Hyper-V Virtual Ethernet Adapter
Physical Address. . . . . : 00-15-5D-34-2B-40
DHCP Enabled. . . . . : No
Autoconfiguration Enabled . . . . . : Yes
Link-local IPv6 Address . . . . . : fe80::7c77:ba89:3b35:2829%32(Preferred)
IPv4 Address. . . . . : 172.20.80.1(Preferred)
Subnet Mask . . . . . : 255.255.240.0
Default Gateway . . . . . :
DHCPv6 IAID . . . . . : 536876381
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
```

```
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
```

```
Command Prompt

rohans-workstation.mshome.net
-----
Record Name . . . . : Rohans-Workstation.mshome.net
Record Type . . . . : 1
Time To Live . . . . : 440681
Data Length . . . . : 4
Section . . . . . : Answer
A (Host) Record . . . : 172.20.80.1

rohans-workstation.mshome.net
-----
No records of type AAAA

uk54.lunrac.com
-----
Record Name . . . . : uk54.lunrac.com
Record Type . . . . : 1
Time To Live . . . . : 9704
Data Length . . . . : 4
Section . . . . . : Answer
A (Host) Record . . . : 217.163.11.154

uk56.lunrac.com
-----
Record Name . . . . : uk56.lunrac.com
Record Type . . . . : 1
Time To Live . . . . : 64862
Data Length . . . . : 4
Section . . . . . : Answer
A (Host) Record . . . : 217.163.11.74

camtasiatudi.techsmith.com
-----
Record Name . . . . : camtasiatudi.techsmith.com
Record Type . . . . : 1
Time To Live . . . . : 440681
Data Length . . . . : 4
Section . . . . . : Answer
A (Host) Record . . . : 127.0.0.1
```

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

```
Command Prompt

C:\Users\shara>ipconfig /flushdns

Windows IP Configuration

Successfully flushed the DNS Resolver Cache.
```

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

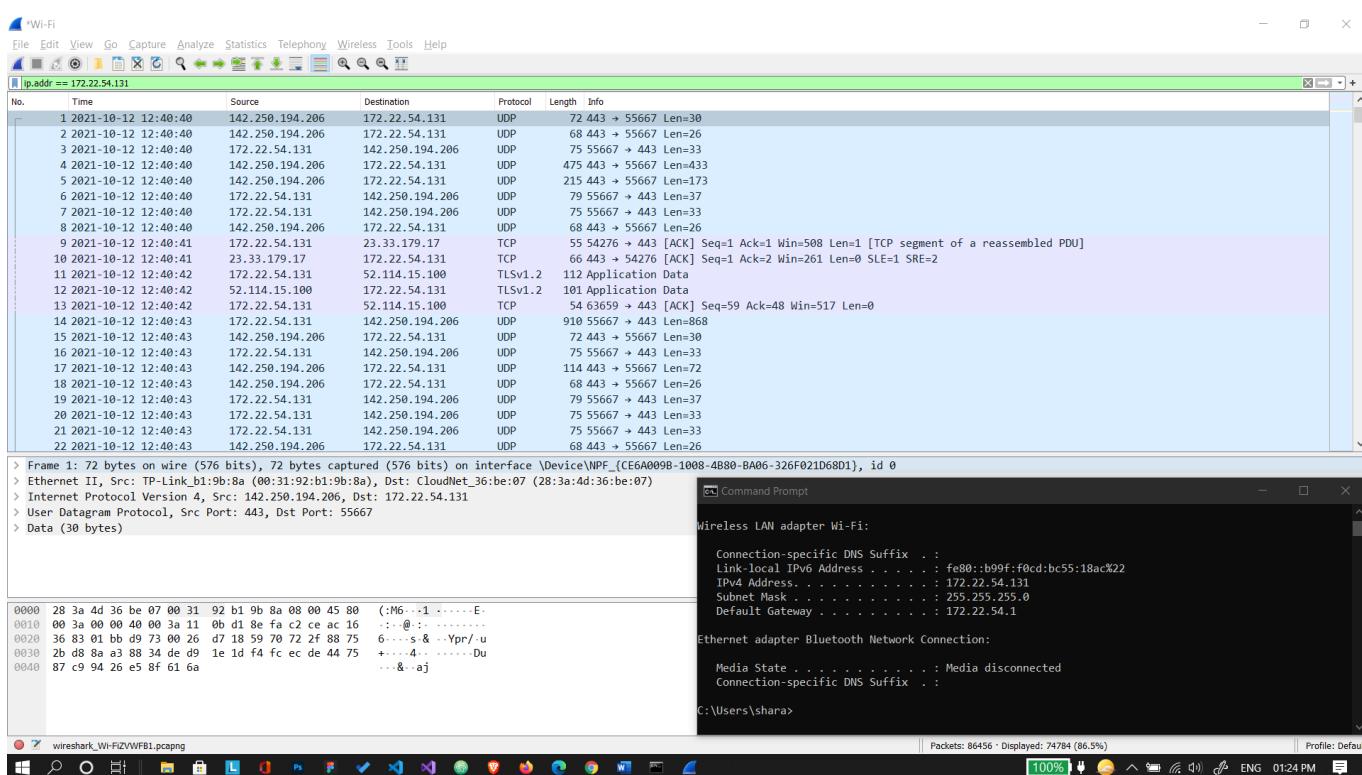
Step 3: Open Wireshark and enter “ip.addr == 172.22.54.131” into the filter, where I've obtained my default ipv4 address using ipconfig from cmd. This filter removes all packets that neither originate nor are destined to my host.

Step 4: Start packet capture in Wireshark.

Step 5: With browser, visit the web page: <http://www.ietf.org>

Step 6: Stop packet capture.

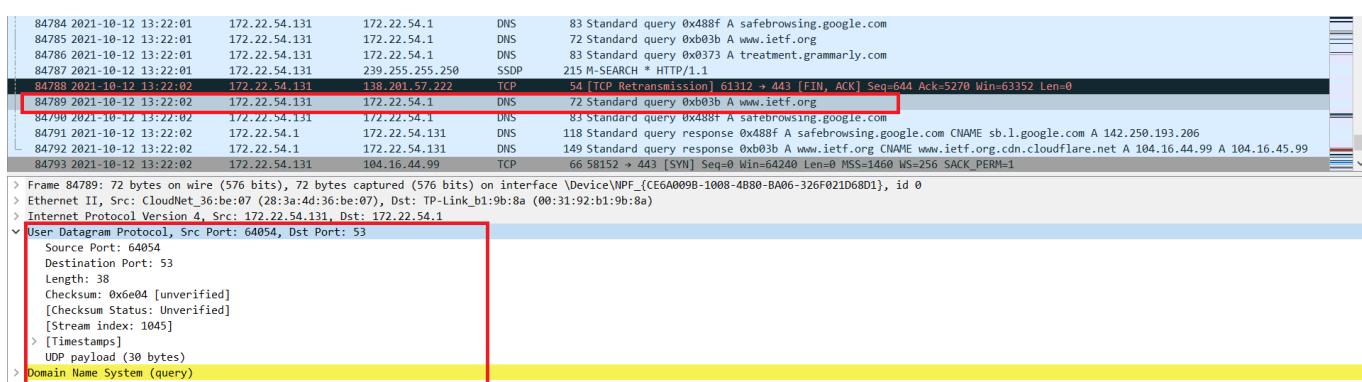
Step 7: Take a screenshot.



Back to questions:

4. Locate the DNS query and response messages. Are they 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

```
> Frame 84789: 72 bytes on wire (576 bits), 72 bytes captured (576 bits) on interface \Device\NPF_{CE6A009B-1008-4B80-BA06-326F021D68D1}, id 0
> Ethernet II, Src: CloudNet_36:be:07 (28:3a:4d:36:be:07), Dst: TP-Link_b1:9b:8a (00:31:92:b1:9b:8a)
> Internet Protocol Version 4, Src: 172.22.54.131, Dst: 172.22.54.131
> User Datagram Protocol, Src Port: 64054, Dst Port: 53
  Source Port: 64054
  Destination Port: 53
Length: 38
Checksum: 0x6e04 [unverified]
[Checksum Status: Unverified]
[Stream index: 1045]
> [Timestamps]
  UDP payload (30 bytes)
> Domain Name System (query)

0000  00 31 92 b1 9b 8a 28 3a 4d 36 be 07 08 00 45 00  -1... (: M6... E-
0010  00 3a 4d 44 00 00 8a 11 28 be ac 16 36 83 ac 16  .:MD... (. -6...
0020  36 01 fa 36 00 35 00 26 6e 04 b0 3b 01 00 00 01 6...6-5&n...;....
0030  00 00 00 00 00 00 03 77 77 04 69 65 74 66 03 .....w wwww.ietf.
0040  6f 72 67 00 00 01 01 org.....

```

User Datagram Protocol (udp), 8 bytes

Packets: 86456 · Displayed: 76002 (87.9%) · Dropped: 0 (0.0%)

100% 🔋 ENG 01:49 PM

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).

```
> Frame 84784 2021-10-12 13:22:01 172.22.54.131 172.22.54.1 DNS 83 Standard query 0x488f A safebrowsing.google.com
> Frame 84785 2021-10-12 13:22:01 172.22.54.131 172.22.54.1 DNS 72 Standard query 0xb03b A www.ietf.org
> Frame 84786 2021-10-12 13:22:01 172.22.54.131 172.22.54.1 DNS 83 Standard query 0x0373 A treatment.grammarly.com
> Frame 84787 2021-10-12 13:22:01 172.22.54.131 239.255.255.250 SSDP 215 M-SEARCH * HTTP/1.1
> Frame 84788 2021-10-12 13:22:02 172.22.54.131 128.201.57.222 TCP 54 [TCP_Retransmission] 61312 → 443 [EIN, ACK] Seq=644 Ack=5270 Win=63352 Len=0
> Frame 84789 2021-10-12 13:22:02 172.22.54.131 172.22.54.1 DNS 72 Standard query 0xb03b A www.ietf.org
> Frame 84790 2021-10-12 13:22:02 172.22.54.131 172.22.54.1 DNS 83 Standard query 0x488f A safebrowsing.google.com
> Frame 84791 2021-10-12 13:22:02 172.22.54.131 172.22.54.1 DNS 118 Standard query
> Frame 84792 2021-10-12 13:22:02 172.22.54.131 172.22.54.131 DNS 149 Standard query
> Frame 84793 2021-10-12 13:22:02 172.22.54.131 104.16.44.99 TCP 66 58152 → 443 [SYN]
> Frame 84794 2021-10-12 13:22:02 172.22.54.131 142.250.193.206 QUIC 1392 Initial, DCID=1d4
> Frame 84795 2021-10-12 13:22:02 172.22.54.131 142.250.193.206 QUIC 119 0-RTT, DCID=1d4
> Frame 84796 2021-10-12 13:22:02 138.201.57.222 172.22.54.131 TCP 54 443 → 61312 [ACK]
> Frame 84797 2021-10-12 13:22:02 172.22.54.131 54.192.166.11 TCP 54 [TCP_Retransmission]
> Frame 84798 2021-10-12 13:22:02 104.16.44.99 172.22.54.131 TCP 66 443 → 58152 [SYN]
> Frame 84799 2021-10-12 13:22:02 104.16.44.99 172.22.54.131 TCP 66 65304 → 443 [SYN]

> Frame 84781: 72 bytes on wire (576 bits), 72 bytes captured (576 bits) on interface \Device\NPF_{CE6A009B-1008-4B80-BA06-326F021D68D1}
> Ethernet II, Src: CloudNet_36:be:07 (28:3a:4d:36:be:07), Dst: TP-Link_b1:9b:8a (00:31:92:b1:9b:8a)
> Internet Protocol Version 4, Src: 172.22.54.131, Dst: 172.22.54.131
> User Datagram Protocol, Src Port: 64054, Dst Port: 53
  Source Port: 64054
  Destination Port: 53
Length: 38
Checksum: 0x6e04 [unverified]

Windows Command Prompt
C:\Windows\system32> ipconfig
Wireless LAN adapter Wi-Fi:
  Connection-specific DNS Suffix . :
  Link-local IPv6 Address . . . . . : fe80::b99f:f0cd:bc55:18ac%22
  IPv4 Address . . . . . : 172.22.54.131
  Subnet Mask . . . . . : 255.255.255.0
  Default Gateway . . . . . : 172.22.54.1

Ethernet adapter Bluetooth Network Connection:
  Connection-specific DNS Suffix . :
  Media State . . . . . : Media disconnected
  Connection-specific DNS Suffix . :
  C:\Windows\system32>
```

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”.

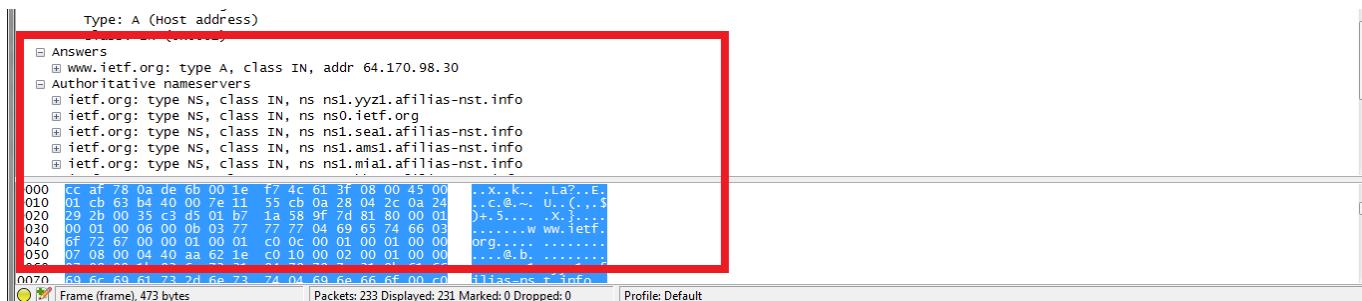
```
> Frame 84789 2021-10-12 13:22:02 172.22.54.131 172.22.54.1 DNS 72 Standard query 0xb03b A www.ietf.org
> Frame 84790 2021-10-12 13:22:02 172.22.54.131 172.22.54.1 DNS 83 Standard query 0x488f A safebrowsing.google.com
> Frame 84791 2021-10-12 13:22:02 172.22.54.131 172.22.54.131 DNS 118 Standard query response 0x488f A safebrowsing.google.com CNAME sb.l.google.com A 14
> Frame 84792 2021-10-12 13:22:02 172.22.54.131 172.22.54.131 DNS 149 Standard query response 0xb03b A www.ietf.org CNAME www.ietf.org.cdn.cloudflare.net

> User Datagram Protocol, Src Port: 64054, Dst Port: 53
> Domain Name System (query)
  Transaction ID: 0xb03b
    Flags: 0x0100 Standard query
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  > Queries
    > www.ietf.org: type A, class IN
    _Response In: 84/94

Ready to load or capture
Packets: 86456 · Displayed: 76002 (87.9%) · Dropped: 0 (0.0%)
```

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.

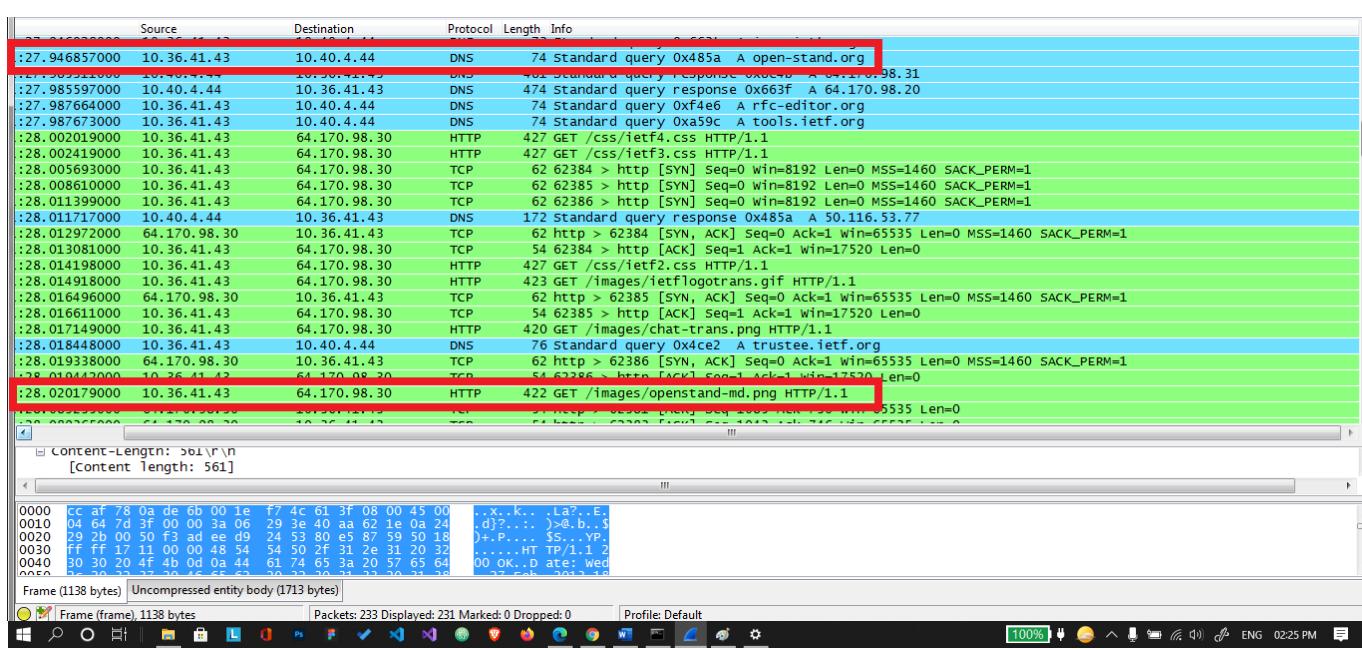


9. Consider the subsequent TCP SYN packet sent by your host. Does the destination IP address of the SYN packet correspond to any of the IP addresses provided in the DNS response message?

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.

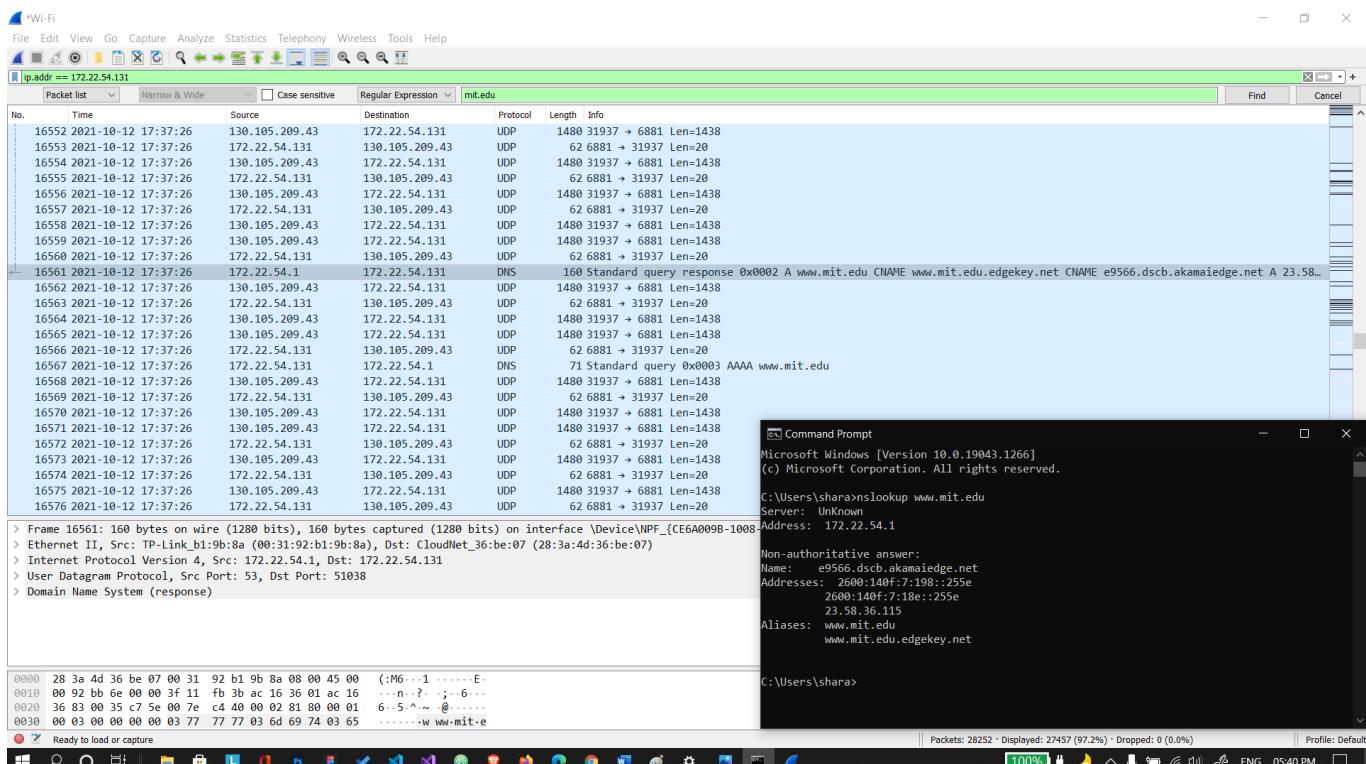
10. This web page contains images. Before retrieving each image, does your host issue new DNS queries?

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:](#)

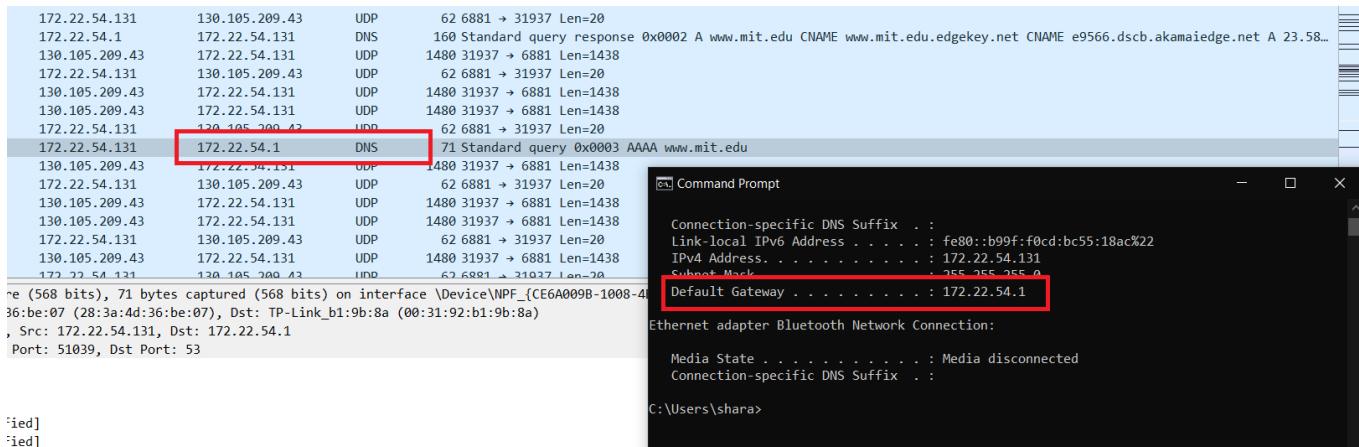
11. What is the destination port for the DNS query message? What is the source port of DNS response message?

Destination Port: 53
Source Port: 51039

Time	Date	Source IP	Destination IP	Protocol	Details
165666	2021-10-12 17:37:26	172.22.54.131	130.105.209.43	UDP	62 6881 → 31937 Len=20
16567	2021-10-12 17:37:26	172.22.54.131	172.22.54.1	DNS	71 [Standard query 0x0003 AAAA www.mit.edu
16568	2021-10-12 17:37:26	130.105.209.43	172.22.54.131	UDP	1480 31937 → 6881 Len=1438
16569	2021-10-12 17:37:26	172.22.54.131	130.105.209.43	UDP	62 6881 → 31937 Len=20
16570	2021-10-12 17:37:26	130.105.209.43	172.22.54.131	UDP	1480 31937 → 6881 Len=1438
16571	2021-10-12 17:37:26	130.105.209.43	172.22.54.131	UDP	1480 31937 → 6881 Len=1438
> Frame 16567: 71 bytes on wire (568 bits), 71 bytes captured (568 bits) on interface \Device\NPF_{CE6A009B-1008-4B80-BA06-326F021D68D1}, id 0					
> Ethernet II, Src: CloudNet_36:b7:07 (28:3a:4d:36:b7:07), Dst: TP-Link_b1:9b:8a (00:31:92:b1:9b:8a)					
> Internet Protocol Version 4 Src: 172.22.54.131 Dst: 172.22.54.1					
> User Datagram Protocol, Src Port: 51039, Dst Port: 53					
Source Port: 51039					
Destination Port: 53					
Length: 37					
Checksum: 0x05b0 [unverified]					
[Checksum Status: Unverified]					
[Stream index: 11]					
> [Timestamps]					
UDP payload (29 bytes)					
> Domain Name System (query)					
0000	00 31 92 b1 9b 8a 28 3a	4d 36 be 07 08 00 45 00	1:....(M6....E		
0010	00 39 75 57 00 80 11 00	ac ac 16 36 83 ac 16	9uM.....6...		
0020	36 01 c7 5f 00 35 00 25	05 b9 00 03 01 00 00 01	6....5 %		
0030	00 00 00 00 00 03 77	77 77 03 6d 69 74 03 65www.mit.e		
Ready to load or capture					
Packets: 28252 · Displayed: 27457 (97.2%) · Dropped: 0 (0.0%)					
Profile: Default					

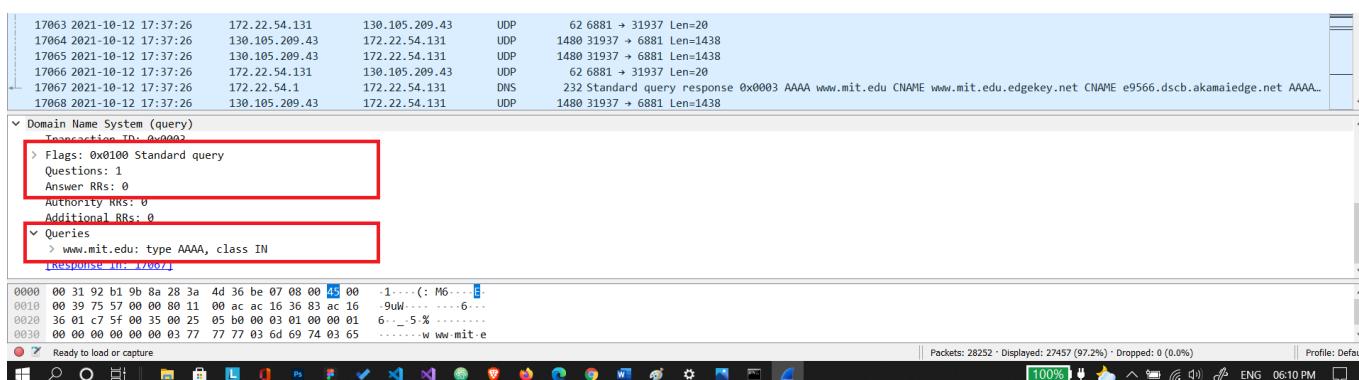
12. To what IP address is the DNS query message sent? Is this the IP address of your default local DNS server?

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.

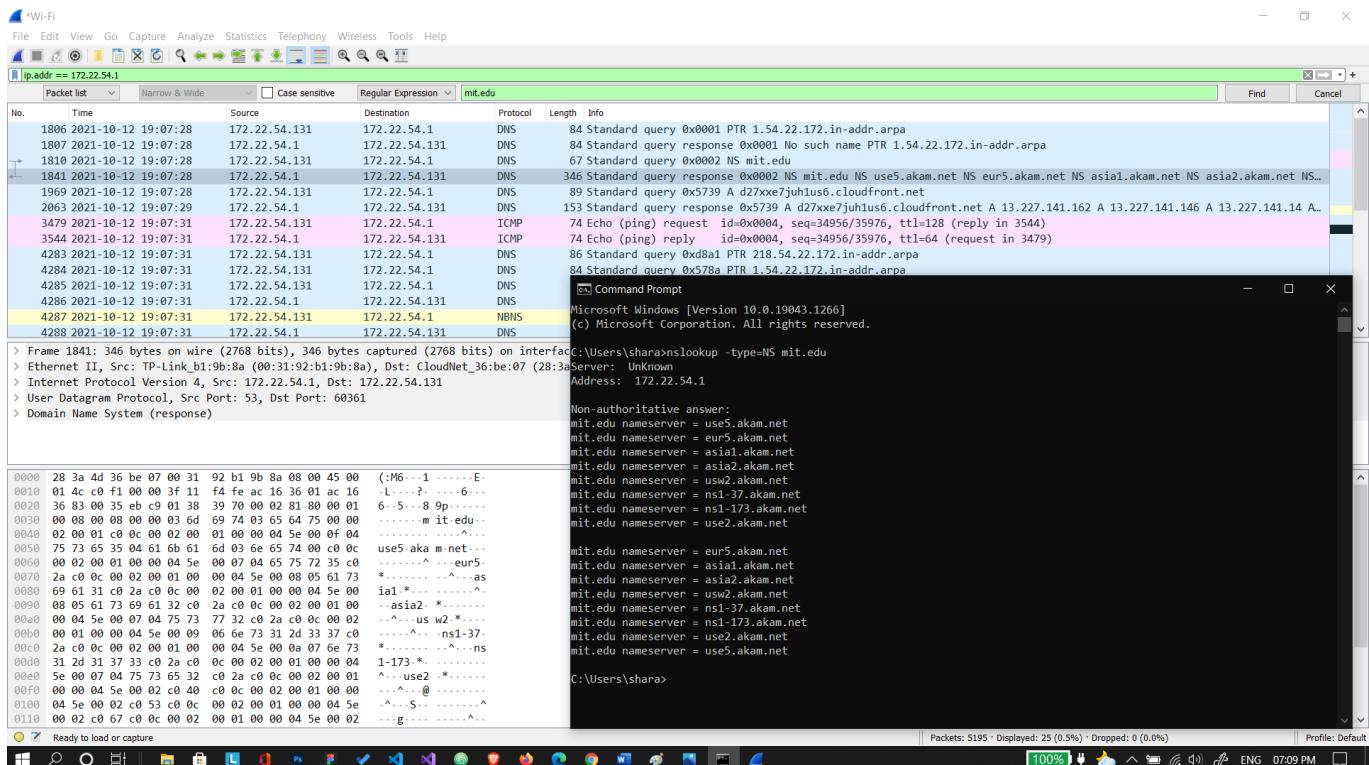
Provided every step of the way.

Section 2.1 – Steps:

Step 1: Repeat the previous experiment.

Step 2: But instead issue the command: `nslookup -type=NS mit.edu`

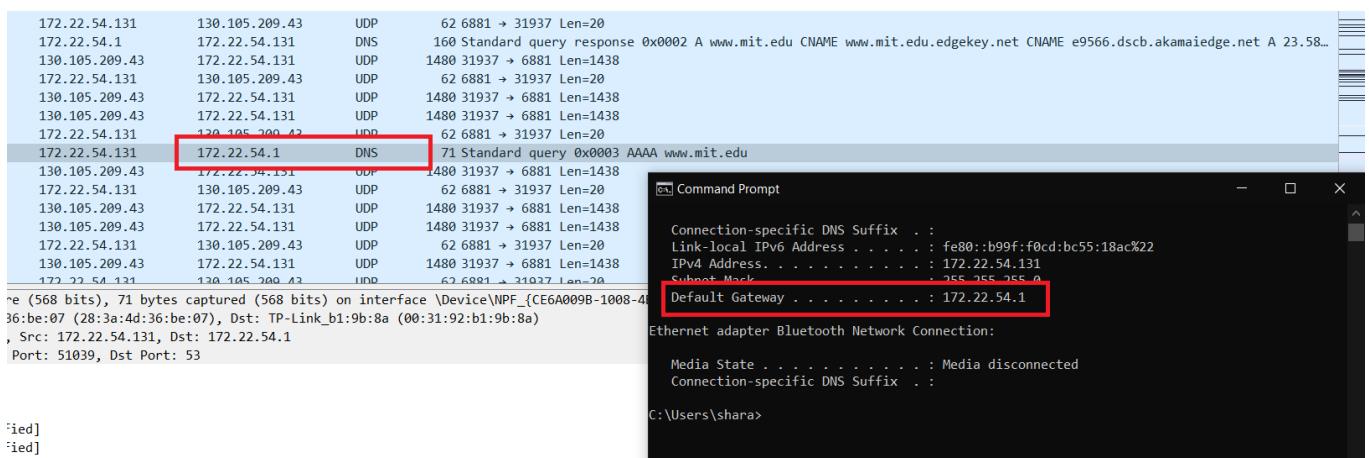
Step 3: Take a screenshot.



Back to questions:

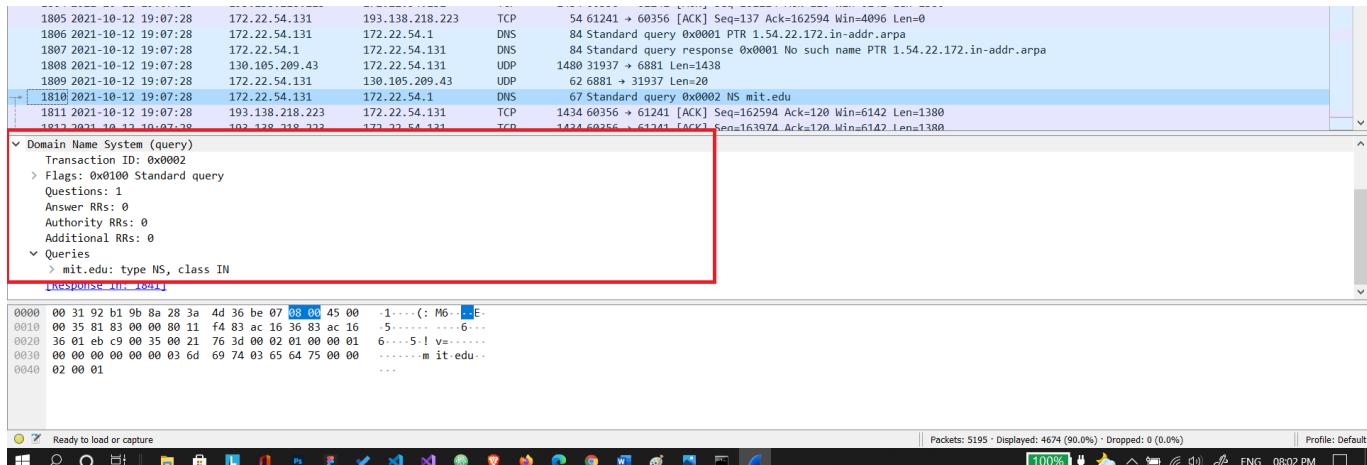
16. To what IP address is the DNS query message sent? Is this the IP address of your default local DNS server?

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).



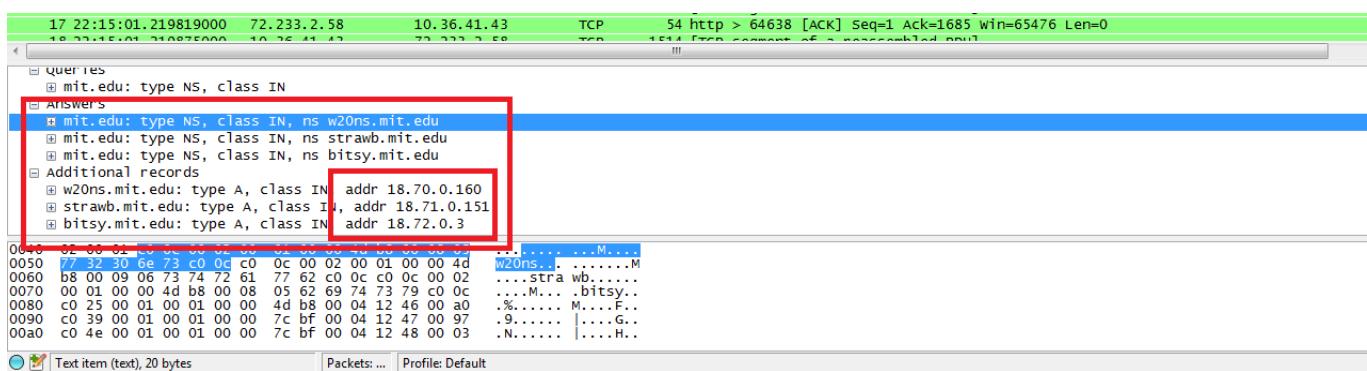
17. Examine the DNS query message. What “Type” of DNS query is it? Does the query message contain any “answers”?

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.

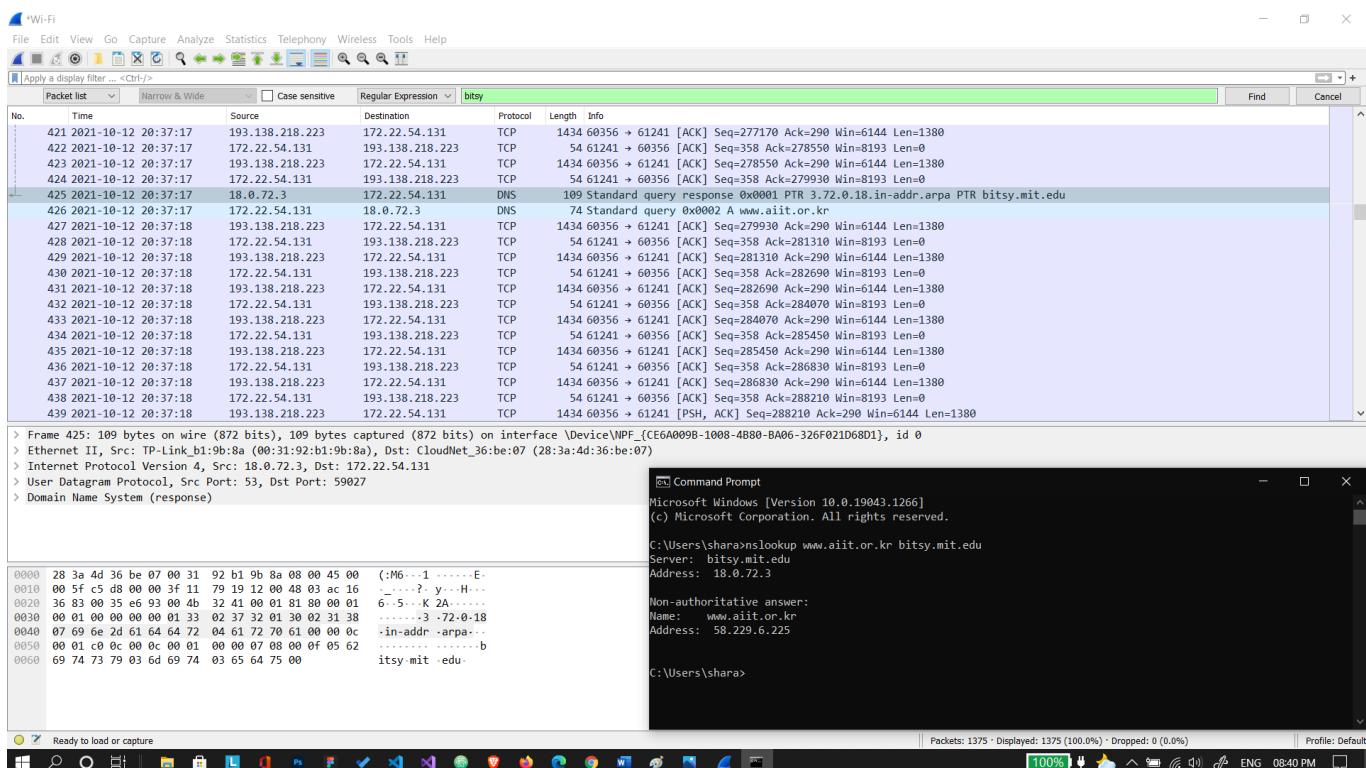
Provided every step of the way.

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`

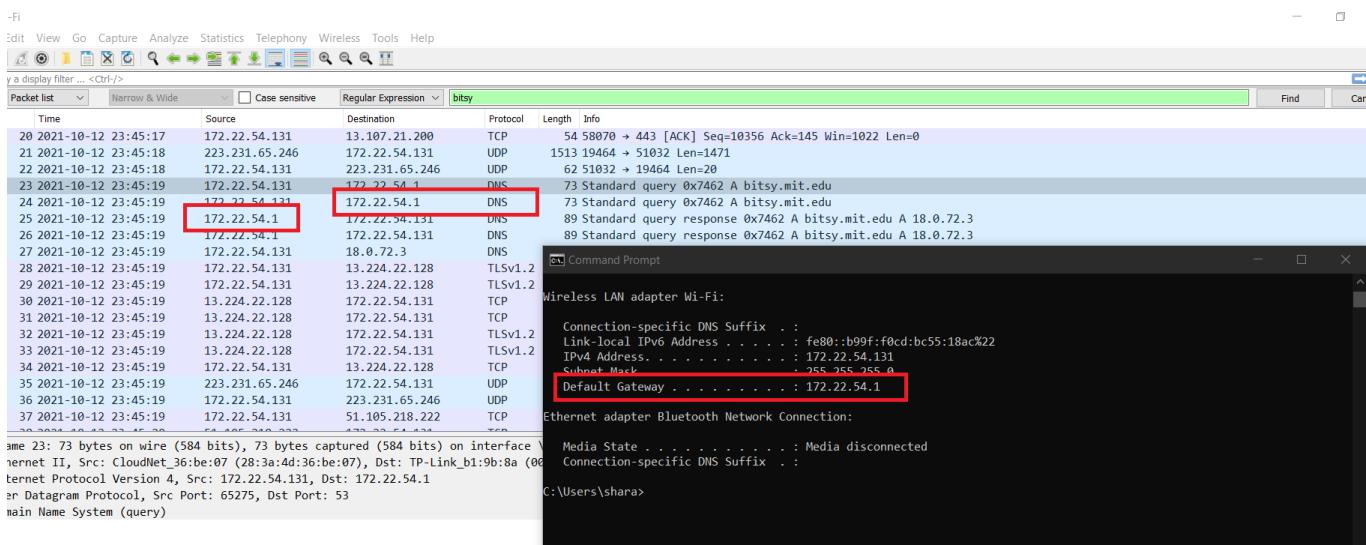
Step 3: Take a screenshot.



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 2021-10-12 23:45:18      172.22.54.131      223.231.65.246      UDP      62 51032 → 19464 Len=20
23 2021-10-12 23:45:19      172.22.54.131      172.22.54.1      DNS      73 Standard query 0x7462 A bitsy.mit.edu
24 2021-10-12 23:45:19      172.22.54.131      172.22.54.1      DNS      73 Standard query 0x7462 A bitsy.mit.edu
25 2021-10-12 23:45:19      172.22.54.1      172.22.54.131      DNS      89 Standard query response 0x7462 A bitsy.mit.edu A 18.0.72.3
26 2021-10-12 23:45:19      172.22.54.1      172.22.54.131      DNS      89 Standard query response 0x7462 A bitsy.mit.edu A 18.0.72.3
27 2021-10-12 23:45:19      172.22.54.131      18.0.72.3      DNS      82 Standard query response 0x0001 PTR 3.72.0.18.in-addr.arpa

UDP payload (31 bytes)
└ Domain Name System (query)
  └ Transaction ID: 0x7462
    > Flags: 0x0100 Standard query
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
    < Queries
      > bitsy.mit.edu: type A, class IN
        Name: bitsy.mit.edu
        [Name Length: 13]
        [Label Count: 3]
        Type: A (Host Address) (1)
        Class: IN (0x0001)
        [Retransmitted request. Original request in: 23]
        [Retransmission: True]

```

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 2021-10-12 23:45:19      172.22.54.131      172.22.54.1      DNS      73 Standard query 0x7462 A bitsy.mit.edu
24 2021-10-12 23:45:19      172.22.54.131      172.22.54.1      DNS      73 Standard query 0x7462 A bitsy.mit.edu
25 2021-10-12 23:45:19      172.22.54.1      172.22.54.131      DNS      89 Standard query response 0x7462 A bitsy.mit.edu A 18.0.72.3
26 2021-10-12 23:45:19      172.22.54.1      172.22.54.131      DNS      89 Standard query response 0x7462 A bitsy.mit.edu A 18.0.72.3
27 2021-10-12 23:45:19      172.22.54.131      18.0.72.3      DNS      82 Standard query response 0x0001 PTR 3.72.0.18.in-addr.arpa
28 2021-10-12 23:45:19      172.22.54.131      13.224.22.128      TLSv1.2      121 Application Data
29 2021-10-12 23:45:19      172.22.54.131      13.224.22.128      TLSv1.2      93 Application Data
30 2021-10-12 23:45:19      13.224.22.128      172.22.54.131      TCP      54 443 → 59639 [ACK] Seq=1 Ack=68 Win=133 Len=0

└ Domain Name System (response)
  Transaction ID: 0x7462
  < Flags: 0x8180 Standard query response, No error
    1... .... .... = Response: Message is a response
    .000 0.... .... = Opcode: Standard query (0)
    .... 0.... .... = Authoritative: Server is not an authority for domain
    .... 0.... .... = Truncated: Message is not truncated
    .... 1.... .... = Recursion desired: Do query recursively
    .... 1.... .... = Recursion available: Server can do recursive queries
    .... 0.... .... = Z: reserved (0)
    .... 0.... .... = Answer authenticated: Answer/authority portion was not authenticated by the server
    .... 0.... .... = Non-authenticated data: Unacceptable
    .... 0.... .... = 0000 = Reply code: No error (0)
  Questions: 1
  Answer RRs: 1
  Authority RRs: 0
  Additional RRs: 0
  < Queries
    > bitsv.mit.edu: type A, class TN
    < Answers
      > bitsy.mit.edu: type A, class IN, addr 18.0.72.3
      [Request In: 23]
      [Time: 0.157684000 seconds]

```

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>!

A screenshot of the Wireshark application. The main window displays a list of network packets captured over a Wi-Fi interface. The packet list shows several HTTP requests and responses. The details pane below the list provides a breakdown of the captured frame (Frame 914) and its components. The bottom half of the screen shows the raw hex and ASCII dump of the captured file content, which corresponds to the 'HTTP-wireshark-file1.html' file mentioned in the browser screenshot.

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

Frame 914: 541 bytes on wire (4328 bits), 541 bytes captured (4328 bits) on interface \Device\NPF_{CE6A009B-1008-4B80-BA06-326F021D68D1}, id 0
 Ethernet II, Src: CloudNet_36:be:07 (28:3a:4d:36:be:07), Dst: TP-Link_b1:b9:8a (00:31:92:b1:b9:8a)
 Internet Protocol Version 4, Src: 172.22.54.131, Dst: 128.119.245.12
 Transmission Control Protocol, Src Port: 64352, Dst Port: 80, Seq: 1, Ack: 1, Len: 487
 Hypertext Transfer Protocol
 > GET /wireshark-labs/HTTP-wireshark-file1.html HTTP/1.1\r\n
 > [Expert Info (Chat/Sequence): GET /wireshark-labs/HTTP-wireshark-file1.html HTTP/1.1\r\n]
 Request Method: GET
 Request URI: /wireshark-labs/HTTP-wireshark-file1.html
 Request Version: HTTP/1.1
 Host: gaia.cs.umass.edu\r\n
 Connection: keep-alive\r\n
 Upgrade-Insecure-Requests: 1\r\n
 Accept-Language: en-US,en-GB;q=0.9,en;q=0.8\r\n

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.

Hypertext Transfer Protocol
 > GET /wireshark-labs/HTTP-wireshark-file1.html HTTP/1.1\r\n
 Host: gaia.cs.umass.edu\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/94.0.4606.81 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-GB;q=0.9,en;q=0.8\r\n

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

Frame 914: 541 bytes on wire (4328 bits), 541 bytes captured (4328 bits) on interface \Device\NPF_{CE6A009B-1008-4B80-BA06-326F021D68D1}, id 0
 Ethernet II, Src: CloudNet_36:be:07 (28:3a:4d:36:be:07), Dst: TP-Link_b1:b9:8a (00:31:92:b1:b9:8a)
 Internet Protocol Version 4, Src: 172.22.54.131, Dst: 128.119.245.12
 Transmission Control Protocol, Src Port: 64352, Dst Port: 80, Seq: 1, Ack: 1, Len: 487
 Hypertext Transfer Protocol
 > GET /wireshark-labs/HTTP-wireshark-file1.html HTTP/1.1\r\n
 > [Expert Info (Chat/Sequence): GET /wireshark-labs/HTTP-wireshark-file1.html HTTP/1.1\r\n]
 Request Method: GET
 Request URI: /wireshark-labs/HTTP-wireshark-file1.html
 Request Version: HTTP/1.1
 Host: gaia.cs.umass.edu\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/94.0.4606.81 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-GB;q=0.9,en;q=0.8\r\n

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

The status code returned was “200 OK”.

Frame 914: 541 bytes on wire (4328 bits), 541 bytes captured (4328 bits) on interface \Device\NPF_{CE6A009B-1008-4B80-BA06-326F021D68D1}, id 0
 Ethernet II, Src: CloudNet_36:be:07 (28:3a:4d:36:be:07), Dst: TP-Link_b1:b9:8a (00:31:92:b1:b9:8a)
 Internet Protocol Version 4, Src: 172.22.54.131, Dst: 128.119.245.12
 Transmission Control Protocol, Src Port: 64352, Dst Port: 80, Seq: 1, Ack: 1, Len: 487
 Hypertext Transfer Protocol
 > GET /wireshark-labs/HTTP-wireshark-file1.html HTTP/1.1\r\n
 > [Expert Info (Chat/Sequence): GET /wireshark-labs/HTTP-wireshark-file1.html HTTP/1.1\r\n]
 Request Method: GET
 Request URI: /wireshark-labs/HTTP-wireshark-file1.html
 Request Version: HTTP/1.1
 Host: gaia.cs.umass.edu\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/94.0.4606.81 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-GB;q=0.9,en;q=0.8\r\n

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

```

964 2021-10-13 14:05:41 128.119.245.12 172.22.54.131 HTTP 540 HTTP/1.1 200 OK (text/html)

> Transmission Control Protocol, Src Port: 80, Dst Port: 64352, Seq: 1, Ack: 488, Len: 486
└ Hypertext Transfer Protocol
  < HTTP/1.1 200 OK\r\n
    > [Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n]
      Response Version: HTTP/1.1
      Status Code: 200
      [Status Code Description: OK]
      Response Phrase: OK
      Date: Wed, 13 Oct 2021 08:35:41 GMT\r\n
      Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.2k-fips PHP/7.4.24 mod_perl/2.0.11 Perl/v5.16.3\r\n
      Last-Modified: Wed, 13 Oct 2021 05:59:01 GMT\r\n
      ETag: "80-5ce35a8aa83d"\r\n
      Accept-Ranges: bytes\r\n

```

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

128 bytes of content are being returned.

```

*Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
http
No. Time Source Destination Protocol Length Info
914 2021-10-13 14:05:41 172.22.54.131 128.119.245.12 HTTP 541 GET /wireshark-labs/HTTP-wireshark-file1.html HTTP/1.1
964 2021-10-13 14:05:41 128.119.245.12 172.22.54.131 HTTP 540 HTTP/1.1 200 OK (text/html)

└ Hypertext Transfer Protocol
  < HTTP/1.1 200 OK\r\n
    > [Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n]
      Response Version: HTTP/1.1
      Status Code: 200
      [Status Code Description: OK]
      Response Phrase: OK
      Date: Wed, 13 Oct 2021 08:35:41 GMT\r\n
      Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.2k-fips PHP/7.4.24 mod_perl/2.0.11 Perl/v5.16.3\r\n
      Last-Modified: Wed, 13 Oct 2021 05:59:01 GMT\r\n
      ETag: "80-5ce35a8aa83d"\r\n
      Accept-Ranges: bytes\r\n
    < Content-Length: 128\r\n
      [Content length: 128]
      Keep-Alive: timeout=5, max=100\r\n
      Connection: Keep-Alive\r\n
      Content-Type: text/html; charset=UTF-8\r\n
      \r\n
      [HTTP response 1/2]
      [Time since request: 0.340907000 seconds]

```

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.

Congratulations again! Now you've downloaded the file lab2-2.html.
This file's last modification date will not change.

Thus if you download this multiple times on your browser, a complete copy will only be sent once by the server due to the inclusion of the IN-MODIFIED-SINCE field in your browser's HTTP GET request to the server.

Wireshark capture details:

- Frame 4297: 541 bytes on wire (4328 bits), 541 bytes captured (4328 bits) on interface \Device\NPF_{CE6A009B-1008-4B80-BA06-326F021D68D1}, id 0
 - Ethernet II, Src: CloudNet_36:be:07 (28:3a:4d:36:be:07), Dst: TP-Link_b1:9b:8a (00:31:92:b1:9b:8a)
 - Internet Protocol Version 4, Src: 172.22.54.131, Dst: 128.119.245.12
 - Transmission Control Protocol, Src Port: 61143, Dst Port: 80, Seq: 1, Ack: 1, Len: 487
- Protocol: Hypertext Transfer Protocol

```

0000  00 31 92 b1 9b 8a 28 3a 4d 36 b0 07 08 00 45 00 ··1···(; M6···E·
0010  02 0f 6e 82 40 00 80 06 32 49 a0 16 36 83 80 77 ··n @·.. 21· 6··w
0020  f5 0c ee d7 00 50 81 11 d0 b2 b0 40 04 4f 50 18 ···P· .. @OP·
0030  02 01 c3 e0 00 47 45 54 20 2f 77 69 72 65 73 ···GE T /wires
0040  68 61 72 6b 2d 6c 61 62 73 2f 48 54 50 2d 77 hark-lab s/HTTP-w
0050  69 72 65 73 68 61 72 6b 2d 66 69 6c 65 32 2e 68 ireshark -file2.h
0060  74 6d 6c 20 48 54 50 50 2f 31 2e 31 0d 0a 48 6f tml HTTP /1.1-Ho
0070  73 74 3a 20 67 61 69 61 2e 63 73 2e 75 60 61 73 st: gala .cs.umas
0080  73 2e 65 64 75 0d 0a 43 6f 6e 66 65 63 74 69 6f s.edu -C onnectio
0090  6e 3a 20 6b 65 65 70 2d 61 6c 69 76 65 0d 0a 55 n: keep- alive·U
00a0  70 67 72 61 64 65 2d 49 66 73 65 63 75 72 65 2d pgrade-I nsecure-
00b0  52 65 71 75 65 73 74 73 3a 20 31 0d 0a 55 73 65 Requests : 1·Use
00c0  72 2d 41 67 65 66 74 3a 20 4d 6f 7a 69 6c 6c 61 r-Agent: Mozilla

```

Packets: 16954 - Displayed: 4 (0.0%) - Dropped: 0 (0.0%) | Profile: Default

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.

A screenshot of the Wireshark interface. The packet list shows several HTTP requests and responses. The second response (packet 5042) is selected. In the "Http" pane, the "Line-based text data: text/html (10 lines)" section is expanded and highlighted with a red box. It displays the HTML content of the file, including the message "Congratulations again! Now you've downloaded the file lab2-2.html." and instructions about file modification dates.

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.

A screenshot of the Wireshark interface. The packet list shows several HTTP requests and responses. The second request (packet 5042) is selected. In the "Http" pane, the raw text content of the GET request is shown, including the "If-Modified-Since" header with the value "Wed, 13 Oct 2021 05:59:01 GMT".

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.

* Wi-Fi File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

http

No.	Time	Source	Destination	Protocol	Length	Info
2063	2021-10-13 16:33:50	172.22.54.131	128.119.245.12	HTTP	541	GET /wireshark-labs/HTTP-wireshark-file3.html HTTP/1.1
2357	2021-10-13 16:33:42	128.119.245.12	172.22.54.131	HTTP	535	HTTP/1.1 200 OK (text/html)
4705	2021-10-13 16:33:46	172.22.54.131	128.119.245.12	HTTP	654	GET /wireshark-labs/HTTP-wireshark-file3.html HTTP/1.1
4890	2021-10-13 16:33:46	128.119.245.12	172.22.54.131	HTTP	294	HTTP/1.1 304 Not Modified
7135	2021-10-13 16:33:50	172.22.54.131	172.22.54.1	HTTP	138	GET /qyyjn/rootDesc.xml HTTP/1.1
7149	2021-10-13 16:33:50	172.22.54.131	172.22.54.1	HTTP	138	GET /qyyjn/rootDesc.xml HTTP/1.1
7150	2021-10-13 16:33:50	172.22.54.131	172.22.54.1	HTTP	138	GET /qyyjn/rootDesc.xml HTTP/1.1
7152	2021-10-13 16:33:50	172.22.54.131	172.22.54.1	HTTP	138	GET /qyyjn/rootDesc.xml HTTP/1.1
7153	2021-10-13 16:33:50	172.22.54.131	172.22.54.1	HTTP	138	GET /qyyjn/rootDesc.xml HTTP/1.1
7155	2021-10-13 16:33:50	172.22.54.131	172.22.54.1	HTTP	1359	HTTP/1.1 200 OK
7162	2021-10-13 16:33:50	172.22.54.131	172.22.54.1	HTTP	138	GET /qyyjn/rootDesc.xml HTTP/1.1
7163	2021-10-13 16:33:50	172.22.54.131	172.22.54.1	HTTP	138	GET /qyyjn/rootDesc.xml HTTP/1.1
7173	2021-10-13 16:33:50	172.22.54.1	172.22.54.131	HTTP	1359	HTTP/1.1 200 OK
7180	2021-10-13 16:33:50	172.22.54.1	172.22.54.131	HTTP	1359	HTTP/1.1 200 OK

```
> Frame 2063: 541 bytes on wire (4328 bits), 541 bytes captured (4328 bits) on interface \Device\NPF_{CE6A009B-1008-4B80-BA06-326F021D68D1}, id 0
> Ethernet II, Src: CloudNet_36:be:07 (28:3a:4d:36:be:07), Dst: TP-Link_b1:9b:8a (00:31:92:b1:9b:8a)
> Internet Protocol Version 4, Src: 172.22.54.131, Dst: 128.119.245.12
> Transmission Control Protocol, Src Port: 63114, Dst Port: 80, Seq: 1, Ack: 1, Len: 487
> Hypertext Transfer Protocol
```

```
0000 00 31 92 b1 9b 8a 28 3a 4d 36 be 07 08 00 45 00 -·1···(: M6···E·
0010 02 0f 0e 9f 40 00 80 06 32 2c ac 16 36 83 80 77 ··n@···2···6···w
0020 f5 0c f6 8a 00 50 2b 1f 5a 4d 5e 2e a2 5c 50 18 ····P+· ZM···P·
0030 02 01 50 89 00 47 45 54 20 2f 77 69 72 65 73 ··P···GE T /wires
0040 68 61 72 6b 2d 6c 61 62 73 2f 48 54 54 50 2d 77 hark-lab s/HTTP-w
0050 69 72 65 73 68 61 72 6b 2d 66 69 6c 65 33 2e 68 ireshark -file3.h
0060 74 6d 6c 20 48 54 54 50 2f 31 2e 31 0d 0a 48 6f tml HTTP /1.1 - Ho
0070 73 74 3a 20 67 61 69 61 2e 63 73 2e 75 6d 61 73 st: gaia .cs.umass
0080 73 2e 65 64 75 0d 0a 43 6f 6e 66 65 63 74 69 6f .edu - C onnectio
0090 6e 3a 20 66 65 65 70 2d 61 6c 69 76 65 0d 0a 55 n: keep - alive ·U
00a0 70 67 72 63 64 65 2d 49 6d 73 65 63 75 72 65 2d pgrade -I nsecure-
00b0 52 65 71 75 65 73 74 73 3a 20 31 0d 0a 55 73 65 Requests : 1 ·Use
00c0 72 2d 41 67 65 6e 74 3a 20 4d 6f 7a 69 6c 6c 61 r-Agent: Mozilla
```

HyperText Transfer Protocol: Protocol

Packets: 8077 - Displayed: 58 (0.7%) - Dropped: 0 (0.0%)

Profile: Default

100% ENG 04:37 PM

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.

No.	Time	Source	Destination	Protocol	Length	Info
2063	2021-10-13 16:33:42	172.22.54.131	128.119.245.12	HTTP	541	GET /wireshark-labs/HTTP-wireshark-file3.html HTTP/1.1
2357	2021-10-13 16:33:42	128.119.245.12	172.22.54.131	HTTP	535	HTTP/1.1 200 OK (text/html)
4705	2021-10-13 16:33:46	172.22.54.131	128.119.245.12	HTTP	654	GET /wireshark-labs/HTTP-wireshark-file3.html HTTP/1.1
4890	2021-10-13 16:33:46	128.119.245.12	172.22.54.131	HTTP	294	HTTP/1.1 304 Not Modified
7135	2021-10-13 16:33:50	172.22.54.131	172.22.54.1	HTTP	138	GET /qyyjn/rootDesc.xml HTTP/1.1
7149	2021-10-13 16:33:50	172.22.54.131	172.22.54.1	HTTP	138	GET /qyyjn/rootDesc.xml HTTP/1.1
7150	2021-10-13 16:33:50	172.22.54.131	172.22.54.1	HTTP	138	GET /qyyjn/rootDesc.xml HTTP/1.1
7152	2021-10-13 16:33:50	172.22.54.131	172.22.54.1	HTTP	138	GET /qyyjn/rootDesc.xml HTTP/1.1
7153	2021-10-13 16:33:50	172.22.54.131	172.22.54.1	HTTP	138	GET /qyyjn/rootDesc.xml HTTP/1.1
7159	2021-10-13 16:33:50	172.22.54.1	172.22.54.131	HTTP/...	1359	HTTP/1.1 200 OK

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.

No.	Time	Source	Destination	Protocol	Length	Info
2063	2021-10-13 16:33:42	172.22.54.131	128.119.245.12	HTTP	541	GET /wireshark-labs/HTTP-wireshark-file3.html HTTP/1.1
2357	2021-10-13 16:33:42	128.119.245.12	172.22.54.131	HTTP	535	HTTP/1.1 200 OK (text/html)
4705	2021-10-13 16:33:46	172.22.54.131	128.119.245.12	HTTP	654	GET /wireshark-labs/HTTP-wireshark-file3.html HTTP/1.1
4890	2021-10-13 16:33:46	128.119.245.12	172.22.54.131	HTTP	294	HTTP/1.1 304 Not Modified
7135	2021-10-13 16:33:50	172.22.54.131	172.22.54.1	HTTP	138	GET /qyyjn/rootDesc.xml HTTP/1.1
7149	2021-10-13 16:33:50	172.22.54.131	172.22.54.1	HTTP	138	GET /qyyjn/rootDesc.xml HTTP/1.1
7150	2021-10-13 16:33:50	172.22.54.131	172.22.54.1	HTTP	138	GET /qyyjn/rootDesc.xml HTTP/1.1

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

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

No.	Time	Source	Destination	Protocol	Length	Info
2063	2021-10-13 16:33:42	172.22.54.131	128.119.245.12	HTTP	541	GET /wireshark-labs/HTTP-wireshark-file3.html HTTP/1.1
2357	2021-10-13 16:33:42	128.119.245.12	172.22.54.131	HTTP	535	HTTP/1.1 200 OK (text/html)
4705	2021-10-13 16:33:46	172.22.54.131	128.119.245.12	HTTP	654	GET /wireshark-labs/HTTP-wireshark-file3.html HTTP/1.1
4890	2021-10-13 16:33:46	128.119.245.12	172.22.54.131	HTTP	294	HTTP/1.1 304 Not Modified
7135	2021-10-13 16:33:50	172.22.54.131	172.22.54.1	HTTP	138	GET /qyyjn/rootDesc.xml HTTP/1.1
7149	2021-10-13 16:33:50	172.22.54.131	172.22.54.1	HTTP	138	GET /qyyjn/rootDesc.xml HTTP/1.1
7150	2021-10-13 16:33:50	172.22.54.131	172.22.54.1	HTTP	138	GET /qyyjn/rootDesc.xml HTTP/1.1
7152	2021-10-13 16:33:50	172.22.54.131	172.22.54.1	HTTP	138	GET /qyyjn/rootDesc.xml HTTP/1.1
7153	2021-10-13 16:33:50	172.22.54.131	172.22.54.1	HTTP	138	GET /qyyjn/rootDesc.xml HTTP/1.1

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.

```
Type: IPv4 (0x0800)
> Internet Protocol Version 4, Src: 128.119.245.12, Dst: 172.22.54.131
> Transmission Control Protocol, Src Port: 80, Dst Port: 63114, Seq: 4381, Ack: 488, Len: 481
> [4 Reassembled TCP Segments (4861 bytes): #2354(1460), #2355(1460), #2356(1460), #2357(481)]
  [Frame: 2354, payload: 0-1459 (1460 bytes)]
  [Frame: 2355, payload: 1460-2919 (1460 bytes)]
  [Frame: 2356, payload: 2920-4379 (1460 bytes)]
  [Frame: 2357, payload: 4380-4860 (481 bytes)]
  [Segment count: 4]
  [Reassembled TCP length: 4861]
  [Reassembled TCP Data: 485454502f312e3120323030204f4b0d0a446174653a205765642c203133204f63742032...]
> Hypertext Transfer Protocol
> Line-based text data: text/html (98 lines)

Frame (535 bytes)  Reassembled TCP (4861 bytes)
● 🔗 Source Port (tcp.srcport), 2 bytes
```

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.

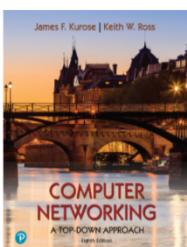
Step 3: Enter the following URL in the browser: <http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file4.html>. The browser should display a short HTML file with two images. These two images are referenced in the base HTML file. That is, the images themselves are not contained in the HTML; instead, 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.

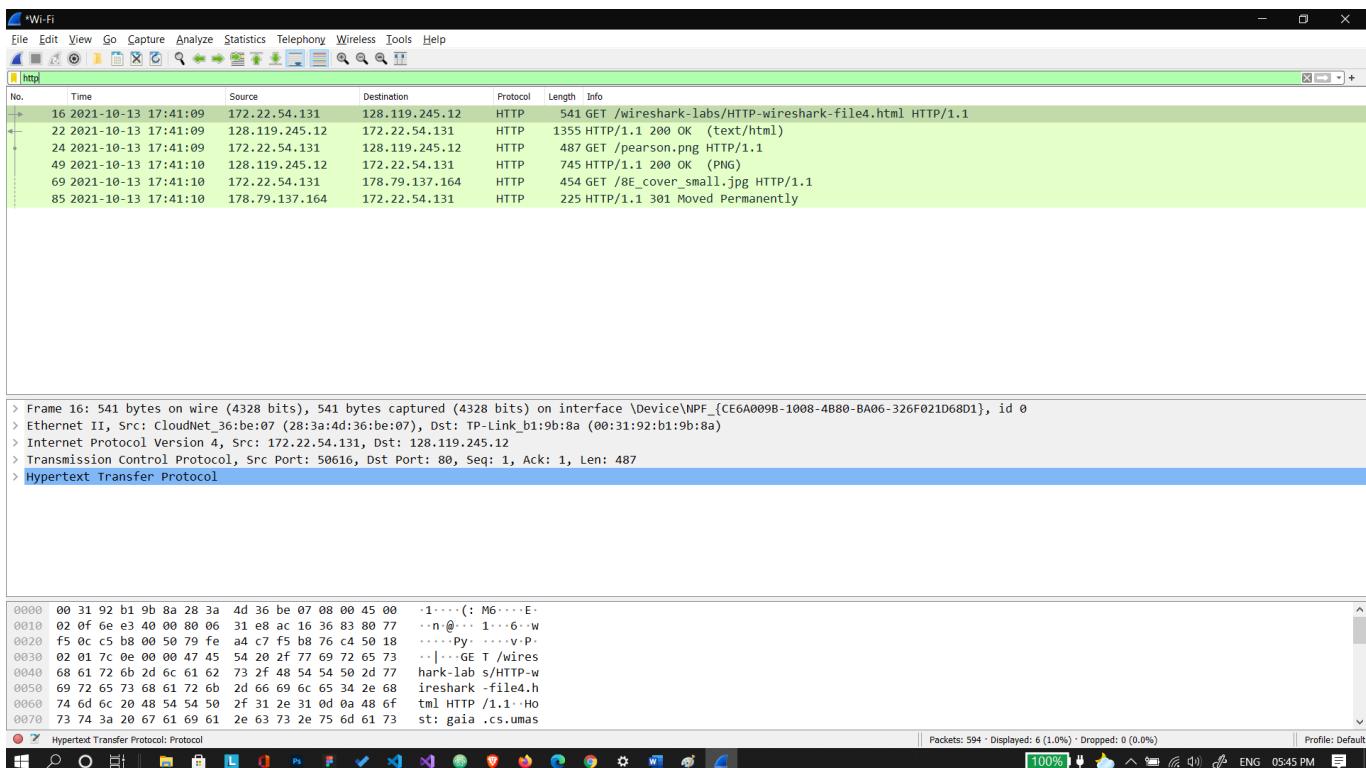
Step 5: Take a screenshot.



This little HTML file is being served by gaia.cs.umass.edu. It contains two embedded images. The image above, also served from the gaia.cs.umass.edu web site, is the logo of our publisher, Pearson. The image of our 8th edition book cover below is stored at, and served from, a WWW server kurose.csslash.net in France:



And while we have your attention, you might want to take time to check out the available open resources for this book at http://gaia.cs.umass.edu/kurose_ross.



Back to questions:

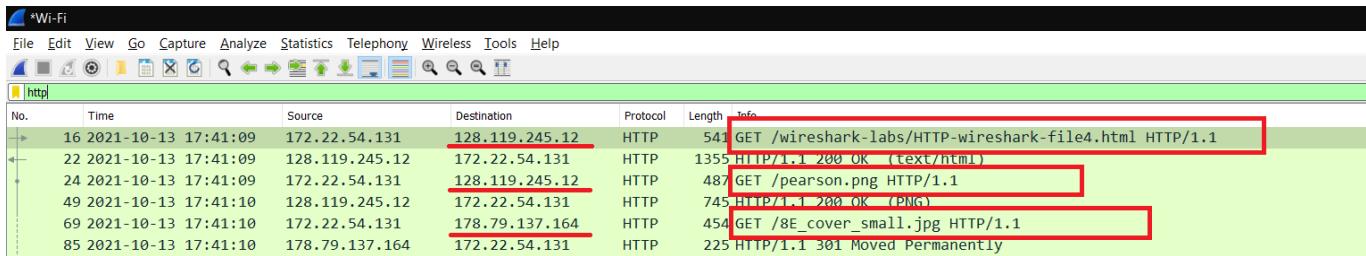
16. How many HTTP GET request messages did your browser send? To which Internet addresses were these GET requests sent?

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



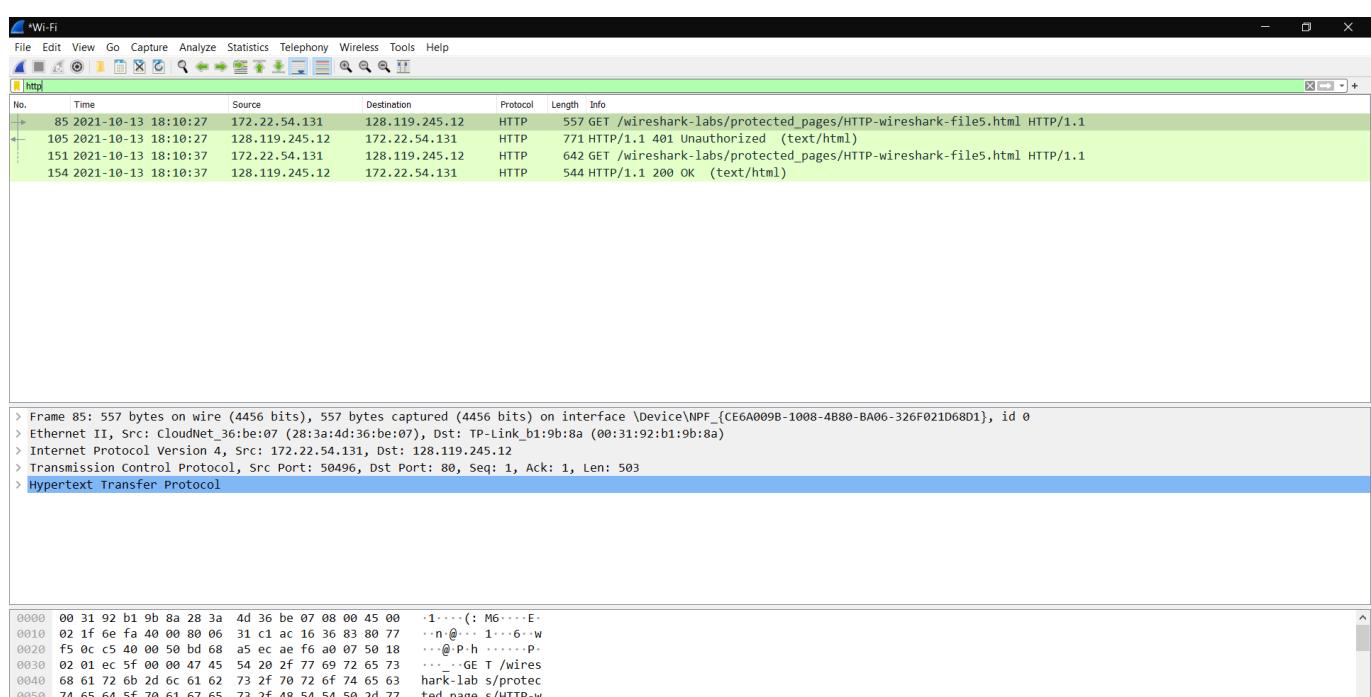
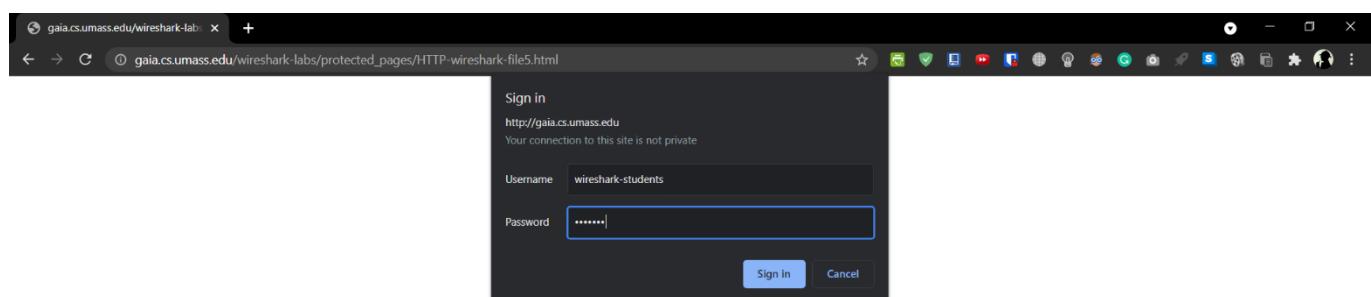
17. Can you tell whether your browser downloaded the two images serially, or whether they were downloaded from the two web sites in parallel? Explain.

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".

No.	Time	Source	Destination	Protocol	Length	Info
85	2021-10-13 18:10:27	172.22.54.131	128.119.245.12	HTTP	557	GET /wireshark-labs/protected_pages/HTTP-wireshark-file5.html HTTP/1.1
105	2021-10-13 18:10:27	128.119.245.12	172.22.54.131	HTTP	771	HTTP/1.1 401 Unauthorized (text/html)
151	2021-10-13 18:10:37	172.22.54.131	128.119.245.12	HTTP	642	GET /wireshark-labs/protected_pages/HTTP-wireshark-file5.html HTTP/1.1
154	2021-10-13 18:10:37	128.119.245.12	172.22.54.131	HTTP	544	HTTP/1.1 200 OK (text/html)

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.

```

151 2021-10-13 18:10:37 172.22.54.131 128.119.245.12 HTTP 642 GET /wireshark-labs/protected_pages/HTTP-wireshark-file5.html HTTP/1.1
154 2021-10-13 18:10:37 128.119.245.12 172.22.54.131 HTTP 544 HTTP/1.1 200 OK (text/html)

> Frame 151: 642 bytes on wire (5136 bits), 642 bytes captured (5136 bits) on interface \Device\NPF_{CE6A009B-1008-4B80-BA06-326F021D68D1}, id 0
> Ethernet II, Src: CloudNet_36:be:07 (28:3a:4d:36:be:07), Dst: TP-Link_b1:9b:8a (00:31:92:b1:9b:8a)
> Internet Protocol Version 4, Src: 172.22.54.131, Dst: 128.119.245.12
> Transmission Control Protocol, Src Port: 52631, Dst Port: 80, Seq: 1, Ack: 1, Len: 588
Hypertext Transfer Protocol
> GET /wireshark-labs/protected_pages/HTTP-wireshark-file5.html HTTP/1.1\r\n
Host: gaia.cs.umass.edu\r\n
Connection: keep-alive\r\n
Cache-Control: max-age=0\r\n
Authorization: Basic d2lyZXNoYXJrLXN0dWRlbmRz0m5ldHdvcms=\r\n
    Credentials: wireshark-students:network
Upgrade-Insecure-Requests: 1\r\n
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/94.0.4606.81 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-GB;q=0.9,en;q=0.8\r\n
\r\n
[Full request URI: http://gaia.cs.umass.edu/wireshark-labs/protected_pages/HTTP-wireshark-file5.html]
[HTTP request 1/1]
[Response in frame: 154]
```

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.



If you have followed the instructions for the TCP Wireshark Lab, you have already downloaded an ASCII copy of Alice and Wonderland from <http://gaia.cs.umass.edu/wireshark-labs/alice.txt> and you also already have the Wireshark packet sniffer running and capturing packets on your computer.

Click on the Browse button below to select the directory/file name for the copy of alice.txt that is stored on your computer.

alice.txt

Once you have selected the file, click on the "Upload alice.txt file" button below. This will cause your browser to send a copy of alice.txt over an HTTP connection (using TCP) to the web server at gaia.cs.umass.edu. After clicking on the button, wait until a short message is displayed indicating the the upload is complete. Then stop your Wireshark packet sniffer - you're ready to begin analyzing the TCP transfer of alice.txt from your computer to gaia.cs.umass.edu!



Congratulations!

You've now transferred a copy of alice.txt from your computer to gaia.cs.umass.edu. You should now stop Wireshark packet capture. It's time to start analyzing the captured Wireshark packets!

No.	Time	Source	Destination	Protocol	Length	Info
193	2021-10-13 19:21:04	172.22.54.131	128.119.245.12	TCP	1514	62791 → 80 [ACK] Seq=148179 Ack=1 Win=131328 Len=1460 [TCP segment of a reassembled PDU]
194	2021-10-13 19:21:04	172.22.54.131	128.119.245.12	TCP	1514	62791 → 80 [ACK] Seq=149639 Ack=1 Win=131328 Len=1460 [TCP segment of a reassembled PDU]
195	2021-10-13 19:21:04	172.22.54.131	128.119.245.12	TCP	1514	62791 → 80 [ACK] Seq=151099 Ack=1 Win=131328 Len=1460 [TCP segment of a reassembled PDU]
196	2021-10-13 19:21:04	172.22.54.131	128.119.245.12	HTTP	539	POST /wireshark-labs/lab3-1-reply.htm HTTP/1.1 (text/plain)
197	2021-10-13 19:21:04	142.250.192.234	172.22.54.131	UDP	70	443 → 64478 Len=28
198	2021-10-13 19:21:04	52.114.132.178	172.22.54.131	TCP	66	[TCP Dup ACK 125#1] 443 → 49306 [ACK] Seq=337 Ack=178 Win=2050 Len=0 SRE=178
199	2021-10-13 19:21:04	128.119.245.12	172.22.54.131	TCP	60	80 → 62791 [ACK] Seq=1 Ack=98543 Win=183296 Len=0
200	2021-10-13 19:21:04	128.119.245.12	172.22.54.131	TCP	60	80 → 62791 [ACK] Seq=1 Ack=100003 Win=182528 Len=0
201	2021-10-13 19:21:04	128.119.245.12	172.22.54.131	TCP	60	80 → 62791 [ACK] Seq=1 Ack=101463 Win=181632 Len=0
202	2021-10-13 19:21:04	128.119.245.12	172.22.54.131	TCP	60	80 → 62791 [ACK] Seq=1 Ack=104383 Win=189056 Len=0
203	2021-10-13 19:21:04	128.119.245.12	172.22.54.131	TCP	60	80 → 62791 [ACK] Seq=1 Ack=107303 Win=194944 Len=0
204	2021-10-13 19:21:04	128.119.245.12	172.22.54.131	TCP	60	80 → 62791 [ACK] Seq=1 Ack=108763 Win=197888 Len=0
205	2021-10-13 19:21:04	128.119.245.12	172.22.54.131	TCP	60	80 → 62791 [ACK] Seq=1 Ack=110223 Win=200704 Len=0

> Frame 196: 539 bytes on wire (4312 bits), 539 bytes captured (4312 bits) on interface '\Device\NPF_{CE6A009B-1008-4B80-BA06-326F021D68D1}', id 0
> Ethernet II, Src: CloudNet_36:be:07 (28:3a:4d:36:be:07), Dst: TP-Link_b1:9b:8a (00:31:92:b1:9b:8a)
> Internet Protocol Version 4, Src: 172.22.54.131, Dst: 128.119.245.12
> Transmission Control Protocol, Src Port: 62791, Dst Port: 80, Seq: 152559, Ack: 1, Len: 485
> [108] Reassembled TCP Segments (153043 bytes): #29(722), #30(1460), #31(1460), #33(1460), #34(1460), #35(1460), #36(1460), #37(1460), #38(1460), #53(1460), #54(1460), #55(1460), #
> Hypertext Transfer Protocol
> MIME Multipart Media Encapsulation, Type: multipart/form-data, Boundary: "----WebKitFormBoundaryHwS9k9Bcc7POZos"

0000 00 31 92 b1 9b 8a 28 3a 4d 36 be 07 08 00 45 00 ·1...(: M6...E·
0010 02 0d 6f b0 a0 00 80 06 31 1d ac 16 36 83 80 77 ·o@... 1...6...w
0020 f5 0c f5 47 00 50 3e ff 5a 54 ee af b8 87 50 18 ...G P... ZT...P·
0030 02 01 85 74 00 00 72 20 79 65 61 72 73 2c 20 74 ...t...r years, t
0040 68 65 20 73 69 6d 70 6c 65 20 61 6e 64 0d 0a 6c he simpl...e and...l
0050 6f 76 69 6e 67 20 68 65 61 72 74 20 6f 66 20 68 owing he art of h
0060 65 72 20 63 68 69 6c 64 68 6f 64 3a 20 20 61 er child hood: a
Frame (539 bytes) -> Reassembled TCP (153043 bytes)
wireshark_Wi-Fi03A1.cappng

Part 2: A first look at the captured trace

- 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.

Frame 196: 539 bytes on wire (4312 bits), 539 bytes captured (4312 bits) on interface \Device\NPF_{CE6A009B-1008-4B80-BA06-326F021D68D1}, id 0
 > Ethernet II, Src: CloudNet_36:be:07 (28:3a:4d:36:be:07), Dst: TP-Link_b1:9b:8a (00:31:92:b1:9b:8a)
 > Internet Protocol Version 4, Src: 172.22.54.131, Dst: 128.119.245.12
 ✓ Transmission Control Protocol, Src Port: 62791, Dst Port: 80, Seq: 152559, Ack: 1, Len: 485
 Source Port: 62791
 Destination Port: 80
 [Stream index: 1]

- 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.

Frame 196: 539 bytes on wire (4312 bits), 539 bytes captured (4312 bits) on interface \Device\NPF_{CE6A009B-1008-4B80-BA06-326F021D68D1}, id 0
 > Ethernet II, Src: CloudNet_36:be:07 (28:3a:4d:36:be:07), Dst: TP-Link_b1:9b:8a (00:31:92:b1:9b:8a)
 > Internet Protocol Version 4, Src: 172.22.54.131, Dst: 128.119.245.12
 ✓ Transmission Control Protocol, Src Port: 62791, Dst Port: 80, Seq: 152559, Ack: 1, Len: 485
 Source Port: 62791
 Destination Port: 80
 [Stream index: 1]

- 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.

Frame 229: 831 bytes on wire (6648 bits), 831 bytes captured (6648 bits) on interface \Device\NPF_{CE6A009B-1008-4B80-BA06-326F021D68D1}, id 0
 > Ethernet II, Src: TP-Link_b1:9b:8a (00:31:92:b1:9b:8a), Dst: CloudNet_36:be:07 (28:3a:4d:36:be:07)
 > Internet Protocol Version 4, Src: 128.119.245.12, Dst: 172.22.54.131
 ✓ Transmission Control Protocol, Src Port: 80, Dst Port: 62791, Seq: 1, Ack: 153044, Len: 777
 Source Port: 80
 Destination Port: 62791
 [Stream index: 1]
 [TCP Segment Len: 777]

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.

Wireshark Screenshot showing network traffic. A red box highlights the first TCP SYN segment at index 7 with sequence number 0. Another red box highlights the TCP header details for this segment, including Source Port: 62791, Destination Port: 80, and Sequence Number: 0.

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.

Wireshark Screenshot showing network traffic. A red box highlights the TCP SYNACK segment at index 25 with sequence number 1 and acknowledgement number 1. Another red box highlights the TCP header details for this segment, including Source Port: 80, Destination Port: 62791, and Sequence Number: 1.

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.

286 2021-10-13 19:21:08 172.22.54.218 224.0.0.251 IGMPV2 46 Membership_Report_group_224.0.0.251	196 2021-10-13 19:21:04 172.22.54.131 128.119.245.12 HTTP 539 POST /wireshark-labs/lab3-1-reply.htm HTTP/1.1 (text/plain)	14 2021-10-13 19:21:02 142.250.192.142 172.22.54.131 QUIC 1392 Protected Payload (KPO)
▼ Transmission Control Protocol, Src Port: 62791, Dst Port: 80, Seq: 152559, Ack: 1, Len: 485		
Source Port: 62791		
Destination Port: 80		
[Stream index: 1]		
[TCP Segment Len: 485]		
Sequence Number: 152559 (relative sequence number)		
Sequence Number (raw): 1050922196		
[Next Sequence Number: 153044 (relative sequence number)]		
Acknowledgment Number: 1 (relative ack number)		
Acknowledgment number (raw): 4004493447		

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	Odd601f	0.359375	0.703125	0.34375	0.34375
2	777	Odd6042	0.359375	0.734375	0.375	0.3475
3	2203	Odd609d	0.703125	1.062500	0.359375	0.3489
4	3629	Odd60f9	0.703125	1.093750	0.390625	0.3541
5	5055	Odd60f9	0.734375	1.109375	0.375	0.3567
6	6481	Odd61af	0.734375	1.140625	0.40625	0.3628

$$\text{EstimatedRTT} = 0.875 * \text{EstimatedRTT} + 0.125 * \text{SampleRTT}$$

$$\text{EstimatedRTT of Segment 1} = 0.34375$$

$$\text{EstimatedRTT of Segment 2} = 0.875 * 0.34375 + 0.125 * 0.375 = 0.3475$$

$$\text{EstimatedRTT of Segment 3} = 0.875 * 0.3475 + 0.125 * 0.359375 = 0.3489$$

$$\text{EstimatedRTT of Segment 4} = 0.875 * 0.3489 + 0.125 * 0.390625 = 0.3541$$

$$\text{EstimatedRTT of Segment 5} = 0.875 * 0.3541 + 0.125 * 0.375 = 0.3567$$

$$\text{EstimatedRTT of Segment 6} = 0.875 * 0.3567 + 0.125 * 0.40625 = 0.3628$$

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.

tcp segment of a reassembled pdu						
No.	Time	Source	Destination	Protocol	Length	Info
281	2021-10-13 19:21:07	172.22.54.131	52.231.207.240	TCP	55	56903 → 443 [ACK] Seq=1 Ack=1 Win=517 Len=1 [TCP segment of a reassembled PDU]
307	2021-10-13 19:21:09	172.22.54.131	13.224.22.202	TCP	55	58070 → 443 [ACK] Seq=1 Ack=1 Win=512 Len=1 [TCP segment of a reassembled PDU]
288	2021-10-13 19:21:08	172.22.54.131	52.231.207.240	TCP	55	59598 → 443 [ACK] Seq=1 Ack=1 Win=512 Len=1 [TCP segment of a reassembled PDU]
26	2021-10-13 19:21:02	172.22.54.131	128.119.245.12	TCP	54	62791 → 80 [ACK] Seq=1 Ack=1 Win=131328 Len=0
144	2021-10-13 19:21:04	172.22.54.131	128.119.245.12	TCP	1514	62791 → 80 [ACK] Seq=10003 Ack=1 Win=131328 Len=1460 [TCP segment of a reassembled PDU]
145	2021-10-13 19:21:04	172.22.54.131	128.119.245.12	TCP	1514	62791 → 80 [ACK] Seq=101463 Ack=1 Win=131328 Len=1460 [TCP segment of a reassembled PDU]
146	2021-10-13 19:21:04	172.22.54.131	128.119.245.12	TCP	1514	62791 → 80 [ACK] Seq=102923 Ack=1 Win=131328 Len=1460 [TCP segment of a reassembled PDU]
147	2021-10-13 19:21:04	172.22.54.131	128.119.245.12	TCP	1514	62791 → 80 [ACK] Seq=104383 Ack=1 Win=131328 Len=1460 [TCP segment of a reassembled PDU]
148	2021-10-13 19:21:04	172.22.54.131	128.119.245.12	TCP	1514	62791 → 80 [ACK] Seq=105843 Ack=1 Win=131328 Len=1460 [TCP segment of a reassembled PDU]
149	2021-10-13 19:21:04	172.22.54.131	128.119.245.12	TCP	1514	62791 → 80 [ACK] Seq=107303 Ack=1 Win=131328 Len=1460 [TCP segment of a reassembled PDU]
150	2021-10-13 19:21:04	172.22.54.131	128.119.245.12	TCP	1514	62791 → 80 [ACK] Seq=108763 Ack=1 Win=131328 Len=1460 [TCP segment of a reassembled PDU]
37	2021-10-13 19:21:02	172.22.54.131	128.119.245.12	TCP	1514	62791 → 80 [ACK] Seq=10943 Ack=1 Win=131328 Len=1460 [TCP segment of a reassembled PDU]
151	2021-10-13 19:21:04	172.22.54.131	128.119.245.12	TCP	1514	62791 → 80 [ACK] Seq=110223 Ack=1 Win=131328 Len=1460 [TCP segment of a reassembled PDU]
152	2021-10-13 19:21:04	172.22.54.131	128.119.245.12	TCP	1514	62791 → 80 [ACK] Seq=111683 Ack=1 Win=131328 Len=1460 [TCP segment of a reassembled PDU]

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.

```
> Flags: 0x010 (ACK)
Window: 512
[Calculated window size: 131328]
[WINDOW SIZE SCALING FACTOR: 256]
Checksum: 0x53b0 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
```

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).

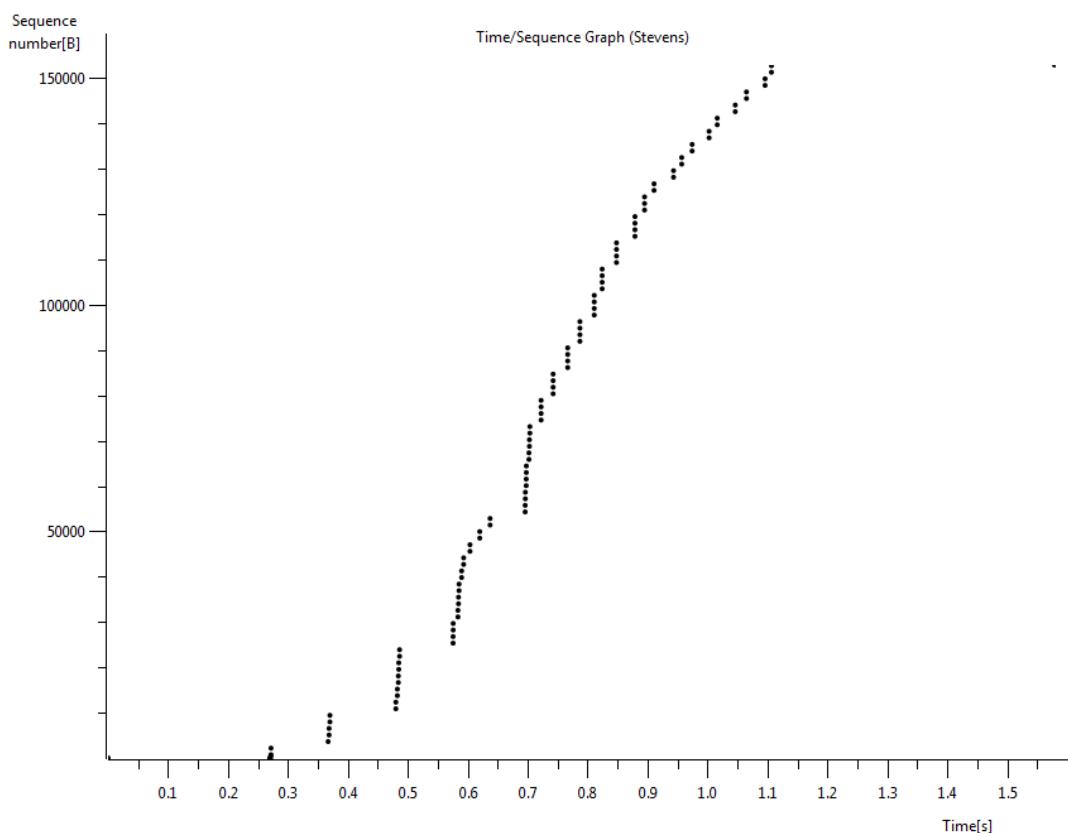
12. What is the throughput (bytes transferred per unit time) for the TCP connection? Explain how you calculated this value.

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.