Natalie Petrosian CSE 150 Computer Networking Professor Chen Qian 28 October 2020

Lab 2: HTTP, DNS, and TCP

#### **Prelab Questions**

https://www.ietf.org/rfc/rfc2616.txt https://www.ietf.org/rfc/rfc1035.txt

# **HTTP Questions**

#### 1. Choose 5 HTTP status codes and describe each one.

200 OK: the request succeeded, and the requested object is later in the message

301 Moved Permanently: the request object is moved, and the new location is specified later in the message

400 Bad Request: the request message is not understood by the server

404 Not Found: the requested document is not found on this server

505 HTTP Version Not Supported

## 2. List the 8 HTTP 1.1 methods and explain what they do.

OPTIONS: allows the client to determine the options/requirements of a resource or the capabilities of a server without initiating resource retrieval.

GET: retrieve the information by the request.

HEAD: the same as GET except the server does not return the body of the message.

POST: asks the origin server to accept the data in the request

PUT: asks that the enclosed data be stored under the requested URI

DELETE: asks the origin server to delete the specified resource

TRACE: echos back input to the user

CONNECT: starts a two-way communication with a requested resource

3. Use wget on example.com to view the last modified date of the webpage. What was the HTTP return status given and what command was used to do this?

The HTTP return status is 200 OK. The GET command was used to do this.

4. Look up the telnet command. Use telnet to connect to towel.blinkenlights.nl. What does this telnet server do?

This telnet server plays a text and character-based Star Wars animation.

## **DNS Questions**

5. In your own words describe what a DNS resource record (RR) is. Now using the command line tool nslookup find the MX resource record of ucsc.edu. What does this resource record mean?

A DNS resource record (RR) refers to the unit of information in DNS zone files, and they are utilized to solve DNS queries. MX stands for mail exchange, and it directs email

messages to a mail server. The MX record shows how the emails should be routed with SMTP.

# 6. What does the command nslookup -type=ns . do? Explain its output.

The command nslookup means "name server look up", and is used to locate the IP address that corresponds to a host or domain name which corresponds to an IP address. The parameter -type=ns . denotes the query type to be name server.

```
mininet@mininet-vm:~$ nslookup -type=ns .
                192.168.1.1
Server:
Address:
                192.168.1.1#53
Non-authoritative answer:
        nameserver = a.root-servers.net.
        nameserver = b.root-servers.net.
        nameserver = c.root-servers.net.
        nameserver = d.root-servers.net.
        nameserver = e.root-servers.net.
        nameserver = f.root-servers.net.
        nameserver = g.root-servers.net.
        nameserver = h.root-servers.net.
        nameserver = i.root-servers.net.
        nameserver = j.root-servers.net.
        nameserver = k.root-servers.net.
        nameserver = l.root-servers.net.
        nameserver = m.root-servers.net.
Authoritative answers can be found from:
```

#### **TCP Questions**

7. How can multiple application services running on a single machine with a single IP address be uniquely identified?

Multiple application services running on a single machine with a single IP address can be uniquely identified by their port number.

8. What is the purpose of the window mechanism in TCP?

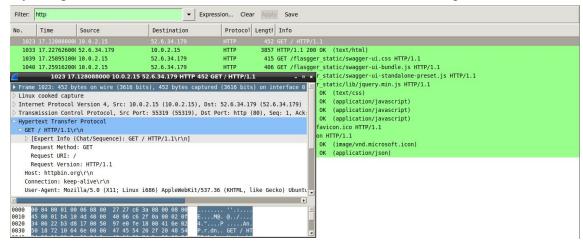
The purpose of the window mechanism in TCP is to control the flow of data packets between two computers or networks, and requires an acknowledgement that the data has been received.

9. What is an MTU? What happens when a packet is larger than the MTU?

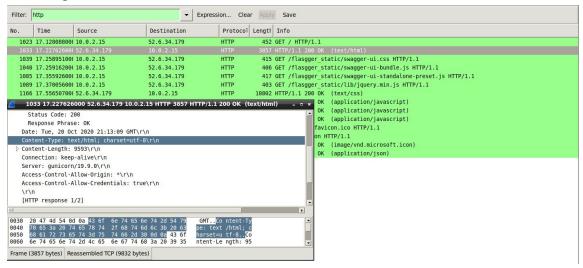
A maximum transmission unit (MTU) is the size of the largest data unit communicated in a single network layer transaction. When a packet is larger than the MTU, the packets will be divided into smaller units and then reassembled to their original size after being received.

Lab Questions
Part 1: HTTP

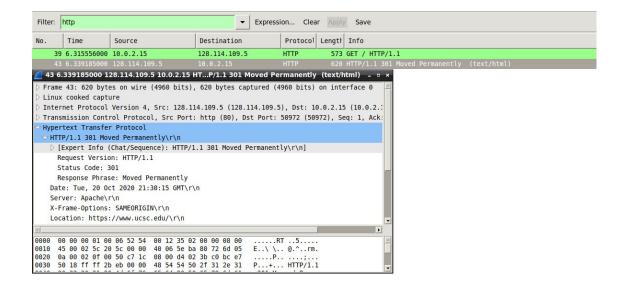
1. My computer used the HTTP method "GET" in order to make this request.



2. The server returned a status code of 200 OK. The content type of the response the server is sending back is "text/html".



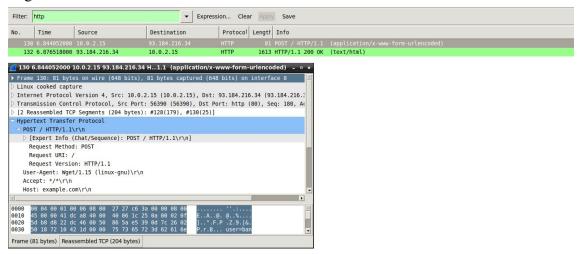
3. The difference here is that we are getting a 301 Moved Permanently status code instead of a 200 OK. The packet indicates the new location of the URL as well. Chromium also redirects you to the correct URL, which is <a href="https://www.ucsc.edu">https://www.ucsc.edu</a>.



4. I generated an HTTP POST method by issuing the following command from the Mininet VM:

wget --post-data 'user=banana&password=slug' <a href="http://example.com/">http://example.com/</a>

I initiated a request to post data on example.com with a username banana and password slug.



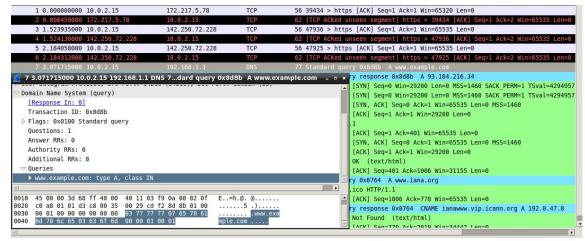
NOTE: I used StackOverflow to learn how to generate a POST command with wget.

#### Part 2: DNS

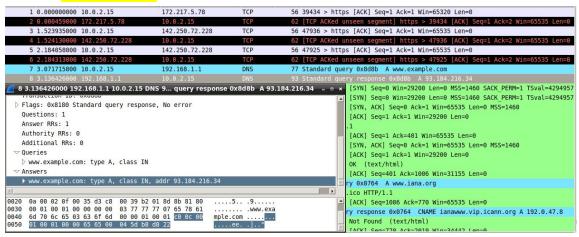
5. Yes, there were steps taken by my computer before the web page was loaded. The DNS server was queried for the IP address of <a href="www.example.com">www.example.com</a>. The query was responded to by a packet that contained the IP address of <a href="93.184.216.34">93.184.216.34</a>. Finally, an HTTP packet with

a GET method was sent to that IP address, and a 200 OK was returned from example.com.

Screenshot of the DNS request for the IP address for example.com.



Screenshot of the response from the DNS server with the IP address of example.com, which is **93.184.216.34.** 



6. We use the command "wget 93.184.216.34 --header 'Host: <a href="www.example.com">www.example.com</a>" in order to download the same content of <a href="www.example.com">www.example.com</a> with its IP address without sending DNS requests.

1 0.000000000 10.0.2.15	93.184.216.34	TCP	76 52618 > http [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=
2 0.022175000 93.184.216.34	10.0.2.15	TCP	62 http > 52618 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
3 0.022243000 10.0.2.15	93.184.216.34	TCP	56 52618 > http [ACK] Seq=1 Ack=1 Win=29200 Len=0
4 0.022585000 10.0.2.15	93.184.216.34	HTTP	169 GET / HTTP/1.1
5 0.022853000 93.184.216.34	10.0.2.15	TCP	62 http > 52618 [ACK] Seq=1 Ack=114 Win=65535 Len=0
6 0.045020000 93.184.216.34	10.0.2.15	HTTP	1663 HTTP/1.1 200 OK (text/html)
7 0.045040000 10.0.2.15	93.184.216.34	TCP	56 52618 > http [ACK] Seq=114 Ack=1608 Win=31240 Len=0
8 0.046694000 10.0.2.15	93.184.216.34	TCP	56 52618 > http [FIN, ACK] Seq=114 Ack=1608 Win=31240 Len=0
9 0.046930000 93.184.216.34	10.0.2.15	TCP	62 http > 52618 [ACK] Seq=1608 Ack=115 Win=65535 Len=0
10 0.073179000 93.184.216.34	10.0.2.15	TCP	62 http > 52618 [FIN, ACK] Seq=1608 Ack=115 Win=65535 Len=0
11 0.073210000 10.0.2.15	93.184.216.34	TCP	56 52618 > http [ACK] Seq=115 Ack=1609 Win=31240 Len=0

The screenshot does not have any DNS packets, as expected. This is because we have utilized the IP address of the site, and therefore it does not go through the DNS.

7. The following is a screenshot with packets corresponding to 'nslookup -type A www.google.com'

```
1 0.000000000 10.0.2.15 192.168.1.1 DNS 76 Standard query 0x9255 A www.google.com
2 0.015846000 192.168.1.1 10.0.2.15 DNS 92 Standard query response 0x9255 A 142.250.72.228
```

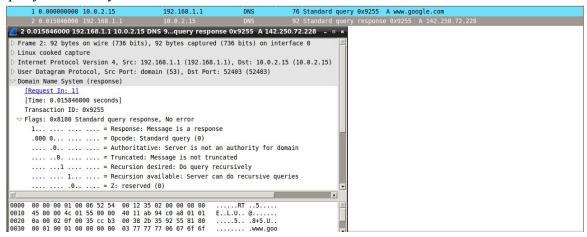
The following is a screenshot indicating the response from the server:

```
mininet@mininet-vm:~$ nslookup -type=A www.google.com
Server: 192.168.1.1
Address: 192.168.1.1#53

Non-authoritative answer:
Name: www.google.com
Address: 142.250.72.228
```

Since the request was resolved, the IP address I was given for <u>www.google.com</u> is **142.250.72.228.** 

8. My computer wanted to complete the request recursively. From the packet content, we can open the "Flags" option and navigate to the section that says "Recursion desired: Do query recursively".



9. Here is a screenshot with the packets corresponding to "nslookup -type=A ucsc.edu", followed by another screenshot indicating the response from the server.

```
1 0.000000000 10.0.2.15 192.168.1.1 DNS 70 Standard query 0x8415 A ucsc.edu
2 0.023337000 192.168.1.1 10.0.2.15 DNS 86 Standard query response 0x8415 A 128.114.109.5

mininet@mininet-vm:~$ nslookup -type=A ucsc.edu

Server: 192.168.1.1
Address: 192.168.1.1#53

Non-authoritative answer:
Name: ucsc.edu
Address: 128.114.109.5
```

The request was resolved, and the IP address I was given for ucsc.edu is 128.114.109.5.

10. In order to find the authoritative name server for the ucsc.edu domain, we must add the parameter "type=soa". I learned of this parameter from StackOverflow. Thus, our

command becomes "nslookup type=soa ucsc.edu". From the screenshot below, the authoritative name server for the ucsc.edu domain is **adns1.ucsc.edu**.

```
mininet@mininet-vm:~$ nslookup -type=soa ucsc.edu

Server: 192.168.1.1

Address: 192.168.1.1#53

Non-authoritative answer:
ucsc.edu
    origin = adnsl.ucsc.edu
    mail addr = hostmaster.ucsc.edu
    serial = 20647867
    refresh = 10800
    retry = 900
    expire = 2419200
    minimum = 900

Authoritative answers can be found from:
```

### Part 3: TCP

The following is a screenshot of the corresponding packets with the SYN, SYN-ACK, and ACK.

```
17 13.19269000(10.0.2.15 80.249.99.148 TCP 76 46112 > http [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSVal=4294942 18 13.33800100(80.249.99.148 10.0.2.15 TCP 62 http > 46112 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 19 13.33803000(10.0.2.15 80.249.99.148 TCP 56 46112 > http [ACK] Seq=1 Ack=1 Win=29200 Len=0
```

The initial window size that my computer advertised to the server is **29200**.

```
> Frame 17: 76 bytes on wire (608 bits), 76 bytes captured (608 bits) on interface 0
Delinux cooked capture
▽ Transmission Control Protocol, Src Port: 46112 (46112), Dst Port: http (80), Seq: 0, Len: 0
  Source port: 46112 (46112)
  Destination port: http (80)
  [Stream index: 0]
  Sequence number: 0
                   (relative sequence number)
  Header length: 40 bytes
 Flags: 0x002 (SYN)
  Window size value: 29200
  [Calculated window size: 29200]

    ○ Checksum: 0xc0ca [validation disabled]

 Doptions: (20 bytes), Maximum segment size, SACK permitted, Timestamps, No-Operation (NOP)
```

The initial window size that the server advertised is 65535.

12. The following screenshot indicates a packet from the download with a source of the server and a destination of my computer. The source port number is 80 and the destination port is 46112.

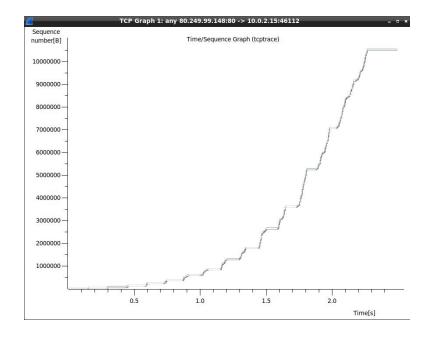
```
22 13.488312000 80.249.99.148 10.0.2.15 T...896 [TCP segment of a reassembled PDU]
Frame 22: 2896 bytes on wire (23168 bits), 2896 bytes captured (23168 bits) on interface
Linux cooked capture

    □ Internet Protocol Version 4, Src: 80.249.99.148 (80.249.99.148), Dst: 10.0.2.15 (10.0.2.1)

¬ Transmission Control Protocol, Src Port: http (80), Dst Port: 46112 (46112), Seq: 1, Ack:

   Source port: http (80)
   Destination port: 46112 (46112)
   [Stream index: 0]
   Sequence number: 1 (relative sequence number)
   [Next sequence number: 2841 (relative sequence number)]
   Acknowledgment number: 139 (relative ack number)
   Header length: 20 bytes
 Flags: 0x010 (ACK)
   Window size value: 65535
    [Calculated window size: 65535]
    [Window size scaling factor: -2 (no window scaling used)]
  Checksum: Oxcbce [validation disabled]
   FOED (1.0) 1 1 1
```

The following screenshot captures the tcptrace graph with the selected packet. The x-axis represents time and the y-axis represents the sequence numbers. The graph indicates the progression of sequences as time went by during the download. It took approximately 2.4 seconds to download the entire file.



13. When we start the download of the file, loss is set to 0%. This means the packet transmission is successful. At approximately 3 seconds, the graph plateaus for a 10-second duration. This occurs at the moment we set the loss to be 100%, and the download progress is halted. After a brief pause, we change the loss back to 0%, and the graph sharply trends upward. The download of the file took approximately 15 seconds in this scenario.

0% loss is shown between the 2.5 second and 13 mark, as indicated by the plateau. TCP is slow-start between 0 and 2.5 seconds.

TCP is in congestion avoidance from 13-14 seconds, as it is recovering from the 0% loss period

