

Prelab 2

1. 5 HTTP status codes:
 - a. 407 – proxy authentication required
 - b. 408 – request time-out
 - c. 500 – internal server error
 - d. 502 – bad gateway
 - e. 504 – gateway time-out
 2. 8 HTTP 1.1 methods:
 - a. GET – used to retrieve information from the given server using a given URI.
 - b. HEAD – same as GET but transfers the status line and header section only.
 - c. POST – used to send data to the server.
 - d. PUT – replaces all current representations of the target resource with the uploaded content.
 - e. DELETE – removes all current representations of the target resource given by a URI.
 - f. CONNECT – establishes a tunnel to the server identified by a given URI.
 - g. OPTIONS – describes the communication options for the target resource.
 - h. TRACE – performs a message loop-back test along the path to the target resource.
 3. The HTTP status returned for *example.com* was a 200 code, meaning OK. The command used to do this was GET, which retrieves the information from the given server.
 4. After using *telnet* to connect to the given server, it started playing Star Wars in ASCII, scene by scene.
 5. DNS resource records are the basic building blocks of host-name and IP information, they provide all the useful data for a given DNS server. The MX resource record is a mail exchanger record.
 6. The command returns a list of non-authoritative name servers along with authoritative ones. It's returning a list of authoritative servers it believes to be.
-
1. This can be achieved through application layer protocol, which will allow for communication without having to worry about the details of how the communication is accomplished.
 2. The purpose of the window mechanism in TCP is a method that controls the flow of packets between two computers or network hosts. It will send one or more data segments and the receiver will acknowledge one or all segments.

3. MTU stands for maximum transmission unit. If a packet is larger than the MTU it will cause jabbering, which when unaddressed can disrupt a network.
4. The following screenshot is information using the command *wget* on *example.com*. You'll notice that there is an HTTP packet containing a TCP segment, which is piggybacking an ACK with a length of 109.

Wireshark 1.10.6 (v1.10.6 from master-1.10)

Filter: tcp

No.	Time	Source	Destination	Protocol	Length	Info
151	32.999557000	127.0.0.1	127.0.0.1	TCP	56	6633 > 60776 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
156	33.954597000	10.0.2.15	93.184.216.34	TCP	76	41631 > http [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=46489 TSecr=0 WS=128
157	33.970962000	93.184.216.34	10.0.2.15	TCP	62	http > 41631 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
158	33.971045000	10.0.2.15	93.184.216.34	TCP	56	41631 > http [ACK] Seq=1 Ack=1 Win=29200 Len=0
159	33.971279000	10.0.2.15	93.184.216.34	HTTP	165	GET / HTTP/1.1
160	33.972126000	93.184.216.34	10.0.2.15	TCP	62	http > 41631 [ACK] Seq=1 Ack=110 Win=65535 Len=0
161	33.991138000	93.184.216.34	10.0.2.15	TCP	1504	[TCP segment of a reassembled PDU]
162	33.991207000	10.0.2.15	93.184.216.34	TCP	56	41631 > http [ACK] Seq=110 Ack=1449 Win=31240 Len=0
163	33.991309000	93.184.216.34	10.0.2.15	HTTP	200	HTTP/1.1 200 OK (text/html)
164	33.991327000	10.0.2.15	93.184.216.34	TCP	56	41631 > http [ACK] Seq=110 Ack=1593 Win=34080 Len=0
165	33.993523000	10.0.2.15	93.184.216.34	TCP	56	41631 > http [FIN, ACK] Seq=110 Ack=1593 Win=34080 Len=0
166	33.993932000	93.184.216.34	10.0.2.15	TCP	62	http > 41631 [ACK] Seq=1593 Ack=111 Win=65535 Len=0
167	34.000555000	127.0.0.1	127.0.0.1	TCP	76	60778 > 6633 [SYN] Seq=0 Win=43690 Len=0 MSS=65495 SACK_PERM=1 TSval=46500 TSecr=0 WS=128
168	34.000578000	127.0.0.1	127.0.0.1	TCP	56	6633 > 60778 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0

Frame 159: 165 bytes on wire (1320 bits), 165 bytes captured (1320 bits) on interface 0

Linux cooked capture

Internet Protocol Version 4, Src: 10.0.2.15 (10.0.2.15), Dst: 93.184.216.34 (93.184.216.34)

Transmission Control Protocol, Src Port: 41631 (41631), Dst Port: http (80), Seq: 1, Ack: 1, Len: 109

Source port: 41631 (41631)

Destination port: http (80)

[Stream index: 69]

Sequence number: 1 (relative sequence number)

[Next sequence number: 110 (relative sequence number)]

Acknowledgment number: 1 (relative ack number)

Header length: 20 bytes

Flags: 0x018 (PSH, ACK)

Window size value: 29200

[Calculated window size: 29200]

[Window size scaling factor: -2 (no window scaling used)]

Checksum: 0x4271 [validation disabled]

[SEQ/ACK analysis]

Hypertext Transfer Protocol

0000 00 04 00 01 00 06 08 00 27 27 c6 3a 00 00 08 00 ''.....
0010 45 00 00 95 b7 2c 40 00 40 06 41 4d 0a 00 02 0f E.....@.AM....
0020 5d b8 d8 22 a2 9f 00 50 b8 b9 02 b4 00 09 c4 02].....P.....
0030 50 18 72 10 42 71 00 00 47 45 54 20 2f 20 48 54 P..r.Bq.. GET / HT

any: <live capture in progress>... Packets: 314 · Displayed: 296 (94.3%) Profile: Default

mininet@minine... Capturing from a...

18:47