CMPE 207 HW1 Pradeep Patil SID 011483277

Part 1 WEBCLIENT

1 a. Apache2 Server Status


```
pradeep@pradeep-VirtualBox: ~
pradeep@pradeep-VirtualBox:~$ systemctl status apache2
● apache2.service - LSB: Apache2 web server
   Loaded: loaded (/etc/init.d/apache2; bad; vendor preset: enabled)
   Drop-In: /lib/systemd/system/apache2.service.d
            └─apache2-systemd.conf
   Active: active (running) since Wed 2018-09-19 11:47:39 PDT; 9h ago
     Docs: man:systemd-sysv-generator(8)
  Process: 3072 ExecReload=/etc/init.d/apache2 reload (code=exited, status=0/SUCCESS)
  Process: 1031 ExecStart=/etc/init.d/apache2 start (code=exited, status=0/SUCCESS)
   CGroup: /system.slice/apache2.service
           ├─1097 /usr/sbin/apache2 -k start
           ├─3179 /usr/sbin/apache2 -k start
           └─3180 /usr/sbin/apache2 -k start

Sep 19 11:47:36 pradeep-VirtualBox systemd[1]: Starting LSB: Apache2 web server...
Sep 19 11:47:36 pradeep-VirtualBox apache2[1031]:  * Starting Apache httpd web server apache2
Sep 19 11:47:38 pradeep-VirtualBox apache2[1031]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name, usi
Sep 19 11:47:39 pradeep-VirtualBox apache2[1031]:  *
Sep 19 11:47:39 pradeep-VirtualBox systemd[1]: Started LSB: Apache2 web server.
Sep 19 11:52:25 pradeep-VirtualBox systemd[1]: Reloading LSB: Apache2 web server.
Sep 19 11:52:25 pradeep-VirtualBox apache2[3072]:  * Reloading Apache httpd web server apache2
Sep 19 11:52:26 pradeep-VirtualBox apache2[3072]:  *
Sep 19 11:52:26 pradeep-VirtualBox systemd[1]: Reloaded LSB: Apache2 web server.
lines 1-22/22 (END)
```

1 b.   Program Exicution


```
pradeep@pradeep-VirtualBox: ~/Desktop/H1 compare
pradeep@pradeep-VirtualBox:~/Desktop/H1 compare$ ./webclient http://localhost/
CMPE 207 HW1 webclient Bapugouda urf Pradeep Patil 277
HTTP/1.1 200 OK
Date: Wed, 19 Sep 2018 09:30:28 GMT
Server: Apache/2.4.18 (Ubuntu)
Last-Modified: Sun, 16 Sep 2018 22:33:37 GMT
ETag: "2c39-57604a5ca8126"
Accept-Ranges: bytes
Content-Length: 11321
Vary: Accept-Encoding
Connection: close
Content-Type: text/html

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <!--
    Modified from the Debian original for Ubuntu
    Last updated: 2014-03-19
    See: https://launchpad.net/bugs/1288690
  -->
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>Apache2 Ubuntu Default Page: It works</title>
    <style type="text/css" media="screen">
  * {
    margin: 0px 0px 0px 0px;
    padding: 0px 0px 0px 0px;
  }

  body, html {
    padding: 3px 3px 3px 3px;

    background-color: #D8DBE2;

    font-family: Verdana, sans-serif;
    font-size: 11pt;
    text-align: center;
  }
```

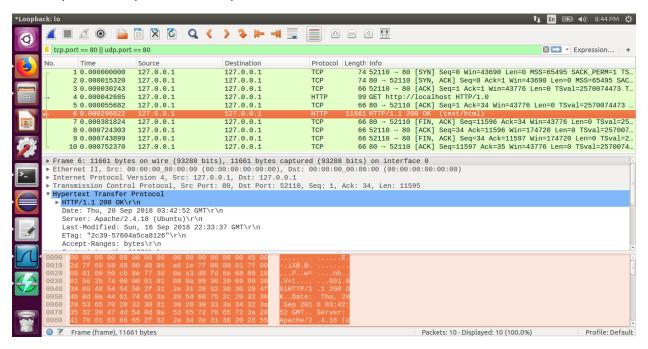# CMPE 207 HW1 Pradeep Patil SID 011483277
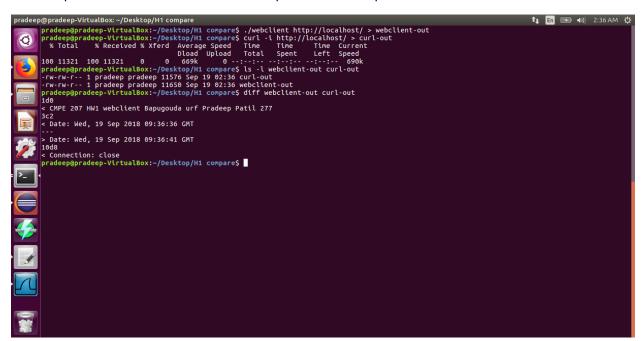
1 c. 1 List of All the Packets

1 c. 2 Capture of Request Packet Expanded

1 c. 2 Capture of Response Packet Expanded

1 d. Comparison between the Webclient output and Curl Output.

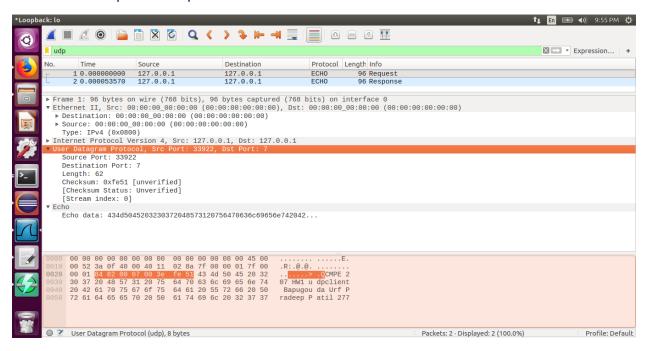# CMPE 207 HW1 Pradeep Patil SID 011483277

Part 2 UDPCLIENT

2 a. XINETD Server Status

2 b. Program Exicution udpclient

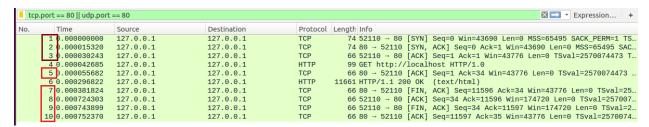# CMPE 207 HW1 Pradeep Patil SID 011483277

2 c. WireShark Capture of Request

2 c. Wireshark Capture of Response
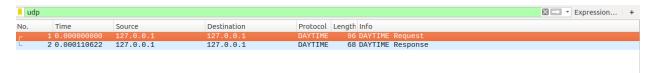
Part 3 Capture Compare

3. (4 pts each, 20 pts)Use your wireshark screenshots for TCP and UDP, and compare the differences of client-server packet exchange between TCP and UDP. In particular, other than the request and response packets, explain the following "additional" packets of TCP (when compared with UDP):
a. Initiate connection
b. Destroy connection
c. After the client sends a request to the server but before server sends the response
d. After the server sends a response to the client

TCP Packets

UDP Packets

a. **Initiate connection**
The first 3 packets are responsible the TCP 3 way handshake to create a connection between the client and server. To establish a connection, TCP uses a three-way handshake. Before a client attempts to connect with a server, the server must first bind to and listen at a port to open it up for connections: this is called a passive open. Once the passive open is established, a client may initiate an active open. Resulting 3 packets 1. [SYN] 2. [SYN,ACK] 3. [ACK]

b. **Destroy connection**
Once the transfer is completed the active connection between client and server is closed and all the associated resources are released. The connection termination phase uses a four-way handshake, with each side of the connection terminating independently. When an endpoint wishes to stop its half of the connection, it transmits a FIN packet, which the other end acknowledges with an ACK. Therefore, a typical tear-down requires a pair of FIN and ACK segments from each TCP endpoint. After the side that sent the first FIN has responded with the final ACK, it waits for a timeout before finally closing the connection, during which time the local port is unavailable for new connections.

c. After the client sends a request to the server but before server sends the response
This packet is an acknowledgement sent from server to client to assert that the request was successfully received. As TCP is connection oriented all the incoming packets are acknowledged by the receiving end.

d. After the server sends a response to the client
After the response is sent we have two cases, I. if all the data requested by the client is sent, the server sends a signal to indicate the requested. II. If data is sent partially the server sends the number of bytes sent as a parameter and then sends the remaining data in parts and continues. All the packets from the server are acknowledged by the client. Once the transfer is complete one of the end points initiate the close of the connection and the connection is closed as explained in b.