**CS 441**

**Program #3**

**Due 4/16 Thursday at 1:15pm online to Blackboard**

**100pts**

**TCP overhead**

TCP provides a higher Quality of Service (QoS) than UDP since TCP provides reliable service while UDP provides only unreliable service. There is a cost to higher QoS which is the additional overhead incurred by TCP. There is both additional load placed on the network and additional delay. In this program we will evaluate the overhead cost of using TCP for file transfers of varying sizes using wireshark

We will use TCP sockets to communicate between two different CSUEB computers as we did in program 2.

**Goal**:

**The goal of the assignment is to find out if overhead increases at a fixed rate when file sizes are increased.**

**We define overhead as packet headers and control packets such as: syns, acks,retransmitted packets or duplicates). You will be able to find these when you import your tcp dump file into wireshark**

Really there is not much new coding for this assignment beyond what you did already for program 2. Particularly here we are looking at the tcpdump capture to evaluate overhead as defined above

We are going to compare **overhead** in terms of extra bytes needed to transfer 4 files of 10, 20, 40, 60KB (where KB = 1000bytes not 1024 for simplicity).

This time the server will send packets of 1000 bytes each.

**Step 1 – You have probably done all this already**

* Download and install Putty or some other ssh program
* Download and install WinScp or other FTP program – See Blackboard for download URL
* Download and install Wireshark at <http://www.wireshark.org/download.html> It is also recommend that you read the man page for Wireshark which is here: <http://www.wireshark.org/docs/wsug_html_chunked/ChWorkDisplayPopUpSection.html#ChWorkPacketListPanePopUpMenu>

**Step 2 – you probably also have done this for program 2, but I have reposted the program we used in program 2 for you.**

* Download, compile, and run the TCP programs provided – see Programs/Program3 on Blackboard

**Coding for server**

* Modify the server port number to a value that is greater than 50000 and less than 50500
* Change the **buf** **array** to a size **1000**
* Set the send **buflen** to **1000** (the value you use in the send() or sendto() call.
* When the client connects, the server will open one of the four files provided and send it in the buffer size chunks (1000 bytes).
* Compile your code

**Coding for client**

* Change **buf array** to **1000** chars
* Set the receive buffer size to the this value (the value you use in the recv() or recvfrom() call.)
* After the client connects to the server, the server should then start sending the file requested.
* The client should print the packets that arrived to the terminal screen
* Compile your code

**Running the programs:**

* 1. You will need to open three putty/terminal windows. In one window you will run the server, the second you will run the client, and the third you will run tcpdump (note tcpdump should be running on the same machine that your client code is running on.
  2. When you run the programs, **FIRST start the Server program.**
  3. Note the port number the server has.
  4. **SECOND** **Start up the tcpdump program** Type tcpdump ‘port port#’ –w capture.pcap for example:

**tcpdump ‘port 50100’** **-w capture.pcap** *note* *quotes are needed*

* 1. **THIRD** start the **client** program up – see program comments for how to start it if your forgot
  2. End TCP dump with **control C**
  3. You can then use **winscp or SFTP** to transfer the **capture.pcap** file to your own computer.
  4. Start up the **Wireshark** program and choose **File/Open** and open your **capture.pcap** file

**Reading the Wireshark capture:**

1. Find TCP setup packets for your session (SYN), right click on the packet and choose Follow TCP stream.
2. You may also choose to use filter (type in the filter box tcp.port == xxx where xxx is the port number your programs are using
3. Select from main menu, **Statistics-> summary**.  Here you can find **total bytes**and**total packets** of displayed packets
4. The summary packet lengths include the data link layer header, and the measured elapsed time consists of protocol processing delay, queuing delay, transmission delay, and propagation delay.
5. Using your capture, **calculate the number of bytes of overhead for each file transfer** – use this to create your table and graph for results. Overhead is headers, and any other control packets syn ack, fin, resends
6. Using your capture, calculate the number of bytes of overhead. Do the above 4 times for each different sized file. Use your data to create your table and graph for results

**Table: complete the table below and the questions**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 10KB file | 20KB file | 40KB file | 60KB |
| total of all bytes sent |  |  |  |  |
| total of overhead bytes |  |  |  |  |

**Question**: Did the amount of overhead for TCP increase proportionally as the file size increased? Why or why not? **Provide calculations** to support your answers – please do not make up numbers for the table above as I will be looking at the captures to compare.

**Grading: Please turn in the following to Blackboard on the due date and time**

* Source Code for both programs – 55pts
* Screenshot of your programs running – 10pts
* Screenshot of your Wireshark capture summary page for four transfers– 10pts
* Your table filled out and a graph (payload vs overhead) – 10pts
* Answers to above questions, **provide calculations** to support. – 15pts