Aaron Goldsmith
1/27/17

Lab 2 - Network Bandwidth Analysis

	Size of packet	CPU Time (Sys+User)	User	Sys
Run 1	0 bits	**0.047**	0.037	0.01
	51200000 bits	**5.501**	0.449	5.052
	102400000 bits	**10.798**	0.916	9.882
	153600000 bits	**16.297**	1.355	14.942
	204800000 bits	**21.581**	1.461	20.12
	256000000 bits	**26.911**	1.984	24.927
	307200000 bits	**32.752**	2.74	30.012
	358400000 bits	**37.925**	2.993	34.932
	409600000 bits	**43.708**	4.055	39.653
Run 2	0 bits	**0.047**	0.035	0.012
	51200000 bits	**5.527**	0.533	4.994
	102400000 bits	**10.928**	0.858	10.07
	153600000 bits	**16.389**	1.652	14.737
	204800000 bits	**21.609**	1.758	19.851
	256000000 bits	**27.064**	2.057	25.007
	307200000 bits	**32.712**	2.519	30.193
	358400000 bits	**38.359**	3.154	35.205
	409600000 bits	**43.634**	3.491	40.143

Each iteration, an additional 5,120,000 bits are sent along with the packet.
This is equivalent to 6.4 MB in physical memory.
The largest packet sent was ~50MB and it took an average 43.671s
The bandwidth is the bits/sec, therefore the bandwidth connection was 1.172 MB/s

As larger packets are sent, it will take more time to successfully send packets. The total CPU time is calculated by taking the Sys Time and adding it to the User time.
The data that was sent to the school server grew in size with each successful run. It started by sending a packet of 0 bits, and then continued to append 51,200,000 bits after each run.

Script.sh

The script that I used to send packets was created by my TA T.Chen. Each time the script completed successfully, a larger packet would prepare to be sent

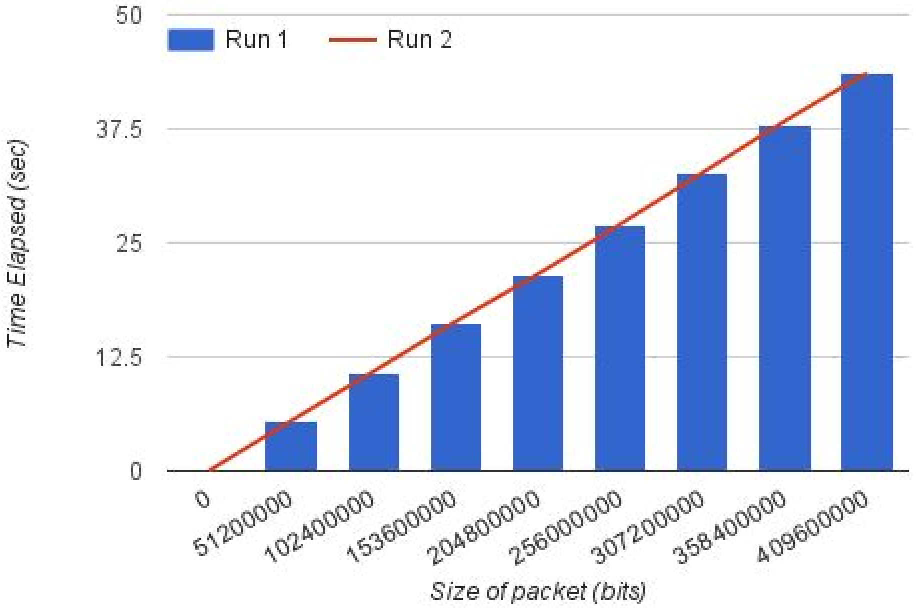
The functionality of this script reminded me of what a DDoS attack might intend to do. Since the script only runs following a successful (smaller) attempt, the system's bandwidth can quickly become occupied assuming a packet of greater than 0 bits successfully reaches its destination address.

Problems that I encountered:

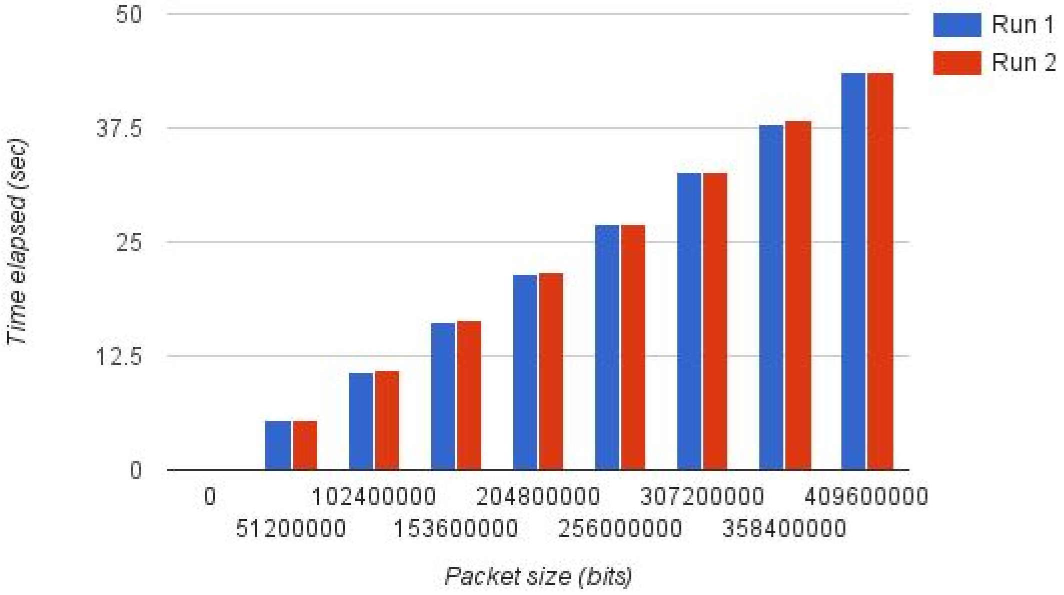
1. I was sick on the day of lab, so had to do everything on my own, at home.
2. The first issue I encountered was in setting up the ~/.ssh directory. I found that it wasn't as straightforward as the instructions made it seem. Apparently I had duplicate files, and overwriting them created more problems. After deleting the old `authorized_keys`, the script worked!
3. The main problem was that I had to enter in my password three times before a package was sent. I understood that this problem would occur if the public / private key has not been properly set up.
4. I returned to the design center lab to redo the packet sending and data collecting, and this time the time graphs made a lot more sense.

Sample Output:

1)	0 bits	user 0m0.037s sys 0m0.010s	**7)**	307200000 bits	user 0m2.740s sys 0m30.012s
2)	51200000 bits	user 0m0.449s sys 0m5.052s	**8)**	358400000 bits	user 0m2.993s sys 0m34.932s
3)	102400000 bits	user 0m0.916s sys 0m9.882s	**9)**	409600000 bits	user 0m4.055s sys 0m39.635s
4)	153600000 bits	user 0m1.355s sys 0m14.942s	* Omitted the *real* time, since it wasn't used in the bandwidth calculation		
5)	204800000 bits	user 0m1.461s sys 0m20.126s			
6)	256000000 bits	user 0m1.984s sys 0m24.927s			



Time elapsed vs Packet size



Time elapsed vs Packet size