

## COL374/672 Computer Networks: 2019-20 semester I

### Assignment 3: Packet trace analysis

In this assignment, you are given packet traces for three days of FTP connections on port 21 to a set of FTP servers. The traces were captured by sniffing the network to which the servers are connected, and therefore contain both incoming and outgoing packets to/from the servers.

<http://www.cse.iitd.ernet.in/~aseth/temp/col334-traces/lbni.anon-ftp.03-01-11.csv>

<http://www.cse.iitd.ernet.in/~aseth/temp/col334-traces/lbni.anon-ftp.03-01-14.csv>

<http://www.cse.iitd.ernet.in/~aseth/temp/col334-traces/lbni.anon-ftp.03-01-18.csv>

Note that FTP port 21 is used by clients to initiate an FTP connection and send commands to the server. The actual data is transferred over connections on port 20, for which we do not have traces.

The traces on port 21 are given as CSV files which you need to parse for several analyses. Each row in the CSV has the following structure:

*Packet number (artificially created) | Time of packet capture (relative time, starting from 0) | source IP | destination IP | protocol | packet length | Information*

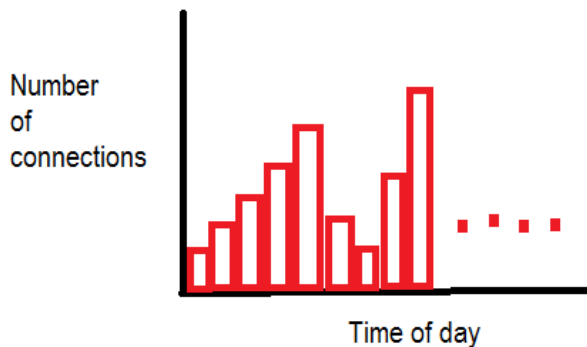
The *Protocol* field is either TCP or FTP. Rows for TCP show in the *Information* field, information about the TCP header such as the source port and destination port, and flags turned ON in the TCP header (eg. whether the packet is a SYN packet, whether the ACK field is active, etc). Rows for FTP show information about a few initial bytes of the actual content in the packets, such as commands issued by the user or responses sent by the server.

What different FTP commands and response codes do you see? Use the FTP RFC to describe these commands briefly.

You next need to write programmes to parse these packet trace files and answer the following questions. Answer them for each of the three days for which the packet traces are provided.

1. How many unique server IPs do you see? How many unique client IPs? Hint: Check all the SYN packets.
2. How many unique TCP flows do you see? Hint: A TCP flow is identified uniquely by its source IP, destination IP, source port, and destination port.
3. Draw a plot of the number of connections (TCP flows) opened to any FTP server within 60min windows over the 24-hour duration, ie. within hours 00:00-01:00, 01:00-02:00, 02:00-03:00, etc. This will give you the traffic profile in any given day, and you can check if the profiles for the three days are similar to one another. A sketch is below of the kind of plot that is expected.

How can you use these traffic profiles to detect if the system is under a DoS attack?



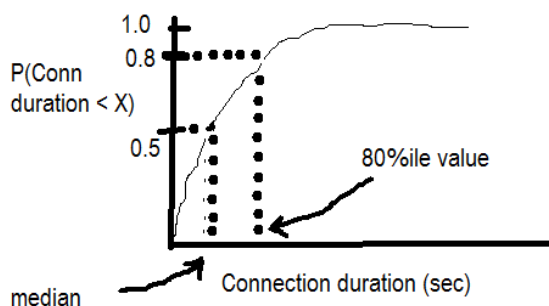
4. For all the connections, find the duration over which a connection was kept open, and plot the CDF (cumulative distribution function) of these connection durations. Hint: To compute a connection duration, use a SYN packet to denote the start of the connection and a FIN or RST packet to denote the end of the connection. A sketch is below of the kind of plot that is expected. Also report the mean and median of the connection durations.

Note that SYN packets will always be sent by the client, but FIN or RST packets could be sent by either/both the client or the server. You will therefore have to be careful when mapping these packets to the same flow, because the source-IP/destination-IP/etc will flip depending upon whether you are looking at a packet going from a client to a server, or vice versa.

Also note that some flows might have started before the packet capture started, and some flows might be continuing beyond the end of the packet capture. Ideally you should ignore such boundary flows from your calculation.

Another important point to keep in mind is that it is possible that some port numbers might have been reused, ie. a flow between some source/ destination pair using some source/ destination port combination could have ended, and sometime later a new flow could come up between the same pair and using the same port numbers. You should try to identify such cases and handle them as different flows, or ignore both of them. Counting them as one flow which spans a long interval of time would be a mistake, and could cause your PDF/CDF plots to include outliers that will make it hard to understand the distribution.

Notice how most connections are of a short duration. Why do you think so?

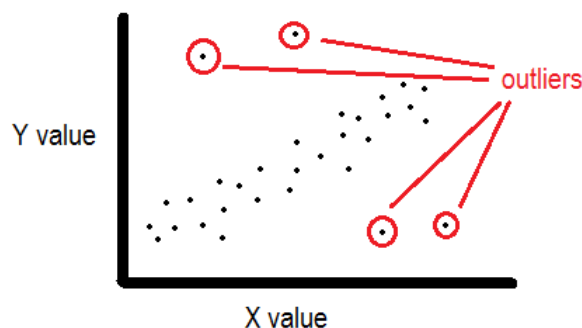


Here, the Y-axis range is from 0.0 - 1.0, and shows the probability that connection durations are less than the corresponding X-axis value.

5. On the same lines as above, for all the connections find the number of bytes sent and received over the connection, and check if there is a correlation between the connection duration (as computed in question 4 above, ie. the duration over which the connection was kept open) and the number of bytes sent to the servers. Similarly check if there is a correlation between the number of bytes sent to the servers and number of bytes received from the servers. Hint: To do this, you will need to sum the packet lengths for all packets belonging to the same flow, identified uniquely with the four tuple (source IP, destination IP, source port, destination port).

To find the correlation, you can use the Pearson's correlation coefficient assuming that it is likely to find a linear relationship between the variables. You should also make a scatterplot of the two variables, a sketch is shown below of what it might look like.

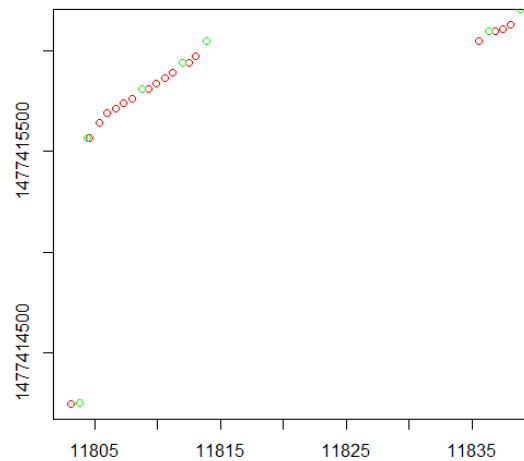
Do you find that there is a correlation? Does the scatterplot help you identify any outliers? If you eliminate the outliers, does your correlation improve?



6. Next, plot the CDF of the inter-arrival times between two consecutive connections being opened, along with the mean and median values. For example, if you see SYN packets for new connections opened at times 0, 10, 15, 17, 20... then the inter-arrival times are (10, 5, 2, 3...).
7. Similarly, plot the CDF of the inter-arrivals times of incoming packets to the servers, and report the mean and median values.

Again notice that most interarrival times are short, but the range is quite wide between the maximum and minimum interarrival times. Why do you think this is the case?

8. Plot the CDF of the packet lengths for the incoming and outgoing packets separately. Does the CDF appear to be clustered around a few values? Why do you think this is the case?
9. Write a tool that given a flow identified by the four tuple, you can generate a sequence number plot for the flow, ie. a trace of the sequence number of a packet and the time at which it was dispatched, along with the acknowledgement for the packet and the time at which the acknowledgement was seen. We will do this only for data sent by the server, ie. use the sequence number on packets sent by the servers, and acknowledgement numbers on packets sent by the clients. A sketch is below of the kind of sequence number plot that is expected.



The Y-axis shows the sequence numbers and ack numbers, and the X-axis is the time. Sequence numbers are shown in red and ack numbers in green

Show the sequence number plots for any two of the most data-intensive connections identified in question 5 above that appear interesting.

Identify some shorter connections where packet retransmissions took place, and show the sequence number plots for two of these connections. Explain what the retransmissions look like in the plot. A retransmission can be spotted by checking if the same sequence number is transmitted more than once. You can write a separate analysis tool to identify flows in which retransmissions happened, and then show the sequence number plot for these flows.

Similarly, identify some connections where spurious retransmissions happened, and show the sequence number plots for two of these flows. Explain the plots too. You can spot spurious retransmissions in this trace because it has captured both acknowledgements as well as data packets. A spurious retransmission is when a packet is retransmitted that wasn't required to be retransmitted, ie. an ACK was seen acknowledging the packet but even then it was retransmitted. Perhaps this ACK packet later got dropped or corrupted and hence a retransmission happened, or the ACK reached the sender late before the retransmission timeout expired. You can write a separate analysis tool to identify flows in which spurious retransmissions happened.

In the same way, show two sequence number plots for flows where duplicate acknowledgements are seen, and explain the plots.

Show two sequence number plots for flows where out-of-order delivery has happened, and explain the plots.

Note that sometimes the plots may appear strange with missing packets and acks. This can happen because the packet capture may not have captured all the packets. This is normal and can happen if the machine that is capturing all packets and saving them to the disk is not able to keep up.

- For questions 6 and 7 above, we next want to fit probability distributions to the data. This is useful to model the workload for theoretical analysis of the system under different conditions. You can do this as follows using R:

- a. Download R from <https://cran.r-project.org/bin/windows/base/>
- b. Write a program which analyses the packet trace files and outputs the packet interarrival times into a data file. This data file can have a simple format, with one interarrival time value on each line. Similarly you can write another program to create a data file for the interarrival time between connections
- c. Open R and install a package for fitting distributions: fitdistrplus. More about this is explained on the following pages if you are interested, otherwise the description given below should be enough to get the job done

<http://www.di.fc.ul.pt/~jpn/r/distributions/fitting.html>

<https://cran.r-project.org/web/packages/fitdistrplus/vignettes/paper2JSS.pdf>

- d. The following code which you can copy-paste into the R console will read your data and show the CDF and PDF for the data. You can also use this for questions 4/6/7/8 above, to plot the CDFs. The sequence number plot shown above was also generated using R.

```
df <- read.csv("C:/courses/col334/packetanalysis/interarrivals.csv", header=TRUE)
```

```
library("fitdistrplus")
```

```
plotdist(df$X, histo=TRUE, demp=TRUE)
```

- e. The following can help you spot outliers. You can remove them from your data files and check how the statistics change

```
plot(df$X)
```

```
descdist(df$X)
```

- f. The following can fit your data to the exponential distribution

```
par(mfrow=c(2,2))
```

```
fe <- fitdist(df$X, "exp")
```

```
fe
```

```
denscomp(list(fe), legendtext=c("exp"))
```

```
cdfcomp(list(fe), legendtext=c("exp"))
```

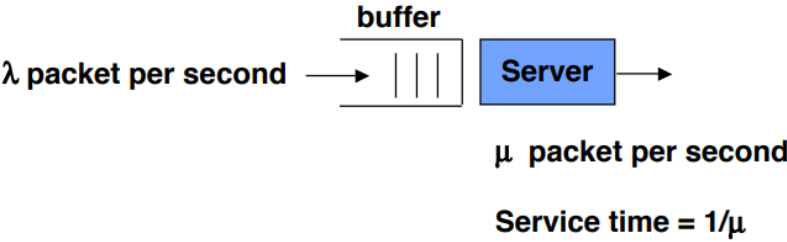
```
qqcomp(list(fe), legendtext=c("exp"))
```

- g. In the code above, the par(..) method specifies the number of rows and columns in which to divide the graphics pad to show multiple graphs together. The fitdist(..) method fits the data to a distribution, and shows the best fitted distribution parameters. The exponential distribution only has a single parameter: The rate. For interarrival times, this rate is simply the expected number of packets per second that most closely fits the data.

The rest of the methods are different ways to compare how good a fit is the exponential distribution to the data:

- i. `denscomp` compares the PDF of the data with the PDF of the fitted distribution
  - ii. `cdfcomp` similarly compares the CDF
  - iii. `qqplot` is the quantile-quantile plot to compare the quantile of the data with the quantile of the fitted distribution, ie. it plots the 10%ile of the data with the 10%ile of the distribution, the 20%ile of the data with the 20%ile of the distribution, etc. A perfect fit would lie along the X=Y line.
  - h. Similarly you can try fitting other distributions. Repeat the code above for “norm” instead of “exp” for the normal distribution, “lnorm” for the log-normal distribution, etc, and report the plots and parameter estimates. You are likely to find that the exponential distribution often provides the best fit. For that reason, and also some interesting mathematical properties of the exponential distribution, it is often used to model interarrival times for various processes.
11. Assuming that the packet interarrival times are exponentially distributed, we will next try to understand the system behavior for different parameter values of the distribution. Let us begin with first understanding the exponential distribution:

The PDF of the IA (interarrival) times is given by $P(\text{IA} = x) =$	$\begin{cases} \lambda e^{-\lambda x} & x \geq 0, \\ 0 & x < 0. \end{cases}$
The CDF is given by $P(\text{IA} \leq x) =$	$\begin{cases} 1 - e^{-\lambda x} & x \geq 0, \\ 0 & x < 0. \end{cases}$
The parameter <i>lambda</i> of the exponential distribution is called the rate, and is measured in units of <i>packets per second</i> in this case. It is the expected number of packets per second, or in other words the inverse of the expected interarrival time between packets. Note that the value obtained by fitting an exponential distribution may not exactly tally with the mean interarrival time as measured from the data, and that is because the exponential distribution that is obtained using the <i>lambda</i> parameter may not be an exact fit to the data.	
The exponential distribution is memoryless, ie. the probability of receiving a packet after a certain interarrival time from the last packet is independent of when the last packet was received.	
The figure below shows a single queue, where packets enter the queue at a rate of <i>lambda</i> packets per second, and are serviced at a rate of <i>mu</i> packets per second. On a link, <i>mu</i> would be nothing other than the rate of the link, in terms of packets per second. The service time is the time needed to service one packet, ie. its transmission delay.	
<i>Lambda</i> / <i>mu</i> is called the utilization factor <i>rho</i> = $\rho = \lambda / \mu$	

	
The probability of having a queue size of $n$ can be shown to =	$P(n) = \rho^n (1 - \rho)$
The average queue size $N =$	$N = \sum_{n=0}^{\infty} n P(n) = \sum_{n=0}^{\infty} n \rho^n (1 - \rho) = \frac{\rho}{1 - \rho}$ $N = \frac{\rho}{1 - \rho} = \frac{\lambda / \mu}{1 - \lambda / \mu} = \frac{\lambda}{\mu - \lambda}$
The average waiting time for a packet in the queue =	$W = \frac{1}{\mu - \lambda} - \frac{1}{\mu}$

Use the value of *lambda* as estimated by the exponential fit for the packet interarrival times for the incoming packets.

To calculate the value of *mu*, assume that the link can transmit data at 128Kbps. Then use the mean value of packet length as found in question 8 above to calculate *mu* in terms of packets that can be serviced per second.

What is the utilization factor *rho*? For this utilization factor, what is the average queue size and the average waiting time for a packet in the queue?

Of course, since the trace only includes FTP control packets, the utilization factor and other quantities are likely to be low. The actual utilization factor with FTP data traffic and other applications is likely to be much higher.

Now graph the queue size and the average waiting time (on separate graphs) for different values of *lambda*, starting from 0 to getting close to *mu*. What does this tell you about how the queue sizes and delays change with an increasing load in the network? This is an example where actual network data is used to validate whether the workload can be modelled using some mathematical distributions, and then the distributions are used to examine system behavior under different network conditions.