ECS 132 Fall 2015
Project 1 Part 2

November 24, 2015

1 Introduction

The goal of this part of the project is to learn about how to detect timing channels and also how to engineer them. This is an experimental project and this is a draft write up. Some revisions are likely to arise but this should help you to start to think of the different steps.

2 Detection

In this section we learn about how to detect the timing channel. Recaping Part 1 of the project, Alice and Bob are secretly trying to embed a secret message in the inter-packet delays (IPDs) of packets generated by their innocuous Skype call. They do so by following a well-known statistical model of the Skype. This type of covert timing channels are called Model-based Covert Timing Channels (MBCTCs). Eve is the warden and she wants to detect the timing channel. Eve knows the model of the IPDs of the (unmodified) Skype traffic. She observes the covert traffic and uses `qqplot` which can be used to see if two data sets come from a population with a common distribution.

1. See the following URL `http://www.itl.nist.gov/div898/handbook/eda/section3/qqplot.htm` to understand `qqplot`.
2. Using the R help function for `qqplot` learn about the usage of `qqplot`.
3. While `qqplot` works with data sets of different sizes, in this project we will consider data sets that are of equal size.
4. Generating values that follow a specific distribution can be easily done in R. For example, the function `rnorm` can be used to generate values that follow the normal distribution with mean μ and standard deviation σ. Similarly, `runif` for data sets following the Uniform distribution, and `rexp` for data sets following the exponential distribution. Use the R help function to learn their usage.

2.1 Steps

Do the following steps.

1. Generate two different samples of the standard Normal distribution (i.e., mean $\mu = 0$ and $\sigma = 1$) of size $n = 30$ using `rnorm`. Do a `qqplot` of the two data sets.
2. Repeat the above step for $n = 100, 1000$. Give a summary of you observations.

3. Generate two data sets each of size $n = 100$, one for standard Normal and the other for Normal with $\mu = 5$ and $\sigma = 3$. Draw the `qqplot` and record you observation.

4. Do step 2 for Exponential distribution with $\lambda = 1$.

5. Using `qqplot` compare the standard Normal with Exponential with $\lambda = 1$ for sample size $n = 100, 500$.

6. Now consider the naive embedding scheme (Question 1/Step 1 of Part 1). Using `qqplot` compare the distribution of the IPDs of the overt and covert traffic.

7. Now compare the distribution of the IPD of the overt and the covert traffic generated by improved scheme in Question 3 in Part 1 of the project.

8. Finally, the Compare the IPD of the overt and covert traffic for the embedding scheme that you developed for Part 1 of Question 5.

3 Implemenation

In this section, we will work through the implementation issues. The overall sender-side system is shown in

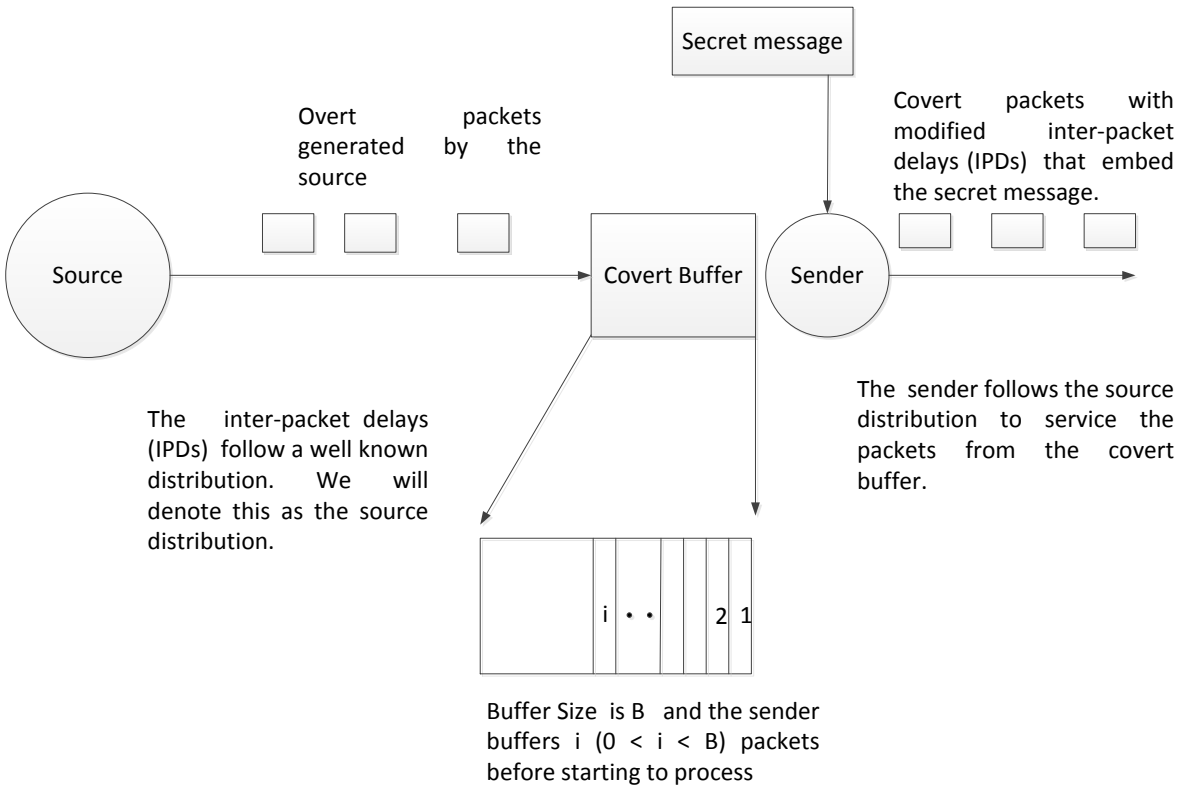

Figure 1: The overall system diagram of the source and the covert sender.

We will make the following assumptions

1. We will consider that the source generates packet following a well-known IPD distribution. The sender also knows this distribution and follows it to inject the delay between the packets to embed the secret message. It is important to note that the source and the sender are independent. Hence, even though they follow the same distribution, the sequence of delays generated by the source will be different from the sequence of delays generated by the sender.
2. To embed a 0, the sender generates a delay between the minimum value (min) and the median. To embed a 1 the sender generates a delay between the median value and the maximum value (max). The median
3. The secret message is a randomly generated sequence of 1s and 0s of size m bits and is given. We will consider two values $m = 16, 32$.
4. The sender has a buffer of size B and initially the sender buffers i packets before starting to send the secret message.
5. Buffer overflow: when a packet arrives at the sender and there are already B packets in the buffer.
6. Buffer underflow: when the sender needs to send a packet but there are no packets in the buffer.

In this project complete the following steps.

1. For buffer size $B = 20$ we want to find out the probability of overflow and underflow, when the IPD follows the Exponential with $\lambda = 1$ and $i = 2, 6, 10, 14, 18$. Use message size $m = 16, 32$ bits.
2. For buffer size $B = 20$ we want to find out the probability of overflow and underflow, when the IPD follows the Uniform distribution in the range (0,1) and $i = 2, 6, 10, 14, 18$. Use message size $m = 16, 32$ bits.
3. Using the Gambler's Ruin problem determine bounds on overflow and underflow probabilities.
4. Propose methods to deal with buffer overflow and underflow.

3.1 Some Notes on Simulating the Implementation

For steps 1 and 2, since the source and the sender are independent processes, a proper way to simulate would be using a discrete event simulation module such as `simmer` in R. However, we can simplify and use basic R. To do this, we can pre-generate the times when the source generates packets and store it in a list. Then we can write the code to simulate the buffer, the encoding scheme, and the sender. This can be done in a single "process." Based on this, following is a very **rough** set of steps to simulate the system.

For each experiment we can break it down to the following steps

1. Generate the random bit pattern of 1s and 0s of size m which is the secret message.
2. Generate a sequence of times when the source will generate the packets. This is based inter-packet delay (IPD) distribution of the packets generate by the source. You can intuit what is the worst case number of packets that you need.
3. For the buffer you need to keep some variables such as B: buffer size, i: the initial buffer size to start sending he secret message bits and CB: current buffer size.
4. For the sender you need to maintain some variables such as the time when the next packet will be sent.
5. For each secret message bit:

(a) Generate a delay following the encoding scheme and hence determine when the next packet will be transmitted.

(b) Update the state of the buffer depending on the number of arrivals during that time.

(c) At appropriate places check for buffer underflow and overflow and break out if it the case.

(d) Appropriately update the current time.

6. Do the experiments multiple times to calculate the different probabilities.

4 Submission Guidelines

Here are the submission guidelines

1. You are required to submit a hardcopy of a report that must contain a) all the results and the analyses/observations for both the detection and the implementation parts and b) a commented R-code for simulating the implementation.

2. One report per group.

3. The report is due 12/7 at 5pm in the homework box in Room 2131 in Kemper Hall.