**Design**

Question #1

overtData <- read.csv(file = "Traffic\_data\_orig.csv", header = TRUE,sep = ",")

secretMessage <- "this is a secret message"

messageLen = as.numeric(nchar(secretMessage))

CharToBinary = function(pchar) {

lhex = charToRaw(pchar)

lbits = rev(as.numeric(rawToBits(lhex)))

return(lbits)

}#Converts a character to Binary equivalent

StringToBinary = function(pstr, pstrlen) {

lbitstream = NULL

ltemp = NULL

for (i in 1:pstrlen) {

ltemp = CharToBinary(substring(pstr,i,i))

lbitstream <- c(lbitstream,ltemp)

}

return(lbitstream)

}#converts a string to Binary vector

#Q1. GENERATING MODIFIED PACKET STREAM

binaryMessage = StringToBinary(secretMessage, messageLen)

lenBinaryMessage = as.numeric(length(binaryMessage))

covertPackets = NULL

timeStream = 0

covertDataDelay = NULL

for (i in 1:lenBinaryMessage) {

if(binaryMessage[i] == 0){

timeStream <- timeStream + 0.25

covertDataDelay <- c(covertDataDelay,0.25)

}#if bit is 0, delay is 0.25

else{

timeStream <- timeStream + 0.75

covertDataDelay <- c(covertDataDelay,0.75)

}#if bit is 1, delay is 0.75

covertPackets <- c(covertPackets,timeStream)

}#loop to generate packet stream from binary message

#Q2. Generating the Histogram

temp = 0 #holds the previous timestamp

overtDataDelay = numeric(dim(overtData)[1])

for( i in 1:dim(overtData)[1])

{

overtDataDelay[i] = overtData[i,2] - temp

temp = overtData[i,2]

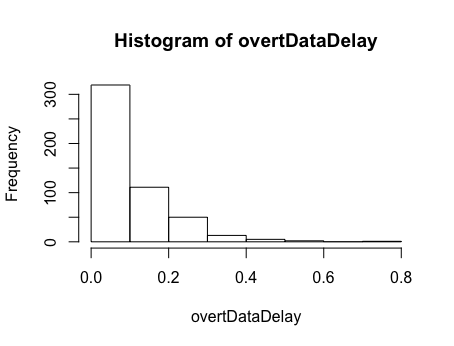
}

hist(overtDataDelay)

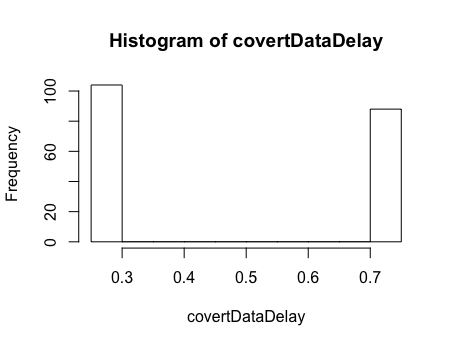
hist(covertDataDelay)

Question #2

**Overt Packet Delay Histogram**

****

**Covert Packet Delay Histogram**



Will Eve be suspicious?

Eve will be suspicious of the 0.75 & 0.25 because it's exactly two unique differences, whereas for the "Overt" data, given to us in the CSV, the histogram looks more like a negative exponential distribution, which is what you would expect of this type of time dependent data

Question #3

#Q3

m = median(overtDataDelay)

min = min(overtDataDelay)

max = max(overtDataDelay)

semiRandomPacketsDelay = NULL

semiRandomPackets = NULL

timeStream = 0

for (i in 1:lenBinaryMessage) {

if(binaryMessage[i] == 0){

timeLapse = runif(1, min, m)

}#if bit is 0, delay is 0.25

else{

timeLapse = runif(1, m, max)

}#if bit is 1, delay is 0.75

timeStream <- timeStream + timeLapse

semiRandomPacketsDelay <- c(semiRandomPacketsDelay,timeLapse)

semiRandomPackets <- c(semiRandomPackets,timeStream)

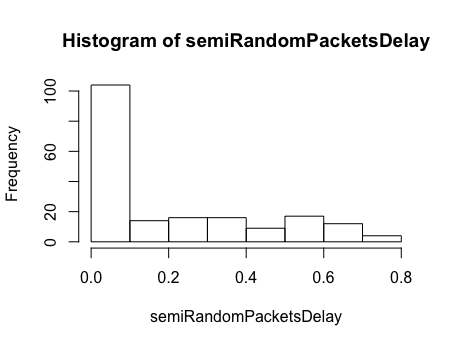
}#loop to generate packet stream from binary message

hist(semiRandomPacketsDelay)

hist(overtDataDelay)

**Note: Code is built up on code from question 1**

Question #4

****

Question #5

1. If we set m from part 3 to the value that represents the third quartile rather than the median, the data appears to have more of a negative exponential distribution, which is the same distribution that the overt packet delay dataset displays. This would appear less suspicious to Eve due to the similarity of the two distributions.
2. If Alice buffers the packets, there will be a moment (in the beginning) when no packets are sent which would disrupt the continuity of the timing channel; this is obviously unrealistic in the case of a Skype call, which is based on real-time interaction. Overall, it would negatively affect the quality of the call, leading Eve to suspicion.
3. If the network modifies the inter packet delay, the message cannot be decoded correctly since the delays are changed and each bit is encoded via delay times. If Bob knows the distribution of the noise within the channel, he can subtract the mean of the noise’s distribution from the covert delay times while decoding to get a somewhat accurate message. This method is obviously flawed as it assumes that the semantic integrity of the message isn’t critical.