Steganography & Steganalysis

INSTRUCTOR: JOHN ORTIZ
SENIOR COMPUTER ENGINEER
UTSA
STEGO@SATX.RR.COM

REQUIRED BACKGROUND

Required Background Overview

- Math
- Information Theory
- Real Random Numbers
- Cryptographic Hashing
- Data Compression Techniques

Sum Fun Math

- The summation symbol simply means to add up a series of numbers

$$\sum_{j=0}^{n-1} X_j$$

- The equivalent in C is:
  - int j, sum = 0;
  - for(j = 0; j < n; j++)
  - { sum = sum + X_j; }
- QUIZ: How many times will this loop?

Sum Fun Math

- n = total number of elements
- P = probability of occurrence of a particular element
- X = element
- j = index
- If P_j is the same for all elements (this is called equiprobable), then what is a simple, everyday name for 'a'?

$$a = \sum_{j=0}^{n-1} P_j X_j = \sum P_j X_j$$

- Sometimes, 'n' is not shown so it just means to SUM over the entire set, whatever the size

Exponentially Fun Math

- Definition: $\log_b N = x$ is such that $b^X = N$
- Read as "the logarithm base b of N equals x"
- You have two logarithm buttons on your calculator
  - "log" and "ln"
  - log is base 10 and ln is base e (e = ~ 2.71)
  - ln is the natural log and has numerous applications
- You also have three inverse log buttons
  - 10^X, e^X, y^x
- Unfortunately, we will deal with log base 2
  - Most likely, you do not have this button available ☹

Exponentially Fun Math

- However, there is a formula to help you out ☺
- $\log_b N = \log_a N / \log_a b$
- The value of 'a' is arbitrary, so you can use either log base 10 or log base 'e'
- For log base 2, we will use the notation "lg"
  - $\log_2 N = \lg N$
- Unless otherwise indicated, log N will imply a base of 10

Exponentially Fun Math

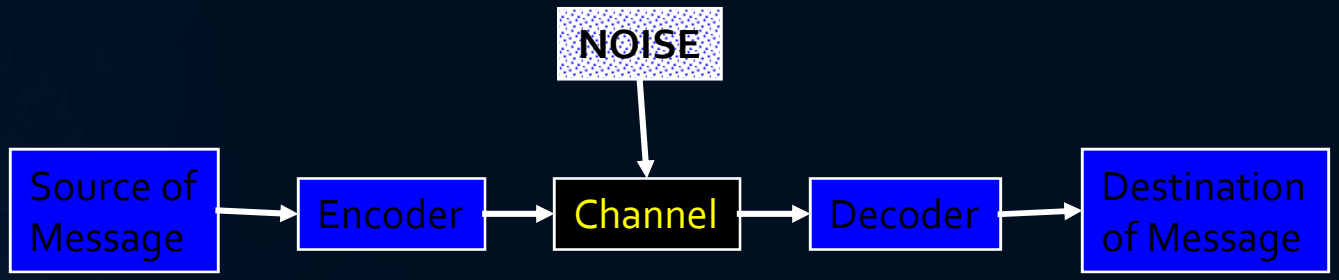
- Some other properties of logarithms (base is irrelevant)
  - $\log (N * M) = \log N + \log M$
    - note: $10^x * 10^y = 10^{(x+y)}$
  - $\log (N / M) = \log N - \log M$
  - $\log N^M = M \log N$
  - $\log_b b = 1$ (base matches number)
  - $\log 1 = 0$
  - $\log 0 =$ undefined

Exponentially Fun Math

- A few other properties:
  - Log N > 0 if N > 1
  - Log N = 0 if N = 1
  - Log N < 0 if N > 0 && N < 1
- What about the inverse lg?
- Example, what is the inverse lg of 8?
- inv log 6?
- inv log 0?
- inv lg 12?

Information Theory

- Information theory is a branch of science that deals with the analysis of a communications system
- We will study digital communications – using a file (or network protocol) as the channel



- Claude Shannon Published a landmark paper in 1948 that was the beginning of the branch of information theory
- We are interested in communicating information from a source to a destination

Information Theory

- In our case, the messages will be a sequence of binary digits
  - Does anyone know the term for a binary digit?
- One detail that makes communicating difficult is noise
  - Noise introduces uncertainty

Information Theory

- Suppose I wish to transmit one bit of information what are all of the possibilities?
  - tx 0, rx 0 - good
  - tx 0, rx 1 - error
  - tx 1, rx 0 - error
  - tx 1, rx 1 - good
- Two of the cases above have errors – this is where probability fits into the picture
- In the case of steganography, the "noise" may be due to attacks on the hiding algorithm

Information Theory

- Probability may be defined in several ways
- It may be the likelihood of outcomes of a certain set of events, for example, the flipping of a coin
  - How many possible outcomes are there?
- Another way of looking at probability is that it is a measure of human surprise
- More surprise, more information, lower probability
- Probability is always between zero and one
  - Probability = zero means it will NEVER happen
    - (is this EVER true?)
  - Probability = one means it will ALWAYS happen

Information Theory

- Claude Shannon introduced the idea of self-information

$$I(X_j) = \lg \frac{1}{P(X_j)} = \lg \frac{1}{P_j} = -\lg P_j$$

- Suppose we have an event X, where X_i represents a particular outcome of the event

Information Theory

- Consider flipping a fair coin, there are two equiprobable outcomes:
  - say X_0 = heads, P_0 = 1/2, X_1 = tails, P_1 = 1/2
- The amount of self-information for any single result is 1 bit
- In other words, the number of bits required to communicate the result of the event is 1 bit

Information Theory

- When outcomes are equally likely, there is a lot of information in the result
- The higher the likelihood of a particular outcome, the less information that outcome conveys
- For instance, if the coin is biased such that it lands heads up 99% of the time, there is not much information conveyed when we flip the coin and it lands on heads
  - There is not much human surprise either
- Note: We use log base 2 to get the units in bits

Information Theory

- Suppose we have an event X, where X_i represents a particular outcome of the event
- Consider flipping a coin, however, let's say there are 3 possible outcomes: heads (P = 0.49), tails (P=0.49), lands on its side (P = 0.02) – (likely MUCH higher than in reality)
  - Note: the total probability MUST ALWAYS add up to one

$$I(X_j) = \lg \frac{1}{P(X_j)} = \lg \frac{1}{P_j} = -\lg P_j$$

- The amount of self-information for either a head or a tail is 1.02 bits
- For landing on its side: 5.6 bits

Information Theory

- Entropy is the measurement of the average uncertainty of information
  - We will skip the proofs and background that leads us to the formula for entropy, but it
  - was derived from required properties
  - Keep in mind that this is a simplified exp[illegible]tion
- H – entropy
- P – probability
- X – random variable with a discrete set of possible outcomes
  - (X_0, X_1, … X_{n-1}) where n is the total number of possibilities

I KNOW it looks hard, but it <u>really</u> is EASY! (Once you figure it out.)

$$Entropy = H(X) = -\sum_{j=0}^{n-1} P_j \lg P_j = \sum_{j=0}^{n-1} P_j \lg \frac{1}{P_j}$$

Information Theory

- A computer file is represented by a sequence of bytes
- If each of the 256 bytes is equiprobable, H = 8, therefore it will require 8 bits/byte to TX the information
- In other words, this file can NOT be compressed
- Determining the true probability is more difficult than simply counting the number of times a symbol appears in a file
  - But it's pretty close!

Information Theory

- Entropy is the average uncertainty
- Purely random data has the greatest possible uncertainty, i.e. entropy
- Entropy is the theoretical limit of how well something could be compressed
- Compressed data has high entropy, but there is still some redundancy
- Encrypted data and compressed data can, in most cases, be distinguished by entropy

Information Theory

- Take Away:
  - Have an intuitive idea of the concept of entropy
  - Entropy is the average uncertainty of information
  - Typically, we can only estimate it
  - Equiprobable == highest entropy

Really Random Numbers

- A truly random number has 3 properties:
  - Uniform distribution over the sample space
  - Independence – no prediction can be made about a future symbol based upon the latest symbol
  - Irreproducible

Really Random Numbers

- A computer can't generate a truly random number
  - Uniform distribution can be achieved, but independence and irreproducibility are not possible with an algorithm
  - Computer generated "random" numbers are called pseudo-random
- PGP gets some randomness by having the user repeatedly press keys
  - it measures the time difference between keystrokes
- Radioactive decay and noise are even better sources of random numbers

Cryptographic Hashing

- A *hash function H* is a transformation that takes an input m and returns a fixed-size string, which is called the hash value *h* (that is, $h = H(m)$).
- Hash functions with *only* this property have a variety of general computational uses
  - Indexing and retrieving items in a database
  - Generally, it is faster to compute a hash and search based on it
  - A hash function that works well for database indexing would not be the best for cryptographic purposes
- A Cryptographic hash function has additional requirements

Cryptographic Hashing

- The basic requirements for a cryptographic hash function are as follows.
  - The input can be of any length
  - The output has a fixed length
  - $H(x)$ is relatively easy to compute for any given x
  - $H(x)$ is one-way
    - given x it is easy to determine H(x), but if given H(x) it is difficult/imposisble to determine x
  - $H(x)$ is collision-free
    - different inputs will produce different hashes

Cryptographic Hashing

- A hash function *H* is said to be *one-way* if it is hard to invert,
  - ``hard to invert'' means that given a hash value *h*, it is computationally infeasible to find some input *x* such that *H*(*x*) = *h*

Cryptographic Hashing

- The hash value exactly represents the document from which it was computed
  - This value is called the *message digest*
  - A message digest is a "digital fingerprint" of the larger document
  - It should be unique for every message
- Hashing is also used to authenticate message senders and receivers
  - Get the message hash, send the hash and message
  - At the receiver, compute the hash again
  - Compare to received hash

Cryptographic Hashing

- Examples of secure hash functions are MD5 & SHA
  - MD – Message Digest
  - SHA – Secure Hash Algorithm
- "In December 2008, a group of researchers used this technique to fake SSL certificate validity., and US-CERT now says that MD5 "should be considered cryptographically broken and unsuitable for further use. " and most U.S. government applications now require the SHA-2 family of hash functions."
  - http://en.wikipedia.org/wiki/MD5

Cryptographic Hashing

- SHA-1 has had similar weaknesses
- SHA-2 functions are now the recommended standard
- "In cryptography, SHA-2 is a set of cryptographic hash functions (SHA-224, SHA-256, SHA-384, SHA-512) designed by the National Security Agency (NSA) and published in 2001 by the NIST as a U.S. Federal Information Processing Standard."
  - http://en.wikipedia.org/wiki/SHA-2
- The SHA-3 standard was released by NIST on August 5, 2015
  http://en.wikipedia.org/wiki/SHA-3

Data Compression

- Goal – Reduce size of target file or transmission
- Two main categories of compression:
  - Lossless
  - Decompressed output is *identical* to input
    - Used on files which cannot be approximated such as executables, text, html, word processor files, spreadsheets, etc.
  - Lossy
    - Decompressed output is *approximation* of input
    - Used on files that do not have to be exact – basically files that are subject to human perception such as pictures, sounds, and videos

Data Compression

- Typically, lossy techniques will yield a better compression ratio, but at a lower fidelity
- We will examine several lossless techniques and one common lossy technique
- Real-world programs will likely use a combination of techniques to optimize their overall performance

Data Compression

- There are numerous algorithms for lossless compression
- Lossless compression reproduces the input file exactly
- We will discuss the following:
  - Run-Length
  - Huffman
  - Arithmetic
  - Delta
  - Static Dictionary
  - Lempel, Ziv, Welch (LZW) (adaptive dictionary)

Data Compression

- Each of these techniques has advantages and disadvantages depending upon the type of data being compressed
- The idea of compression is to remove any/all pattern
- A pattern is nothing more than a repetition of data and compression seeks to remove this redundancy

Run-Length Encoding

- Run-Length Encoding is a very simple scheme
- For each symbol, the number of occurrences is counted
- The symbol and the corresponding number of occurrences are output to the file
  - In some cases, the symbol is known, so only the count need be saved
  - Jpeg makes use of this, storing the count of zeros

Run-Length Encoding

- The number of bits used to represent the count must be predetermined
- For example, if you have only a 5 kilobyte file, the count could never exceed 5 kilobytes, right?
- How many bits would be sufficient to hold 5 kb as a maximum count?

Run-Length Encoding

- The number of bits used to represent the count must be predetermined
- For example, if you have only a 5 kilobyte file, the count could never exceed 5 kilobytes, right?
- How many bits would be sufficient to hold 5 kb as a maximum count?
  - 13 bits. 5 x 1024 = 5120
  - 2^{12} = 4096 (too small)
  - 2^{13} = 8192 (plenty of room)

Run-Length Encoding

- If you're compressing only ASCII text, you could get away with 7 bits
- If you restrict the text to upper case letters and a few delimiters, you could do 5 bits
- You could do the count at the bit level; i.e. the number of zeros and ones
- What if your count exceeds your range?
- If you're using 5 bits for the count, your max range is only 1 - 32
  - The value "zero" could represent a count of 32
  - OR, it could mean 31 + next count

Huffman Encoding

- Huffman encoding is named after David A. Huffman, a student at M.I.T. in the 1950s
- He was in the first Information Theory class ever taught
- Huffman encoding uses variable length code words

Huffman Encoding

- The main concept is to give shorter code words to symbols with the highest probability, and longer code words to those symbols with lesser probability
  - Can anyone name another common, popular code which uses this same concept?
- Another characteristic of a Huffman code is that the two symbols that occur least frequently have a code word of equal length

Arithmetic Encoding

- Arithmetic Coding is good with small alphabets and highly skewed probabilities
- Arithmetic coding has been shown to have a 5 – 10 % improvement over Huffman, however, it is more complex
- Huffman codes require that all symbols in the source alphabet have a code, regardless of whether that symbol is used

Static Dictionary

- Another approach to compression is to make a table of frequently occurring patterns and use small code words to represent those patterns
- This table is called a dictionary
- English text is appropriate because the probabilities of various words are not equal
- "the" and "then" appear more frequently then "entropy"

Static Dictionary

- A static dictionary technique is efficient when prior knowledge about the source is available
- The larger the dictionary, the more bits required to encode the words, but if the words are long and frequent the compression will be successful
- Could combine with Huffman and use variable sized code words!

Adaptive Dictionary

- An adaptive dictionary is a scheme that requires no prior knowledge of the source
- The dictionary is constructed on-the-fly
- The dictionary must be included in the output file so that the receiver can decode it
- There are numerous variations
- LZ77 is an approach developed by Abraham Lempel and Jacob Ziv in 1977
- LZ78 solved some of the inefficiencies of LZ77
  - LZ78 had some drawbacks as well
- Terry Welch made modifications resulting in the LZW algorithm

Lempel-Zev-Welch (LZW)

- LZW is a general purpose algorithm
  - does not guarantee optimal performance for all data
  - gets reasonable performance for most data
- LZW uses a code table, typically with 4096 entries
  - how many bits does this require?
- The first 256 entries are used to encode single bytes from the input data
  - Note: this is a loss since no more than 8 bits are required to encode single bytes
- Codes 256 – 4095 are used to represent a sequence of bytes
- Each time a sequence is repeatedly encountered, the code word can be used to represent the sequence

Lossy Data Compression

- Lossy compression algorithms attempt to remove unimportant details to achieve compression
- Since details are missing from the output file, mathematically speaking, the quality is reduced
- The goal is to remove as much detail as possible without rendering the data useless or unappealing

Lossy Data Compression

- Practically, lossy techniques are combined with lossless techniques to achieve excellent compression
- Jpeg typically achieves > 75% compression
- Mp3 can reduce a 50MB file down to 4 MB which is > 90% savings
- Video achieves ~ 90% compression
  - In both cases there is little if any perceived loss of quality

Lossy Data Compression

- Lossy techniques typically take advantage of weakness in human perception
- Many studies have been done to determine what we can see/hear well, and what we can't
- For audio applications, if two similar frequencies occur close together, and one is >> amplitude, the second will be masked (i.e. we can't hear it)
  - Mpeg layer 3 uses a psychoacoustic model
- In video, we cannot differentiate between shades of color with slight differences, especially if they are near each other

Lossy Data Compression

- A 24-bit bitmap image contains 8 bits for each of three primary colors: Red, Green, Blue
- A simple lossy compression technique could discard the Least Significant Bit (LSB) for each of these colors
  - This would result in approximately 12.5% compression
- It could discard 2 or 3 least significant bits
  - At what point can you notice?
  - At what point does it matter?

Lossy Data Compression

- The answers are that it depends on the person and the purpose of the image
  - For secret missile plans, all that matters is that they can be read
  - For treasured family memories, fidelity is more important
- Why does compression matter for steganography?

- If you can discard it … you can hide in it!

HANDS - ON

- We'll use the wbh program to examine files

Questions &Comments