Introduction to Cryptography

July 15, 2019

Cryptography is defined in various ways. Some definitions will say that it is the study of writing and solving codes, but more recently the study has come to encompass user authentication, differential privacy, and secure multiparty computation. Today we are going to look at some of the earliest developments in cryptography, and see how these codes have been broken, redesigned, and broken time and time again.

Codes

General Codes

- Consist of a codebook which maps words to codewords.
- Examples:
  - Encode each letter of the alphabet as its corresponding letter: A=1, B=2, ..., Z=26.
  - If we want to encode the word "CRYPTO," we get "3-18-25-16-20-15."

Morse Code

- First used in 1844.
- Can communicate each letter in the alphabet as a series of dots and dashes.
- Example:

$$Y = -.-- \qquad A = .- \qquad L = .-.. \qquad E = .$$

- It's easy to use; you can send messages with any medium that has digital signaling (switching a light on and off or sounding a horn).
- Problems with Morse code:
  - Can only transmit one message per wire at a time, so if we want to send multiple messages, we either have to wait for each message to be transmitted in sequence or use multiple wires.
  - Not everyone knew Morse code at the time it was introduced - it was like having to learn a new language.
  - NOT prefix free! (AE = U = dot dot dash). This can lead to confusion and many miscoded messages.

Prefix-Free Codes

- A word w_1 is a prefix of another word w_2 if w_2 begins with w_1.
- Prefix-free codes: No codeword contains another codeword as its prefix.

Ciphers

- Require a *key* and an efficient algorithm for encryption and decryption.

Shift Cipher

- Enumerate the alphabet: A = 1, B = 2, ..., Z = 26.
- Pick a key k from 1 to 25 and shift each of the letters in the alphabet by k.
- To decrypt, subtract k.
- Why this is not secure: There are only 25 possible keys (26 if you can send the message "in the clear") which is few enough to try them all. You can also learn something about the message based on which letters repeat themselves and where the spaces are, as well as how far apart we know certain letters to be. For instance, if you see a letter by itself, it is likely `'A'` or `'I'` – once this letter is uncovered, so are the rest because they are all shifted by the same key.

Substitution (Monoalphabetic) Cipher

- Again, enumerate the alphabet. This time, our key is a *permutation* of the alphabet (take all the letters and switch them around). For example, the key may be:

  `D X S F Z E H C V I T P G A Q L K J R U O W M Y B N`,

  indicating that A maps to D, B maps to X, and so on...
- To encrypt, look up each letter in the permutation and switch it to that letter.
- To decrypt, reverse the process of encryption using the given key.
- This is better than Caesar's cipher because even if you break one letter, you cannot infer the rest of the letters as easily. In addition, there are *many* more possible keys (26*25*24*...*2*1).
- This is still weak because we can use a letter frequency attack to decrypt (either look for a specific word or phrase in the encryption, or look at the frequencies of each letter – we know that `'E'` and `'T'` occur the most frequently).
- It also requires a much larger key which may be harder to communicate to the recipient of the message.

One-Time Pad

- Enumerate the alphabet and select a message. Now, generate a key as long as the message where each component of the key is draw *uniformly at random* from 0 to 25. Shift each letter of the message by the corresponding shift in the key.
- Must securely communicate the key for each message.
- Must hide the key / destroy it properly after sending and decrypting.

Considerations

At this point, we have seen some very different strategies for enciphering, each with their own pros and cons. Some elements we should consider when choosing a code:

- Strength of cipher: If we want strong "security," we do not want to use a shift cipher as this is easy to attack.
- Key size: The shift cipher has the smallest key size, one-time pad has the longest (which means higher cost to securely transmit the key each time because the key must be as long as the message!).