Section-1 (Classical Cryptography)

Welcome to Learners' Space: Introduction to Cryptography!

The roots of cryptography date back to history when they were first used for being able to send messages to another party such that even if an enemy intercepted the message, they couldn't make sense of it. This is the principle of confidentiality: no one except the authorised parties should be able to make sense of the data. Classical ciphers were used to encrypt a plaintext message (in a particular language, we usually use English in our examples) into unintelligible ciphertext which cannot be decrypted without knowledge of a certain secret key known only to the authorised participants. Of course, "cannot be decrypted" is an idealisation and classical ciphers, despite trying their best to be hard to decrypt without a key, are vulnerable to all sorts of attacks that would enable an attacker to decrypt a ciphertext without knowledge of the secret key. Analysing cryptographic protocols rigorously is the field of cryptanalysis. In this section, we explore a few classical ciphers. We then move on to study the two most common forms of encoding which, though aren't technically part of cryptography, are invaluable to solving practical challenges in modern cryptography which would be covered in the subsequent sections.

Classical Ciphers

Caesar Cipher

The Caesar Cipher, famously used by Julius Caesar in wartime (though he technically used a fixed value of the key), is one of the simplest and most popular classical ciphers out there.

- https://crypto.interactive-maths.com/caesar-shift-cipher.html
  ▸ **EXERCISE:** Decrypt *HAAHJR HA KHDU* [Caesar; Key=7]

ROT13 Cipher

The ROT13 is nothing but the Caesar cipher with key=13. It's special because the encryption and decryption functions are the same - making it a 'mirror cipher' or a 'reciprocal cipher'. It's commonly used to obfuscate answers to newspaper puzzles.

- https://gchq.github.io/CyberChef/#recipe=ROT13(true,true,true,13)&input=QUJDRA
  ▸ **EXERCISE:** Decrypt *EVAT NEBHAQ GUR EBFVR* [ROT13]

Vigenère Cipher

The Vigenère Cipher, also known as the Polyalphabetic Cipher, was once considered one of the hardest ciphers to crack; so much so that it was also called le chiffrage indéchiffrable (the indecipherable cipher). Unfortunately for the cipher though, it's very easy to crack using statistical analysis.

- https://pages.mtu.edu/~shene/NSF-4/Tutorial/VIG/Vig-Base.html
  ▸ **EXERCISE:** Decrypt *Eiec fwijd bf Qpamvrvcs Isaasxi'd ugzc oa tsc uzqpgayydgkif oq Taownrrp aaxzee?* [Vigenère; KEY=ANALYSIS]

Atbash Cipher

If you have ever had to study Mental Ability for any competitive examinations (trauma moments, I know), then you must have met the Atbash Cipher innumerable times!

- https://crypto.interactive-maths.com/atbash-cipher.html
  ▸ **EXERCISE:** Decrypt *Ivzw Gsv Wz Ermxr Xlwv?* [Atbash]

Monoalphabetic Substitution Cipher

The monoalphabetic substitution cipher is a great example of a cipher that has a forbidding large key space, something that would be inhibitive to even the beefiest of computers, yet can be cracked very easily using statistical methods.

- https://crypto.interactive-maths.com/monoalphabetic-substitution-ciphers.html

Cracking Monoalphabetic Substitution Cipher - Frequency Analysis

Using frequency analysis to crack the monoalphabetic substitution cipher is Crypto101 and one of the most basic examples of cryptanalysis.

- https://www.youtube.com/watch?v=iPymNpwwjtk&ab_channel=WebCraftie

Transposition Ciphers

Another interesting class of classical ciphers is the transposition ciphers. As the name suggests, they transpose (move around) parts of the plaintext to create the ciphertext. - https://crypto.interactive-maths.com/simple-transposition-ciphers.html

Morse Code

Morse Code is NOT a cipher! It's an encoding format, albeit one used quite often in Classical Cryptography.

- https://en.wikipedia.org/wiki/Morse_code#Letters,_numbers,_punctuation,_prosigns_for_Morse_code_and_non-Latin_variants
  ▸ **EXERCISE:** Decode -- --- .-. ... . / .... .- ... / .- / ...- . .-. -.-- / .. -. - . .-. . ... - .. -. --. / ... . - / -- --- ..-. / ..- ... .- --. . ... / .. -. / .... - --- .-. -.-- [Morse]

The Enigma Machine

The Enigma Machine is a specific instance of a set of mechanical cryptographic machines called rotor machines. The Enigma Machine is undoubtedly the most popular of the rotor machines. The Germans used Enigma machines during World War II to communicate messages secretly and many countries invested huge capital funding to try and break the Enigma. It seemed almost impossible to crack the Enigma. Enter Alan Turing: he created an electro-mechanical device called the Bombe to crack the Enigma (along with his group of cryptographers at Bletchley Park). The Allied victory is doubtful to have occurred if not for the cracking of the Enigma.

- https://www.khanacademy.org/computing/computer-science/cryptography/crypt/v/case-study-ww2-encryption-machines
- https://www.youtube.com/watch?v=ybkkiGtJmkM&ab_channel=JaredOwen
- https://www.imdb.com/title/tt2084970/

Encodings

Note: *Encodings are not ciphers, and not technically a part of cryptography per se. They are just ways to describe data in different formats.*

Hexadecimal (Hex)

Hexadecimal encoding is achieved by taking the ASCII value of each character and converting that to its hex equivalent and concatenating all the formed "hex characters" together.

- https://dev.to/neumaneuma/decoding-the-confusing-world-of-encodings-part-1-3oke
- https://linuxhint.com/string-to-hexadecimal-in-python/
- https://codeigo.com/python/convert-hex-to-string
  ▸ **EXERCISE:** Decode

4865786164656369 6D616C206973206A757374206F6E65206F66207365766572616C2028696E66696E6974656C792920706F737369626C65207261646963657321 [Hex]

Base64

Base64 is a kind of encoding where each character represents one of 64 integers. In Base64 encoding, another special character "=" is used to represent padding. This is needed because ASCII strings are always multiples of 8 bits whereas Base64 strings are multiples of 6 bits, hence whenever we wish to encode an ASCII string that's not a multiple of 6 bits, we need to pad the Base64 encoding to make the encoding-length a "multiple of 8 bits".

- https://dev.to/neumaneuma/decoding-the-confusing-world-of-encodings-part-2-4lo
- https://stackabuse.com/encoding-and-decoding-base64-strings-in-python/
  ▸ **EXERCISE:** Decode *QmFzZTY0IGhhcyBhIGxvdCBvZiBjb3VzaW5zIHN1Y2ggYXMgQmFzZTMyLCBCYXNlNTgsIGV0Yy4=* [Base64]

Discussions among students are encouraged.

Created with :heart: by CSeC