Linnéuniversitetet
Kalmar Växjö

Linnaeus University

1DV700 - Computer Security
Assignment 1

Student: Leopold Huber
Personal number: 920806-0997
Student ID: lh223sf@student.lnu.se

Setup Premises

Explain your setup such as, OS, web browser, tools being used, development environment, and whatever else is necessary…

Windows 10
Chrome (use console log for hasing testing, open the inspector of your web-browser)
NodeJS (download on nodejs.org)
Type “npm install” in the main-folder in your terminal, wait for dependencies of the project to download/install, then type “npm start” to start the app in the browser (React app)

Task 1

a)
Symmetric encryption vs Asymmetric encryption: [1].
- Symetric encryption uses one key for both encryption and decryption of the data.
- Assymmetric uses one public key for the encryption, and one private key for the decryption.

Encryption algorithms vs Hash algorithms: [2]
- Encryption algorithms are used to hide the contents of a message by for example substituting letters and changing their placements. I could for example talk with my friend in and encrypted way by always using the alphabet index of the characters instead of the actual characters. A would be 1, B would be 2 and so one. So if I wanted to say "ABC" I could say "123" instead (substitution). I could also decide to always say the last words in a meaning first (transportation), so in that case "Nice to meet you" would be "You nice to meet". These are 2 very simple and easy to break made up crypto algorithms.

- Hash algorithms are used to create a hash-value, also known as a checksum. This value can be used to indicate that the data has not been tampered with. If person 1 sends a message with the hash-value of 95 then if the hash-value is 98 when person b receives it the contents of the message has changed. Hash algorithms are also one-way functions, meaning that you cannot compute the plain text from the hashed value. The hash algorithm are the rules employed to create the hash value.

Compression vs Hashing [3]
- Compression can be either lossy or lossless compression. The former removes certain data to reduce file-size, the later only reduces redundant, repeated data to reduce file-size. With lossless compression the original file can always be restored from the compressed file.

- Hashing is the activity of taking a file and reduce it to a hashed value. This value can not be used to get data about the file since hashing algorithms are one-way functions. A hashed value cannot reproduce the data that was used for creation of the hashed value.

b)
Comparing Steganography, Encryption and Digital Watermarking
Steganography – Hide a message in one of the least significant bits of an image. Two images may for example look identical for the viewer, but one may contain secret data in the bits that it is made of of. One use-case could be to send secret information by attaching an image in an email. The receiver of this image could use the bits that its composed of to read the secret message.

Encryption – A process used to conceal a message by giving other meaning to characters, words, symbols, numbers etc. I could for example say “study” when I mean taking narcotics. This way I could communicate with my peers (who understands my algorithm and what words has what meaning) when I plan on taking my narcotics, and nobody but us would understand my true intentions.

Digital Watermarking [4] – Can be of two sorts. First, visible watermarking is a visible text or image on top of a media, such as an image. It can be used to make it clear who is the copyright owner of the image. Invisible watermarking can also embed this information on a media, but nobody but the creator who has the algorithm/keys can see the watermark. This type of watermarking may be used to prove who the owner is, since the owner is the only one who can make the watermark visible. Since the watermark is invisible people using it without authorization will be caught when the owner makes the watermark visible in the images that they have used without authorization.

Task 2

a)
Decryption of "HKPUFCMHY BHDDXZH"
"encrypted message"

b)
Decryption of "QMJ BPZ B XPJZ RZWJPAXQ LAD"
"you are a true security wiz"
I solved this by using an online-tool [5] with help of the auto-solve option. It would have been hard for me otherwise. After "solving" it I had a talk with Adam who gave me some hints and now I understand that the crypto analysis I could have done is matching the cipher characters with the most common characters used, and the most common character combinations.

For example, B can only be I or A since it is alone. And QMJ and BPZ I were able to guess as "are" and "you" since they are two of the most frequent 3 letters words according to thaigi.com [6]. After this I had filled many of the blanks in the sentence and could understand how to simply guess the last words. "LAD" would be hard to guess, but since "Lad" somewhat means boy it could also be guessed that way.

c)
I find it incredibly time-consuming to do it "by hand". I can however understand how automated tools running on strong machines reduces the security of encryption algorithms with help of brute force attacks checking all different combinations of keys and algorithms. To mitigate this risk I think 2 factor authentication together with "number of tries" per IP-address locks will reduce the viability more and more in the future of brute force attacks.

Task 3

Please open the main folder and run npm install and then npm start. To test the app you can encrypt the plainText.txt with either leoCipher or caesarCipher. After this you can download the file and use the corresponding decryption methods to get the original message. One note is that I've intentionally removed formatting and star signs to make the message harder to guess, these will be removed when encrypting the original file.

leoCipher – Changes alphabetical characters into numbers. The key is an alphabet of numbers, divided by the character x. The cipher-output is a string of numbers divied by the character z.

The key "4x18x22x23x15x13x24x10x5x7x21x26x11x1x17x2x3x6x8x9x12x14x16x19x20x25x" for example indicates that we can exchange A for 4, B for 18, C for 22 and so on. So the plaintext ABC with this key will have the cipher-text "z4z18z22".

caesarCipher – Changes the start of the alphabet. Key is a number indicating on what character the alphabet starts. The characters before the number will come after z instead.

The key "7" for the plaintext ABC will therefore have the ciphertext "hij".

Task 4

Please see the file “LeopoldHuber.txt”. It is uploaded and also in this file. The leoCipher described in Task 3 was used to encrypt the “Plaintext.txt” file with the key “4x18x22x23x15x13x24x10x5x7x21x26x11x1x17x2x3x6x8x9x12x14x16x19x20x25x”.

Task 5

I decrypted the file "ElinaComstedt_substitution.txt". I started by checking for similarities between that file and the "plaintext.txt" template and noticed that the cypher over the *** signs must represent this text "Secret message - Top Secret" and "May only be read by security passed personnel"

With this in mind i could compare the plain text with the cypher-text over the *** signs to find out what every plain-text character is encrypted to. Using this method I was able to find out the plaintext-cypher pair for around 70% of the characters in the alphabet.

Now I guessed the remaining pairs by checking what values would make sense for words that still had plaintext characters that I had no cypher character for.

After filling in the gaps only the plaintext Q and Z was not decrypted yet, but by looking at the other decrypted characters I realized that they must be represented by the cypher characters H and Q, because of the order of the cypher characters. I only now noticed that the encryption used was a Ceaser cipher (the "cypher alphabet" started on "R" instead of "A"). After using an online-tool [5] and inputting the key I had guessed I noticed that the plaintext was about harry-potter. The key I guessed by the process above is posted below:

PLAIN	**CYPHER**
A	r
B	s
C	t
D	u
E	v
F	w
G	x
H	y
I	z
J	a
K	b
L	c
M	d
N	e
O	f
P	g
Q	H
R	i
S	j
T	k
U	l
V	m
W	n
X	o
Y	p
Z	Q

Task 6

a)
Please enter desired value in the “Paste text to be hashed” input box. Your hashed value will be under the “Start Hashing” button. I’ve tried replicating the pearson hashing function but it’s not perfect. More on that below.

b)
First upload one of the five “stringsToTest” files in the main folder. Then press the “Test Hashing” button. Open the inspector to check the console log output. You can also check the results.png of the results of the strings in strings.png (in main folder).

Every file consists of 216 strings that only have 1 8-bit character that they don’t have in common.

Testing the 5 different files I usually get around 50-100 collisions per file (the array with 255 different possible hash values are shuffled on every page refresh).

Since 50-100 collisions on 216 strings accounts for around 25-50% collisions I think the collision rate is high since I then cannot reliably ensure that the input and output strings are in fact the same, since two different strings many times produces the same hash-value.

I’ve taken inspiration of a python implementation of Pearson hashing [7]. According to the article linked from Wikipedia [8] pearson hashed strings that only differ by 1 character can never collide. This leads me to think that the collisions may be related to the Javascript language or my coding abilities. But since the scope of this assignment is creating custom simple hashing function I am okay with this many collisions for now.

C):

A cryptographic hash function makes it harder to generate a string that would force a collision [9]. It needs to be resistant to Preimage attacks [10], Collision attacks [11], and have an Avalanche effect [12], so that one small change in input makes a big change in the output. Many hashing functions exist without these properties, but without these properties they are less secure.

Here is a quick distilled version of Preimage attacks, Collision attacks and Avalanche effect intended for a reader with no previous knowledge of Computer Science.

Preimage attacks – Cryptographic hash-functions trying to find a message with a specific hash-value.

Collision attack – Find two messages that produce the same hash-value. A good explanation of use-case for this is explained in the Wikipedia article [10], quoted below:

"The usual attack scenario goes like this:

1. Mallory creates two different documents A and B that have an identical hash value, i.e., a collision. Mallory seeks to deceive Bob into accepting document B, ostensibly from Alice.

2. Mallory ***sends document A to Alice****, who agrees to what the document says, signs its hash, and sends the signature to Mallory.*

3. Mallory attaches the signature from document A to document B.

4.*Mallory then* ***sends the signature and document B to Bob****, claiming that Alice signed B. Because the digital signature matches document B's hash, Bob's software is unable to detect the substitution."*

Avalanche effect – A desirable property of a cryptographic algorithm that changes the output significantly even if the input is only changed slightly, for example by a single bit.

Why my custom hash function isn't secure:

According to this article [13] a cryptographic function should be collision resistant among other things to separate it from an unsecure hashing function. By only running the tests above we can immediately discard my custom hash function as a secure cryptographic hash function since we had 25 – 50% collision rate.

Bibliography

[1]
Exploring the Differences Between Symmetric and Asymmetric Encryption - Cyware Social
https://cyware.com/news/exploring-the-differences-between-symmetric-and-asymmetric-encryption-8de86e8a

[2]
Fundamental Difference Between Hashing and Encryption Algorithms - Baeldung CS
https://www.baeldung.com/cs/hashing-vs-encryption

[3]
Unit 1.3.1 Compression, Encryption and Hashing - Wikibooks
https://en.wikibooks.org/wiki/A-level_Computing/OCR/Unit_1.3.1_Compression,_Encryption_and_Hashing#Lossy_Compression

[4]
Differences Between Watermarking and Steganography – Differencebetween.net
http://www.differencebetween.net/business/product-services/differences-between-watermarking-and-steganography/

[5]

Substitution Cipher Solver Tool - Boxentriq
https://www.boxentriq.com/code-breaking/cryptogram

[6]
Cryptograms: New Vision - The thaigi group
http://www.thiagi.com/instructional-puzzles-original/2015/2/13/cryptograms

[7]
Pearson hashing - Wikipedia
https://en.wikipedia.org/wiki/Pearson_hashing

[8]
The universality of iterated hashing over variable-length strings – ScienceDirect
https://www.sciencedirect.com/science/article/pii/S0166218X11004276?via%3Dihub

[9]
What is the difference between cryptographic and noncryptographic hashes? - Quora
https://www.quora.com/What-is-the-difference-between-cryptographic-and-noncryptographic-hashes

[10]
Preimage attack – Wikipedia
https://en.wikipedia.org/wiki/Preimage_attack

[11]
Collision attack - Wikipedia
https://en.wikipedia.org/wiki/Collision_attack

[12]
Avalanche effect – Wikipedia
https://en.wikipedia.org/wiki/Avalanche_effect

[13]
Anderson Dadario - Startup & Information Security Blog
https://dadario.com.br/cryptographic-and-non-cryptographic-hash-functions/