Linnéuniversitetet
Kalmar Växjö

1DV700 - Computer Security

Assignment 1

Encryption

Author: Rashed QAZIZADA
Area: Computer Science
Supervisor: Ola Flygt
Course code: 1DV700
Semester: Autumn 2020

Abstract

This page was intentionally left unchanged.

Keywords

Place your keywords here

Contents

1 Task one 1

1.1 Differences between the following pairs of methods; 1

1.1.1 Symmetric encryption vs Asymmetric encryption 1

1.1.2 Encryption algorithms – Hash algorithms 1

1.1.3 Compression - Hashing 2

1.2 Differences between Steganography, Encryption and Digital Watermarking 2

1.3 Purpose and usage of Steganography 2

1.4 Purpose and usage of Encryption 2

1.5 Purpose and usage of Digital Watermarking 2

2 Different types of Steganography 2

2.1 Explaining how the information is hidden using the tools on this *link* . . 2

2.2 Other ways to hide information using Steganography 3

2.3 Secret.bmp from MyMoodle 4

3 Decryption with/out key 5

3.1 Decrypting the message “RK ERKT EHURMXD” with the given key (cipher line) 5

3.2 Decryption without the key 5

3.3 Another short encrypted message without the key 6

4 Java program implementation 7

5 plaintext.txt 8

6 Perform crypto analysis 8

7 Hash function 9

7.1 Simple hash function and Statistical tests 9

7.2 Difference between normal hash and secure functions 10

Lab Report

1 Task one

1.1 Differences between the following pairs of methods;

1.1.1 Symmetric encryption vs Asymmetric encryption

Symmetric encryption is a type of encryption where we are using exact the same key to encrypt and decrypt the data, such as the DES standard, each pair communication side share a single key that deliver as both an encryption key and a decryption key. Furthermore, this Key needs to be stored securely and if the third party gain access to it. They will be able to decrypt messages passing between the pair.
When exchanging data(sender and receiver) each uses the same key, K, which means that: M = decrypt (K, encrypt(K,M)). Therefore, there must be a secure channel to transfer the key.

Asymmetric encryption key is also known as public key cryptography used by RSA standard. A Asymmetric encryption system assigns each entity a pair of keys. private key and public key. In Private key system,the key must be keep save and nobody should gain access to it. However, Public key can be available for anybody to use.

Symmetric encryption considered as less secure than the Asymmetric encryption. In practice, Symmetric encryption is much faster then Asymmetric encryption because Symmetric uses only one key for both encrypt and decrypt of the data and also the difference between Symmetric and Asymmetric encryption is that with Asymmetric encryption there must two keys for the process. If a third party tries to encrypt data with your public key, they would not be able to decrypt it. The only way to decrypt it is using your private key.

To summarise, with Asymmetric encryption the private key do not need to be transferred. There is no chance to be intercepted by third party. However, Symmetric encryption key must transferred[1].

1.1.2 Encryption algorithms – Hash algorithms

Hashing is a method of one-way encryption , and on the hand, Encryption is a two-way process and uses an algorithm to encode its data. There are two most commonly used types of encryption algorithms Symmetric encryption and Asymmetric encryption.

The main difference between Encryption algorithms , and Hash algorithms is that Hash algorithm uses an algorithm to map data of any size to a fixed length, and this is called a hash value. In addition, Hash algorithm is more useful for storing and retrieving data, like password. If any changes occurs in the data the Hash algorithm will make a significant change in the data which make it harder for the cryptanalysis to steal the data. Whereas, encryption algorithm is used for sending and receiving the data in the form of cipher.

In summary, It all depends on the data integrity that you are sending or receiving, sometimes Hash algorithm can be used over the encryption algorithm or the other way around.

1.1.3 Compression - Hashing

Compression algorithms refer to a technique that is used to reduces the data size or compress data. Reducing the data size reduces the time required for transmission and there are two types of compression Lossy and Lossless. During the compression data can be lost in the Lossy compression technique whereas, all data is retained exactly the same as the original compressed version in Lossless.

Hashing is the process of converting the data into another value known as the hash value. Hash values are impossible to revert back to the original data.

In the final analysis, the key difference is that with Hashing, data can never be decrypted whereas with compression data might be lost, damaged, or stolen. Furthermore, Compressed data can be decrypted depending on the compressed factors However, Hashing cannot be decrypted.

1.2 Differences between Steganography, Encryption and Digital Watermarking

Steganography is an ancient technique for hiding secret data within another file, such as a picture[2]. Whereas, Encryption is the technique that encodes data into an unreadable format(Cipher-text). On the other hand, Digital Watermarking is a process of inserting digital information into digital files such as signals or pictures.

The key difference is that Steganography is invisible whereas, encryption is always visible. However, Digital watermarking can be both visible or invisible.

1.3 Purpose and usage of Steganography

The purpose of Steganography is hiding the existence of data inside a cover(video, audio, or text). It can be used in communication for confidential information.

1.4 Purpose and usage of Encryption

The purpose of encryption is concealing the data by encoding it into unreadable text(cipher text). It can be used to protect private communication, sensitive data so that the data remains hidden from unauthorized parties.

1.5 Purpose and usage of Digital Watermarking

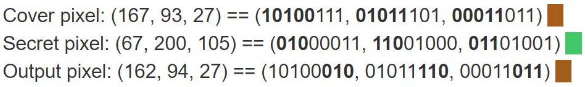
The purpose of digital watermarking is, the act of hiding a message related to a digital signal such as an audio, video or image data. It is mostly used to protect copyright of multimedia data and to identify ownership of the copyright of such signal.

2 Different types of Steganography

2.1 Explaining how the information is hidden using the tools on this *link*

First of all, they explain that each channel (red, green, blue) of each pixel in an image is represented by an 8-bit value[3]. In addition, channel can refer to the sets of numbers inside an image[4].

To hide a secret picture inside another picture or the original picture they replace the least most significant bits of the original picture pixel value with the same number of most significant bits from the secret picture pixel value. Figure 1 is an example of the processing Secret pixel[3].



Cover pixel: (167, 93, 27) == (**10100**111, **01011**101, **00011**011) 🟫
Secret pixel: (67, 200, 105) == (**01**000011, **11**001000, **01**101001) 🟩
Output pixel: (162, 94, 27) == (10100**010**, 01011**110**, 00011**011**) 🟫

Figure 1: Using 3 hidden bits [3]

The output is unidentifiable from the cover colour. However, at this point the output contains approximation of the secret pixel value. To extract the secret pixel pad the missing bits with zeros (**010**00000,**110**00000, **011**00000), the result will be approximately same as the first stage. Furthermore, they stated that, using more least most significant bits results a better quality. However, it makes it easier to spot that the hidden image is there[3]

2.2 Other ways to hide information using Steganography

Steganography can be traced back to 440 BC. Through out the centuries emperors has been widely used Steganography for exchanging secret message in different forms[5].

- In Physical Technique Steganography has been used to hide messages on paper using invisible ink, messages that was sent with a certain rule or key, Messages were written in Morse code on yarn...[5]
- In Digital messages technique hiding messages within the lowest bits of an image, colour, sound.., hiding data within encrypted data, Pictures embedded in video material...[5].
- In Digital text technique making text the same as the original color...[5]
- Concealing an image withing a sound file
- Least significant bit/s
- Social Steganography
- Steganography in streaming media
- Cyber-physical system
- Printed
- network
- using puzzles

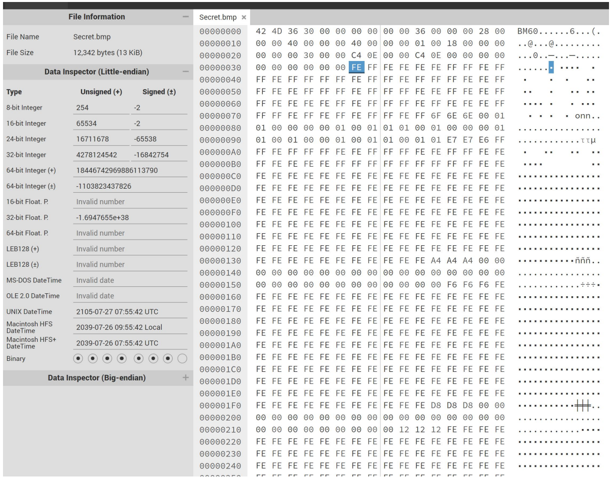
To summarize, there many different ways to hide information using Steganography. It depends the type of information and the technique suit best for the purpose.

2.3 Secret.bmp from MyMoodle

First, I played around with the web link provided in Mymoodle. I understood that the data was hidden in the least most significant bit of the secret.bmp file. Second, I opened the secret.bmp on this url.https://hexed.it/ after some reading and googling I figured it out that the first 54 bytes are the header information. Out of the first 14-bytes are the header, the remaining 40 bytes are DIB header, each of these header has information in it that tells about the make up the of the file. for instance in bitmap file the first 2 bytes will always be in hex 42, 4d is the the identity "BM" in Dump area grid. These 2 bytes tells the computer that this is bitmap file. The next 4 bytes are the size of the bitmap file, which is 36,30,00,00,00,00 etc.

Row 4, column 6, that is the first bit of information or the blue value for this bitmap file. By taking the least most significant bit of the secret.bmp such as the first 8 byte as follow FE = 1111111**0**, FF = 1111111**1**, FE = 1111111**0**, FE = 1111111**0**, FE = 1111111**0**, FE = 1111111**0**, FF = 1111111**1**, FF = 1111111**1** [5]. I managed to reveal the first byte of the secret message. This binary **01000011** is equivalent to Decimal **67**, Hexadecimal of **43** and ASCII/Unicode text character C. I continued converting rest of the Hexadecimals until I reach to FF ≡ 255 ≡ "empty or space" in ASCII/Unicode text and FE ≡ 0 in ASCII/Unicode text NULL [6]. The secret message in secret.bmp file was the word“Congratulations!”.

Automate conversion Analysis



File Information

File Name	Secret.bmp
File Size	12,342 bytes (13 KiB)

Data Inspector (Little-endian)

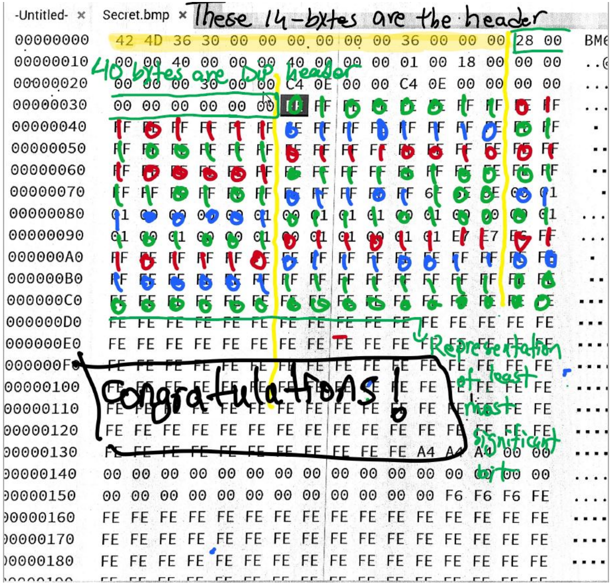
Type	Unsigned (+)	Signed (±)
8-bit Integer	254	-2
16-bit Integer	65534	-2
24-bit Integer	16711678	-65538
32-bit Integer	4278124542	-16842754
64-bit Integer (+)	18446742969886113790	
64-bit Integer (±)	-1103823437826	
16-bit Float. P.	Invalid number	
32-bit Float. P.	-1.6947655e+38	
64-bit Float. P.	Invalid number	
LEB128 (+)	Invalid number	
LEB128 (±)	Invalid number	
MS-DOS DateTime	Invalid date	
OLE 2.0 DateTime	Invalid date	
UNIX DateTime	2105-07-27 07:55:42 UTC	
Macintosh HFS DateTime	2039-07-26 09:55:42 Local	
Macintosh HFS+ DateTime	2039-07-26 07:55:42 UTC	
Binary	● ● ● ● ● ● ● ○	

Data Inspector (Big-endian)

Secret.bmp

```
00000000  42 4D 36 30 00 00 00 00  00 00 36 00 00 00 28 00   BM60......6...(.
00000010  00 00 40 00 00 00 40 00  00 00 01 00 18 00 00 00   ..@...@.........
00000020  00 00 00 30 00 00 C4 0E  00 00 C4 0E 00 00 00 00   ...0..─...─.....
00000030  00 00 00 00 00 00 FE FF  FE FE FE FE FF FF FE FF   ......▪ ▪▪▪▪  ▪ 
00000040  FF FE FF FF FF FF FE FF  FF FE FF FF FF FE FE FF    ▪    ▪  ▪   ▪▪ 
00000050  FF FE FE FF FF FF FE FF  FF FF FE FE FF FE FE FF    ▪▪   ▪   ▪▪ ▪▪ 
00000060  FF FE FE FE FE FF FE FF  FF FF FE FF FE FE FE FF    ▪▪▪▪ ▪   ▪ ▪▪▪ 
00000070  FF FF FE FF FE FF FE FF  FF FE FF 6F 6E 6E 00 01     ▪ ▪ ▪  ▪ onn..
00000080  01 00 00 00 00 01 00 01  01 01 00 01 00 00 00 01   ................
00000090  01 00 01 00 00 01 00 01  01 00 01 01 E7 E7 E6 FF   ............ττµ 
000000A0  FF FE FF FF FF FE FE FF  FF FF FE FE FF FF FE FE    ▪   ▪▪   ▪▪  ▪▪
000000B0  FF FE FE FE FE FF FF FF  FF FF FF FF FF FF FE FE    ▪▪▪▪         ▪▪
000000C0  FE FE FE FE FE FE FE FE  FE FE FE FE FE FE FE FE   ▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪
000000D0  FE FE FE FE FE FE FE FE  FE FE FE FE FE FE FE FE   ▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪
000000E0  FE FE FE FE FE FE FE FE  FE FE FE FE FE FE FE FE   ▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪
000000F0  FE FE FE FE FE FE FE FE  FE FE FE FE FE FE FE FE   ▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪
00000100  FE FE FE FE FE FE FE FE  FE FE FE FE FE FE FE FE   ▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪
00000110  FE FE FE FE FE FE FE FE  FE FE FE FE FE FE FE FE   ▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪
00000120  FE FE FE FE FE FE FE FE  FE FE FE FE FE FE FE FE   ▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪
00000130  FE FE FE FE FE FE FE FE  FE FE FE A4 A4 A4 00 00   ▪▪▪▪▪▪▪▪▪▪▪ñññ..
00000140  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ................
00000150  00 00 00 00 00 00 00 00  00 00 00 00 F6 F6 F6 FE   ............÷÷÷▪
00000160  FE FE FE FE FE FE FE FE  FE FE FE FE FE FE FE FE   ▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪
00000170  FE FE FE FE FE FE FE FE  FE FE FE FE FE FE FE FE   ▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪
00000180  FE FE FE FE FE FE FE FE  FE FE FE FE FE FE FE FE   ▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪
00000190  FE FE FE FE FE FE FE FE  FE FE FE FE FE FE FE FE   ▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪
000001A0  FE FE FE FE FE FE FE FE  FE FE FE FE FE FE FE FE   ▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪
000001B0  FE FE FE FE FE FE FE FE  FE FE FE FE FE FE FE FE   ▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪
000001C0  FE FE FE FE FE FE FE FE  FE FE FE FE FE FE FE FE   ▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪
000001D0  FE FE FE FE FE FE FE FE  FE FE FE FE FE FE FE FE   ▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪
000001E0  FE FE FE FE FE FE FE FE  FE FE FE FE FE FE FE FE   ▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪
000001F0  FE FE FE FE FE FE FE FE  FE FE FE D8 D8 D8 00 00   ▪▪▪▪▪▪▪▪▪▪▪╫╫╫..
00000200  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ................
00000210  00 00 00 00 00 00 00 00  00 12 12 12 FE FE FE FE   ............▪▪▪▪
00000220  FE FE FE FE FE FE FE FE  FE FE FE FE FE FE FE FE   ▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪
00000230  FE FE FE FE FE FE FE FE  FE FE FE FE FE FE FE FE   ▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪
00000240  FE FE FE FE FE FE FE FE  FE FE FE FE FE FE FE FE   ▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪▪
```

Figure 2: Hexed conversion [5]


-Untitled-
Secret.bmp
These 14-bytes are the header
40 bytes are DIP header
Representation of least most significant bit
Congratulations!

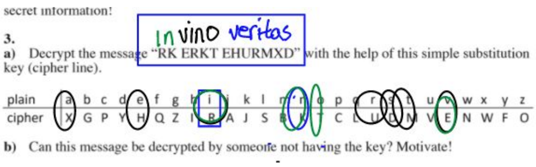
```
00000000  42 4D 36 30 00 00 00 00 00 00 36 00 00 00 28 00   BM6
00000010  00 00 40 00 00 00 40 00 00 00 01 00 18 00 00 00   ..@
00000020  00 00 00 30 00 00 C4 0E 00 00 C4 0E 00 00 00 00   ...
00000030  00 00 00 00 00 00 FE FF FE FE FE FE FF FF FE FF   ...
00000040  FF FE FF FF FF FF FE FF FF FE FF FF FF FE FE FF   .
00000050  FF FE FE FF FF FF FE FF FF FF FE FE FF FE FE FF   ..
00000060  FF FE FE FE FE FF FE FF FF FF FE FF FE FE FE FF   ..
00000070  FF FF FE FF FE FF FE FF FF FE FF FF FE FE 00 01   .
00000080  01 00 00 00 00 01 00 01 01 01 00 01 00 00 00 01   ...
00000090  01 00 01 00 00 01 00 01 01 00 01 01 E7 E7 E6 FF   ...
000000A0  FF FE FF FF FF FE FE FF FF FF FE FE FF FF FE FE   .
000000B0  FF FE FE FE FE FF FF FF FF FF FF FF FF FF FE FE   ..
000000C0  FE FE FE FE FE FE FE FE FE FE FE FE FE FE FE FE   ...
000000D0  FE FE FE FE FE FE FE FE FE FE FE FE FE FE FE FE   ...
000000E0  FE FE FE FE FE FE FE FE FE FE FE FE FE FE FE FE   ....
000000F0  FE FE FE FE FE FE FE FE FE FE FE FE FE FE FE FE   ....
00000100  FE FE FE FE FE FE FE FE FE FE FE FE FE FE FE FE   ....
00000110  FE FE FE FE FE FE FE FE FE FE FE FE FE FE FE FE   ....
00000120  FE FE FE FE FE FE FE FE FE FE FE FE FE FE FE FE   ....
00000130  FE FE FE FE FE FE FE FE FE FE FE A4 A4 A4 00 00   ....
00000140  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ....
00000150  00 00 00 00 00 00 00 00 00 00 00 00 F6 F6 F6 FE   ....
00000160  FE FE FE FE FE FE FE FE FE FE FE FE FE FE FE FE   ....
00000170  FE FE FE FE FE FE FE FE FE FE FE FE FE FE FE FE   ....
00000180  FE FE FE FE FE FE FE FE FE FE FE FE FE FE FE FE   ....
```

Figure 3: Revealed Message calculations [5]

3 Decryption with/out key

3.1 Decrypting the message "RK ERKT EHURMXD" with the given key (cipher line)

The decryption message was IN VINO VERITAS or "In wine, there is truth". The decryption was easy. I just match the cipher with the plain I revealed the message.



secret information!

3.
a) Decrypt the message "RK ERKT EHURMXD" with the help of this simple substitution key (cipher line).

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

b) Can this message be decrypted by someone not having the key? Motivate!

Figure 4: Revealed Message

3.2 Decryption without the key

The decryption process without the key would take a lot of time but it is possible to decrypt the message.

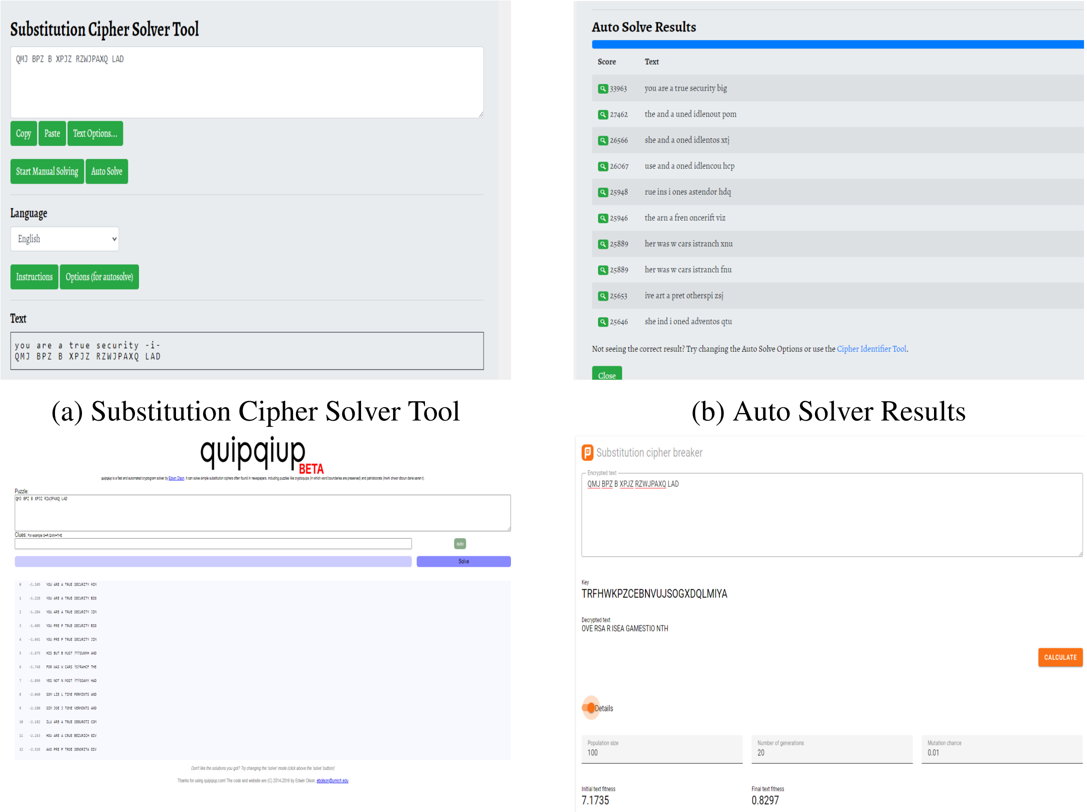
First, try to understand the decrypted message and use the most complicated way to guess the cipher line. Second, use the frequency of the using the regularities of the language. For instance the letter E is the most used one among the other letters in English language. For this decrypted message it is difficult to find a candidate for the letter E or T the second possible candidate. It does not make huge different even the letter R appeared 3 times.

To summarize, decryption without the key would be difficult. The decrypted message could be any language. Therefore, you must look for all possible way to break the code.

3.3 Another short encrypted message without the key

My all attempts to decrypt the text failed. I tried the Crypto analysis of English language manually. However, this encrypted message was difficult to come up with guessing letters like on this encrypted message "QMJ BPZ B XPJZ RZWJPAXQ LAD". There are 3 possible letters that have appeared 3 times J, P, and Z they could be a candidate for letters E, T, O or N. I tried to write all the characters in substitution method 0 to +3 and 0 to -3 and still could not make anything meaningful out of it.

Besides, my manual analyses I also searched some other automated tools and the results are as following *1. CryptogramSolver.com* , *2. guballa.com* , *3. quipquip.com* , *4. planetcalc.com* , and *5. cryptiidecoder.com*



```
QMJ BPZ B XPJZ RZWJPAXQ LAD
```

```
you are a true security -i-
QMJ BPZ B XPJZ RZWJPAXQ LAD
```

(a) Substitution Cipher Solver Tool

Auto Solve Results

Score	Text
33963	you are a true security big
27462	the and a uned idlenout pom
26566	she and a oned idlentos xtj
26067	use and a oned idlencou hcp
25948	rue ins i ones astendor hdq
25946	the arn a fren oncerift viz
25889	her was w cars istranch xnu
25889	her was w cars istranch fnu
25653	ive art a pret otherspi zsj
25646	she ind i oned adventos qtu

Not seeing the correct result? Try changing the Auto Solve Options or use the Cipher Identifier Tool.

(b) Auto Solver Results

quipqiup is a fast and automated cryptogram solver by Edwin Olson. It can solve simple substitution ciphers often found in newspapers, including puzzles like cryptoquips (in which word boundaries are preserved) and patristocrats (inwhi chwor dboun darie saren t).

Puzzle:

```
QMJ BPZ B XPJZ RZWJPAXQ LAD
```

Clues: For example G=R QVW=THE

```
0   -1.105  YOU ARE A TRUE SECURITY HIM
1   -1.228  YOU ARE A TRUE SECURITY BIG
2   -1.284  YOU ARE A TRUE SECURITY JIM
3   -1.605  YOU PRE P TRUE SECURITY BIG
4   -1.661  YOU PRE P TRUE SECURITY JIM
5   -1.675  HIS BUT B MUST ?T?SUNMH AND
6   -1.748  FOR WAS W CARS ?S?RAHCF THE
7   -1.899  YES NOT N MOST ?T?SOAMY HAD
8   -2.040  SOM LIE L TIME PERMINTS AND
9   -2.180  SIM JOE J TOME VERMONTS AND
10  -2.182  ILU ARE A TRUE SEGUROTI COM
11  -2.243  HOU ARE A CRUE BEZURICH GIV
12  -2.526  AWO PRE P TROE SENORITA DIV
```

Don't like the solutions you got? Try changing the 'solve' mode (click above the 'solve' button)!

Thanks for using quipqiup.com! The code and website are (C) 2014-2016 by Edwin Olson, ebolson@umich.edu

(c) 12 possible match

Encrypted text: QMJ BPZ B XPJZ RZWJPAXQ LAD

Key: TRFHWKPZCEBNVUJSOGXDQLMIYA

Decrypted text: OVE RSA R ISEA GAMESTIO NTH

Population size: 100

Number of generations: 20

Mutation chance: 0.01

Initial text fitness: 7.1735

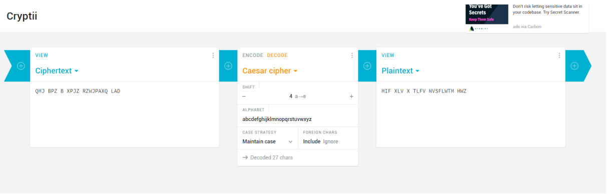
Final text fitness: 0.8297

(d) 20 possible match

Figure 5: Statistical tests

Cryptii Ciphertext decoder makes it crystal clear that this not English ciphertext. Because the brute-force cryptanalysis tried 25 possible keys. Still the text was unreadable "encrypted". Apart from all other approaches I made, there are other possible decryption methods available which is out of scope of this course like Monoalphabetic Cipher, Polyalphabetic Cipher.

To conclude, the encrypted text could be in another language, I guess, it be could Spanish, Or the other possibility is the message was encrypted using combine both methods.



Cryptii

Ciphertext: QMJ BPZ B XPJZ RZWJPAXQ LAD

ENCODE DECODE — Caesar cipher

SHIFT: 4 a→e

ALPHABET: abcdefghijklmnopqrstuvwxyz

CASE STRATEGY: Maintain case

FOREIGN CHARS: Include Ignore

→ Decoded 27 chars

Plaintext: MIF XLV X TLFV NVSFLWTM HWZ

VIEW
VIEW
You've Got Secrets
Keep Them Safe
Don't risk letting sensitive data sit in your codebase. Try Secret Scanner.
ads via Carbon

Figure 6: Ciphertext decoder Cryptii

4 Java program implementation

I am implementing the two simple encryption and decryption methods Substitution and Transposition[1]. For Substitution I use Caesar Cipher and for Transposition I use simple permutation.

Caesar Cipher encryption, one of the most used Substitution technique that substitute one character at the time. The Caesar Cipher Substitution method will replaced each letter by adding the given key to their ASCII value. such as **“1DV700 computer security HT-20”**. For this encryption the key value is 2. The Caesar Cipher Substitution method encrypts the text as follow **“3FX922"eqorwvgt"ugewtkva"JV/423”**.
The Caesar Cipher Substitution method check the first char and the first char is numeric value and it is a number then the Substitution method add the key value to its ASCII value. Now the result should be 2 which is correct because the ASCII value of 1 according to *asciitable.com* is **49**. Hence, 49+2 = 51 and 51 is 3 according to *asciitable.com* . The second char is D and it is a letter then the key is added to the letter D’s ASCII value. Now the letter should be F because ASCII value of D is 44 and 44 + 2 the key value = 46 and 46’s ASCII value is F which is correct.

For Transposition technique, I am using simple permutation one of the most used transposition technique. This technique is an alternative to substitution ciphers. Instead of changing the coding of the characters (block) in the plaintext, it rearranges the text and also the effect is that cipher text and the plaintext contains the same symbols [7]
The simple permutation method divides the plaintext into blocks and decides on a permutation order then it rearrange the blocks according the given key. for instance if the plaintext is “1DV700 computer security HT-20”, the key is 1423 and the Cipher text the Java method output as the following “7VD1 02-TH ytiruces retupmoc 00”
Note: The encryption implementation is limited and it accepts keys between 0-255 numbers as the key value.

For the decryption method, for both Substitution and Transposition you must know the key to be able to decrypt the file. Both Substitution and Transposition methods will try to substitute the key the user provides. In all time the decryption method will try to decrypt the file, regardless of the valid or invalid key. If the file was encrypted with more than 8-bits neither the substitution method nor the transposition method could decrypt the file. Furthermore, the time complexity depends on the length of the key even for this 8-bit key length implementation it will take a lot time to find the right key. It might take quite some time to decrypt it.

[1] How the program runs! The program will ask the user to choose the file path then one of the two Option 1. Encryption 2. Decryption and third step the user must select methods of their choice 1. Substitution 2. Transposition. Forth step the user must select where to save the file. Note the file type must be **.txt** and the final step is to enter an encryption key between 0 to 255 integer value.

The Substitution decryption method will substitute every letter by subtracting the key with the character's ASCII value. The process is the same as the encryption. However, it only differs when dealing with the ASCII value. Here the method is taking the key and subtracting it from the character's ASCII value. For instance **"3FX922"eqorwvgt" ugewtkva"JV/423"** for this encryption the key value is 2. To decrypt the file the method takes one character at the time and subtracting it with key. Hence, 3 is the first character and its ASCII value is 51. Now, 51 - 2 = 49, 49 is 1 according the *asciitable.com* so then end result will be as follow **"1DV700 computer security HT-20"**.

The Transposition simple permutation method divides the plaintext into blocks and decides on a permutation order according the key. Then it rearrange the blocks.

5 plaintext.txt

Secret message has been posted a page long with the Encrypted version on Mymoodle on my name "Rashed Qazizada" and I encrypted the Plaintext using Substitution method with the key number 20.

6 Perform crypto analysis

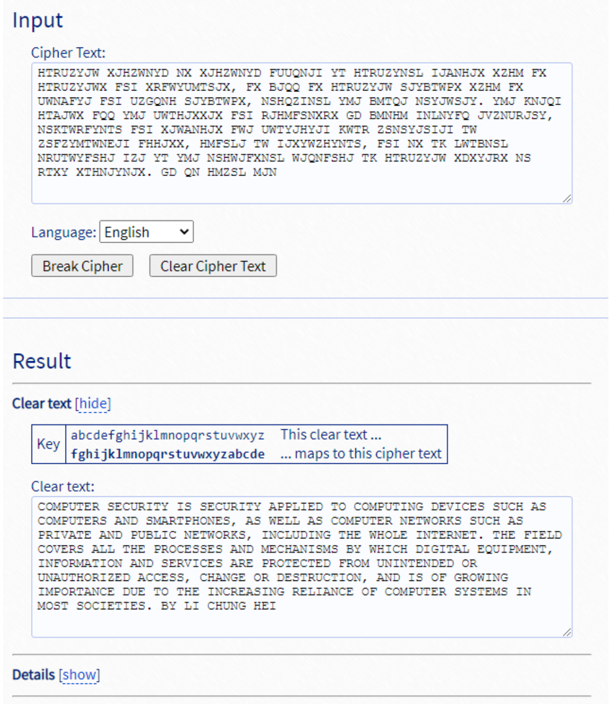
First step, I had to analysis the type of encryption by checking if the encryption is a Plaintext, or what kind of encryption algorithm is used?, what it could be the possible key?, what kind of Ciphertext it is?, is it Substitution or Transposition?. Furthermore, I checked the type of operations used for transforming plaintext to Ciphertext. Is it Symmetric or Asymmetric?. If it is Substitution technique?. what type of Substitution, is it Caesar Cipher?, is it Monoalphabetic?,Playfair cipher or polyalphabetic cipher?. On the other hand, I also checked if the Cipher text was encrypted using transposition techniques? and if it is what type of transposition, is it simple permutation? or other type of Transposition or even I checked that it could be a combination of both methods. finally, there was no other possible technique left for me not to check the encrypted files, therefore, I manage to decrypt 3 students encrypted files.

Second technique I tried was that I copy paste the Cipher text on *substitution method solver code breaker* and I manage to find the possible key.

Final step, I executed the Cipher text using my Substitution decryption method to break the Cipher text. The first, Ciphertext I manage to decrypt belongs to Adam Rashdan with the key number 7. I kept trying to break the code with the substitution method. As there is only 26 possible combination for English letter so the method broke the ciphertext in 7th tries starting from 0.

The second, Cipher text I manage to decrypt belongs Munish Sharma with the key 3. After the 4th try starting from 0 the ciphertext broke and the secret message was revealed ,and the third Ciphertext I manage to decrypt belongs to Li Chung Hei with the key 5. This Ciphertext was the easiest one to break, I did not do much analyses. I Just copy pasted the ciphertext on *guballa.com* then I figured it out that the letter H is C and there is only 5 letters shift and then I put the key 5 in my substitution method program and I managed to break it.

To conclude, for all the ciphertext I did the manual analyses and then I tried to inputted in sites like *1. CryptogramSolver.com* , *2. guballa.com* , *3. quipquip.com* , *4. planetcalc.com* , and *5. cryptiidecoder.com* . Trying these website gave me a hint and then I ran it into my Substitution methods multiple times to get the right key. I could break some of the ciphertext. In total I tried 8 students' ciphertext and I managed to break 3.



Cipher Text:

```
HTRUZYJW XJHZWNYD NX XJHZWNYD FUUQNJI YT HTRUZYNSL IJANHJX XZHM FX
HTRUZYJWX FSI XRFWYUMTSJX, FX BJQQ FX HTRUZYJW SJYBTWPX XZHM FX
UWNAFYJ FSI UZGQNH SJYBTWPX, NSHQZINSL YMJ BMTQJ NSYJWSJY. YMJ KNJQI
HTAJWX FQQ YMJ UWTHJXXJX FSI RJHMFSNXRX GD BMNHM INLNYFQ JVZNURJSY,
NSKTWRFYNTS FSI XJWANHJX FWJ UWTYJHYJI KWTR ZSNSYJSIJI TW
ZSFZYMTWNEJI FHHJXX, HMFSLJ TW IJXYWZHYNTS, FSI NX TK LWTBNSL
NRUTWYFSHJ IZJ YT YMJ NSHWJFXNSL WJQNFSHJ TK HTRUZYJW XDXYJRX NS
RTXY XTHNJYNJX. GD QN HMZSL MJN
```

Language: English

Key	abcdefghijklmnopqrstuvwxyz	This clear text ...
	fghijklmnopqrstuvwxyzabcde	... maps to this cipher text

Clear text:

```
COMPUTER SECURITY IS SECURITY APPLIED TO COMPUTING DEVICES SUCH AS
COMPUTERS AND SMARTPHONES, AS WELL AS COMPUTER NETWORKS SUCH AS
PRIVATE AND PUBLIC NETWORKS, INCLUDING THE WHOLE INTERNET. THE FIELD
COVERS ALL THE PROCESSES AND MECHANISMS BY WHICH DIGITAL EQUIPMENT,
INFORMATION AND SERVICES ARE PROTECTED FROM UNINTENDED OR
UNAUTHORIZED ACCESS, CHANGE OR DESTRUCTION, AND IS OF GROWING
IMPORTANCE DUE TO THE INCREASING RELIANCE OF COMPUTER SYSTEMS IN
MOST SOCIETIES. BY LI CHUNG HEI
```

Figure 7: Substitution Solver

7 Hash function

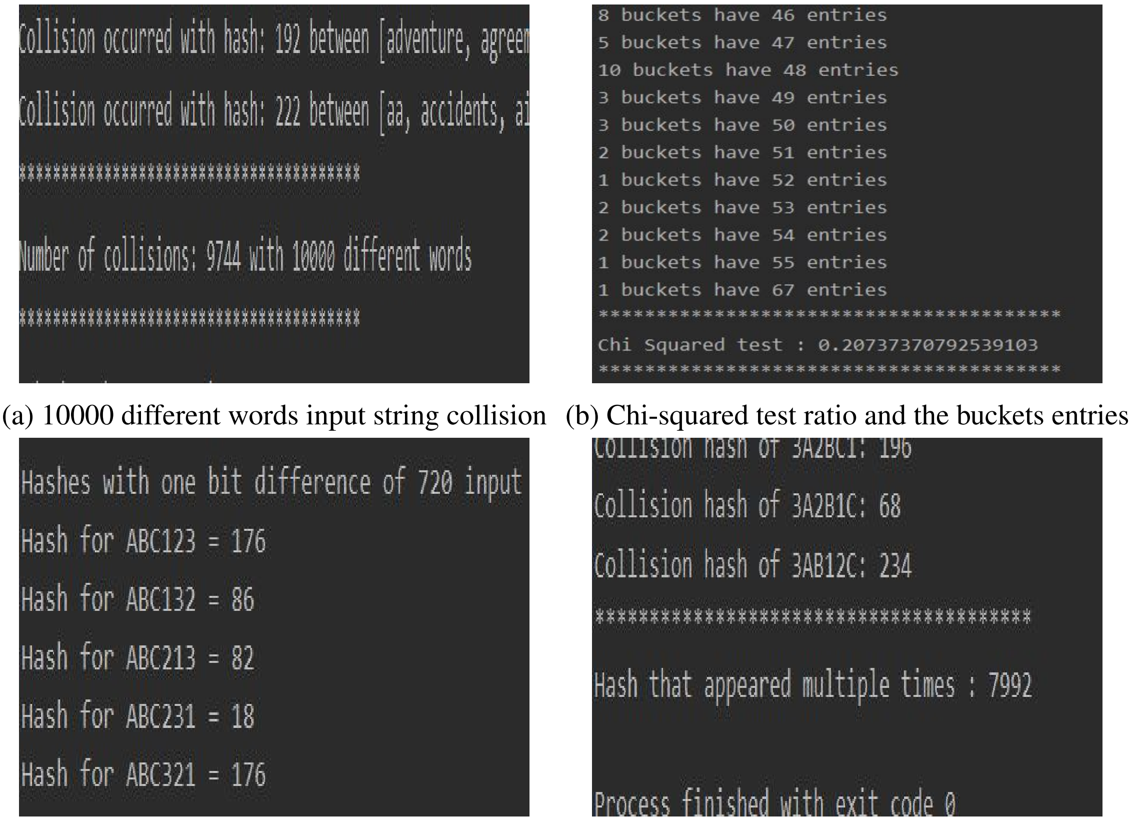
7.1 Simple hash function and Statistical tests

I have implemented a simple hash function that accepts different input size and produces a hash value of 8-bits. By taking each character ASCII value in the string, then XOR each byte of the character, second multiply it by prime number and finally, the result or the return value will be always a positive integer hash value.

My statistical implementation tests: The first hash function"diffHashValue" that takes an array of strings and run tests for uniformity. When testing a hash function, according to Wikipedia the uniformity of the distribution of hash values can be evaluated by chi-squared test with the ratio within the interval of (0.95 - 1.05) [8]. However, the "diffHashValue" function has ratio within the interval of **0.207**. Therefore, "diffHash-Value" hash function can be evaluated uniform distribution. The second hash function "one1000InputTry" that tests 720 Swedish Registration certificates with the input strings that are very similar. It only differ in one bit. Running 720 tests with input strings on this hash function the main distribution I get is the collision of hash values.

When I analyse the results I can say that these hash functions are not secure enough because they do not produce a fixed length output depending on whatever the input is. On the other hand, a good has function should map the expected inputs as evenly as possible.

To summarize, For 10000 different word input strings there is 9744 collisions out of 10000 input strings. In mean time, for 720 word input strings with one bit different there are 7992 collision. The reason is that the number collisions pairs of inputs increases..



```
Collision occurred with hash: 192 between [adventure, agreem
Collision occurred with hash: 222 between [aa, accidents, ai
***************************************
Number of collisions: 9744 with 10000 different words
***************************************
```

(a) 10000 different words input string collision

```
8 buckets have 46 entries
5 buckets have 47 entries
10 buckets have 48 entries
3 buckets have 49 entries
3 buckets have 50 entries
2 buckets have 51 entries
1 buckets have 52 entries
2 buckets have 53 entries
2 buckets have 54 entries
1 buckets have 55 entries
1 buckets have 67 entries
***************************************
Chi Squared test : 0.20737370792539103
***************************************
```

(b) Chi-squared test ratio and the buckets entries

```
Hashes with one bit difference of 720 input
Hash for ABC123 = 176
Hash for ABC132 = 86
Hash for ABC213 = 82
Hash for ABC231 = 18
Hash for ABC321 = 176
```

(c) 720 input strings

```
Collision hash of 3A2BC1: 196
Collision hash of 3A2B1C: 68
Collision hash of 3AB12C: 234
***************************************
Hash that appeared multiple times : 7992

Process finished with exit code 0
```

(d) 7992 Hash appeared multiple times.

Figure 8: Statistical tests

7.2 Difference between normal hash and secure functions

There is a difference between normal hash functions and secure has functions. What is that difference? Prove that your hash function is not a secure hash function. What is the easiest way to prove this? The main difference between normal hash function and secure hash function is that the Simple hash function has less properties then secure hash function. First, I am trying to explain Simple hash function and secure hash function so that I can prove the my hash function not a secure hash function.

Simple hash function or normal hash function Should be able to take the message and put it into a matrix table and it should fill one block in the matrix table see the Simple hash function figure. If we fill one block and it is not enough, the table should use the next block. Then it should compute bitwise XOR on the column at the same time to create the hash code at the end of the table. Regardless of the size of message, simple hash function should add a block and it should create a fixed output even if we input different values. This simple hash function will create different hash values. However, it will have collisions.
In addition, if we try to make this simple hash function more complex we can do that as well. For instance, we can do One-bit circular shift on the hash value after each block is processed, this would improve the code however, it will be still a very simple hash

function. On the other hand, for hash function to be secure we require more properties and its purpose of the hash function is to produce a “fingerprint”.
If we consider the following six properties that are needed for secure hash function and then compare it with simple hash function we can easily prove that why our simple hash function is not secure.

1. It should be able to use any data size[7].

    - The simple hash function can handle any data size too. so we can add just add more rows

2. It should produce a fixed length output[7].

    - The simple hash function can handles this property as well any data size too. so we can add just add more rows

3. The hash value should be easy to compute for any given value message[7].

    - The simple hash function meet this requirements as well

4. For any given hash value “h”, it should be computationally infeasible to find x such that H(x) =h. It is almost impossible to find a message that will give the same hash code, So we cannot come up with a message that will give a certain hash value in secure hash function. However, this properties does not meet the Simple hash function because in the Simple hash function it is possible to create a hash value of similar. I can referrer to my java implementation where I inputted 720 words that give 7992 collision which means that we had 7992 similar hash values[7].

    - The simple hash function does not meet this requirements.

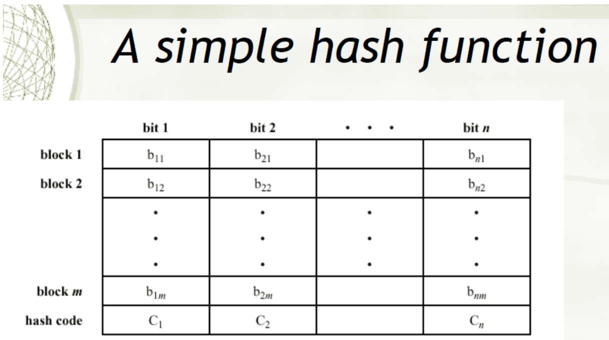
5. For any given block x, it is computationally infeasible to find y $\neq$ x with H(y) = H(x). What this property trying to prove is that if we have a message and this property gives a certain hash value for the given message and this property make sure that we won’t be able to find another one that has the same hash value it is almost impossible to find collisions. However, we know that there will be a lot of collisions because the hash code is small but to find a collision it is infeasible. It needs a lot of efforts to find collision. For instance, if we would try the brute-force approach and just test different messages, we might find that it works but it will take million of years to do that and that is why it is computational infeasible[7].

    - The simple hash function does not meet this requirements.

6. It is computationally infeasible to find any pair (x,y) such that H(x) = H(y). This property prove that we cannot come up with any pair that has the same hash value, there will be no collision. Obviously our simple hash function approach does not meet for 4,5 and 6 properties [7].

There are number of hash function that has been developed that more or less secure. like SHA, MD4....

To conclude, the hash value in no mean should be reversible. The simple hash function does not handle all these properties. I can easily say that it is not a secure hash function. To have a secure function there must be at least these six properties of secure function. End of the Assignment 1



A simple hash function

	bit 1	bit 2	• • •	bit *n*
block 1	b_{11}	b_{21}		b_{n1}
block 2	b_{12}	b_{22}		b_{n2}
	⋮	⋮	⋮	⋮
block *m*	b_{1m}	b_{2m}		b_{nm}
hash code	C_1	C_2		C_n

Figure 9: Ola Flygt Course material snipped shot

References

[1] D. E. Comer, "Computer networks and internets," 2015.

[2] NIST, "How does steganography work and does it threaten enterprise data?" *revision 4 of SP 800-53*, 2013.

[3] J. Stanley, "Image steganography." [Online]. Available: https://incoherency.co.uk/image-steganography/ [Kontrollerad: 17.11.2020]

[4] M. S. Limited, "Images and channels." [Online]. Available: https://networkencyclopedia.com/synchronous-transmission/ [Kontrollerad: 17.11.2020]

[5] HexEd.it, "Data inspector bitmap to hex value." [Online]. Available: https://hexed.it/ [Kontrollerad: 17.11.2020]

[6] ASCIITABLE.COM, "Ascii table and description." [Online]. Available: http://www.asciitable.com/ [Kontrollerad: 17.11.2020]

[7] O. Flygt, "Encryption, lecture 2," 2020.

[8] Wikipedia, "Hash function." [Online]. Available: https://en.wikipedia.org/wiki/Hash_function#Properties [Kontrollerad: 20.11.2020]