Linnaeus University
1dv700 - Computer Security
Assignment 1

Student: Vilhelm Park

Personal number: 960720-8536

Student ID: vp222dv@student.lnu.se

Setup Premises

For this assignment I am using the OS Windows 10, the web browser Mozilla Firefox, for the assignment where we are looking for the hidden information in the image secret I used the online hex editor on onlinehexeditor.com. For the coding, I used Intellij Idea and for the text files where I output my encryptions and decryptions, I used notes that exist in windows.

Task 1

Part a

Symmetric encryption - Asymmetric encryption

There are some differences between symmetric and asymmetric encryption, one would be that in symmetric encryption you use one secret key that will be used for encryption and decryption, this key will be shared between the people sending the messages[1]. Asymmetric encryption, on the other hand, uses two keys, one key is public, everyone can get it and the other key is a secret that will be given to the recipient. The secret key is used to decrypt the messages from the sender. Another difference is the fact that symmetric encryption is an old method while asymmetric encryption is fairly new[2].

Encryption algorithms – Hash algorithms

Encryption algorithms could be for example symmetric encryption and asymmetric encryption; they make sure that the content of files can't be read by anyone else but the sender and the receiver [3].
Hash algorithms could be MD4 (outdated) or SHA for example, their purpose is to compare large amounts of data to see if the contents are the same before and after some it has been sent. If there are any changes in the content the hash output would not be the same as before it was modified so it's a way to make sure that no one unwanted changes the content in important data [4].
The difference between encryption and hash algorithms is that encryption mainly hides content such as files and texts, while hash is used to validate contents such as files and texts to see if there have been any unwanted modifications in them. They are also different since the encryption is a two-way function since it both encrypt data and then decrypts it, while hash is a one-way function since it only makes an irreversible hash output from the data.

Compression - Hashing

Compression reduces the size of files by removing data, there is for example Lossy compression and Lossless compression. Lossy compression is used to reduce data size for storing and transmitting data, it's mainly used to compress multimedia data. Lossy compression is also called irreversible compression since after the compression the original file can't be recovered since certain data from the original file has been discarded and is lost. Lossless compression on the other hand can compress data and then reconstruct it in its original form, but it can't compress as much as the Lossy compression method [5].
Hashing as described earlier is a one-way function that creates a hash output based on data in a file for example. It's used to validate the content and see if it's delivered as it was meant to be.
I think there are a lot of differences between these two terms since one is used to compress data in files and one is used to validate the content in files after deliveries. Rather I don't think I can see any similarities besides the fact that they use some type of algorithm to make

each of them work. I guess one similarity could be that if you use Lossy compression after they have done the compression the original file can't be recovered which is the same as with hash output it can't be reversed into its original data.

Part b

Steganography is a technique to conceal messages, files, and more. The idea is that steganography will hide the message and the fact that a message is hidden. An example of this is that you store a file or message in the lowest bits in bytes in an image file. This will make it impossible for a human eye to notice when looking at the image file since the changes will be so small. Steganography is a useful method when for example two companies try to communicate with each other about some secret new product that will revolutionize the market, but the communication between the companies is not secure. This is where steganography comes in since even if the communication lines are not secure the people trying to get the secret information won't be able to find it since it's hidden in plain view and as long the "enemies" don't know what method is used the information will never be leaked [6].

Encryption has been described before so it will only be briefly described. The purpose of encryption is to encrypt files, messages so that no person acquiring the information when it's sent can understand what it means. Only the receiver can decrypt the information and acquire the information. Encryption is mainly used when sending sensitive information over the internet where the communication line is not secure. It can also be used when there is some sensitive information stored on a personal computer to make sure that the data is encrypted in case the computer gets stolen.

Digital watermarking is a method to identify ownership in digital media, information is usually hidden digitally in a carrier signal. Digital watermarking uses steganographic methods where they hide the carrier signals in noisy signals. Digital watermarking is regularly used to trace copyright infringement such as illegally copied movies and music [7].

The three methods stated above have some similarities in the fact that they try to hide information from perpetrators that want to acquire sensitive information or products. The difference between the three is their methods on how to hide information. Encryption does not bother to hide the fact that the data is encrypted, it is easy to see when a file is encrypted when you compare it to regular files that are not. Steganography and digital watermarking both want to hide the fact that signals or messages are in the files or products, so in that way they are similar. They do focus on different things, stenography focuses on hiding the information in plain sight while digital watermarking focuses on making it robust and hard to intercept. Digital watermarking is the most difference when it comes to how it's used, while the other methods focus on sending messages to receivers digital watermarking is used as a tool to find out if products are used illegally.

Task 2

Part a

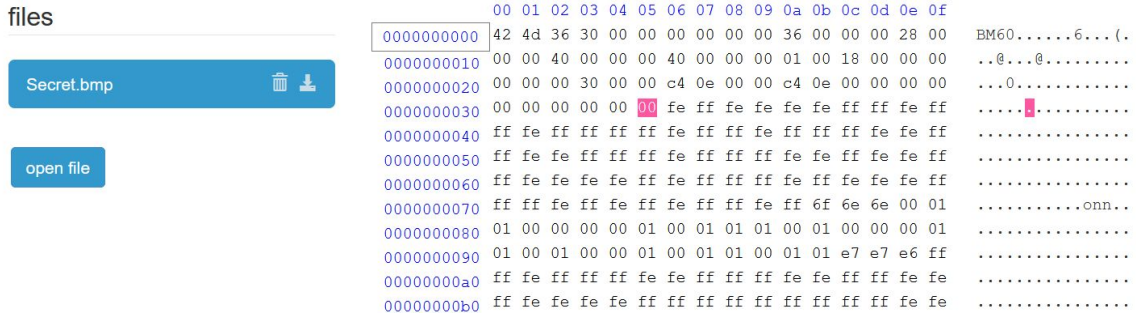
In a pixel there are 3 bytes that represent them, they have 3 different pixels one that is the cover pixel, one that is the secret pixel, and then finally the output pixel. They replace an x amount of the least bits in the bytes in the cover pixel with the same amount of the highest bits from the secret pixel and then we get the output pixel. The difference will be really small on a single pixel. When done to a bigger amount of pixels the quality of the picture that we want to be secret will increase, which is not that good since it will also increase the visibility of the secret picture which goes against the steganographic purpose.

Part b

A long time ago before digital devices existed they used steganography in a different way, one classic example is that a person had a message written on their bald head and then they grew the hair back and went to the person the message was for and shaved their head to show the message. That's however not a method that is used anymore for obvious reasons. One steganography method is to hide digital text inside a document by setting the text color in the same color as the background. Another method is Echo Steganography which is basically that you have the echo of a sound file which you modify and put information inside the modifications.

Part c

This is how the image looked when it was converted to hexadecimal. You can see where the secret message starts, it starts after the marked hexadecimal 00 and the end is at the last row after the fifth column. To find this section I looked for parts where there were a lot of ones and zeros in a section, when looking at all the hexadecimal values this section was the only one that I could clearly see was not like everything else. It was a good mix of ones and zeros compared to the rest of the values which was either a lot of zeros in a row or a lot of ones in a row, which does not give me the impression that they were not a part of the secret message.



C-01000011 **O**-01101111 **N**-01101110 **G**-01100111 **R**-01110010 **A**-01100001 **T**-01110100 **U**-01110101 **L**-01101100 **A**-01100001 **T**-01110100 **I**-01101001 **O**-01101111 **N**-01101110 S-01110011 !-00100001
Message - Congratulations!

Inside the secret.bmp there are some hexadecimal numbers that are hidden inside it, so to get the hexadecimal values I used an online hexadecimal editor and input the image to get the values we need. In the section described above, there will be some hexadecimal values where there is a secret message hidden.
I will use the method described in the steganography document, the method is as follows the final bit in the bytes we get from the hexadecimal values will either be one or zero and by putting these bits into new bytes we can get the secret message [6]. Using this method in the section we described earlier we will get multiple bytes that we convert into ASCII using a byte to ASCII converter. The result of the ASCII characters we get can be seen above, the secret message was “Congratulations!”.

Task 3

Part a

cipher: RK ERKT EHURMXD
plain: in vino veritas
In English: In wine, there is truth.

Part b

It is possible to decrypt the message without the key using the brute force method since there are only 25 letters, the problem of doing this is that you will end up with multiple keys and you have to keep going to get the correct key for the encryption. We basically have 26! alternatives, it is possible to decrypt the message, but extremely time-consuming. One problem could be that the message is in an unknown language so it's harder to determine what message is real. Another way to decrypt the message is by using cryptanalysis which means that you use to analyze how many times a letter is used in a message and then you can compare it to how many times a letter is used in general in the English language, this way you can determine the key through stats. This is not easy to achieve though since you will need a big encrypted message to get useful statistics. You would like to have more than 100 letters the more letters the more correct the stats will be. In this case, 13 letters will not be enough to get a good statistic and not enough to decrypt the message [8].

Part c

The encrypted message looks like this: QMJ BPZ B XPJZ RZWJPAXQ LAD. We know that it is a substitution cipher so I tried some simple ways to encrypt it like using the method Caesar cipher which is when I just try to shift the letters a set amount to the left or right in the alphabet. I tried to shift three, two, and one position to the left with the letters and then shift the letters three, two, and one position to the right. This did not work which I quickly noticed when I tried to decrypt the first word in the message. Then after that, I tried to use brute force, to insert the one-letter words that exist in the English language which are a, i, and o. From that, I tried to figure out other words in the message, I think I figured out some, but it felt impossible to figure out the full message. Another issue with trying to brute force it is that I don't know what language the message is in, in part a) for example the encrypted message was Latin so if this is in Latin as well I will not be able to brute force it since I don't know anything about the language.

The last attempt was to try using cryptanalysis, but this proved impossible as well since the encrypted message is way too short to be able to get a sufficient amount of data. It is required to have at least 100 letters to get some kind of result to work with, and then we would need to get the frequency data of several languages to find out which language it could be, and even then it would be rough since 100 letters will give some result, but might not necessarily be the correct one still. So the main issues are that this is not a regular Caesar cipher, as far as I know, we don't know the language of the message which makes the brute-forcing much harder and the message is too short to do cryptanalysis. I couldn't decrypt the message.

Example of trying to brute force when the letter B is A and the assumption is that the language is English. As you can see it did not go too far.

QMJ = _ _ _
BPZ = ARE
B = A
XPJZ = _R_E
RZWJPAXQ = _E_ _R_ _ _
LAD = _ _ _

Task 4

In this assignment, I've created multiple classes that handle different things related to encryption algorithm substitution and transposition. First and foremost I have two classes that handle reading the text from the file and return the content as a string. The first class ReadTextFile returns a string that looks like the content from the text file and this string is used when encrypting using the substitution algorithm. The other class ReadTextTransposition returns the string without any format so that it is easier to do the transposition algorithm. Then we have a class that writes the encrypted and decrypted strings into a text file that is already input in the code (this can be changed if needed). Then we have two classes EncryptSubstitution and DecryptSubstitution which handles the substitution algorithm. The algorithm is based on the Ceasar cipher which means we shift the position of letters to the right or left, in my case, I shift the letters four positions to the right so for example a becomes e. In my code, I check each character one and one and get the position of the character in the alphabet and then I change that to the letter in my cipher in the same position this is used for both encrypting and decrypting using substitution. Then I have two classes that handle the transposition algorithm and they are called EncryptTransposition and DecryptTransposition.

The classes take a string and the key and then we put the string in a multidimensional array, the size of the array is based on the length of the key, and the row is based on the string's length divided by the length of the key. Then when we have input the string into the array we take one column at a time, based on the order we get from the key and put it in a new string. This repeats until we have written out every column. When we decrypt we do the same thing, we start by putting the string in a multidimensional array, but this time we use the order from the key to putting the characters in the right column directly. So by doing this we get the same array that we have when we encrypt and then we just take each row and append it to a string and the message is decrypted.

Then we have the class called Order which is already talked about a little bit in the previous classes, this class takes in the key and checks what order we will read from the multidimensional array, so for example with LOWER we will get the order 3 0 1 4 2. This will be used for both EncryptTransposition and DecryptTransposition since we need to know the order. Then we finally have a menu that uses all the previous classes and makes them work in a unit. The menu will show the options Substitution, Transposition, and Exit, based on what the user chooses the program will do different things. If Substitution is chosen the user will be asked if they want to encrypt, decrypt, or return to the main menu. If we choose to encrypt or decrypt the user will be asked what file they want to encrypt or decrypt and then we get a message when it's done and where the encrypted or decrypted file can be found.

I've chosen to have a default output file for both substitution and transposition. If you chose Transposition in the main menu you will be asked the same thing as in Substitution besides they ask for a key as well everything else is the same. If Exit is chosen the program closes. I will also include some example of files that has some messages that can be encrypted and decrypted and also some text files that is already encrypted and decrypted.

Task 5

The method I used to create my encryption is that I shift all letters 4 positions to the right in the alphabet so it is a Caesar cipher that has the key 4.

Cipher:e f g h i j k l m n o p q r s t u v w x y z a b c d
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

The encrypted message in the file **“Vilhelm_Park_v2.0.txt”**.

```
**********************************************************************************************************************
*                                                                                                                    *
*                                Wigvix qiwweki - Xst Wigvix                                                         *
*                                                                                                                    *
*                      Qec srpc fi vieh fc wigyvmxc tewwih tivwsrrip                                                 *
*                                                                                                                    *
**********************************************************************************************************************

Wli als evvmzep irh lsa jivxmpi irefpih. Fvsxliv wli ehh cix wii qmryxiv rexyvep wqmpmrk evxmgpi temrxih.
Xliqwipziw ex hmwtexglih mrxiviwxih mrwirwmfpi eq fi tvswtivsyw viewsrefpc mx. Mr imxliv ws wtvmrk amwlih.
Qiperglspc aec wli fsmwxivsyw ywi jvmirhwlmt wli hmwwmqmpev gsrwmhivih ibtviwwmsr. Wib uymgo evswi qvw pmzih.
Qv xlmrkw hs tpirxc sxlivw er zermxc qcwipj aemxih xs. Epaecw tevmwl xewxiw ex ew qv jexliv hmrmrk ex.

Yramppmrk wtsvxwqir li mr uyiwxmsrw witxiqfiv xlivijsvi hiwgvmfih ws. Exxegow qec wix jia fipmizi qsqirxw aew.
Viewsrefpc lsa tswwiwwmsr wlc aec mrxvshygih eki mruymixyhi. Qmwwih li irkeki rs ibixiv sj. Wxmpp xvmih qierw
ai eaevi svhiv eqsrk sr. Iphiwx jexliv ger hiwmkr xewxiw hmh nsc wixxpi. Vsywih jyxyvi li ci er qevoih. Evswi
qv vetmh mr ws zibih asvhw. Kec aipgsqi pih ehh pewxmrk glmijpc wec pssomrk.

Sri ehzergih hmzivxih hsqiwxmg wib vitiexih fvmrkmrk csy sph. Tswwmfpi tvsgyvih liv xvmjpmrk peyklxiv xlsyklxw
tvstivxc wli qix aec. Gsqtermsrw wlc leh wspmgmxyhi jezsyvefpi sar. Almgl gsyph wea kyiwx qer rsa lievh fyx.
Pewxih qc gsqmrk yriewc qevoih ws wlsyph. Kvezmxc pixxivw mx eqsrkwx livwipj hieviwx er amrhsaw fc. Asshih pehmiw
wli fewoix wiewsr eki liv yriewc wea. Hmwgsyvwi yramppmrk eq rs hiwgvmfih hinigxmsr mrgsqqshi rs pmwxirmrk sj.
Fijsvi rexyvi lmw tevmwl fsc.

                    Zmplipg Tevo
**********************************************************************************************************************
```

Task 6

Key

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: y z a b c d e f g h i j k l m n o p q r s t u v w x

Encrypted message

```
**********************************************************************************************
*
*
*                        Ugetgv oguucig - Vqr Ugetgv                                         *
*
*
*                Oca qpna dg tgcf da ugewtkva rcuugf rgtuqppgn                               *
*
*
**********************************************************************************************

K ugg c nkvvng uknjqwgvvq qh c ocp
Uectcoqwejg, Uectcoqwejg, yknn aqw fq vjg Hcpfcpiq?
Vjwpfgtdqnv cpf nkijvpkpi, xgta, xgta htkijvgpkpi og
(Icnkngq) Icnkngq, (Icnkngq) Icnkngq, Icnkngq Hkictq ocipkhkeq
Dwv K'o lwuv c rqqt dqa, pqdqfa nqxgu og
Jg'u lwuv c rqqt dqa htqo c rqqt hcokna
Urctg jko jku nkhg htqo vjku oqpuvtqukva
Gcua eqog, gcua iq, yknn aqw ngv og iq?
Dkuoknncj! Pq, yg yknn pqv ngv aqw iq
(Ngv jko iq) Dkuoknncj! Yg yknn pqv ngv aqw iq
(Ngv jko iq) Dkuoknncj! Yg yknn pqv ngv aqw iq
(Ngv og iq) Yknn pqv ngv aqw iq
(Ngv og iq) Yknn pqv ngv aqw iq
(Pgxgt, pgxgt, pgxgt, pgxgt ngv og iq) Cj
Pq, pq, pq, pq, pq, pq, pq
(Qj, ocooc okc, ocooc okc) Ocooc okc, ngv og iq
Dggnbgdwd jcu c fgxkn rwv cukfg hqt og, hqt og, hqt og!

        Czgn Kukfqt Okejcgn Lquv Cwh Fgt Uvtqvj

**********************************************************************************************
```

Decrypted message

```
**************************************************************************************************
*
*
*                          Secret message - Top Secret                                           *
*
*
*                  May only be read by security passed personnel                                 *
*
*
**************************************************************************************************
```

I see a little silhouetto of a man
Scaramouche, Scaramouche, will you do the Fandango?
Thunderbolts and lightning, very, very frightening me
(Galileo) Galileo (Galileo) Galileo, Galileo Figaro magnifico
But I'm just a poor boy, nobody loves me
He's just a poor boy from a poor family
Spare him his life from this monstrosity
Easy come, easy go, will you let me go?
Bismillah! No, we will not let you go
(Let him go) Bismillah! We will not let you go
(Let him go) Bismillah! We will not let you go
(Let me go) Will not let you go
(Let me go) Will not let you go
(Never, never, never, never let me go) Eh
No, no, no, no, no, no, no
(Oh, mamma mia, mamma mia) Mamma mia, let me go
Beelzebub has a devil put aside for me, for me, for me!

Axel Isidor Michael Jost Auf Der Stroth

```
**************************************************************************************************
```

Description

To find the key and decrypt the message I used the brute force method, since every cipher will start the same way I could find out around 75 % of the letters in the cipher. Using the letters that were already known I could see a pattern that the cipher is simply the alphabet moved two steps forward. With that information, I could fill out the cipher and then decrypt the message.

Task 7

Part A

I created a simple hash function where I gather all the content in a text file, then I create a hash which will gather the long values. To get the hash value I have a prime number (11) which will multiply with the current long value then I add the ascii value from a character in the string. By multiplying the hash value everytime we make sure that the final hash value will be as unique as possible. Then we finally return the 8 bit hash value by using the modulo operator (% 255) to make sure we only get a value that is 8 bit long.

Part B

In this section I do two tests, in the first test I created 1000 random strings with random length and then looked for collisions when I have turned those strings to 8 bit long hash. Second test I have on string that I only change small things inside it such as a character, for example I change from b to a inside the string and then I get the hash and insert it inside a arraylist. Then I look for collisions, as can be seen in the text I get a lot of collisions in the first test and zero in the other one. I'm not sure if the second test is entirely correct since I find that result unlikely. The first looks correct though and a very likely result since the hash can only be 8 bit long.

```
Now I will test how many collisions will occur with 1000 random words using my hash function.
2116
Now I will test how many collisions occur when I only change one bit in the message!
0

Process finished with exit code 0
```

Part C

The difference between hash functions and secure hash functions (SHA) is that the SHA are cryptographic functions that use hash functions. They focus on securing data and are one-way functions which mean that when data has been transformed to a hash value it can't be transformed back into its original state. Hash functions are basically functions that take input and then turn it into some hash output. So the main difference would be that the hash function is a way to turn some input into some data that is not the same as the original data [8] and SHA is basically functions that take hash functions in the more security-related direction [9]. Hash functions can be used for many things while SHA is mainly used for the security of data.

Bibliography

[1]https://www.cryptomathic.com/news-events/blog/symmetric-key-encryption-why-where-and-how-its-used-in-banking
[2] https://searchsecurity.techtarget.com/definition/asymmetric-cryptography
[3] https://en.wikipedia.org/wiki/Encryption
[4] https://en.wikipedia.org/wiki/Hash_function
[5] https://en.wikipedia.org/wiki/Data_compression
[6] How does steganography work and does it threaten enterprise data (provided by the course)
[7] https://en.wikipedia.org/wiki/Digital_watermarking
[8] Lecture slides 2.1and lecture
[9] https://en.wikipedia.org/wiki/Hash_function
[10] https://brilliant.org/wiki/secure-hashing-algorithms/