

2/21/2019

B00099344 Jason Swords

School of Informatics

Department of Informatics and Engineering

Technology University Dublin

Blanchardstown Campus

Blanchardstown

Dublin 15.

Secure-Communications

Bachelor of Science

Computing in Information Technology

24/08/2014

Table of Contents

[Introduction 1](#_Toc2270169)

[Classic Ciphers 2](#_Toc2270170)

[Exercise-1 2](#_Toc2270171)

[Exercise 2. 2](#_Toc2270172)

[Exercise 3. 3](#_Toc2270173)

[Exercise 4. 4](#_Toc2270174)

[Revision – Extra decoding lab 5](#_Toc2270175)

[Number Station 7](#_Toc2270176)

[Password Cracking 8](#_Toc2270177)

[Conclusion 12](#_Toc2270178)

[Bibliography 13](#_Toc2270179)

[Appendices 14](#_Toc2270180)

# Introduction

This report will consist of completed lads conducted during the second semester of third year.

# Classic Ciphers

## Exercise-1

**Caesar cipher**

Looking at the cipher, it is a sentence with the letters either shifted or substituted. Using the first scenario, and cryptii (Cryptii, 2019) to cycle through each possible (3) shift, the true sentence is revealed. Traditionally, Caesar cipher is strictly 3 shifts.

Wkh ruljlqdo Fdhvdu flskhu dozdBv xvhg d vkliw ri wkuhh

the original caesar cipher always used a shift of three

**Rot 13**

Rot 13 is a variation of the Caesar cipher which shifts the letters 13 times, e.g A becomes N. Again using cryptii (Cryptii, 2019) to reveal the text straight away.

EBG 13 JNF HFRQ OL ZVPEBFBSG SBE RAPBQVAT JVAQBJF ERTVFGEL RAGEVRF

rot 13 was used by microsoft for encoding windows registry entries

**Shift ciphers.**

Identifying the cipher as another variation of a shift cipher, cryptii (Cryptii, 2019) quickly identified the true text after shifting through many variations until at shift 16, the text is revealed.

1. MYWWYX FKBSKDSYXC YP DRO CRSPD KBO YPDOX ECON VSUO NSPPOBOXD UOIC

common variations of the shift are often used like different keys

This cipher was revealed using the same method, but at a shift of 2.

1. RFCQC AGNFCPQ YPC RPGTGYJ RM APYAI YLB QMJTC CTCL ZW FYLB

these ciphers are trivial to crack and solve even by hand

This cipher is different from the previous two, this one includes numbers and special characters, which means it’s a Rot 47 cipher. Using cryptii to identify the encrypted text.

1. (:E9 6IA6C:6?46 J@F H:== DE2CE E@ C64@8?:D6 E96 492C24E6CD6ED @7 6249 6?4@5:?8]

With experience you will start to recognise the character sets of each encoding.

## Exercise 2.

**Substitution ciphers**

**Atbash:**

This cipher is Atbash. This works by replacing A with Z, B with Y, C with X and so on. So, using Crypto corner (Crypto Corner, 2019) to reveal the true text.

zgyzhs xrksvi dzh z hfyhgrgfgrlm xrksvi gszg ivevihvw gsv zokszyvg

With experience you will start to recognise the character sets of each encoding.

**Kamasultra:**

KBJHOESNWYCVI Cipher text = Irtg tg kf qwkjbvq

APMRZQGFXDULT Decrypted = THIS IS AN EXAMPLE

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1st | K | B | J | H | O | E | S | N | W | Y | C | V | I |
| 2nd | A | P | M | R | Z | Q | G | F | X | D | U | L | T |

Irtg tg kf qwkjbvq = THIS IS AN EXAMPLE

Knowing the cipher is Kamasutra cipher(Iqmal, Iqmal and profile, 2019), each letter in the cipher is encrypted and decrypted using the same process. To decrypt the cipher text provided, the first letter (I) is found in the table. The letter opposite, either below or above it is noted, in this case, the letter (T). the next letter is (R), opposite that is (H) and so on until the entire cipher text is revealed.

**Substitution cipher.**

This cipher proved a little more complex than the previous ciphers. Initially thinking maybe Caesar cipher or Rot 13, but after cycling through each variation, this was quickly disproved. So obviously some form of a substitution cipher, so using quipquip.com(Quipqiup.com, 2019) to reveal the encrypted text. Quipquip uses frequency analysis to determine the possible character set.

reqdhqxol mxmclfuf uf gtq bql gy neqmbuxs fhnfgughguyx ouztqef gtq cyxsqe gtq wqffmsq gtq qmfuqe ug nqoywqf

frequency analysis is the key to breaking substitution ciphers the longer the message the easier it becomes

## Exercise 3.

**Vigenere Cipher**

Identifying this cipher as a Vigenere cipher deems frequency analysis ineffective due the use of a repeated key value being used. Fortunately, this value can be brute forced using decode.fr (Cipher et al., 2019), and the cipher text can be decrypted, and the key found. In this case the key is MARKMARK.

mS jeNSkSfUkSaN TSbHVbe BVMMMV ZaPlVMR, gOaPcO NEXKZ TiiUNX da

AkdMCb KZD UOOOUO fHVW, MNU dTE nOMKeOeSVc MSjYOIRdQD nSfH kRQM

jdMRkOP Tf LQCfWQ KeYiN, jOhEiKX CZZTEic QMVbSEU gUTY dTE jKYE ZNQA fP

fRpSZG kY NRVKW TYO QFWOOTZfQNVce OW PdEheQNTi MNRVkSZc MNU

RQNTO YAbO fHV MUPYOdS dYdE jOOUiO. FHV WMIe SPER gMS kY OOdO gP

nSfH dOfHfN eO kRMT R ZXAZXfEod XEkdQR USPN’k KXWRie MRZ fO kRQ SRWQ

CZZTEidQXk VQTkOd (SpWNOc), da ATRUEmO fHZc fHV MUPYOdS lcQD R

bQPVKfIeQ WEp. lk FRb fHV WaSk gQLc-UZOnX aF kRQSV MUPYOdS Zc fHV

FUGVXèdE TSbHVb NUk dTEiO MRV WMNp YfHVbe WZdT VVbk SZWULRb

PEjSSNj.

Key = MARKMARK

AS SUBSTITUTION CIPHERS BECAME POPULAR PEOPLE BEGAN TRYING TO ATTACK AND DECODE THEM AND THE WEAKNESSES ASSOCIATED WITH THEM STARTED TO BECOME KNOWN SEVERAL CIPHERS EMERGED WITH THE SAME IDEA OF TRYING TO BREAK THE EFFECTIVENESS OF FREQUENCY ANALYSIS AND HENCE MAKE THE CIPHERS MORE SECURE THE MAI

## Exercise 4.

**Transposition cipher**

This cipher proved slightly more difficult than previous ciphers. Initially trying quipquip.com for a frequency analysis but this returned nothing of value. Using decode.fr (Cipher et al., 2019) and searching for column transposition cipher decoder, the text was quickly revealed.

NRAATIOSSPCOTNIRHIEPCSAOLDLAELMLCUORPNESTTMAUSNIIOHEACTEQNUIACTHOHENTGEREDOHFRTOTEETLNSEIRTEAXTAPBLYINCGIGNTAIDRI

A TRANSPOSITION CIPHER ALSO CALLED COLUMN SPERMUTATION IS A TECHNIQUE TO CHANGE THE ORDER OF THE LETTERS IN A TEXT BY PLACING IT IN A GIRD

# Revision – Extra decoding lab

**Message – 1**

V2VsbCBkb25lIHlvdSBzb2x2ZWQgcGFydCAx

Using a hash analyser (Tunnelsup.com, 2019) to identify the hash type, the results show base 64. Using CyberChef to decode the string. The results are:

Plain text = Well done you solved part 1

**Message – 2**

d8578edf8458ce06fbc5bb76a58c5ca4

Using hash analyser again, the results show MD4 or MD5. Putting the hash into google shows the results are:

Plain text = qwerty

**Message – 3**

Pna lbh oryvrir Zvpebfbsg hfrq guvf gb rapbqr vgf ertvfgel inyhrf!

Judging by the absence of numbers and spaces in the cypher, this looks like a type of shift cypher. To reveal the message, Quipquip.com is used. The results are:

Plain text = Can you believe Microsoft used this to encode its registry values!

**Message – 4**

(6== 5@?6[ E9:D :D 2 >F49 =6DD @7E6? FD65 6?4@5:?8 D496>6

Noticing the presence of special characters and alphanumeric characters, the observation of ROT 47 in use, is use as basis to try this cypher: The results are:

Plain text = Well done, this is a much less often used encoding scheme

**Message – 5**

q293VFOcoKOlMKAmnKMyVUEbnKZtq2SmVUMypaxtqUWcL2g5YPO3MJkfVTEiozHtn

J4tp29fqzyh MlO0nTymYPOiozk5VT9hMFOgo3WyVUEiVTqiYPOao29xVTk1L2fAPt==

Noticing the two == signs at the end of the cypher indicates that BASE64 is in use. After attempting to decode using CyberChef, this failed to reveal any message. A cryptic hint saying, “it doesn’t have to be just once!” indicate that the cypher may have been encrypted more than once. After attempting to run two and three rounds of BASE64, still no legible message. After many attempts of using different cypher combinations, the solution was found using ROT13 and BASE64 together. The results are:

Plain text = wow! impressive this was very tricky, well done in solving this, only one more to go, good luck

**Message – 6**

EVIB DVOO WLMV LM HLOERMT GSV URMZO KILYOVN, SLKVUFOOB RG

WRWM'G GZPV BLF GLL OLMT ZMW BLF VMQLBVW RG.

Again, noticing the absence of numbers and the presence of spaces, this could be another shift cypher. This assumption proved valid. The results are:

Plain text = VERY WELL DONE ON SOLVING THE FINAL PROBLEM, HOPEFULLY IT DIDN'T TAKE YOU TOO LONG AND YOU ENJOYED IT.

# Number Station

Number stations are short-wave radio stations repeatedly broadcasting streams of number and letters using voice or Morse code (Users.telenet.be, 2019). This method was used to transmit encrypted messages to the intended people listening.

Before decrypting the message, two types of encryption can be used, one the encrypt the message and another the encrypt the cipher text. In this instance, Mod 10 addition is used to decrypt the cipher text before using another technique to calculate the message.

Mod 10 is simply adding two number and only keeping the remainder. E.g., 6 + 6 %10 = 2.

In this cipher, the first digit from the key is added to the first digit of the cipher text and the remainder is used.

Key = 66153-77185-10800-54937-48159-83271-12892-07132-34987-53954-23074

Cipher = 66475-19274-92028-78494-24146-68542-17507-39398-32348-59378-70636

Mod10= 22528-86359-02828-22321-62295-41713-29399-36420-66225-02222-93600

To complete the second step, a method known as the Straddling Checkerboard (Practicalcryptography.com, 2019) is used to decrypt the message. The method consists of several steps to complete the decryption.

* Layout grid as below, number top row 0-9
* Insert straddling checkerboard key (AT\_ONE\_SIR)
* Populate empty spaces with reaming letters of the alphabet
* Number the two rows using the index of each space in the key. ie. 2 and 6
* Using the decrypted text above, taking the first block – 22528
  + 2 = use line number 2
  + Number immediately after is the letter index
    - 2 = D
  + Next number is 5. 5 = E
  + Next number is two, again jump to line number 2
    - 8 = L
* The same process is used to complete the remaining blocks of numbers

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|  | A | T | \_ | O | N | E | \_ | S | I | R |
| 2 | B | C | D | F | G | H | J | K | L | M |
| 6 | P | Q | U | V | W | X | Y | Z | # | / |

* Number 6 is similar to 2. Jump to line number 6

Encrypted -22528-86359-02828-22321-62295-41713-29399-36420-66225-02222-93600

Decrypted - DEL – IVER - ALL - DOC - UME -NTSTO-MORR-OWB -YDE -ADD -ROPA

Formatted - DELIVER ALL DOCUMENTS TOMORROW BY DEAD DROP A

# Password Cracking

This lab presented a document with six hashes and some information about how the hashes are formed.

* Password after 2010 are alphanumeric 5 – 7 characters long.
* Passwords before this consist of only digits 5 – 7 digits long.
* All passwords contain the same salt.
* From mySQL database.
* Sites domain [www.exploringsecurity.com](http://www.exploringsecurity.com).
* Salt format believed to be: unnamed hash($salt, $hash).

The first step is to identify the hash type in use by the database. To aid this, an online hash identifier was used (Tunnelsup.com, 2019). This site concluded SHA1 was in use. Reinforcing this conclusion, another online resource (Hashc.co.uk, 2019) was used.

Now that the hash type is known, the salt is the next important piece of data needed before any attempted cracking can be conducted.

Conducting a Google search of the topic, It quickly became apparent that the salt may be a string stored separately in the data base unique to each user or the same string for every user. Considering this database was constructed in 2009 – 2010, we could firstly eliminate that maybe salts were unique to each password or that one salt was used throughout the database.

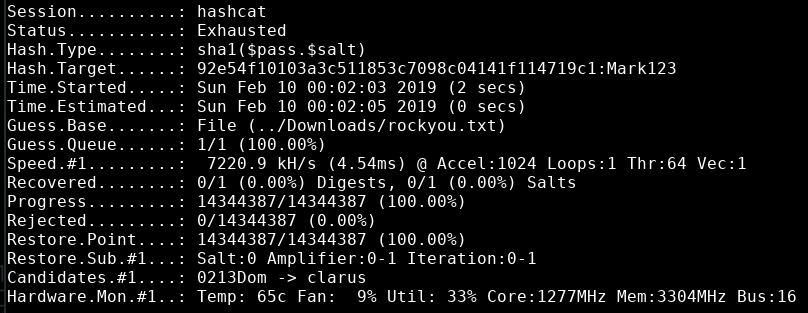
|  |  |  |  |
| --- | --- | --- | --- |
| Date | Username | Hash (Sha1) | Role |
| 2009 | Sparky | 2834da08d58330d8dafbb2ac 1c0f85f6b3b135ef | Admin |
| 2010 | Mark123 | 92e54f10103a3c511853c709 8c04141f114719c1 | User |
| 2009 | superman | 437fbc6892b38db6ac5bdbe2 eab3f7bc924527d9 | User |
| 2010 | security | fafa4483874ec051989d53e1 e432ba3a6c6b9143 | User |
| 2009 | Tomtom | 06f6fe0f73c6e197ee43eff4e5 f7d10fb9e438b2 | User |
| 2010 | JillC | f44f3b09df53c1c11273def13 cacd8922a86d48c | User |

To test this, it would be safer to eliminate any other info as the salt. The first unique string suitable for the salt is the user name considering the remaining information is dates and roles assigned to each user.

To test the salt and try crack the hashes, a password recovery tool named Hashcat will be used. For Hashcat to run correctly, the correct format must be used. Using the information supplied, $salt - $pass, we have the format of the hash type, now the format to specify the salt is needed. This is found on Hashcats website (Hashcat.net, 2019)giving examples as to how a hash should be formatted for each hash type. Sha1 should be password:salt, separated with a colon.

**92e54f10103a3c511853c709 8c04141f114719c1: Mark123**

**$hashcat -m 120 hash-1 Downloads.rockyou.txt**

The hash format was accepted but no hash was found.

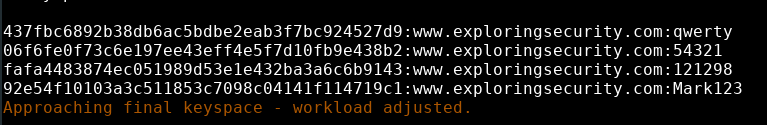
As no more unique information from the datbase could be used, another type of salt must have been used. The domain name of the website, mentioned in the brief, could also be a possibility. This will be the next attempt.

**92e54f10103a3c511853c709 8c04141f114719c1:www.exploringsecurity.com**

**$hashcat -m 120 hash-1 Downloads/rockyou.txt**

This found the password. Assuming this format is used on all passwords, all six hashes are formatted into a text file and ran against the rockyou.txt wordlist which contains over 14 million passwords.

**$hashcat -m 120 hashes Downloads/rockyou.txt**

This time four passwords were recovered using the wordlist.

The remaining two passwords will have to be recovered using other methods. According to the brief, only alphanumeric characters are in the password, this eliminates sixty-five special characters, leaving sixty-two characters to use in passwords.

Many tools exist for creating password lists using specified characters. One such tool is Crunch. This tool can create custom password lists containing any characters specified.

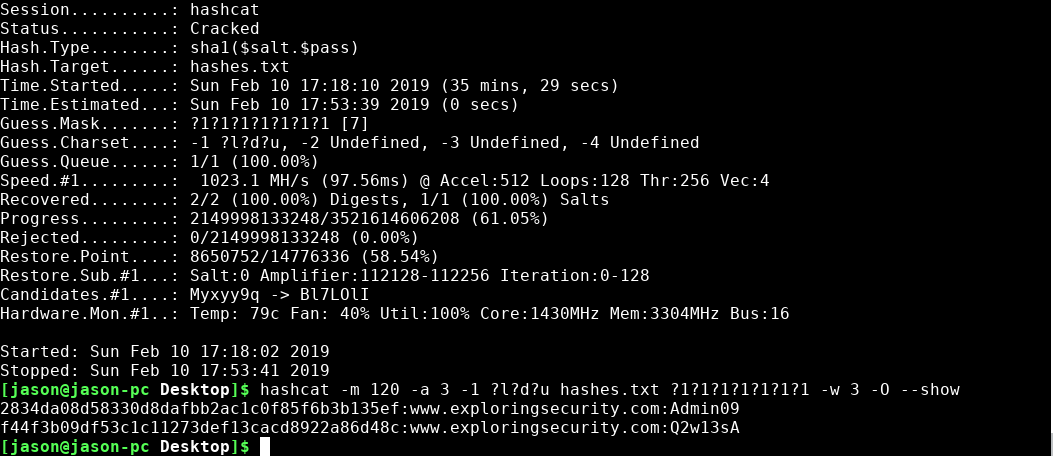
**$crunch 5 7 abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789 | hashcat -m 120 hashes**

This command proved extremely slow, with the computer running for several hours without any values found. Considering the possibility that maybe Crunch is slowing down Hashcat, the size of creating a text file with all possibilities was considered, until it became apparent how big the file would be. Twenty-Five terabytes was the predicted file size, which is not a logical direction considering the time and space available on the computer.

Another method is Hashcat’s built in mask attack(Hashcat.net, 2019), which is similar to using Crunch, but as the tool is built in, it should perform much faster than the previous attempt. This attack allows the user to specify which character set should be applied, and how long the password is.

**$hashcat -m 120 -a 3 -1 ?l?u?d hashes.txt ?1?1?1?1?1?1?1 -w 3 -O**

As this is a intensive process, another computer was used to speed the process up, taking advantage of the GPU (Graphics Processing Unit) which reduced the time significantly (12hours and 23mins down to 35 mins).

After running this command, the program quickly produced the remaining passwords by trying every conceivable possibility within the constraints used.

The entire list of passwords is:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Date | Username | Hash (Sha1) | Role | Salt | Password |
| 2009 | Sparky | 2834da08d58330d8dafbb2ac 1c0f85f6b3b135ef | Admin | www.exploringsecurity.com | Admin09 |
| 2010 | Mark123 | 92e54f10103a3c511853c709 8c04141f114719c1 | User | www.exploringsecurity.com | Mark123 |
| 2009 | superman | 437fbc6892b38db6ac5bdbe2 eab3f7bc924527d9 | User | www.exploringsecurity.com | qwerty |
| 2010 | security | fafa4483874ec051989d53e1 e432ba3a6c6b9143 | User | www.exploringsecurity.com | 121298 |
| 2009 | Tomtom | 06f6fe0f73c6e197ee43eff4e5 f7d10fb9e438b2 | User | www.exploringsecurity.com | 54321 |
| 2010 | JillC | f44f3b09df53c1c11273def13 cacd8922a86d48c | User | www.exploringsecurity.com | Q2w13sA |

# Conclusion

# Bibliography

Users.telenet.be. (2019). Numbers Stations. [online] Available at: http://users.telenet.be/d.rijmenants/en/numbers.htm [Accessed 8 Feb. 2019].

Quipqiup.com. (2019). quipqiup - cryptoquip and cryptogram solver. [online] Available at: https://quipqiup.com/ [Accessed 8 Feb. 2019].

Practicalcryptography.com. (2019). Practical Cryptography. [online] Available at: http://practicalcryptography.com/ciphers/straddle-checkerboard-cipher/ [Accessed 8 Feb. 2019].

Cipher, V., Cipher, P., Cipher, V., cipher?, H., cipher?, H., ciphertext?, H., key?, H., plaintext?, H., cipher?, W., ?, W., ?, W., ?, W., ?, W. and invented?, W. (2019). Vigenère Cipher - Decoder, Encoder, Solver, Translator. [online] Dcode.fr. Available at: https://www.dcode.fr/vigenere-cipher [Accessed 8 Feb. 2019].

Iqmal, A., Iqmal, A. and profile, V. (2019). Kamasutra Cipher. [online] Programmingcode4life.blogspot.com. Available at: https://programmingcode4life.blogspot.com/2015/10/kamasutra-cipher.html [Accessed 8 Feb. 2019].

Crypto Corner. (2019). Atbash Cipher. [online] Available at: https://crypto.interactive-maths.com/atbash-cipher.html [Accessed 8 Feb. 2019].

Cryptii. (2019). Caesar cipher: Encode and decode online. [online] Available at: https://cryptii.com/pipes/caesar-cipher [Accessed 8 Feb. 2019].

Tunnelsup.com. (2019). Hash Analyzer - TunnelsUP. [online] Available at: https://www.tunnelsup.com/hash-analyzer/ [Accessed 9 Feb. 2019].

Hashc.co.uk. (2019). hashC - hash identifier & hashcat modes lister. [online] Available at: https://hashc.co.uk/hashid [Accessed 9 Feb. 2019].

Hashcat.net. (2019). example\_hashes [hashcat wiki]. [online] Available at: https://hashcat.net/wiki/doku.php?id=example\_hashes [Accessed 9 Feb. 2019].

Hashcat.net. (2019). mask\_attack [hashcat wiki]. [online] Available at: https://hashcat.net/wiki/doku.php?id=mask\_attack [Accessed 10 Feb. 2019].

Tunnelsup.com. (2019). Hash Analyzer - TunnelsUP. [online] Available at: https://www.tunnelsup.com/hash-analyzer/ [Accessed 14 Feb. 2019].

Cipher, C., Cipher, T., Cipher, C., cipher?, H., cipher?, H., ciphertext?, H. and key?, H. (2019). Columnar Transposition Cipher - Decoder, Encoder, Solver, Translator. [online] Dcode.fr. Available at: https://www.dcode.fr/columnar-transposition-cipher [Accessed 28 Feb. 2019].

# Appendices