Decrypting an Unknown Caesar Cipher Using Brute Force

Farrel Farandieka Fibriyanto – 13520054[1]

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
[1]13520054@std.stei.itb.ac.id

***Abstract*— Ciphers, or cryptography in general, has played an integral role in our life. Whether it is to encrypt a document so only some knows the content or maybe as simple as sending a text message, we have used cryptography, either consciously or subconsciously. One of the oldest most basic cipher known is the Caesar Cipher, named after the inventor, Julius Caesar. The cipher involves shifting the letters of each character by a certain amount, the amount being the key to the ciphertext, forward to generate the ciphertext, and shifting it back the said amount. As the operation involved is basic compared to modern day cryptography, it is easy to crack any ciphertext encrypted using the Caesar Cipher through brute force.**

Keywords—cryptography, cipher, caesar, bruteforce, decryption

I. Introduction

Cryptography is a technique for achieving confidentiality of a message. This is done through converting the message being sent to a jumbled mess of letters, which we call a ciphertext, using a certain predefined method. The receiver then converts the ciphertext back to a readable message by going backwards through the predefined method said before. If one does not know the said method, they would find the sent message to just be a collection of random letters. Thus, only those who knows the predefined method could read the message.

One of the earliest known ciphers is the Caesar Cipher, also known as Caesar's Cipher. It is named after the inventor, Julius Caesar, who used it in his private correspondence [1]. The first form of the cipher used a shift of three, meaning every letter is shifted three letters forward. If the letter is shifted outside of the alphabet range, it would wrap back around to the front. To decrypt the ciphertext, one would need to shift it backwards with the amount of shift according to the key used to encrypt it.

Nowadays, through the power of computing, cryptography uses a much more sophisticated method of encrypting a message. This is because basic forms of cryptography like the Caesar Cipher is easily cracked through bruteforce within a matter of seconds. To give an example on how easy it is to decrypt an older form of cipher, this paper will discuss a way of decrypting one of the basic ciphers, the Caesar Cipher.

II. Theoretical Foundation

A. Cryptography

Cryptography is the practice and study of securing communications between two or more parties. The word is derived from the greek word "*kryptós*" and "*graphein*" which means "Hidden, secret" and "To write", respectively[2]. Taken from Cambridge's online dictionary, cryptography means the practice of creating and understanding codes that keep information secret.

Cryptography's earlier forms were synonymous to encryption, a process in which we convert information from a state that is readable to a state which to the unknowledged would just be unintelligible nonsense. The sender and the received would first need to agree on the method in which they encrypt and decrypt their messages. The sender would take the readable message and run it through a certain method or algorithm, to turn it into a ciphertext in a process called encryption. Ciphertext is the forementioned unintelligible nonsense text. The sender is now able to send the message to the receiver. If by any chance the message is intercepted, the interceptor would only see the ciphertext, which to them might be an unknown language or even trash messages since it is filled with letters that makes no sense. If the receiver finally received the message, they could then decrypt, a process that reverts the encryption process producing readable message, the ciphertext. Thus, only those who knows the method or algorithm of encryption could decrypt the message. By making sure only the sender and receiver knows the message, we can preserve confidentiality of the message.

B. Cryptanalysis

Cryptanalysis is the pratice of studying cryptographic systems to look for weaknesses or leaks of information [3]. This could mean it is used for breaching cryptographic security systems or to gain access to contents of encrypted messages even if the cryptographic key used is unknown.

Some methods of cryptanalysis are [4][5]:

a) Ciphertext-only attack

The attacker would have information to some ciphertexts. They would then attempt to discover the key used in encryption and readable message.

b) Known-plaintext attack

The attacker would have some plaintext pairs that have been collected earlier, moreover the intercepted ciphertext that it wants to break. They would then attempt to discover the key used and decrypt the message.

c) Chosen-plaintext attack

The attacker would know either the encryption algorithm or have access to the encryption device. They would then try to work backwards with the message and find informations about the key used in the encyption.

d) Man-in-the-middle attack

An attacker would compromise a secure communication channel and find out more about the message or key being used in the communication.

e) Dictionary attack

An attacker would encrypt all the words in their dictionary, then check whether the resulting ciphertext generated matches the intercepted ciphertext.

C. *Caesar Cipher*

Caesar Cipher, also known as Caesar's Cipher or Caesar Shift, is one of the most basic encryption technique. The method is named after Julius Caesar who is the inventor of the technique. It involves shifting the letters in the message with another letter on the alphabet based on the key to produce an intangible ciphertext. If the letter shifted goes out of range on the alphabet, it will loop back around to the front. So if 'Z' is shifted by one it would become 'A'.

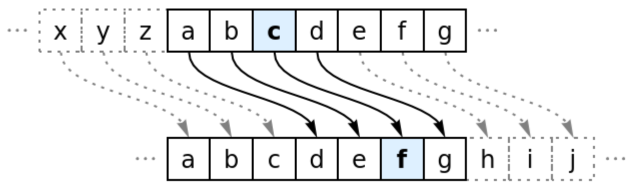


Fig. 1 Example of encryption using a Caesar Cipher with a key of 3.
Taken from https://resources.wolfram.com

To decrypt the ciphertext, one would need knowledge of the key first. The letters in the ciphertext then needs to be shifted back based on the key.

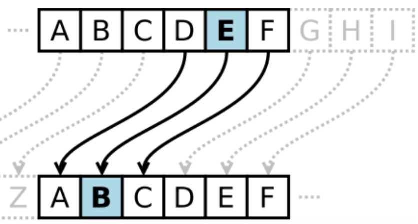


Fig.2 Example of decryption using a Caesar Cipher with a key of 3.
Taken from https://upload.wikimedia.com

D. *Brute Force Algorithm*

Brute force algorithm, also called the "naïve" algorithm, is one of the simplest algorithm that can be used to solve virtually any problem. The algorithm solves a problem through exhaustion, or going through all possible choices until a solution is found. The way the algorithm solves a problem is very simple and intuitive, at the cost of being slow.

E. *Pattern Matching*

Pattern matching is the act of finding or locating a substring from a text that matches a given pattern. Assuming the given text (T) is longer than the pattern (P), the algorithm will iterate through the given text to find occurences of the pattern. Different versions of pattern matching might stop if it finds the first occurrence, or continue to find all occurences.

```
   NOBODY NOTICED HIM
1  NOT
2   NOT
3    NOT
4     NOT
5      NOT
6       NOT
7        NOT
8         NOT
```

Fig.3 Visual example of pattern matching through brute force
Taken from Algorithm Strategies teaching material [6].

There exists different algorithms for pattern matching such as the Brute force algorithm, the Knuth-Morris-Pratt algorithm (KMP), the Boyer-Moore algorithm, Damerau-Levehstein distance, etc. Pattern matching is used in many applications such as in text editors' find function, search engines, file searching, bioinformatics, etc. Different applications might need to use different algorithms to satisfy the needs of their functions.

F. *Regular Expression*

Regular expression is a notation that expresses a search pattern in a text. Regular expression, or regex for short, is able to efficiently match a pattern within a text. Regex is most often used for "find" or "find for replace" operations in a text or for validating inputs.

Regular expression syntax usually comprises of Character Classes, Assertions, Group and Ranges, and Quantifiers [7].

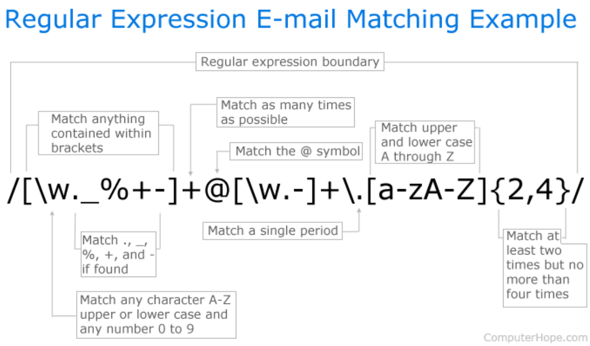


Fig.4 Example of a regex syntax.
Taken from https://paulvanderlaken.com

G. *Commonly Used Words Dictionary*

For every language there exists a set of words that are used more often than other words. If one goes through and count the frequency of words in many literature, they might be able to get the most commonly used words for that language. If they then get a new unique literature on their hands and compare the most commonly used words previously compiled against it, they would be able to ascertain if the literature is in that specific language or not.

Word	PoS	Freq
the	Det	61847
of	Prep	29391
and	Conj	26817
a	Det	21626
in	Prep	18214
to	Inf	16284
it	Pron	10875
is	Verb	9982
to	Prep	9343
was	Verb	9236

Fig.5 The 10 most commonly used words in english
Taken from BNC's Word Frequency [8].

III. Proposed Method

The following proposed method is based on the fact that we know the cipher being used to encrypt a plaintext is the Caesar Cipher, with an unknown key, that is only encrypted once. Thus, this will be a Chosen-plaintext, dictionary attack on a given ciphertext.

There is a need to check if the key being checked by the brute force algorithm is the key used for encryption. Here is where we need our dictionary of most used words. The dictionary will be used to check for the amount of occurences in the ciphertext after being encrypted using a key. The theory being the key with the most occurences is the key that is most likely being used to encrypt the ciphertext.

In this paper, I will be expecting english as the language that the message uses, so there will be a need for an english word dictionary. Since the use of words changes year by year, the latest word frequency list is preferred.

After getting our dictionary, we need to check how many occurences of the generated shifted dictionary are in the ciphertext.

After getting our word dictionary and the regular expression syntax, we are now able to do the proposed algorithm. Outline of the proposed method is as follows:

1. Let *i* be from 0 to 25
2. Make a temporary dictionary filled with our original dictionary and shift it with the Caesar Cipher by *i*
3. Find all occurences of dictionary in the ciphertext by using our regular expression syntax generator and running it through a regular expression finder function.
4. Save the amount of occurences for the given *i* in an array
5. Repeat steps 2 to 4 until *i* reaches 25
6. Find the index that contains the highest occurrence in the array.
7. Let that index be our most likely decryption key

After getting our decryption key, if we need to get the encryption key we will just need to put it into the following equation:

$$\text{encryption\_key} = (26 - \text{decryption\_key}) \bmod 26 \qquad (1)$$

A. *Word Dictionary*

For the word dictionary, I suggest using the 2018 word frequency data generated from OpenSubtitle[9]. I propose to use the 100 most used words from that list for the word dictionary. The dictionary will be saved in either an external file or hard-coded into the program since the dictionary will be static.

If an alternative dictionary is needed, I recommend using the word frequency list based on the British National Corpus. The downside of using the word frequency based on the corpus is that the corpus has not been updated since I was born. Thus, it will be outdated and might not work as effective as newer lists.

B. *Regular Expression Syntax*

For our regex syntax, we need to make sure the match we get from running the regex finder is one that is a full word match instead of a partial word match. To do this we will wrap our dictionary's encrypted word one by one with the word boundary syntax (\b) to generate the regular expression syntax. The use of word boundary is to make sure we do not get any partial match from another word.

Fig.6 Comparison between using word boundary and without

C. *Pseudocode*

1) *Caesar Cipher*

```
FUNCTION do_caesar(text: str, shift: int) -> str:
    IF(shift = 0):
        RETURN text
    ENDIF
    result <- ""
    FOR char IN text:
        newAscii <- get_ascii(char) + shift
        IF newAscii > 90 AND ord(char) < 91: {capital
letter loopback}
            newAscii -= 26
        ELSEIF (newAscii > 122 AND get_ascii(char) < 123):
{noncapital letter loopback}
            newAscii -= 26
        ENDIF
        IF ((get_ascii(char) < 65) or (ord(char) > 90 AND
get_ascii(char) < 97) or (get_ascii(char) > 122)):
{nonletter ignore}
            newAscii <- get_ascii(char)
        ENDIF
        result.addLetter(to_ascii(newAscii))
    ENDFOR
    RETURN result
ENDFUNCTION


FUNCTION arr_do_caesar(texts: List[str], shift: int) ->
List[str]:
    result <- []
    FOR text IN texts:
        result.append(do_caesar(text, shift))
    ENDFOR
    RETURN result
ENDFUNCTION
```

2) *Occurrence Checker*

```
FUNCTION get_occurence(dict: List[str], text: str) -> int:
    occurence <- 0
    FOR word IN dict:
        exp <- "\b" + word + "\b"
        occurence += len(re.findall(exp, text))
    ENDFOR
    RETURN occurence
ENDFUNCTION
```

3) *Bruteforcing Program*

```
FUNCTION bruteforce_caesar(ciphertext):
    dict <- get_dict()
    occurence_list <- []
    FOR i TRAVERSE (0..24):
        occurence_list.append(get_occurence(dict,
ciphertext)
        dict <- arr_do_caesar(dict, 1)
    ENDFOR
    mostLikelyShift <- index_of(occurence_list,
max(occurence_list))
    mostLikelyKey <- (26 - mostLikelyShift) % 26
    mostLikelyDecipheredText <- do_caesar(ciphertext,
mostLikelyKey)
    RETURN mostLikelyDecipheredText
ENDFUNCTION
```

IV. EXPERIMENT RESULTS

The proposed method is programmed using the programming language Python and some experiments to decrypt a Caesar Cipher is performed. The ciphertext is generated by running a snippet of a news article through a Caesar Cipher encryptor with a random key. The news article snippets are gathered from The Guardian and National Geographic.

A. *Key of 3*

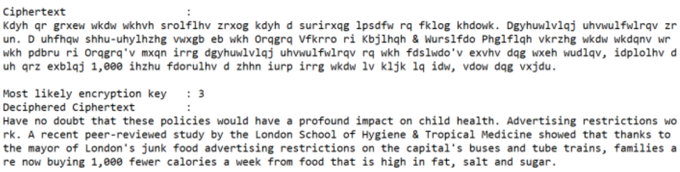
A snippet of text from the article "Boris Johnson's obesity U-turn is a total Eton mess" is taken and put through a Caesar Cipher Encryptor using a key of 3, the plaintext is as follows:

> Have no doubt that these policies would have a profound impact on child health. Advertising restrictions work. A recent peer-reviewed study by the London School of Hygiene & Tropical Medicine showed that thanks to the mayor of London's junk food advertising restrictions on the capital's buses and tube trains, families are now buying 1,000 fewer calories a week from food that is high in fat, salt and sugar.

And the generated ciphertext is as follows:

> Kdyh qr grxew wkdw wkhvh srolflhv zrxog kdyh d surirxqg lpsdfw rq fklog khdowk. Dgyhuwlvlqj uhvwulfwlrqv zrun. D uhfhqw shhu-uhylhzhg vwxgb eb wkh Orqgrq Vfkrro ri Kbjlhqh & Wurslfdo Phglflqh vkrzhg wkdw wkdqnv wr wkh pdbru ri Orqgrq'v mxqn irrg dgyhuwlvlqj uhvwulfwlrqv rq wkh fdslwdo'v exvhv dqg wxeh wudlqv, idplolhv duh qrz exblqj 1,000 ihzhu fdorulhv d zhhn iurp irrg wkdw lv kljk lq idw, vdow dqg vxjdu.

The ciphertext is then put through the decryption program. The program succeeded in finding the original message as well as the encryption key.



```
Ciphertext                  :
Kdyh qr grxew wkdw wkhvh srolflhv zrxog kdyh d surirxqg lpsdfw rq fklog khdowk. Dgyhuwlvlqj uhvwulfwlrqv zr
un. D uhfhqw shhu-uhylhzhg vwxgb eb wkh Orqgrq Vfkrro ri Kbjlhqh & Wurslfdo Phglflqh vkrzhg wkdw wkdqnv wr
wkh pdbru ri Orqgrq'v mxqn irrg dgyhuwlvlqj uhvwulfwlrqv rq wkh fdslwdo'v exvhv dqg wxeh wudlqv, idplolhv d
uh qrz exblqj 1,000 ihzhu fdorulhv d zhhn iurp irrg wkdw lv kljk lq idw, vdow dqg vxjdu.

Most likely encryption key  : 3
Deciphered Ciphertext       :
Have no doubt that these policies would have a profound impact on child health. Advertising restrictions wo
rk. A recent peer-reviewed study by the London School of Hygiene & Tropical Medicine showed that thanks to
the mayor of London's junk food advertising restrictions on the capital's buses and tube trains, families a
re now buying 1,000 fewer calories a week from food that is high in fat, salt and sugar.
```

Fig.7 Program output for key of 3 test case

B. *Key of 5*

A snippet of text from the article "Russia-Ukraine war: UN calls for end to school strikes after nearly 100 child deaths in

April; EU to consider Ukraine’s membership – as it happened” is taken and put through a Caesar Cipher Encryptor using a key of 5, the plaintext is as follows:

> The horrors of Hiroshima and Nagasaki made the whole world afraid of the atomic bomb – even those who might launch one. Today that fear has mostly passed out of living memory, and with it we may have lost a crucial safeguard, Daniel Immerwahr, associate professor of history at Northwestern University, writes.

And the generated ciphertext is as follows:

> Ymj mtwwtwx tk Mnwtxmnrf fsi Sflfxfpn rfij ymj bmtqj btwqi fkwfni tk ymj fytrnh gtrg - jajs ymtxj bmt rnlmy qfzshm tsj. Ytifd ymfy kjfw mfx rtxyqd ufxxji tzy tk qnansl rjrtwd, fsi bnym ny bj rfd mfaj qtxy f hwzhnfq xfkjlzfwi, Ifsnjq Nrrjwbfmw, fxxthnfyj uwtkjxxtw tk mnxytwd fy Stwymbjxyjws Zsnajwxnyd, bwnyjx.

The ciphertext is then put through the decryption program. The program succeeded in finding the original message as well as the encryption key.

```
Ciphertext                    :
Ymj mtwwtwx tk Mnwtxmnrf fsi Sflfxfpn rfij ymj bmtqj btwqi fkwfni tk ymj fytrnh gtrg - jajs ymtxj bmt rnlmy
 qfzshm tsj. Ytifd ymfy kjfw mfx rtxyqd ufxxji tzy tk qnansl rjrtwd, fsi bnym ny bj rfd mfaj qtxy f hwzhnfq
 xfkjlzfwi, Ifsnjq Nrrjwbfmw, fxxthnfyj uwtkjxxtw tk mnxytwd fy Stwymbjxyjws Zsnajwxnyd, bwnyjx.

Most likely encryption key    : 5
Deciphered Ciphertext         :
The horrors of Hiroshima and Nagasaki made the whole world afraid of the atomic bomb - even those who might
 launch one. Today that fear has mostly passed out of living memory, and with it we may have lost a crucial
 safeguard, Daniel Immerwahr, associate professor of history at Northwestern University, writes.
```

Fig.8 Program output for key of 5 test case

C. *Key of 13 (ROT13)*

A snippet of text from the article “Stonehenge builders ate undercooked offal, ancient faeces reveals” is taken and put through a Caesar Cipher Encryptor using a key of 13, also known as ROT13, the plaintext is as follows:

> Stonehenge is thought to have been built around 2,500BC, with evidence suggesting the builders were housed at a settlement known as Durrington Walls, about 2 miles away. The site was predominantly occupied in the winter months, and appears to have been used for between 10 to 50 years.

And the generated ciphertext is as follows:

> Fgbaruratr vf gubhtug gb unir orra ohvyg nebhaq 2,500OP, jvgu rivqrapr fhttrfgvat gur ohvyqref jrer ubhfrq ng n frggyrzrag xabja nf Qheevatgba Jnyyf, nobhg 2 zvyrf njnl. Gur fvgr jnf cerqbzvanagyl bppHcvrq va gur jvagre zbaguf, naq nccrnef gb unir orra hfrq sbe orgjrra 10 gb 50 lrnef.

The ciphertext is then put through the decryption program. The program succeeded in finding the original message as well as the encryption key.

```
Ciphertext                    :
Fgbaruratr vf gubhtug gb unir orra ohvyg nebhaq 2,500OP, jvgu rivqrapr fhttrfgvat gur ohvyqref jrer ubhfrq
ng n frggyrzrag xabja nf Qheevatgba Jnyyf, nobhg 2 zvyrf njnl. Gur fvgr jnf cerqbzvanagyl bpphcvrq va gur j
vagre zbaguf, naq nccrnef gb unir orra hfrq sbe orgjrra 10 gb 50 lrnef.

Most likely encryption key    : 13
Deciphered Ciphertext         :
Stonehenge is thought to have been built around 2,500BC, with evidence suggesting the builders were housed
at a settlement known as Durrington Walls, about 2 miles away. The site was predominantly occupied in the w
inter months, and appears to have been used for between 10 to 50 years.
```

Fig.9 Program Output for key of 13 test case

D. *Key of 16*

A snippet of text from the article “North Korea’s Covid caseload passes 2 million amid global concern about regime’s pandemic plan” is taken and put through a Caesar Cipher Encryptor using a key of 16, the plaintext is as follows:

> The rising caseload and a lack of medical resources and vaccines has led the UN human rights agency to warn of “devastating” consequences for North Korea’s 25 million people, and World Health Organization officials worry an unchecked spread could give rise to deadlier new variants.

And the generated ciphertext is as follows:

> Jxu hyiydw sqiubeqt qdt q bqsa ev cutysqb huiekhsui qdt lqssydui xqi but jxu KD xkcqd hywxji qwudso je mqhd ev “tulqijqjydw” sediugkudsui veh Dehjx Aehuq’i 25 cybbyed fuefbu, qdt Mehbt Xuqbjx Ehwqdypqjyed evvysyqbi mehho qd kdsxusaut ifhuqt sekbt wylu hyiu je tuqtbyuh dum lqhyqdji.

The ciphertext is then put through the decryption program. The program succeeded in finding the original message as well as the encryption key.

```
Ciphertext                    :
Jxu hyiydw sqiubeqt qdt q bqsa ev cutysqb huiekhsui qdt lqssydui xqi but jxu KD xkcqd hywxji qwudso je mqhd
 ev “tulqijqjydw” sediugkudsui veh Dehjx Aehuq'i 25 cybbyed fuefbu, qdt Mehbt Xuqbjx Ehwqdypqjyed evvysyqbi
 mehho qd kdsxusaut ifhuqt sekbt wylu hyiu je tuqtbyuh dum lqhyqdji.

Most likely encryption key    : 16
Deciphered Ciphertext         :
The rising caseload and a lack of medical resources and vaccines has led the UN human rights agency to warn
 of “devastating” consequences for North Korea's 25 million people, and World Health Organization officials
 worry an unchecked spread could give rise to deadlier new variants.
```

Fig.10 Program output for key of 16 test case

E. *Key of 23*

A snippet of text from the article “How do you capture the ‘essence of touch’—especially during a pandemic?” is taken and put through a Caesar Cipher Encryptor using a key of 23, the plaintext is as follows:

> The importance of touch came into focus two years ago as the world coped with the isolation imposed by COVID-19. Months spent avoiding handshakes and hugs for fear of an infectious disease only reinforced how essential these connections are for our overall health.

And the generated ciphertext is as follows:

> Qeb fjmloqxkzb lc qlrze zxjb fkql clzrp qtl vbxop xdl xp qeb tloia zlmba tfqe qeb fplixqflk fjmlpba yv ZLSFA-19. Jlkqep pmbkq xslfafkd exkapexhbp xka erdp clo cbxo lc xk fkcbzqflrp afpbxpb lkiv obfkclozba elt bppbkqfxi qebpb zlkkbzqflkp xob clo lro lsboxii ebxiqe.

The ciphertext is then put through the decryption program. The program succeeded in finding the original message as well as the encryption key.

```
Ciphertext                    :
Qeb fjmloqxkzb lc qlrze zxjb fkql clzrp qtl vbxop xdl xp qeb tloia zlmba tfqe qeb fplixqflk fjmlpba yv ZLSF
A-19. Jlkqep pmbkq xslfafkd exkapexhbp xka erdp clo cbxo lc xk fkcbzqflrp afpbxpb lkiv obfkclozba elt bppbk
qfxi qebpb zlkkbzqflkp xob clo lro lsboxii ebxiqe.

Most likely encryption key    : 23
Deciphered Ciphertext         :
The importance of touch came into focus two years ago as the world coped with the isolation imposed by COVI
D-19. Months spent avoiding handshakes and hugs for fear of an infectious disease only reinforced how essen
tial these connections are for our overall health.
```

Fig.11 Program output for key of 23 test case

F. *Key of 0*

A snippet of text from the article “How do you capture the ‘essence of touch’—especially during a pandemic?” is taken and put through a Caesar Cipher Encryptor using a key of 0, also known as key of 26, the plaintext as well as the ciphertext is as follows:

> We humans rely on a suite of cues to recognize our friends, such as their smiles, their voices, or the way they walk. Biologists have known for several decades that dolphins form close friendships, and that the cetaceans identify pals by their unique whistles. Now new surprising research suggests bottlenose dolphins use their sense of taste to discern their friends' urine from unrelated dolphins.

The ciphertext is then put through the decryption program. The program succeeded in finding the original message as well as the encryption key.

```
Ciphertext                   :
We humans rely on a suite of cues to recognize our friends, such as their smiles, their voices, or the way
they walk. Biologists have known for several decades that dolphins form close friendships, and that the cet
aceans identify pals by their unique whistles. Now new surprising research suggests bottlenose dolphins use
 their sense of taste to discern their friends' urine from unrelated dolphins.

Most likely encryption key   : 0
Deciphered Ciphertext        :
We humans rely on a suite of cues to recognize our friends, such as their smiles, their voices, or the way
they walk. Biologists have known for several decades that dolphins form close friendships, and that the cet
aceans identify pals by their unique whistles. Now new surprising research suggests bottlenose dolphins use
 their sense of taste to discern their friends' urine from unrelated dolphins.
```

Fig.12 Program output for key of 0 test case

G. *Key of 2 of Bahasa Indonesia Plaintext*

A snippet of text from the article "Hanya Menyeberangi Sungai Kecil, Julius Caesar Memulai Perang Panjang" is taken and put through a Caesar Cipher Encryptor using a key of 2, the plaintext is as follows:

> Meski tidak lebih dari sebuah aliran kecil, Rubicon memiliki arti pentingnya bagi Romawi. Sungai ini menandai perbatasan resmi antara Italia dan Cisalpine Gaul, wilayah selatan Pegunungan Alpen yang diperintah oleh Caesar.

And the generated ciphertext is as follows:

> Ogumk vkfcm ngdkj fctk ugdwcj cnktcp mgekn, Twdkeqp ogoknkmk ctvk rgpvkpipac dcik Tqocyk. Uwpick kpk ogpcpfck rgtdcvcucp tguok cpvctc Kvcnkc fcp Ekucnrkpg Icwn, ykncacj ugncvcp Rgiwpwpicp Cnrgp acpi fkrgtkpvcj qngj Ecguct.

The ciphertext is then put through the decryption program. The program failed in finding the original message as well as the correct encryption key.

```
Ciphertext                   :
Ogumk vkfcm ngdkj fctk ugdwcj cnktcp mgekn, Twdkeqp ogoknkmk ctvk rgpvkpipac dcik Tqocyk. Uwpick kpk ogpcpf
ck rgtdcvcucp tguok cpvctc Kvcnkc fcp Ekucnrkpg Icwn, ykncacj ugncvcp Rgiwpwpicp Cnrgp acpi fkrgtkpvcj qngj
 Ecguct.

Most likely encryption key   : 7
Deciphered Ciphertext        :
Hznfd odyvf gzwdc yvmd nzwpvc vgdmvi fzxdg, Mpwdxji hzhdgdfd vmod kziodibitv wvbd Mjhvrd. Npibvd did hziviy
vd kzmwvovnvi mznhd viovmv Dovgdv yvi Xdnvgkdiz Bvpg, rdgvtvc nzgvovi Kzbpipibvi Vgkzi tvib ydkzmdiovc jgzc
 Xvznvm.
```

Fig.13 Program output for key of 2 Bahasa Indonesia text test case

H. *Key of 24 of Bahasa Indonesia Plaintext*

A snippet of text from the article "Hanya Menyeberangi Sungai Kecil, Julius Caesar Memulai Perang Panjang" is taken and put through a Caesar Cipher Encryptor using a key of 24, the plaintext is as follows:

> Prajuritnya menjadi saksi bagaimana Caesar mengasah keterampilannya sebagai ahli strategi militer dan politik, serta menaklukkan Gaul. Caesar memperluas batas Republik Romawi sejauh Rhine dan sepanjang waktu menopang pengaruhnya kembali di Romawi.

And the generated ciphertext is as follows:

> Npyhspgrlwy kclhybg qyiqg zyeygkyly Aycqyp kcleyqyf icrcpykngjyllwy qczyeyg yfjg qrpyrceg kgjgrcp byl nmjgrgi, qcpry kclyijsiiyl Eysj. Aycqyp kckncpjsyq zyryq Pcnszjgi Pmkyug qchysf Pfglc byl qcnylhyle uyirs kclmnyle ncleypsflwy ickzyjg bg Pmkyug.

The ciphertext is then put through the decryption program. The program failed in finding the original message as well as the correct encryption key.

```
Ciphertext                   :
Npyhspgrlwy kclhybg qyiqg zyeygkyly Aycqyp kcleyqyf icrcpykngjyllwy qczyeyg yfjg qrpyrceg kgjgrcp byl nmjgr
gi, qcpry kclyijsiiyl Eysj. Aycqyp kckncpjsyq zyryq Pcnszjgi Pmkyug qchysf Pfglc byl qcnylhyle uyirs kclmny
le ncleypsflwy ickzyjg bg Pmkyug.

Most likely encryption key   : 20
Deciphered Ciphertext        :
Tvenyvmxrce qirnehm weowm fekemqere Geiwev qirkewel oixiveqtmperrce wifekem elpm wxvexikm qmpmxiv her tspmx
mo, wivxe qireopyooer Keyp. Geiwev qiqtivpyew fexew Vityfpmo Vsqeam wineyl Vlmri her witernerk aeoxy qirste
rk tirkevylrce oiqfepm hm Vsqeam.
```

Fig.14 Program output for key of 24 Bahasa Indonesia text test case

I. *Analysis*

From the testing done, the program is able to decrypt an English ciphertext encrypted using Caesar Cipher with an unknown key. From 5 random key, the program is able to figure out all of the encryption key used for each ciphertext. The program is also able to ascertain if the input is already in plaintext form as seen in figure 12, this way the program will not misbehave if the input is a regular plaintext. The program is unable to correcly determine the encryption key if the given ciphertext is not in English.

V. Conclusion

In this paper, a method of decrypting an unknown Caesar Cipher has been presented. The method is able to decrypt a Caesar Cipher ciphertext and figure out the encryption key used for encrypting said ciphertext. Experiment results show that if the original message's language matches the program's dictionary, it is able to decrypt the ciphertext accurately.

Video Link at YouTube

https://youtu.be/itvXB_7RZcA

Repository Link At GitHub

https://github.com/Noxira/CaesarBruteforce

Acknowledgment

I would like to express my great appreciation to Dr. Ir. Rinaldi Munir, M.T. for the knowledge he shared throughout the semester. I would also like to thank my families for providing me love and guidance for whatever I pursue, as well as my colleagues and friends for their support.

References

[1] Suetonius, Vita Divi Julii 56.6

[2] Liddell, Henry George; Scott, Robert; Jones, Henry Stuart; McKenzie, Roderick (1984). A Greek-English Lexicon. Oxford University Press

[3] Thomas W. Edgar, David O. Manz, in Research Methods for Cyber Security, 2017

[4] SearchSecurity. 2022. *What is cryptanalysis? Definition from SearchSecurity*. Available at:

https://www.techtarget.com/searchsecurity/definition/cryptanalysis. Accessed 20 May 2022.

[5] GeeksforGeeks. 2022. *Cryptanalysis and Types of Attacks - GeeksforGeeks*. Available at: https://www.geeksforgeeks.org/cryptanalysis-and-types-of-attacks. Accessed 20 May 2022.

[6] Munir, R., 2022. Informatika.stei.itb.ac.id. Available at: https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/stima21-22.htm. Accessed 20 May 2022.

[7] Developer.mozilla.org. 2022. *Regular expression syntax cheatsheet - JavaScript | MDN*. Available at: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Regular_Expressions/Cheatsheet#unicode_property_escapes. Accessed 20 May 2022.

[8] Leech, G., Rayson, P. and Wilson, A., 2001. *Word frequencies in present-day written and spoken English*. Harlow: Longman.

[9] GitHub. 2022. *GitHub - hermitdave/FrequencyWords: Repository for Frequency Word List Generator and processed files*. Available at: https://github.com/hermitdave/FrequencyWords. Accessed 20 May 2022.

LEGAL STATEMENT

I Hereby state that this paper is of my own writing, not an adaptation of another paper, not a translation of another paper, nor a result of plagiarism.

Bandung, 20 Mei 2022

Farrel Farandieka Fibriyanto, 13520054