Traditional Cryptography

Olaf van Waart and Julian Thijssen

February 8, 2015

1 Introduction

The human history has a long tradition of communicating confidential information by encrypting the text which then becomes incomprehensible. Only the intended receiver has the tools to decrypt this message. However traditional ways of encrypting text are not secure especially when attempts to break to code are aided by computational power. In this report we will show this by discussing and breaking several traditional codes.[1]

The goal of a traditional cipher is to scramble a plaintext in such a way that any interceptor of this cipher text can't make heads or tails of it. This encryption process should be designed to have a decryption process which reverses the encryption so the intended receiver can receive the cipher text, decrypt it, and read the original plaintext. A practical consequence of this is that before communicating any encrypted message the sender and receiver should have agreed upon encryption method. To be able to reuse encryption methods, different keys are used which alter the encryption process slightly so any non-intended receiver won't be able to decipher the cipher text easily even if he/she knows the encryption and decryption method. This key should also be communicated between sender and intended receiver and this is one of the weaknesses of a traditional cipher. However in this report we will focus on breaking the cipher text the hard way, without the possibility to intercept the key.

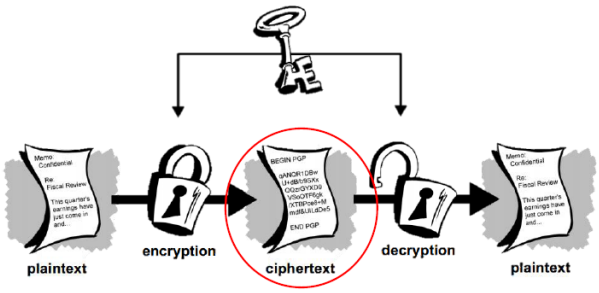


Figure 1: The process of encryption and decryption of plaintext

So an encryption process, given a plaintext (P), a key (K) gives the ciphertext (C). Thus:

$$E(P, K) = C$$

A decryption process is the exact opposite, given ciphertext and the key gives the original plaintext. Thus:

$$D(C, K) = P$$

Also this means that the process is reversible so:

$$D(E(K, P), K) = P$$

1.1 Caesar's Cipher

A well-known historic cipher is the cipher Caesar used to communicate. We can all imagine that Julius Caesar didn't want his tactical information to fall in the wrong hands. To make sure this didn't happen he devised a way to encrypt messages, and Caesar's cipher was born. The Caesar's cipher takes the characters from the plaintext, shifting the alphabet by a fixed number, the key, and replacing these characters by their shifted counterparts. For example a left shift 3 of plaintext "BABE" would become "YXYE".Because of this process the Caesar's cipher is also known as the shift cipher.

Another way to show this would be to give each of the 26 letters in the alphabet a number from 0 to 25 we would then be able to calculate the ciphertext characters (c_i), given the plaintext characters (p_i) and key (K).

$$c_i = (p_i + K) \ mod \ 26$$

There are only 26 possibilities in which this cipher can be shifted so one can imagine that such a cipher is easily crackable by trying all 26 possibilities, even without computational power this is doable.

2 Substitution cipher

The Caesar's cipher is an example of a substitution cipher, both substitute each character from the plaintext by a fixed character to make up the ciphertext. This fixed character is given by the key. In contrast to Caesar's cipher a substitution cipher doesn't have to hold to the original alphabet in the same order and can not only rearrange the alphabet but also use completely different characters like numbers. So while in a Caesar's cipher "BABE" could become "YXYE", or any of the other 25 possibilities, in a substitution cipher that uses only non-alphabetic characters "BABE" could for example become ";!;?".

The ciphertext we tried to break is an English text from the Gutenburg Project[2]. We first looked at the character frequencies in the ciphertext and compared this to the frequencies of the English language, for this we used charachter frequencies we extracted from several text from the Gutenberg Project.

The first question we needed to know does this text contain spaces or not. From comparing the frequencies of the English language it seems apparent that it indeed does contain spaces and that "P" from the ciphertext corresponds to "whitespace", from now on indicated by "␣", In the plaintext. We can also decipher from comparing the frequency tables that "R" most probably corresponds with "e" in the plaintext, because they are both the only characters around 0,10. The following characters all have frequencies close to each other, around

1	2	3
H	E	L
L	O	
W	O	R
L	D	

Table 1: "HELLO WORLD" with blocksize 3

0,07 – 0,06, and are thus too close to each other to reliably match chipertext with plaintext. To be able to decipher the following cipher text we should be looking for other statistic characteristics of the English language. One such following step could be to look for the most common bigrams, a combination of two characters like "he", or most common trigrams, a combination of three characters like "the", or most common words like "the". By looking at the partially deciphered ciphertext we noticed that there were a lot of occurences of "˙19e˙" and that the most common three letter word in English is "the" so from this follows that "1" corresponds to "t", and "9" to "h". From here we went step by step deciphering the ciphertext character by character from frequencies of characters, ngrams and words.

Using this method we were able to decipher the whole ciphertext except for ",U.GC" from which all characters only have a single occurrence.. But after searching for the decipher plaintext on Google we were able to find the original text, a piece from "BLACKWOOD'S Edinburgh MAGAZINE. VOL. LXV.", on the Gutenberg Project website. Then it became clear that ",U.GC" correspond to "1789".

When we started to break this ciphertext we knew that the language of the underlying plaintext was English. If this was not the case we would be able to calculate the index of coincidence which is a unique number for each language to recognize the underlying language. This only holds for a long enough ciphertext, words or even sentences are a lot harder to break this way, like we can see with the example of ",U.GC" which should decode to "1789". While we used a method which mostly was based on 'handwork' one could use statistical characteristics, or language model, to make a program which would automatically decipher a chipertext from a substitution cipher.

3 Permutation cipher

A permutation cipher uses blocks of a size b to chop the plaintext into pieces and then rearrange these individual pieces in the same way. One could best imagine this by visualizing a table with b columns and each row representing a new piece of the plaintext, like in table .. where "HELLO WORLD" is chopped in blocks of size 3.

The cipher would then shuffle the columns of this table, for example in the order of 312 which would give the ciphertext of "LHE LO RWO LD".

Because the plaintext needs to be chopped up in blocks of equal size the blocksize, b, is dependent on the length of the plaintext, L.

$$L \; mod \; b = 0$$

3	1	2
L	H	E
	L	O
R	W	O
	L	D

Table 2: "HELLO WORLD" with blocksize 3

One could also "pad" the plaintext so that $(L + p)\ mod\ b = 0$ but this padding might be easily recognized. Like the substitution cipher this, and all following ciphers we will try to break, was an English text from the Gutenberg Project. To break this cipher we first wanted to know all possible blocksizes we did this by trying all numbers between 1 and the length of the ciphertext, 2128, which gave us the possible blocksizes of:

$$[1, 2, 4, 7, 8, 14, 16, 19, 28, 38, 56, 76, 112, 133, 152, 266, 304, 532, 1064, 2128]$$

We then used the knowledge that "TH" is the most frequent bigram in the English language to calculate the most probable blocksize. We did this by calculating each 'anagram' of the blocksizes:

$$[1, 2, 4, 7, 8]$$

We used only these 5 to start off with to minimize computational power that would be needed to calculate these anagrams. The result of this calculation was:

$$\{1 : 24, 2 : 24, 4 : 24, 7 : 59, 8 : 27\}$$

Which led us to believe that a blocksize of 7 would be most probable, we then looked at the result this optimized on "TH" shuffle actually gave us and it was to our surprise really well readable: "BUT IN THE BRITISH EMPIRE, FOR A CENTURY PAST, IT HAS BEEN THOROUGHLY UNDERSTOOD, BY MEN OF SENSE OF ALL PARTIES, THAT A CHANGE OF DYNASTY ..."

A Google search on this text led us to the original plaintext, another section of the same "BLACKWOOD'S Edinburgh MAGAZINE. VOL. LXV." as the substitution cipher.

To solve this cipher we used a quite inefficient brute force method by calculating all possible ways to arrange each blocksize and then checking them for "TH" frequency. We also developed a method which would do this in a smarter way be rearranging each column according to the place where it results in the highest "TH" frequencies. This would result in a program which would be able to solve permutation ciphers quite reliably, especially when not only optimized for "TH" frequencies but all bigrams or even trigrams.

4 Substitution & Permutation cipher

The combination of the above two mentioned ciphers is called a substitution and permutation cipher, where the plaintext is both substituted and permutated. To break this ciphertext we first looked at the frequencies of the ciphertext. This showed us that "J" in the ciphertext corresponds to ˙ in the plaintext, both

probabilities close to 0,17, and “I” to “e”, both close to 0,10. This frequency analysis also showed that probably either “Z” or “ ’ ” corresponds to “t” we went on a limb and guessed that “Z” would correspond to “t”. So we substituted all “J” with ˙ all “I” with “e” and all “Z” with “t” which gave us a partially deciphered ciphertext which was in the wrong order due to the permutation. We then used the same method as with the permutation cipher but we couldn’t use the “TH” frequencies because we didn’t know which character corresponds to “H”. Instead we used the most common letters at the beginning and end of words which gave us the bigrams t˙, e˙ and ˙t. The length of the ciphertext is 2740 from which we calculated all possible blocksizes:

$$[1, 2, 4, 5, 10, 20, 137, 274, 548, 685, 1370, 2740]$$

We then used the blocksizes $[1, 2, 4, 5]$ to calculate the blocksize and order with the most occurrences of the bigrams: t˙, e˙ and ˙t. This would give us a partially deciphered ciphertext which we assumed would be in the right order, with blocksize 5 and order 43102. We then used the same technique as the substitution cipher to solve the remaining ciphertext. Which gave us a piece from “The Friends; Or the Triumph of Innocence”

5 Polyalphabetic ciphers

5.1 Alberti cipher

So far all the alphabets we have seen were monoalphabetic (i.e., the cipher uses a single alphabet for translation). In about 1466, Leon Battista Alberti invented one of the first polyalphabetic ciphers.[3] This cipher uses two alphabet disks, one stationary and one movable, that can be used to translate a plaintext message into an encrypted cipher text. To translate a message, one would look up each letter of their message on the uppercase alphabet that is listed on the outer disk. This letter would then correspond to a lowercase letter on the inner disk. However, by shifting the inner disk, the alphabet on it gets shifted a couple of places. The Alberti cipher used this mechanism to shift the alphabet along every couple of words, and so the message would be encrypted by using multiple alphabets.

A weakness of this cipher is that somehow the shifts in alphabets would need to be communicated to the reader. Therefore, an index letter was chosen on the inner ring that would indicate on the outer ring which position it corresponded to. This corresponding outer capital letter would then be inserted in the cipher text at the point where the shift happens. The result of this was a cipher text that was primarily written in lowercase, but occasionally contained an uppercase letter.

For someone unaware of the Alberti cipher trying to break this code, this cipher text would be hard to break. However, if the code breaker was aware of the method of the Alberti cipher, it would be plain to see that it was encoded in this way, and therefore extremely simple to break.

Figure 2: The Alberti cipher disk used to easily look up cipher text letters from plaintext letters (source Wikipedia)

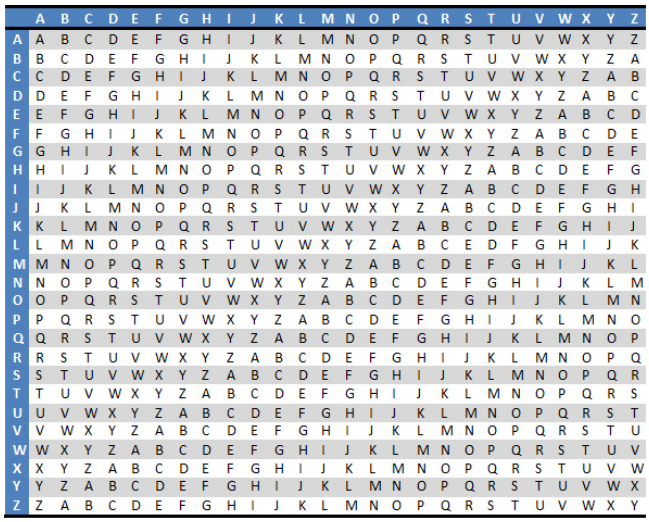
5.2 Trithemius cipher

An improvement to this cipher method was made by Johannes Trithemius is 1508. Trithemius used a square table of alphabets called a Tabula recta. This table contained the Latin alphabet on each row, but shifted by one place per row downwards. This created a table of 26 different alphabets, that can be used like a substitution cipher. By using the alphabet that is in the row below the previous alphabet every time a letter of plaintext was encoded, the security of this cipher was more vast than the simple monoalphabetic ciphers.

An advantage of using a clear system for switching alphabets was also that the sender did not have to communicate to the reader where the alphabet switches happened. The sender and receiver simply agree to use the Trithemius cipher for encryption. This means that any eavesdroppers have a much harder time to figure out the method of encryption, since any indication, like sporadic capital letters, are not present in the cipher text.[4]

6 Periodic, Polyalphabetic cipher

One of the major developments in cryptography happened when in 1553, Giovan Battista Bellaso started using a repeating key word to indicate a switch in cipher alphabet. The system functions in the same way as the Trithemius cipher, in that the alphabet is switched after every letter. However, the next alphabet is not determined by a Tabula recta, but by the corresponding letter in the repeating key word.



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

Figure 3: Tabula recta (source: www.impsglobal.info)

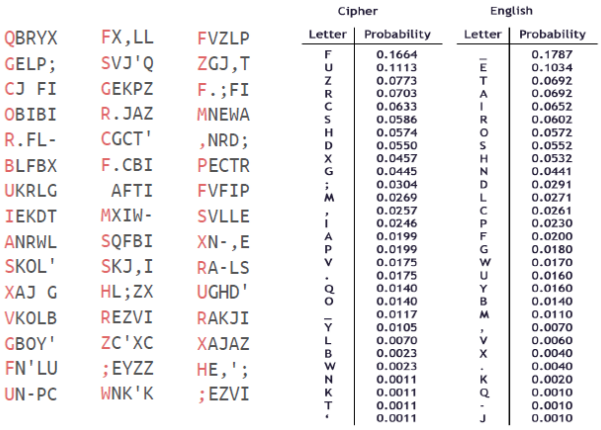
The Vigenère cipher is an example of a polyalphabetic cipher. It is polyalphabetic, because it uses multiple alphabets in order to encode the plaintext. It is periodic, because the key it uses for encryption is repeated multiple times along the plaintext.[4]

Periodic ciphers are generally easier to break than aperiodic ciphers, but the idea of using a key significantly boosts the security of the encryption. If an evil interceptor managed to get a hold of your ciphertext, there would be no visible indication of the encryption algorithm used as with the Alberti cipher. Nor would, by assuming it is encrypted with a Trithemius cipher, the code breaker be able to decode the cipher text immediately. In fact, by increasing the length of the key, the possible alphabet combinations used by this cipher increase dramatically. Given a sufficient length key this cipher would be nigh impossible to break in the early days of cryptography. It is therefore no surprise that the Vigenère cipher earned the title of 'The Indecipherable Cipher'.

The primary concern in breaking a periodic polyalphabetic cipher is finding the period at which the alphabets are repeated. If the original plaintext was written in a natural language, finding the period N would allow a code breaker to take every Nth letter of the cipher text and perform a frequency analysis on it. This analysis should match the frequency distribution of the original plaintext language as long as the message is of at least moderate size. Abusing this weakness in the cipher, one can then solve the cipher in much the same way as a polyalphabetic substitution cipher.

6.1 The breaking of a periodic polyalphabetic cipher

We were kindly provided with a periodic polyalphabetic cipher to break. We followed the above structure after reading some literature on the cipher technique.[5][6]



```
QBRYX   FX,LL   FVZLP
GELP;   SVJ'Q   ZGJ,T
CJ FI   GEKPZ   F.;FI
OBIBI   R.JAZ   MNEWA
R.FL-   CGCT'   ,NRD;
BLFBX   F.CBI   PECTR
UKRLG    AFTI   FVFIP
IEKDT   MXIW-   SVLLE
ANRWL   SQFBI   XN-,E
SKOL'   SKJ,I   RA-LS
XAJ G   HL;ZX   UGHD'
VKOLB   REZVI   RAKJI
GBOY'   ZC'XC   XAJAZ
FN'LU   ;EYZZ   HE,';
UN-PC   WNK'K   ;EZVI
```

Cipher		English	
Letter	Probability	Letter	Probability
F	0.1664	_	0.1787
U	0.1113	E	0.1034
Z	0.0773	T	0.0692
R	0.0703	A	0.0692
C	0.0633	I	0.0652
S	0.0586	R	0.0602
H	0.0574	O	0.0572
D	0.0550	S	0.0552
X	0.0457	H	0.0532
G	0.0445	N	0.0441
;	0.0304	D	0.0291
M	0.0269	L	0.0271
,	0.0257	C	0.0261
I	0.0246	P	0.0230
A	0.0199	F	0.0200
P	0.0199	G	0.0180
V	0.0175	W	0.0170
.	0.0175	U	0.0160
Q	0.0140	Y	0.0160
O	0.0140	B	0.0140
_	0.0117	M	0.0110
Y	0.0105	'	0.0070
L	0.0070	V	0.0060
B	0.0023	X	0.0040
W	0.0023	.	0.0040
N	0.0011	K	0.0020
K	0.0011	Q	0.0010
T	0.0011	-	0.0010
'	0.0011	J	0.0010

Figure 4: Taking every Nth letter from a cipher text can reveal a frequency distribution close to a natural language

The first step was to determine the period. We wrote a program that would step through the complete cipher text and look at repeating sequences of characters. These repeating sequences can be purely accidental or the result of the same slice of plaintext occurring twice over the same slice of key. However, if they were the result of the latter the distances between two repeating sequences would be some multiple of the period used to encode the message. Our program tallies these sequences and ranks them according to how often they occur in the cipher text. As it turned out the most common sequences all had distances that were a multiple of five. By this point we could be quite sure the period used to encode the message would have been five.

The next step was to look at every fifth letter of the cipher text (that would have been encrypted using the same alphabet) and determine their character frequencies. We wrote a program that calculated these frequencies for every fifth letter starting at the offsets 0, 1, 2, 3 and 4. This gave us the frequency distribution of all the five alphabets that were used. It was immediately obvious that we were on the right track, because the character frequencies matched very closely to the frequencies of the English language.

By this point, if a tabula recta were used to encode the plaintext it would have been a breeze to find the original message. This is due to the extreme disparity between the frequency of the whitespace character and the next most common character in the English language. Basically, there was a very clear indication of which character mapped to the whitespace character in each of our deduced alphabets.
Since a tabula recta does no scrambling of the alphabets, once you know the position of one of the characters, you can extrapolate the rest of the alphabet.

However, the alphabets used to encode the message did turn out to be scrambled. This forced us to use the methods normally used to break substitution

ciphers (frequency, n-grams) to break what turned out to be a substitution cipher using five completely random alphabets. Luckily, cracking five alphabets in this way is still doable by hand although quite tedious.

7 Running key cipher

The running key cipher is a bit different from the ciphers previously discussed. Instead of substituting a letter of the plaintext for a letter in some alphabet, an operation is performed between the plaintext and the key that results in a cipher text. Such an operation could be anything, most commonly an "exclusive or" or addition of the ASCII values of the plain text and key. A running key cipher is different from the ciphers seen so far in that it is aperiodic. The key is not short and repeated along the plaintext, but instead is a string of the same length as the plaintext. Since for long messages this string would be hard to remember for the receiving party, it is typically chosen to be a snippet from a book. In this way, the book that is used and the page from which the snippet originates can be shared in advance. From there it is quite easy for the receiving party to find or remember the right key and decipher the message.

The fact that books are written in a natural language unfortunately also forms the downfall of this encryption technique. If the cipher text is sufficiently long, one can assume the original plaintext contains common bigrams and trigrams such as "to", "and", "the". You might also assume that the ciphertext is some kind of addition or modular addition of a plaintext and key by looking at the resulting characters. For example, if the resulting ciphertext is a bunch of unreadable symbols, the ASCII values of the plaintext and key might have been added up.

Due to the process described earlier (if you have the plaintext and ciphertext, you can determine the key), once the eavesdropper knows the manner in which the plaintext is encrypted, he can try putting some of these common n-grams at every position of the plaintext and see parts of the key used to encrypt the original plaintext. What he would be looking for is parts of words that resemble the original plaintext language. If by luck one of these n-grams is put at the same position of the n-gram in the original plaintext, the key at that position is revealed. From there on the code breaker could guess what bigger word this slice would have been part of. If he guesses right this could then by used in reverse fashion to get a slice of the same size as this word of the original plaintext. This technique is called zig-zagging and it is plain to see why.

Now the code breaker might have gathered a bunch of small phrases from both the plaintext and key. Since the key is typically taken from a book, it very simple to find which book it is from, especially in this day and age.
Interestingly if the key of the running key cipher were a completely random sequence, this cipher technique becomes a one-time pad, an encryption that is proven to be unbreakable as long as the key is only used once.

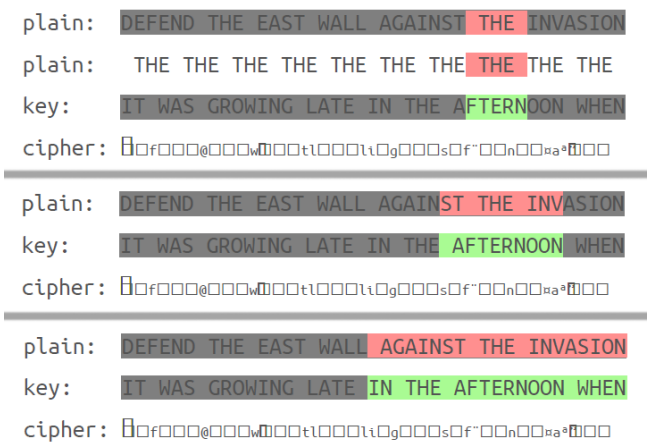


Figure 5: Zig-zagging from a seed word to obtain more information

8 Conclusion

In short, we have summarized the techniques, strengths and weaknesses of some of the most important traditional cryptographic ciphers. In general, to obtain increased security it is important to use a key for encryption. This key should be quite random, and of adequate size.

Many of these technique are inherently unsafe and should not be used alone in practice if the user wants to securely send a message. However, when the principles of the traditional ciphers are combined they can form a much safer way of encryption.

9 Acknowledgements

We would like to thank Mathias Winther Madsen for providing ciphertexts for most of the cipher techniques discussed in this report. These ciphertexts allowed us to explore the strengths and weaknesses in each of these and to learn about ways to break them.

References

[1] Madsen, Mathias *http://informationtheory.weebly.com/presentation-topics.html*

[2] The Gutenberg Project, *http://www.gutenberg.org/*

[3] Servos, William, *Using a genetic algorithm to break Alberti Cipher*. Journal of Computing Sciences in Colleges, 2004.

[4] Borda, Monica, *Fundamentals in information theory and coding pages 132-135*. 2011.

[5] *http://www.umich.edu/ umich/fm-34-40-2/ch8.pdf.*

[6] *http://www.umich.edu/ umich/fm-34-40-2/ch9.pdf.*