Scribe: Cryptography and Network Security (Class.3.C)

Aditya Anand

22-Sep-2020

1 Introduction

In this lecture we mainly look at the cryptanalysis of the Vignere cipher. We shall also look a simple attack on the Hill Cipher. The Vignere cipher is known for being unbreakable for over 300 years since its introduction in 1553, until it was finally broken in 1863. Here we discuss a statistical attack on the Vignere cipher. Yet again, this emphasizes the need for provably secure encryption schemes, and why a seemingly complicated cipher may not be secure in practice.

2 The Vignere Cipher

The Vignere cipher is an extension of the Caesar Shift Cipher - only instead of using a single shift value, it uses a keyword of a fixed length m, each entry of which is (possibly), a different shift value. The keyword is then repeatedly used to encrypt the plaintext. Consider a plaintext with lowercase English alphabets. Suppose that we denote the index of the i^{th} character of the message as m_i, where $m_i \in \{0, 1, ..26\}$. Then the i^{th} character of the ciphertext is given as $c_i = (m_i + k_{i \ mod \ m}) \ (\text{mod } 26)$.

For the purposes of any attack, we will assume that the keyword length m is unknown to the attacker.

3 Cryptanalysis of Vignere Cipher

For any natural number n, we denote by $[n]$ the set $\{1, 2, ...n\}$. The cryptanalysis of the Vignere Ciphere involves two major steps: In the first step, we determine the length m of the keyword, and in the second, we recover the keyword itself, knowing the length.

3.1 Determining the keyword length

First, we attempt to guess the keyword using a simple observation based on how the Vignere Cipher encrypts messages.

Observation 1. *Consider two identical characters at different positions i and j, $i < j$ in the message. If $m \mid j - i$, then the ciphertext characters at these positions must be identical.*

The Kaisinski test aims to exploit this property. We find a large number n of pairs of identical segments in the ciphertext. Suppose that the spacing between their starting positions is δ_i for $i \in [n]$. We then expect that m should divide each δ_i (though it is not guaranteed), and hence that m should divide $gcd((\delta_i)_{i\in[n]})$. Since this test is a heuristic, we need a more foolproof way to determine and verify if an obtained value of m is indeed correct.

For this purpose, we define a useful quantity called the index of coincidence. The index of coincidence captures the distribution of the characters in the text, without concerning the identity of the letters. In particular, this means, if you apply a shift cipher to a given text, then the index of coincidence remains unchanged. The gap between the index of coincidence for typical English text and random text allows us to determine if a given value of m is correct.

Definition 1. *Given a string x, the index of coincidence $I_c(x)$ is the probability that two random elements of x are identical.*

Suppose the frequency of the $(i+1)^{th}$ English letter i in a string x of length n is given by $f_i, i \in \{0, 1, 2, \ ... \ 25\}$. Then $I_c(x) = \frac{\sum_{i=0}^{25} \binom{f_i}{2}}{\binom{n}{2}}$, using elementary probability. We may approximate $I_c(x)$ as $\sum_{i=0}^{25} \frac{f_i^2}{n^2}$. Defining $p_i = \frac{f_i}{n}$, we have $I_c(x) = \sum_{i=0}^{25} p_i^2$.

The probability of occurrence for every character is well known for typical English text and it turns out that I_c value for typical English text is around 0.065. For random text x, $I_c(x) = \sum_{i=0}^{25} (\frac{1}{26})^2 = \frac{1}{26} \sim 0.038$. Thus given a text, we are able to determine, if the letters follow the distribution of typical English text or not, looking at how close the I_c value is, to 0.065.

Now suppose the ciphertext for some candidate value of m is given by $y_0y_1y_2..y_m..y_{2m}..$. We obtain the m strings $s_1 = y_0y_my_{2m}..$, $s_2 = y_1y_{m+1}y_{2m+1}..$ so on till $s_m = y_{m-1}y_{2m-1}...$ Notice that if the value of m is indeed correct, then in each string s_j, $j \in [m]$, the characters have been shifted by the same amount as compared to the original message. In turn this means that the I_c value for each of these strings is likely to be close the typical value of 0.065, since I_c value remains unchanged under the shift operation.

Thus we adopt one of two approaches. We look at candidate values of m from the Kaisinski test, and decide the correct value of m looking at the I_c values. Otherwise, we may completely avoid the Kaisinski test, and iterate over $m = 1, 2, \ \ldots$ and find the right value of m by looking at the I_c values.

3.2 Determining the keyword

Now that we know the length of the keyword, our next goal is to determine the keyword itself. This approach is again based on a statistical analysis using an index of coincidence between two different strings.

Definition 2. *Given two strings x and y, the mutual index of coincidence $MI_c(x, y)$ is the probability that a random of element of x is identical to a random element of y.*

How do we compute $MI_c(x, y)$? Again, it follows from elementary probability. Let us denote by f_i and f_i' the frequencies of the i^{th} letter in x and y respectively (where a is the 0^{th} letter). Further, let n and n' be the length of the strings x and y respectively. Then it follows that

$$MI_c(x, y) = \frac{\sum_{i=0}^{25} f_i f_i'}{nn'}$$

Recall the definitions of the strings $s_1, s_2...s_m$ defined above, where each s_i has been shifted by the same amount. Suppose that the key entry corresponding to string i is k_i, i.e. each element of s_i is shifted by the same amount k_i as compared to the plaintext. The key idea is to compute the mutual index of coincidence for two such strings s_i and s_j.

Notice that if p_i is the probability of occurrence of a letter i in English text, then we may write $MI_c(y_i, y_j) = \sum_{h=0}^{25} p_{h-k_i} p_{h-k_j} = \sum_{h=0}^{25} p_h p_{h+k_i-k_j}$.

In other words, the mutual index of coincidence between s_i and s_j depends only on the difference $k_i - k_j$. Once $k_i - k_j$ is fixed, so is $MI_c(s_i, s_j)$. Now we further observe that if $k_i = k_j$, then $MI_c(s_i, s_j) = \sum_{h=0}^{25} p_h^2 = 0.065$, the typical value for English text. Otherwise if $k_i - k_j \neq 0$, the values are closer to 0.04, the value for random text. This gives us a way to distinguish if $k_i = k_j$. How can we exploit this to find the value of $k_i - k_j$?

The answer to that question is straightforward. Since we can only detect a difference of zero, we manually shift one of the strings (say s_j) by an additional amount $g \in \{0, 1, 2...25\}$. We then find the value of g say g^*, that gives us the value of the mutual index of coincidence closest to 0.065. It follows that $k_i = k_j + g^*$, and hence we find $k_i - k_j = g^*$.

Once we find $k_i - k_j$ for each pair of strings, we can shift them appropriately so that the whole ciphertext now corresponds to a shifted version of the plaintext, and hence we may easily decrypt the ciphertext (this is similar to a Caesar Shift cipher).

4 Attack on Vignere Cipher: an example

Consider the ciphertext:

CHREEVOAHMAERATBIAXXWTNXBEEOPHBSBQMQEQERBWRVXUOAKXAOSXX
WEAHBWGJMMQMNKGRFVGXWTRZXWIAKLXFPSKAUTEMNDCMGTSXMXBT
UIADNGMGPSRELXNJELXVRVPRTULHDNQWTWDTYGBPHXTFALJHASVBFXN
GLLCHRZBWELEKMSJIKNBHWRJGNMGJSGLXFEYPHAGNRBIEQJTAMRVLCRR
EMNDGLXRRIMGNSNRWCHRQHAEYEVTAQEBBIPEEWEVKAKOEWADREMM
TBHHCHRTKDNVRZCHRCLQOHPWQAIIWXNRMGWOIIFKEE

We observe that the text CHR repeats multiple times in the ciphertext. Observing that these occurrences are 165, 235 and 285 positions apart, we get from the Krasinski test that the kyword length must divide their gcd, and hence divide 5. Therefore we suspect that the keyword length is either 1 or 5.

The index of coincidence is closest to 0.065 for $m = 5$,and not close to 0.065 for $m = 1$. We conclude that the keyword length $m = 5$.

To find exactly the key, we write down the strings s_1, s_2, s_3, s_4, s_5, for positions which are $0, 1, 2, 3, 4$ modulo 5. The strings start as $s_1 = CVA...$, $s_2 = HOE....$ etc. At this stage, we find the relative shifts using the MI_c values as described earlier. We get a shift vector relative to the first string s_1 as $(0, 9, 22, 5, 16)$. We then arrange all except the first string so that they are shifted by the same amount relative to the first string. It only remains to find this shift. One way is to go through all 26 shifts and find the best by a brute force. Alternatively, we find the mutual index of coincidence with any known piece of English text by shifting the ciphertext by each integer $g \in \{0, 1, 2..25\}$, and choose that value of g which makes the MI_c value closest to 0.06. We thus find that the ciphertext must be shifted by 17 units to find the plaintext. After decryption, the text looks like this:

The almond tree was in tentative blossom. The days were longer often ending with magnificent evenings of corrugated pink skies. The hunting session was over with hounds and guns put away for six months. The vineyards were busy again as the well organized farmers treated their vines and more lackadaisical neighbors hurried to do the pruning they should have done in November.

5 A simple attack on the Hill cipher

In this section we look at an attack on the Hill cipher. The Hill Cipher works as follows - on input a vector $X = (x_1, x_2..x_m)^T$, compute a vector $Y = (y_1, y_2..y_m)^T$ as the simple matrix multiplication $Y = AX$. All computations are modulo 26, and hence the key matrix A is chosen arbitrarily but for the restriction that $gcd(det(A), 26) = 1$.

The Hill Cipher has a very large key space of 26^{m^2}, since the matrix A has m^2 many entries. The methods that we studied in the previous section, all rely on statistics of English text, allowing possibly for shifts/permutations. But the Hill Cipher does not preserve any such statistical property, and thus this attack does not work either. These difficulties, in general, make a ciphertext only attack against the Hill Cipher difficult.

However, the Hill Cipher is not immune to known plaintext attacks. Suppose we know a set of plaintext row vectors X^* and corresponding ciphertexts Y^*. Then we must have $Y^* = X^*A^T$ and it is easy to compute A^T as $A^T = (X^*)^{-1}Y^*$. The only catch in this otherwise straightforward attack is that X^* may not be invertible, due to linear dependence, in which case we would need more plaintext-ciphertext pairs to find a suitable invertible matrix.

6 Conclusion

In this lecture we looked at cryptanalysis of the Vignere Cipher and a simple attack on the Hill Cipher. The Vignere Cipher attack used statistical methods, and the distribution of English text letters in typical messages. We used a preliminary test called the Kaisinski test, to guess candidate values for the keyword length. We then used the mutual index of coincidence, a statistical property that is an invariant under permutation/shifts to determine exactly the length of the keyword. Next, we defined the mutual index of coincidence, that helped us determine the relative shifts between different positions of the keyword, allowing us to break the Vignere Cipher. Finally, we studied a simple known plaintext attack on the Hill Cipher, by solving a system of linear equations and matrix inversion.