(10)

Cryptanalysis of a Vigenère:

A Vigenère cipher can be broken using the techniques of breaking a shift cipher as soon as the length of the keyword is known. This is because each repeat of a letter in the keyword corresponds to a single shift cipher. There are two ways to determine the keylength.

Kasiski Test (1863 - Major F.W. Kasiski, German cryptologist): Length of keyword is a divisor of the gcd of the distances between identical strings of length at least 3.

Friedman Test (1925, Colonel William Frederick Friedman (1891-1969)) also called the *Kappa Test*:

The **index of coincidence** of a message is the probability that a randomly chosen pair of letters in the message are equal. If the message has length n and n_i denotes the number of occurrences of the i^{th} letter then the index, denoted by I, is given by:

$$I = \frac{\sum_{i=1}^{26} n_i(n_i - 1)}{n(n-1)}$$

Now we can also calculate this index for any language source if we know the probabilities of occurrence of each of the letters. Thus, if p_a is the probability of occurrence of the letter a, for example, then we get:

$$I_{Source} = p_a p_a + p_b p_b + \ldots + p_z p_z = \sum_{i=a}^{z} p_i^2$$

Using our knowledge of these probabilities we can easily calculate that $I_{English} \sim 0.065$ and if we had a random source of English letters then $I_{Random} \sim 0.038$ (= 1/26).

This index can give information about a message. For instance, if a ciphered message was either a transposition or a monoalphabetic substitution then one would expect to have $I_{Message} \sim I_{English}$, but if a polyalphabetic substitution was used then this value should decrease (but

no lower than 0.038) since the polyalphabetic procedure tends to randomize the occurrences of the letters.

Let us now apply this index to a Vigenère ciphertext. If the ciphertext has length n and the keyword has length k (and n >> k) then in the positions corresponding to the same letter of the keyword, the ciphertext has been created with a monoalphabetic substitution, so if one were to calculate the index of just those positions, we should get 0.065. On the other hand if one were to calculate the index using only pairs from different letters of the keyword, the index would be much lower (0.038 if the keyword letters were randomly chosen). We may therefore calculate the expected number (A) of pairs of equal letters in the following way:

Pick a letter from the ciphertext (n choices), there are (n/k - 1) remaining letters that have used the same keyword letter [we are neglecting round-off error] and so,

$$\frac{n(\frac{n}{k}-1)}{2} = \frac{n(n-k)}{2k}$$

pairs of this type. There are (n - n/k) remaining letters that have used a different keyword letter [assuming the keyword letters are all distinct], and so there are

$$\frac{n(n-\frac{n}{k})}{2} = \frac{n^2(k-1)}{2k}$$

pairs of this type. Therefore,

$$A = \frac{n(n-k)}{2k}(0.065) + \frac{n^2(k-1)}{2k}(0.038)$$

and so,

$$I_{\text{Ciphertext}} = \frac{A}{\binom{n}{2}} = \frac{n-k}{k(n-1)}(0.065) + \frac{n(k-1)}{k(n-1)}(0.038) = \frac{1}{k(n-1)}[0.027n + k(0.038n - 0.065)]$$

from which we may solve for k (keyword length):

$$k = \frac{0.027n}{(n-1)I_{Ciphertext} - 0.038n + 0.065}$$

Machines used to encipher and decipher.

- Hagelin (US Army)
- Enigma (German Army and Navy)
- Purple (Japanese Diplomatic)

One-Time Pad (Vernam Cipher)

- Vigenère Cipher with keylength = size of message and the key is formed randomly.
- *Perfect Security*, it can not be broken. Used in the Washington-Moscow Red Phone line.
- Disadvantage, both parties must have the same key. (Books of random numbers)

Modern Cryptanalysis

The "philosophy" of modern cryptanalysis is embodied in Kerckhoffs' principle, which was formulated in the book *La cryptographie militaire* (1883) by the Dutch philologist Jean Guillaume Hubert Victor François Alexandre Auguste Kerckhoffs von Nieuwenhof, as he is called in all his full glory.

Kerckhoffs' Principle: *The security of a cryptosystem must not depend on keeping secret the crypto-algorithm. The security depends only on keeping secret the key.*

Example :

Applying Kasiski Test to attack Vigenere Cipher.

A Kasiski test can be used to attack the Vigenere Cipher. The Kasiski test is used to break polyalphabetic ciphers. In this example the plaintext is encrypted with Vigenere Code using the keyword "tree"

```
TreeTreeTreeTreeTreeTreeTreeTreeTreeTreeTreeTreeTreeTreeTreeTreeT
reeT

In the forest there are many trees with the same height. For example
many

be xav jhiiwm xlxii tii frrc kvixj abkl myi lrqi yimzyx. wsv vbefgpi
derr


        eeTreeTreeTreeTreeTreeT
        trees grow to ten feet.
         wvxvw zisa ks mvr yvix.
```

1) Count the number of spaces between repeated ciphertext. The letter ii appear three times. Between the first and second ii there are eight spaces and four spaces between the second and third ii. By finding the greatest common factor, the keyword is known to be four letters. In order to finish solving, four frequency analyses are done at once.

```
        be xav jhiiwm xlxii tii frrc kvixj  abkl myi lrqi  yimzyx.
wsv vbefgpi derr xvxvw zisa ks mvr yvix.
```

2) Break the ciphertext into four columns putting the first letter in the first column, second letter in second column, etc, leaving empty spaces between ciphertext letters.

```
        1. b a h m x t f   x b m l   z     f   r x z   m y   x = 3  m
= 3, f = 2, b = 2
        2. e v i   i i r k j k y r y y w v g d   v i k v v   i = 4, v
= 4, y = 3, k = 3
        3.     i x i i r v   l i q i x s b p e x w s s r i   i = 6, x
= 3, s = 3, r = 2
        4. x j w l     c i a     i m . v e i r v   a     x   i = 3, v
= 2, a = 2, x = 2
```

3) Since E and T are the most common letters

```
Column 1 x = E
Column 1 m = T

Column 2 v = E
Column 2 k = T

Column 3 i = E
Column 3 x = T

Column 4 i = E
Column 4 x = T

*Note that the capital letters stand for plaintext.
 **Some trial and error is necessary.

be TaE jhiEwT TlEiE tiE frrc TvEEj  abTl TyE lrqE  yEmzyT.
wsv EbefgpE derr TvEEw zisa Ts TEr yEET.
```

4) The second and ninth words look like "the".

```
Column 1 a = H
Column 2 y = H

be THE jhiEwT TlEiE tiE frrc TvEEj  abTl THE lrqE  HEmzHT.
wsv EbefgpE derr TvEEw zisa Ts TEr yEET.
```

5) The 11th word looks like "height". The 17th word looks like "to".

```
Column 1 z = G
Column 2 s = O
Column 4 m = I

be THE jhiEwT TlEiE tiE frrc TvEEj  abTl THE lrqE  HEIGHT.
wOv EbefgpE derr TvEEw GiOa TO TEr yEET.
```

6) The 18th word looks like "ten". The 19th word must be "feet".

```
Column 1 y = F
Column 3 r = N


be THE jhiEwT TlEiE tiE frNc TvEEj  abTl THE lrqE  HEIGHT.
wOv EbefgpE derr TvEEw GiOa TO TEN FEET.
```

7) The fourth word looks like "there".

```
        Column 2 i = R
        Column 4 l = H

        be THE jhREwT THERE tRE frNc TvEEj  abTl THE lrqE  HEIGHT.
wOv EbefgpE derr TvEEw GROa TO TEN FEET.
```

8) Fifth word looks like "are" and 16th word is "grow"

```
        Column 1 t = A
        Column 4 a = W

        be THE jhREwT THERE ARE frNc TvEEj  WbTl THE lrqE  HEIGHT.
wOv EbefgpE derr TvEEw GROW TO TEN FEET.
```

9) The third word is "forest". The 7th and 15th words are "trees".

```
        Column 1 h = O
        Column 2 j = S
        Column 3 w = S
        Column 3 v = R
        Column 4 w = S
        Column 4 v = R
        Column 4 j = F

        be THE FOREST THERE ARE frNc TREES  WbTl THE lrqE  HEIGHT.
wOR EbefgpE derr TREES GROW TO TEN FEET.
```

10) The first word in context is "in" and the 7th word is "with"

```
        Column 1 b = I
        Column 2 e = N
        Column 4 l = H
        IN THE FOREST THERE ARE frNc TREES  WITH THE lrqE  HEIGHT.
wOR EbefgpE derr TREES GROW TO TEN FEET.
```

11) The 12th word is "for" and the 13th word is "example"

```
        Column 1 f = M
        Column 2 w = F
        Column 2 g = P
        Column 3 b = X
        Column 3 p = L
        Column 4 e = A

        IN THE FOREST THERE ARE MrNc TREES  WITH THE lrqE  HEIGHT.
FOR EXAMPLE derr TREES GROW TO TEN FEET.
```

12) The sixth word is "many".

```
        Column 3 l = H
        Column 4 a = W

        IN THE FOREST THERE ARE MANY TREES  WITH THE lAqE  HEIGHT.
FOR EXAMPLE derr TREES GROW TO TEN FEET.
```

13) The tenth word is "same".

```
        Column 1 l = S
        Column 3 q = M

        IN THE FOREST THERE ARE MANY TREES  WITH THE SAME  HEIGHT.
FOR EXAMPLE derr TREES GROW TO TEN FEET.
```

14) The last word is difficult to find. In the context, it is "many".

```
        Column 1 r = Y
        Column 2 d = M
        Column 3 e = A
        Column 4 r = N

        IN THE FOREST THERE ARE MANY TREES  WITH THE SAME  HEIGHT.
FOR EXAMPLE MANY TREES GROW TO TEN FEET.
```

Example:

Index of Coincidences

The breaking of a Beaufort Cipher. The same method can be used for Vigenère – or indeed any time when a repeating is key applied letter by letter.

Suppose that we wished to crack the following message:

```
VKMHG QFVMO IJOII OHNSN IZXSS CSZEA WWEXU
LIOZB AGEKQ UHRDH IKHWE OBNSQ RVIES LISYK
BIOVF IEWEO BQXIE UUIXK EKTUH NSZIB SWJIZ
BSKFK YWSXS EIDSQ INTBD RKOZD QELUM AAAEV
MIDMD GKJXR UKTUH TSBGI EQRVF XBAYG UBTCS
XTBDR SLYKW AFHMM TYCKU JHBWV TUHRQ XYHWM
IJBXS LSXUB BAYDI OFLPO XBULU OZAHE JOBDT
ATOUT GLPKO FHNSO KBHMW XKTWX SX
```

We might begin by trying to establish the key length. This is done by using the 'Index of Coincidences'.

Essentially, the message is 'shifted' along, compared with the original, and the number of matches are counted.

```
Original: VKMHGQFVMOIJOIIOHNSNIZXSSCSZEA...
 Shift 1: KMHGQFVMOIJOIIOHNSNIZXSSCSZEAW...
 Shift 2: MHGQFVMOIJOIIOHNSNIZXSSCSZEAWW...
 Shift 3: HGQFVMOIJOIIOHNSNIZXSSCSZEAWWE...
 Shift 4: GQFVMOIJOIIOHNSNIZXSSCSZEAWWEX...
 Shift 5: QFVMOIJOIIOHNSNIZXSSCSZEAWWEXU...
 Shift 6: FVMOIJOIIOHNSNIZXSSCSZEAWWEXUL...
 Shift 7: VMOIJOIIOHNSNIZXSSCSZEAWWEXULI...
```

When this is done for the full message above, we get the following number of coincidences:

Shift	Coincidences
1	8
2	12
3	11
4	13
5	9
6	25
7	11

Notice the big leap in coincidences at a shift of 6? There is another leap (though not as noticeable) at a shift of 12 too. This implies that the key is 6 letters long.

If this isn't immediately clear, then consider the what is happening. If we match a random string of letters against another random string, we would expect roughly 1 match in 26 letters. So in 260 letters we'd expect roughly 10 coincidences (though we would not be surprised by deviations from this).

English, or any language, does not consist of a random string of letters. A random English sentence compared to another English sentence would see more matches as *some letters are more common* than others.

If we shift the ciphertext against itself by a distance equal to the size of the key, we are comparing letters which are encoded using the same key letter. Therefore 'e' will be encoded the same way each time – as will the other letters. So we'd expect more matches than for an arbitrary shift (where 'e' will have been encoded using different key letters).

Once the key length is established, the rest of the decode should come quite readily – but I'll save this for a later article.