Traditional Symmetric-Key Ciphers

Table of Contents

[General Structure 2](#_Toc133109901)

[Cryptanalysis 2](#_Toc133109902)

[Categories of Traditional Ciphers 3](#_Toc133109903)

[Monoalphabetic Substitution Ciphers 4](#_Toc133109904)

[Additive Ciphers 4](#_Toc133109905)

[Multiplicative Ciphers 5](#_Toc133109906)

[Affine Ciphers 6](#_Toc133109907)

[Polyalphabetic Substitution Ciphers 6](#_Toc133109908)

[Autokey Ciphers 7](#_Toc133109909)

[Playfair Ciphers 7](#_Toc133109910)

[Vigenère Cipher 9](#_Toc133109911)

[Hill Cipher 10](#_Toc133109912)

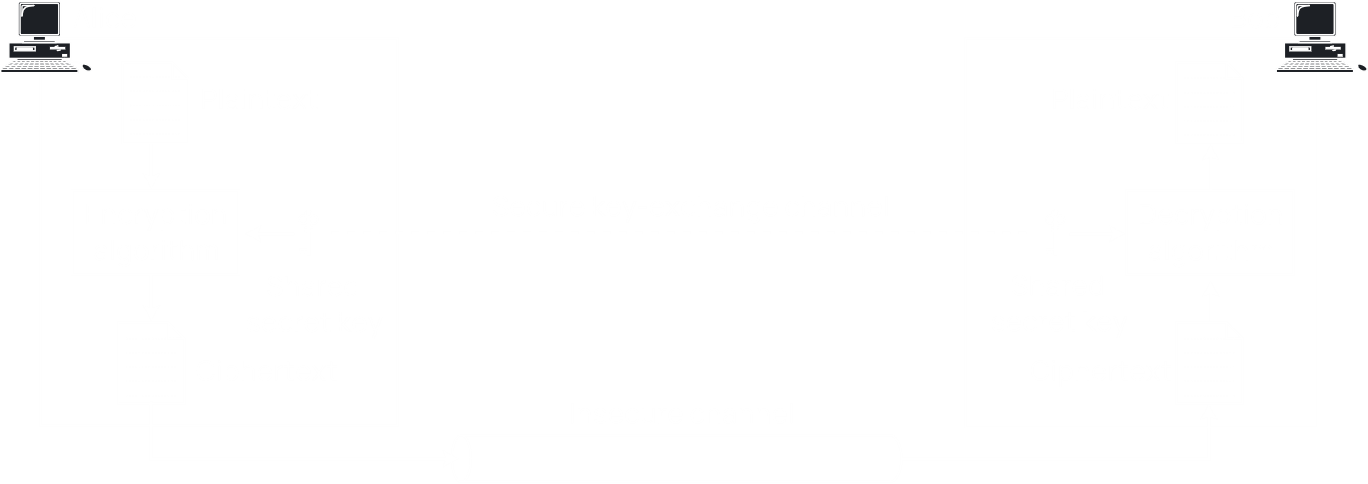
[Transposition Ciphers 11](#_Toc133109913)

[Keyless Transposition Ciphers 11](#_Toc133109914)

[Keyed Transposition Ciphers 12](#_Toc133109915)

[Stream and Block Ciphers 13](#_Toc133109916)

## General Structure



The general structure for a traditional symmetric-key cipher involves a **shared secret key**. This follows **Kerckhoff’s Principle**, which says that every client pair needs their own key. For clients, there are keys. The shared secret key is used to encode the data before sending it and decode the data on the other end.

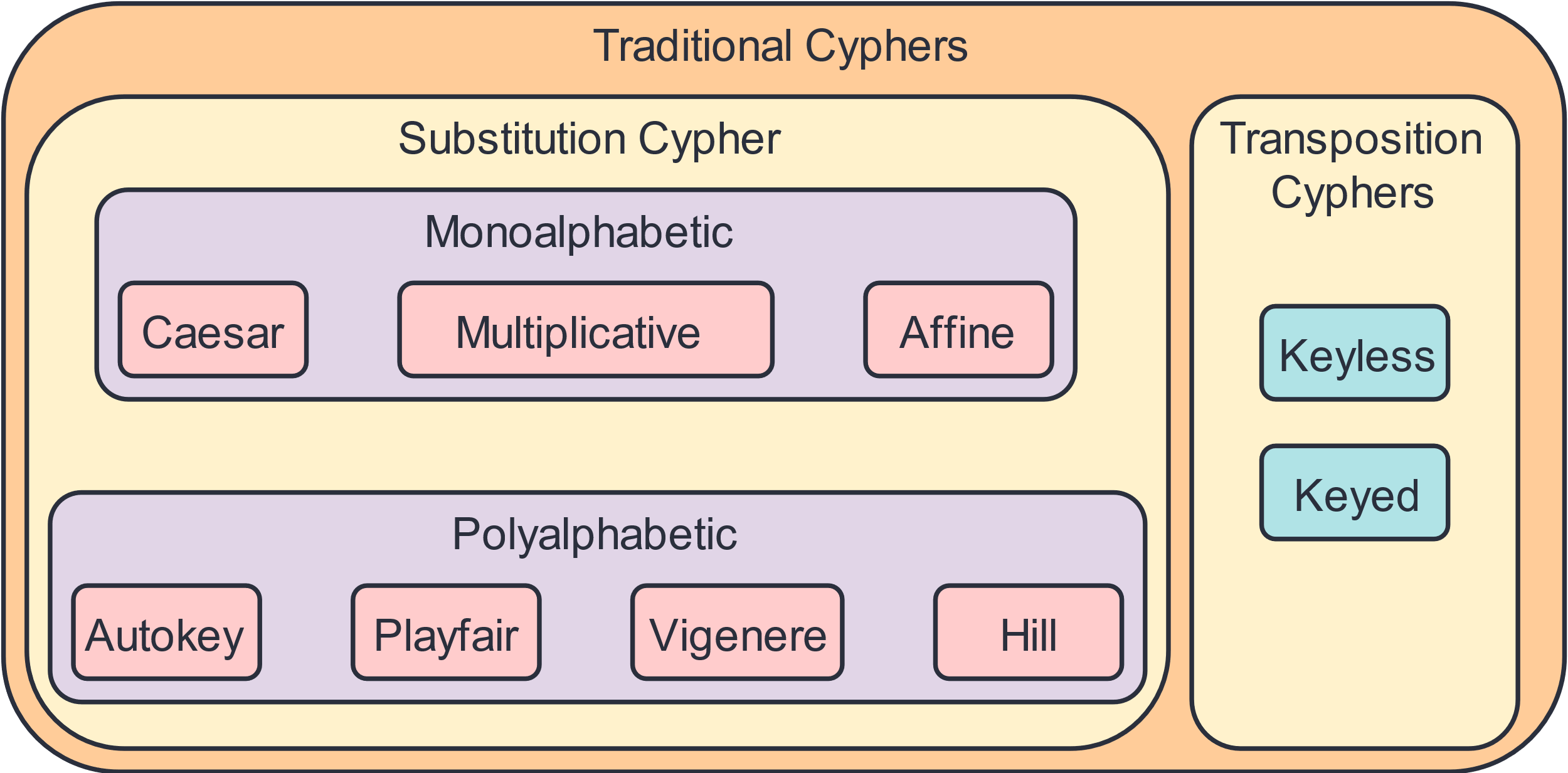
## Cryptanalysis

**Cryptography** is the science and art of creating secret codes. **Cryptanalysis** is the science and art of breaking those codes. This is important to find vulnerabilities in the cryptosystem.

Cryptanalysis attacks are of 4 types.

* **Ciphertext Only** – The attacker only has the ciphertext, not the key to decode it. This is done using brute force attacks, statistical attacks and pattern attacks. Statistical attacks rely on the properties of language to figure out what the text is. Pattern attacks rely on patterns in the cipher.
* **Known Plaintext** – The attacker managed to obtain one of the initial unencrypted plaintexts and the corresponding ciphers. This allows them to decipher all the future ciphertexts.
* **Chosen Plaintext** – The attacker accesses the sender’s computer and specifies the plaintext/ciphertext pair themselves.
* **Chosen Ciphertext** – The attacker accesses the receiver’s computer and specifies the plaintext/ciphertext pair themselves.

## Categories of Traditional Ciphers



## Monoalphabetic Substitution Ciphers

Substitution ciphers are ciphers that rely on replacing one symbol with another. For example, replacing every alphabet with the alphabet which comes 3 characters after it.

Monoalphabetic ciphers are substitution ciphers that replace each character with exactly one character. The replacement character is always the same.

### Additive Ciphers

These are monoalphabetic substitution ciphers and are also called **shift ciphers** or **Caesar ciphers**. The cipher, , is created by adding a key to the plaintext, . It is decrypted by subtracting .

For example,

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Plaintext | h | e | l | l | o |
| Value | 07 | 04 | 11 | 11 | 14 |
|  |  |  |  |  |  |
| Ciphertext | W | T | A | A | D |

In examples, the plaintext should always be in **small letters** while the ciphertext is in **capital letters**. The calculation above should always be shown when encrypting or decrypting text.

Additive ciphers are susceptible to brute force and statistical attacks. This is because the key domain is small (26 keys), and there are lots of common words in English text that occur repeatedly (the, he, ing, etc.).

### Multiplicative Ciphers

Like additive ciphers, **multiplicative ciphers** are also monoalphabetic, substitution ciphers. Instead of adding the key to the plain text, we **multiply** it.

There is, however, a catch. Notice that we are not simply dividing by when decrypting the text. This is because we need to multiply by the **multiplicative inverse**. We did the same thing with additive ciphers, except that the additive inverse of is just . The problem with multiplicative inverses though, is that not all values of have them. The list is provided below.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Number | 1 | 3 | 5 | 7 | 9 | 11 | 15 | 17 | 19 | 21 | 23 | 25 |
| Multiplicative Inverse | 1 | 9 | 21 | 15 | 3 | 19 | 7 | 23 | 11 | 5 | 17 | 25 |

Thus, our key domain has shrunk to just **12 values**.

### Affine Ciphers

**Affine Ciphers** are a combination of additive and multiplicative ciphers. They work by first applying an additive cipher with the key and then applying a multiplicative cipher with the key . Changing the order of the additive and multiplicative ciphers is acceptable. We just need to be careful that whichever order we use, the reverse is done when decrypting the text.

The key domain is of keys.

## Polyalphabetic Substitution Ciphers

In **Polyalphabetic Ciphers**, each occurrence of a character may be mapped to a different substitute character, meaning there is a **one-to-many­** relationship. The substitute character depends on the original character as well as the position of the character. For example, the character ‘a’ could be substituted with ‘N’ or with ‘D’ depending on the position of ‘a’. This requires that our key be a **stream of subkeys**.

### Autokey Ciphers

In **Autokey Ciphers**, the subkey is automatically created from the plaintext characters during the encryption process.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Plaintext | a | t | t | a | c | k | i | s | t | o | d | a | y |
| 00 | 19 | 19 | 00 | 02 | 10 | 08 | 18 | 19 | 14 | 03 | 00 | 24 |
| Key Stream | 12 | 00 | 19 | 19 | 00 | 02 | 10 | 08 | 18 | 19 | 14 | 03 | 00 |
| Ciphertext | 12 | 19 | 12 | 19 | 02 | 12 | 18 | 00 | 11 | 7 | 17 | 03 | 24 |
| M | T | M | T | C | M | S | A | L | H | R | D | Y |

For the plaintext above, we take the plaintext value for each character and use that as the subkey for the next character. For example, the first ‘a’ has the value 00, which is why the key value for the next character, ‘t’, is 00. The **first subkey value** is agreed upon beforehand between the sender and receiver. The final encrypted character’s value is the sum of the key and the plaintext character’s value, the same process as an **additive cipher**.

Cryptoanalysis attacks on autokey ciphers are affected by **brute-force attacks**, due to the small key domain, but they are not affected by single-letter frequency statistical attacks.

### Playfair Ciphers

In the **Playfair Cipher**, the secret key comes from a **5 x 5 matrix**. In the matrix, the characters ‘I’ and ‘J’ are considered to be the same. Different arrangements of the keys in the matrix can create different keys.

Secret Key:

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

To stay on the same page, we will be following 2 rules for creating the matrix:

1. A secret key will be provided beforehand, e.g., Playfair. The alphabets of this key should be placed on the cells from left to right and top to bottom, ignoring repeated characters.
2. The remaining alphabets from the English language should fill the rest of the cells in alphabetic order.

When encrypting a piece of text, such as ‘hello’, we first divide the text into pairs of two characters each, i.e., ‘he’, ‘ll’ and ‘o’. From this, we create the ciphertext.

Before we create the cipher, we need to make sure of two things:

1. If two letters in a pair are the same, they must be separated by a **bogus letter**. Thus, ‘he’, ‘ll’ and ‘o’ becomes ‘he’, ‘lx’ and ‘lo’.
2. If the total number of characters in the plaintext is **odd**, an extra bogus character is added to the last pair to make the characters even. This does not apply to our current example.

In the encryption process, 3 rules are followed:

1. If both letters in a pair are located on the **same row** in the secret key, the encrypted character will be the **next letter to the right** in the same row. If the character is the last in a row, the first character in the next row is used.
2. If both letters in a pair are located on the **same column** in the secret key, the encrypted character will be the **next letter below** in the same column. If the character is a the last in a column, the first character in the next column is used.
3. If both letters in a pair are not in the same row or column, the encrypted character is the one that is on the **same row** as the plaintext character and on the **same column** as the other plaintext character in the pair.

Following these three rules with the above secret key, our pairs become:

* ‘he’ ‘EC’
* ‘lx’ ‘QZ’
* ‘lo’ ‘BX’

Thus, the ciphertext is ECQZBX.

### Vigenère Cipher

This cipher is named after Blaise de Vigenère, a 16th century French mathematician. In this cipher, a **key stream** of length () is chosen and this is repeated across the entire text. As before, the value of the plaintext character is added to the corresponding subkey to generate the ciphertext character.

Suppose we have the plaintext ‘she is listening’ and the key stream is PASCAL (15, 0, 18, 2, 0 11).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Plaintext | s | h | e | i | s | l | i | s | t | e | n | i | n | g |
| 18 | 07 | 04 | 08 | 18 | 11 | 08 | 18 | 19 | 04 | 13 | 08 | 13 | 06 |
| Key Stream | 15 | 00 | 18 | 02 | 00 | 11 | 15 | 00 | 18 | 02 | 00 | 11 | 15 | 00 |
| Ciphertext | 07 | 07 | 22 | 10 | 18 | 22 | 23 | 18 | 11 | 06 | 13 | 19 | 02 | 06 |
| H | H | W | K | S | W | X | S | L | G | N | T | C | G |

### Hill Cipher

Invented by Lester S. Hill, the **Hill Cipher** uses an sized matrix as the key. The plaintext is wrapped into a block with columns and rows (an matrix). The matrices are **multiplied** to give the ciphertext.

Suppose , and we have the following key:

Let the plaintext string be ‘act’. Thus, the ciphertext is generated as:

Notice that the result of the multiplication is brought into the range of 0 to 25.

To get back the plaintext, we simply multiply the ciphertext with the inverse of the key.

Since the process requires multiplication, we need to make sure that the key matrix has a **multiplicative inverse**.

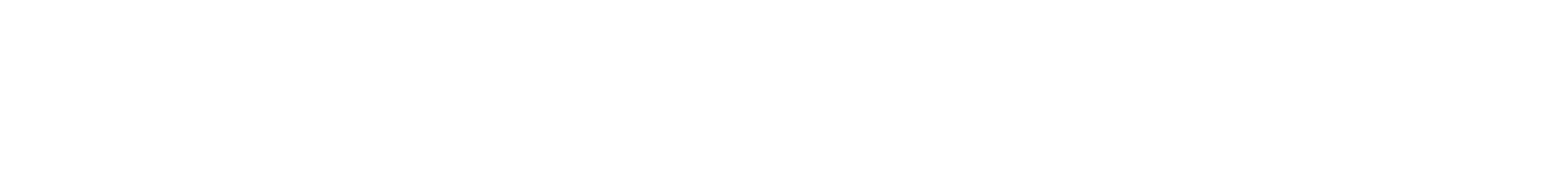
## Transposition Ciphers

Instead of substituting one character for another, **transposition ciphers** rely on changing the location of the characters. For example, a symbol at the 1st position might be exchanged with one at the 10th position.

### Keyless Transposition Ciphers

There are two **Keyless Transposition Ciphers**, Zigzag and Fixed Columns.

In the **Zigzag** method, we split the text in a zigzag fashion into two rows and then create the ciphertext row by row. For the plaintext ‘meet me at the park’, the zigzag would look like this:



The corresponding ciphertext is thus ‘MEMATEAKETETHPR’.

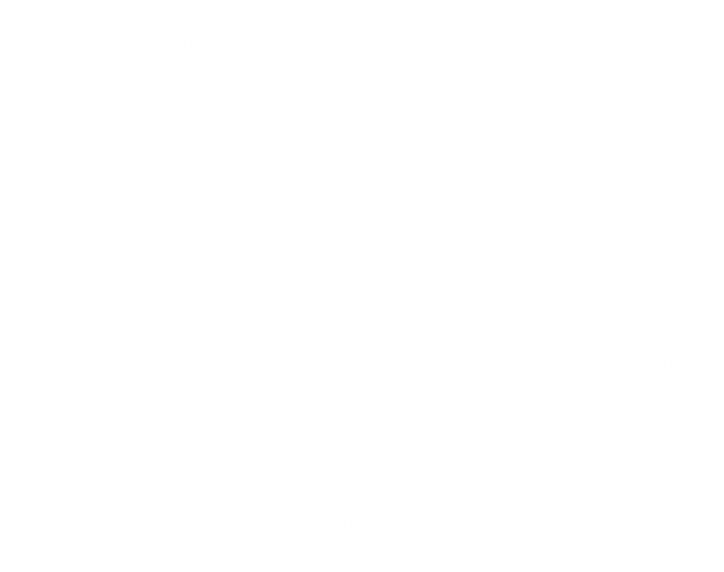
In the **Fixed Columns** method, the number of columns is agreed upon beforehand. The plaintext is wrapped with the fixed number of columns in mind, as below:



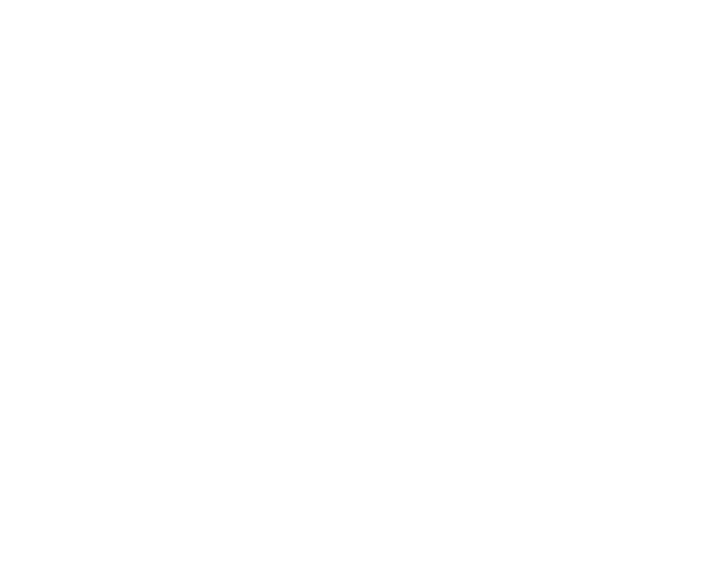
The ciphertext is then generated column by column. For the above text, it is ‘MMTAEEHREAEKTTP’.

### Keyed Transposition Ciphers

In a **Keyed Transposition Cipher**, the plaintext is divided into blocks of characters each. If the last block does not have enough characters, dummy characters are inserted. Suppose the plaintext is ‘enemy attacks tonight’ and . Thus,



For each block, the letters are transposed based on a key. Suppose the key is ‘31452’, meaning the 3rd character should come first, the 1st character should come second and so on. From this, the text becomes:



The ciphertext can be generated from this column by column or row by row. If we do it column by column, the ciphertext is ETTHEAKIMAOTYCNZNTSG. If we do it row by row, the ciphertext is EEMYNTAACTTKONSHITZG. When decrypting the text, we just need to remember to reverse this process properly.

## Stream and Block Ciphers

The ciphers we have seen so far fall into one of two categories, stream ciphers or block ciphers.

**Stream Ciphers** process the plaintext one symbol at a time. The symbol can be a character or a bit. Monoalphabetic ciphers, autokey ciphers and Vigenère ciphers are stream ciphers.

**Block Ciphers** process the plaintext in groups of size , where . The generated ciphers are of the same size. The entire group is encrypted using a **single key**, although the key can consist of multiple subkeys. Playfair ciphers, Hill ciphers and keyed transposition ciphers are block ciphers.