Chapter 1

Classic Ciphers

You are in a maze of twisty little passages, all alike.
— Adventure

1.1 Introduction

Most of this chapter is devoted to introducing terminology and discussing a select few classic "pen and paper" ciphers. Our goal here is not to cover classical cryptography in detail, since there are already many excellent sources of information on such ciphers. For example, Kahn's history [74] has a general discussion of virtually every cipher developed prior to its original publication date of 1967, Barr [7] presents a readable introduction to cryptography, Spillman [139] nicely covers the cryptanalysis of several classic cipher systems and Bauer [8] provides rigorous coverage of a large number of classical crypto topics. The ciphers we discuss in this chapter have been selected to illustrate a few important points that arise in upcoming chapters.

Even if you are familiar with classical cryptosystems, you should read the next two sections where terminology is discussed, since the terminology in cryptography is not always consistent. In addition, the material in Sections 1.4.3 and 1.4.4 is directly referenced in upcoming chapters.

1.2 Good Guys and Bad Guys

In cryptography, it is traditional that Alice and Bob are the good guys who are trying to communicate securely over an insecure channel. We employ Trudy (the "intruder") as our generic bad guy. Some books have a whole cast of bad guys with the name indicating the particular evil activity (Eve, the eavesdropper, for example), but we use Trudy as our all-purpose bad "guy".

Since this is a cryptanalysis book, we often play the role of Trudy. Trudy is an inherently more interesting character than boring old Alice and Bob, and this is part of what makes cryptanalysis so much more fun than cryptography. Trudy does not have to play by any preconceived set of rules. However, it is important to remember that attacks on real systems are almost certainly illegal, so do not attempt to play Trudy in the real world.

1.3 Terminology

Cryptology is the art and science of making and breaking "secret codes." Cryptology can be subdivided into *cryptography* (the art and science of making secret codes) and *cryptanalysis* (the breaking of secret codes). The secret codes themselves are known as *ciphers* or *cryptosystems*. In this book, we are focused on cryptanalysis, but many topics in cryptography naturally arise.

It is common practice to use the term cryptography as a synonym for cryptology, and we generally follow this practice. In fact, we often use *crypto* as shorthand for cryptology, cryptography, cryptanalysis, or any variety of other crypto-related topics. The precise meaning should be clear from the context.

The original readable message is the *plaintext*, while the *ciphertext* is the unreadable text that results from *encrypting* the plaintext. *Decryption* is the inverse process, where the ciphertext is converted into plaintext.

A *key* is used to configure a cryptosystem. All classic systems are *symmetric ciphers*, meaning that the same key is used to encrypt as to decrypt. In so-called *public key cryptography* the encryption and decryption keys are different, which means that the encryption key can be be made public, but the decryption key must remain private. We cover public key cryptosystems in Chapters 6 and 7, while all of the remaining chapters—including the remaining sections of this chapter—deal with symmetric ciphers.

Note that decryption is distinct from cryptanalysis, since cryptanalysis implies an attack of some sort has been used to read the messages, while decryption implies that the plaintext has been retrieved using the key by the expected process. Of course, if Trudy recovers the key via cryptanalysis, then she can simply decrypt a particular ciphertext.

The typical encryption and decryption process is illustrated in Figure 1.1, where P_i is the ith unit of plaintext (which may be a bit, a letter, a word, or a larger block, depending on the particular cipher), C_i is the corresponding unit of ciphertext, and the squiggly line represents the transmission of the ciphertext over an insecure channel.

There are several generic types of attacks on ciphers. In a *ciphertext only* attack, the attacker attempts to recover the key or plaintext from the ciphertext. In particular, in a ciphertext-only attack, the cryptanalyst does

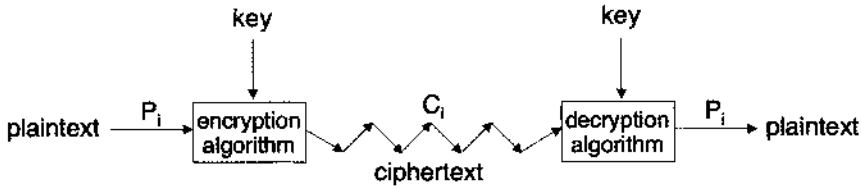

Figure 1.1: Encryption and decryption.

not know any of the underlying plaintext. A basic assumption is that the ciphertext is always available to an attacker. After all, if the ciphertext is not available to the attacker, why bother to encrypt?

In a *known plaintext* attack, Trudy has the ciphertext as well as some of the corresponding plaintext. This might give the attacker some advantage over the ciphertext only scenario—certainly the attacker is no worse off with known plaintext. If Trudy knows all of the plaintext, there is probably not much point in bothering to attack the system, so the implicit assumption is that the amount of known plaintext is relatively limited.

As the name implies, in a *chosen plaintext* attack, an adversary can choose the plaintext and then obtain the corresponding ciphertext. This can only help the attacker, as compared to a known plaintext scenario. Similarly, in a *chosen ciphertext* attack, the cryptanalyst chooses ciphertext and gets to see the corresponding plaintext. There are also *related key* attacks, where the attacker can break the system if two keys are used that happen to be related in some very special way. While this may seem somewhat esoteric, we will see an example of a real-world related key attack in Chapter 3.

In most cases, recovering the key is Trudy's ultimate goal, but there are attacks that recover the plaintext without revealing the key. A cipher is generally not considered secure unless it is secure against all plausible attacks. Cryptographers are, by nature, a paranoid bunch, so "plausible" is usually defined very broadly.

Kerckhoffs' Principle is one of the fundamental concepts underlying cryptography. This principle states that the strength of a cryptosystem depends only on the key and, in particular, the security does not depend on keeping the encryption algorithm secret. This principle is generally construed even more broadly to imply that the attacker knows the protocols and overall system in which a cryptosystem is used. Adherence to Kerckhoffs' Principle should ensure that the security of a cryptosystem does not depend on the much-dreaded "security by obscurity", since the security does not depend on a secret algorithm. Unfortunately, there are many real-world pressures that can lead to violations of Kerckhoffs' Principle, usually with disastrous consequences.

Why do we insist on Kerckhoffs' Principle? After all, the attacker's job certainly must be more difficult if the crypto algorithm is unknown. In part, the answer is that Kerckhoffs' Principle is just a codification of reality – algorithms never remain secret for long so it is far better to find flaws beforehand, rather than after an algorithm is embedded in millions of applications dispersed across the globe. It also happens to be true that designing a secure cipher is not easy, and it is made all the more difficult when efficiency is an issue, which is usually the case. An extensive peer review process is essential before any algorithm can be considered sufficiently secure for use. We will see several real-world examples that illustrate the wisdom of Kerckhoffs in upcoming chapters.

Suppose that Alice encrypts a message and sends the ciphertext to Bob. Figure 1.2 illustrates what information is available to Alice, Bob and the attacker, Trudy. At a minimum we assume that Trudy has access to the ciphertext and, by Kerckhoffs' Principle, she also knows how the crypto algorithm works. In some cases, Trudy may have additional information, such as known plaintext, chosen plaintext, etc.

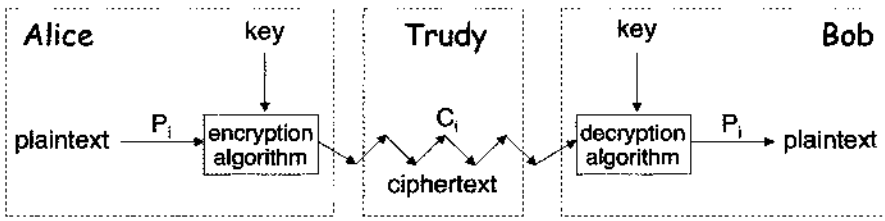

Figure 1.2: Who knows what.

In the next section we highlight a few selected classic crypto topics. We also discuss some important cryptanalytic principles and we provide details on a few specific ciphers that are relevant to later chapters.

1.4 Selected Classic Crypto Topics

If you have done much traveling, you know that it is almost impossible to see everything, and if you try, you are bound to regret it. It is usually far more productive to avoid the "tourist death march" and instead focus on a few specific interesting locations. We will take a similar approach here as we peruse selected classic crypto topics, stopping at a few points of interest, but making no attempt to cover every possible topic along the way. Since our focus in the remainder of the book is cryptanalysis, we emphasize attacks on the classic ciphers that we cover.

Since ancient times, cryptography has been used for military and diplomatic purposes. In the remainder of this chapter we consider a few specific examples of classic ciphers. These ciphers have been carefully selected to illustrate important topics that arise in the study of modern ciphers presented in subsequent chapters.

The history of crypto is itself a fascinating topic, but it is not our focus here. For more crypto history, a good crypto timeline can be found at [104] and there is always Kahn's book [74]. For a more in-depth technical look at classic ciphers, see Bauer's fine book [8].

1.4.1 Transposition Ciphers

Transposition ciphers jumble the letters of the message in a way that is designed to confuse the attacker, but can be unjumbled by the intended recipient. The concept of transposition is an important one and is widely used in the design of modern ciphers, as will be seen in subsequent chapters. Note that the key must provide sufficient information to unscramble the ciphertext.

Scytale

One of the earliest recorded uses of cryptography was the Spartan *scytale* (circa 500 B.C.). A thin strip of parchment was wrapped helically around a cylindrical rod and the message was written across the rod, with each letter on a successive turn of the parchment. The strip was unwound and delivered to the receiver. The message could then be decrypted with the use of an identical cylindrical rod. To anyone who intercepted the message, and did not understand the encryption technique, the message would appear to be a jumble of letters. A clever cryptanalyst with access to a number of rods of various diameters will soon recover the plaintext.

For the scytale cipher, which is an example of a transposition cipher, the key is the rod (or its diameter). This is a very weak cipher since the system could be easily broken by anyone who understands the encryption method.

Columnar Transposition

Suppose we have plaintext SEETHELIGHT and we want to encrypt this using a *columnar transposition cipher*. We first put the plaintext into the rows of an array of some given dimension. Then we read the ciphertext out of the columns. The key consists of the the number of columns in the array. For example, suppose we choose the key to be four, which means that we write the plaintext in four columns as

$$\begin{bmatrix} S & E & E & T \\ H & E & L & I \\ G & H & T & X \end{bmatrix},$$

where the final X is used as to fill out the array. The ciphertext is then read from the columns, which in this case yields SHGEEHELTTIX. The intended recipient, who knows the number of columns, can put the ciphertext into an appropriate-sized array and read the plaintext out from the rows.

Not surprisingly, a columnar transposition is not particularly strong. To perform a ciphertext only attack on this cipher, we simply need to test all possible decrypts using c columns, where c is a divisor of the number of characters in the ciphertext.

Keyword Columnar Transposition

The columnar transposition cipher can be strengthened by using a keyword, where the keyword determines the order in which the columns of ciphertext are transcribed. We refer to this as a *keyword columnar transposition cipher*. For example, consider encrypting the plaintext CRYPTOISFUN using a keyword columnar transposition cipher with keyword MATH, again using four columns. In this case, we get the array

$$\begin{array}{cccc} \multicolumn{4}{c}{\begin{array}{cccc} M & A & T & H \end{array}} \\ \left[\begin{array}{cccc} C & R & Y & P \\ T & O & I & S \\ F & U & N & X \end{array}\right] & . \end{array}$$

The ciphertext is read from the columns in alphabetical order (as determined by the keyword), so that, in this example, the ciphertext is ROUPSXCTFYIN.

Is it possible to conduct a ciphertext-only attack on a keyword columnar transposition cipher? It is certainly not as straightforward as attacking a non-keyword columnar cipher. Suppose we obtain the ciphertext

VOESA IVENE MRTNL EANGE WTNIM HTMEE ADLTR NISHO DWOEH

which we believe was encrypted using a keyword columnar transposition. Our goal is to recover the key and the plaintext. First, note that there are 45 letters in the ciphertext. Assuming the array is not a single column or row, the array could have any of the following dimensions: 9×5, 5×9, 15×3 or 3×15. Suppose that we first try a 9×5 array. Then we have the ciphertext array in Table 1.1.

We focus our attention on the top row of the array in Table 1.1. If we permute the columns as shown in Table 1.2, we see the word GIVE in the first row and we see words or partial words in the other rows. Therefore, we have almost certainly recovered the key.

This method is somewhat ad hoc, but the process could be automated, provided we can automatically recognize likely plaintexts. In this example, we have recovered the encryption key 24013 and the plaintext is

GIVE ME SOMEWHERE TO STAND AND I WILL MOVE THE EARTH.

Table 1.1: Ciphertext Array

0	1	2	3	4
V	E	G	M	I
O	M	E	E	S
E	R	W	E	H
S	T	T	A	O
A	N	N	D	D
I	L	I	L	W
V	E	M	T	O
E	A	H	R	E
N	N	T	N	H

Table 1.2: Permuted Ciphertext Array

2	4	0	1	3
G	I	V	E	M
E	S	O	M	E
W	H	E	R	E
T	O	S	T	A
N	D	A	N	D
I	W	I	L	L
M	O	V	E	T
H	E	E	A	R
T	H	N	N	N

There are many ways to systematically mix the letters of the plaintext. For example, we can strengthen the columnar transposition cipher by allowing the permutation of columns and rows. Since two transpositions are involved, this is known as a *double transposition cipher*, which we briefly describe next.

Double Transposition Cipher

To encrypt with a double transposition cipher, we first write the plaintext into an array of a given size and then permute the rows and columns according to specified permutations. For example, suppose we write the plaintext ATTACKATDAWN into a 3×4 array:

$$\begin{bmatrix} A & T & T & A \\ C & K & A & T \\ D & A & W & N \end{bmatrix}.$$

Now if we transpose the rows according to $(0,1,2) \to (2,1,0)$ and then transpose the columns according to $(0,1,2,3) \to (3,1,0,2)$, we obtain

$$\begin{bmatrix} A & T & T & A \\ C & K & A & T \\ D & A & W & N \end{bmatrix} \longrightarrow \begin{bmatrix} D & A & W & N \\ C & K & A & T \\ A & T & T & A \end{bmatrix} \longrightarrow \begin{bmatrix} N & A & D & W \\ T & K & C & A \\ A & T & A & T \end{bmatrix}.$$

The ciphertext is read directly from the final array:

NADWTKCAATAT.

For the double transposition, the key consists of the size of the matrix and the row and column permutations. The recipient who knows the key can simply put the ciphertext into the appropriate sized matrix and undo the permutations to recover the plaintext.

If Trudy happens to know the size of the matrix used in a double transposition, she can insert the ciphertext into a matrix of the appropriate size. She can then try to unscramble the columns to reveal words (or partial words). Once the column transposition has been undone, she can easily unscramble the rows; see Problem 12 for an example. This attack illustrates the fundamental principle of *divide and conquer*. That is, Trudy can recover the double transposition key in parts, instead of attacking the entire key all at once. There are many examples of divide and conquer attacks throughout the remainder of this book.

In spite of the inherent divide and conquer attack, the double transposition cipher is relatively strong - at least in comparison to many other classic cipher. The interested reader is directed to [88] for a thorough cryptanalysis of the double transposition.

1.4.2 Substitution Ciphers

Like transposition, substitution is a crucial concept in the design of modern ciphers. In fact, Shannon's [133] two fundamental principles for the design of symmetric ciphers are *confusion* and *diffusion*, which, roughly, correspond to the classic concepts of substitution and transposition, respectively. These are still the guiding principles in the design of symmetric ciphers.

In this section we discuss several classic substitution ciphers. We highlight some of the clever techniques that can be brought to bear to attack such ciphers.

Caesar's Cipher

In 50 B.C., Gaius Julius Caesar described the use of a specific cipher that goes by the name of *Caesar's cipher*.[1] In Caesar's cipher, encryption is ac-

[1] Historians generally agree that the Caesar's cipher was named after the Roman dictator, not the salad.

complished by replacing each plaintext letter with its corresponding "shift-by-three" letter, that is, A is replaced by D, B is replaced by E, C is replaced by F, and so on. At the end of the alphabet, a wrap around occurs, with X replaced by A, Y replaced by B and Z replaced by C. Decryption is accomplished by replacing each ciphertext letter with its corresponding left-shift-by-three letter, again, taking the wrap around into account.

Suppose we assign numerical values $0, 1, \ldots, 25$ to the letters $\mathtt{A}, \mathtt{B}, \ldots, \mathtt{Z}$, respectively, Let p_i be the ith plaintext letter of a given message, and c_i the corresponding ith ciphertext letter. Then Caesar's cipher can be mathematically stated as $c_i = p_i + 3 \pmod{26}$ and, therefore, $p_i = c_i - 3 \pmod{26}$. In Caesar's cipher, the key is "3", which is not very secure, since there is only one key—anyone who knows that the Caesar's cipher is being used can immediately decrypt the message.

Simple Substitution

A *simple substitution* (or *mono-alphabetic substitution*) cipher is a generalization of the Caesar's cipher where the key can be any permutation of the alphabet. For the simple substitution, there are $26! \approx 2^{88}$ keys available. This is too many keys for any attacker to simply try them all, but even with this huge number of keys, the simple substitution cipher is insecure. Before we discuss the attack on the simple substitution, we consider a few special types of related ciphers that have been used in the past.

Nomenclator

Circa 1400, a type of cipher known as a *nomenclator* was invented and came into widespread use by trading states in Europe and by the Catholic Church. A nomenclator is a book that describes how letters, syllables, and words are converted into ciphertext and vice versa. In effect, this is a hybrid between a simple substitution and a codebook cipher (described below), and it has a larger number of possible keys than a simple substitution cipher. All else being equal (which it never is), this should make the cryptanalyst's job more difficult.

Poly-alphabetic Substitution

During the Renaissance, the first *poly-alphabetic substitution* cipher was invented by one Leon Battista Alberti (1404–1472). Such a cipher is essentially a variable simple substitution cipher, that is, a different substitution alphabet is used for different parts of the message. In Alberti's cipher, this was accomplished by use of a device that included an inner and outer cipher wheel with the alphabet written in particular ways on each wheel. The inner wheel freely rotated allowing the two alphabets to be aligned in any fashion, with

each alignment generating a different (simple) substitution. As the message was encrypted, differing substitution alphabets could be used, as determined by both parties in advance, or as specified within the message itself.

In his book *Traicté des Chiffres*, Blaise de Vigenère (1585) discusses a poly-alphabetic substitution that uses a 26×26 rectangular array of letters. The first row of the array is A, B, C, ..., Z, and each succeeding row is a cyclic left shift of the preceding one. A keyword can then be used to determine which of the cipher alphabets to use at each position in the text. In this way, all "shift-by-n" simple substitutions are readily available for use. The Vigenère cipher, and its cryptanalysis, is discussed below.

Affine Cipher

An *affine* cipher is a simple substitution where $c_i = ap_i + b \pmod{26}$. Here, the constants a and b are integers in the range 0 to 25 (as are p_i and c_i). To decrypt uniquely—always a nice feature for a cipher system– we must have $\gcd(a, 26) = 1$. Consequently, there are $26 \cdot \phi(26) = 312$ affine ciphers for the English language, where ϕ is the Euler-phi function (see the Appendix for a definition of the ϕ function). The decryption function for the affine cipher is $p_i = a^{-1}(c_i - b) \pmod{26}$, where $aa^{-1} = 1 \pmod{26}$, that is, a^{-1} is the multiplicative inverse of a, modulo 26.

Affine ciphers are weak for several reasons, but the most obvious problem is that they have a small *keyspace*. A ciphertext only attack can be performed by conducting a brute force search of all 312 possible key pairs (a, b). This attack is trivial, provided we can recognize the plaintext when we see it (or, better yet, automatically test for it).

Simple Substitution Cryptanalysis

Trying all possible keys is known as an *exhaustive key search*, and this attack is always an option for Trudy. If there are N possible keys, then Trudy will, on average, need to try about half of these, that is, $N/2$ of the keys, before she can expect to find the correct key. Therefore, the first rule of cryptography is that any cipher must have a large enough keyspace so that an exhaustive search is impractical. However, a large keyspace does not ensure that a cipher is secure. To see that this is the case, we next consider an attack that will work against any simple substitution cipher and, in the general case, requires far less work than an exhaustive key search. This attack relies on the fact that statistical information that is present in the plaintext language "leaks" through a simple substitution.

Suppose we have a reasonably large ciphertext message generated by a simple substitution, and we know that the underlying plaintext is English. Consider the English letter frequency information in Table 1.3, which was

compiled from a 7834-letter sample of written English. By simply computing letter frequency counts on our ciphertext, we can make educated guesses as to which plaintext letters correspond to some of the ciphertext letters. For example, the most common ciphertext letter probably corresponds to plaintext E. We can obtain additional statistical information by making use of digraphs (pairs of letters) and common trigraphs (triples). This type of statistical attack on a simple substitution, is very effective. After a few letters have been guessed correctly, partial words will start to appear and the cipher should then quickly unravel.

Table 1.3: English Letter Frequencies as Percentages

Letter	Relative Frequency	Letter	Relative Frequency
A	8.399	N	6.778
B	1.442	O	7.493
C	2.527	P	1.991
D	4.800	Q	0.077
E	12.15	R	6.063
F	2.132	S	6.319
G	2.323	T	8.999
H	6.025	U	2.783
I	6.485	V	0.996
J	0.102	W	2.464
K	0.689	X	0.204
L	4.008	Y	2.157
M	2.566	Z	0.025

Vigenère Cipher

Recall that a poly-alphabetic substitution cipher uses multiple simple substitutions to encrypt a message. The Vigenère cipher is a classic poly-alphabetic substitution cipher. The World War II cipher machines discussed in Chapter 2 are more recent examples of poly-alphabetic substitutions.

In the Vigenère cipher, a key of the form $K = (k_0, k_1, \ldots, k_{n-1})$, where each $k_i \in \{0, 1, \ldots, 25\}$, is used to encipher the plaintext. Here, each k_i represents a particular shift of the alphabet. To encrypt a message,

$$c_i = p_i + k_{i \pmod n} \pmod{26}$$

and to decrypt

$$p_i = c_i - k_{i \pmod n} \pmod{26}.$$

For example, suppose $K = (12, 0, 19, 7)$, which corresponds to the keyword MATH (since M corresponds to a shift of 12, A corresponds to a shift of 0, and so on). Using this keyword, the the plaintext SECRETMESSAGE is encrypted as EEVYQTFLESTNQ.

Next, we cryptanalyze the Vigenère cipher. But first, note that a poly-alphabetic substitution (such as the Vigenère cipher) does not preserve plaintext letter frequencies to the same degree as a mono-alphabetic substitution. Furthermore, if the number of alphabets is large relative to the message size, the plaintext letter frequencies will not be preserved at all. Therefore, the generic simple substitution attack discussed above will not work on a poly-alphabetic substitution.

However, the Vigenère cipher is vulnerable to a slightly more sophisticated statistical attack. To see how this works, first consider a Vigenère cipher with a small keyword. Suppose that the following ciphertext was created using a Vigenère cipher with a three-lettered keyword:

```
RLWRV MRLAQ EDUEQ QWGKI LFMFE XZYXA QXGJH FMXKM QWRLA
LKLFE LGWCL SOLMX RLWPI OCVWL SKNIS IMFES JUVAR MFEXZ
CVWUS MJHTC RGRVM RLSZS MREFW XZGRY RLWPI OMYDB SFJCT
CAZYX AQ.
```
(1.1)

To recover the key and decrypt the message, we can make use of the fact that the ciphertext is composed of three simple substitutions. To accomplish this, we tabulate the letter frequencies for the sets

$$S_0 = \{c_0, c_3, c_6, \ldots\}, \quad S_1 = \{c_1, c_4, c_7, \ldots\}, \quad \text{and} \quad S_2 = \{c_2, c_5, c_8, \ldots\},$$

where c_i is the ith ciphertext letter. Doing so, we obtain the results in Tables 1.4, 1.5, and 1.6, respectively.

Table 1.4: Letter Frequencies in S_0

Letter	R	Q	U	K	F	E	Y	J	M	L	G	P	C	N	I	Z	W	B
Frequency	10	4	3	1	2	3	2	3	3	4	2	2	4	1	1	1	1	1

Table 1.5: Letter Frequencies in S_1

Letter	L	V	E	W	I	M	X	Q	K	S	H	R	Y	C	A
Frequency	6	5	4	2	4	4	7	1	1	6	1	2	1	1	1

From the S_0 ciphertext in Table 1.4, we might reasonably guess that ciphertext R corresponds to plaintext E, T, N, O, R, I, A or S, which gives us

Table 1.6: Letter Frequencies in S_2

Letter	W	M	A	D	Q	G	L	F	Z	K	C	O	X	S	J	T	Y
Frequency	6	4	5	2	1	3	3	5	4	2	1	3	1	2	1	2	1

candidate values for k_0, namely $k_0 \in \{13, 24, 4, 3, 0, 9, 17, 25\}$. Similarly, for set S_1, ciphertext X might correspond to plaintext E, T, N, O, R, I, A or S, from which we obtain likely values for k_1, and from set S_2, ciphertext W likely correspond to plaintext E, T, N, O, R, I, A or S. The corresponding likely keyword letters are tabulated in Table 1.7.

Table 1.7: Likely Keyword Letters

k_0	k_1	k_2
N	T	S
Y	E	D
E	K	J
D	J	I
A	G	F
J	P	O
R	X	W
Z	F	E

The combinations of likely keyword letters in Table 1.7 yield $8^3 = 2^9$ putative keywords. By testing each of these putative keyword on the first few letters of the ciphertext, we can easily determine which, if any, is the actual keyword. For this example, we find that $(k_0, k_1, k_2) = (24, 4, 18)$, which corresponds to YES, and the original plaintext is

```
THE TRUTH IS ALWAYS SOMETHING THAT IS TOLD, NOT
SOMETHING THAT IS KNOWN. IF THERE WERE NO SPEAKING
OR WRITING, THERE WOULD BE NO TRUTH ABOUT ANYTHING.
THERE WOULD ONLY BE WHAT IS.
```

This attack provides a significant shortcut as compared to trying all possible $26^3 \approx 2^{14}$ keywords.

Knowing the length of the keyword used in a Vigenère cipher helps greatly in the cryptanalysis. If the keyword is known, and the message is long enough, we can simply perform letter frequency counts on the associated sets of ciphertext to begin solving for the plaintext. However, it is not so obvious how to determine the length of an unknown keyword. Next, we consider two methods for approximating the length of the keyword in a Vigenère cipher.

Friederich W. Kasiski (1805–1881) was a major in the East Prussian infantry regiment and the author of the cryptologic text *Die Geheimschriften und die Dechiffer-kunst*. Kasiski developed a test (amazingly, known as the Kasiski Test), that can sometimes be used to find the length of a keyword used in a cipher such as the Vigenère. It relies on the occasional coincidental alignment of letter groups in plaintext with the keyword. To attack a periodic cipher using the Kasiski Test, we find repeated letter groups in the ciphertext and tabulate the separations between them. The greatest common divisor of these separations (or a divisor of it) gives a possible length for the keyword.

For example, suppose we encrypt the plaintext

THECHILDISFATHEROFTHEMAN

with a Vigenère cipher using the keyword POETRY. The resulting ciphertext is

IVIVYGARMLMYIVIKFDIVIFRL.

Notice that the second occurrence of the ciphertext letters IVI begins exactly 12 letters after the first, and the third occurrence of IVI occurs exactly six letters after the second. Therefore, it is likely that the length of the keyword is a divisor of six. In this case, the keyword length is exactly six.

Index of Coincidence

While working at the Riverbank Laboratory, William F. Friedman (1891–1969) developed the *index of coincidence*. For a given ciphertext, the index of coincidence I is defined to be the probability that two randomly selected letters in the ciphertext represent the same plaintext symbol.

For a given ciphertext, let $n_0, n_1, \ldots, n_{25}$ be the respective letter counts of A, B, C, ..., Z in the ciphertext, and set $n = n_0 + n_1 + \cdots + n_{25}$. Then, the index of coincidence can be computed as

$$I = \frac{\binom{n_0}{2} + \binom{n_1}{2} + \cdots + \binom{n_{25}}{2}}{\binom{n}{2}} = \frac{1}{n(n-1)} \sum_{i=0}^{25} n_i(n_i - 1). \tag{1.2}$$

To see why the index of coincidence gives us useful information, first note that the empirical probability of randomly selecting two identical letters from a large English plaintext is

$$\sum_{i=0}^{25} p_i^2 \approx 0.065,$$

where p_0 is the probability of selecting an A, p_1 is the probability of selecting a B, and so on, and the values of p_i are given in Table 1.3. This implies that an (English) ciphertext having an index of coincidence $I \approx 0.065$ is probably

associated with a mono-alphabetic substitution cipher, since this statistic will not change if the letters are simply relabeled (which is the effect of encrypting with a simple substitution).

The longer and more random a Vigenère cipher keyword is, the more evenly the letters are distributed throughout the ciphertext. With a very long and very random keyword, we would expect to find

$$I \approx 26\left(\frac{1}{26}\right)^2 = \frac{1}{26} \approx 0.03846.$$

Therefore, a ciphertext having $I \approx 0.03846$ could be associated with a poly-alphabetic cipher using a large keyword. Note that for any English ciphertext, the index of coincidence I must satisfy $0.03846 \leq I \leq 0.065$.

The question remains as to how to determine the length of the keyword of a Vigenère cipher using the index of coincidence. The main weakness of the Vigenère (or any similar periodic cipher) is that two identical characters occurring a distance apart that is a multiple of the key length will be encrypted identically. In such cryptosystems, the key length k can be approximated by a function involving the index of coincidence I and the length of the ciphertext n. The following example illustrates this technique.

Suppose an English plaintext containing n letters is encrypted using a Vigenère cipher, with a keyword of length k, where, for simplicity, we assume n is a multiple of k. Now suppose that we arrange the ciphertext letters into a rectangular array of n/k rows and k columns, from left to right, top to bottom. If we select two letters from different columns in the array, this would be similar to choosing from a collection of letters that is uniformly distributed, since the keyword is more-or-less "random". In this case, the portion of pairs of identical letters is, approximately,

$$0.03846\binom{k}{2}\left(\frac{n}{k}\right)^2 = 0.03846\,\frac{n^2(k-1)}{2k}.$$

On the other hand, if the two selected letters are from the same column, this would correspond to choosing from ciphertext having a symbol distribution similar to printed English plaintext, since, in effect, a simple substitution is applied to each column. In this case, the portion of pairs of identical letters is approximately

$$0.065\binom{\frac{n}{k}}{2}k = 0.065\,\frac{1}{2}\left(\frac{n}{k}\right)\left(\frac{n}{k}-1\right)k = 0.065\left(\frac{n(n-k)}{2k}\right).$$

Therefore, the index of coincidence satisfies

$$I \approx \frac{0.03846\,\frac{n^2(k-1)}{2k} + 0.065\left(\frac{n(n-k)}{2k}\right)}{\binom{n}{2}} = \frac{0.03846n(k-1) + (0.065)(n-k)}{k(n-1)}. \tag{1.3}$$

The attacker, Trudy, does not know k, but she can solve for k in (1.3) to obtain

$$k \approx \frac{0.02654n}{(0.065 - I) + n(I - 0.03846)}. \tag{1.4}$$

Then given n and I, which are easily computed from the ciphertext, Trudy can approximate k, the number of letters in the keyword of the underlying Vigenère cipher.

The index of coincidence was a cryptologic breakthrough, since it can be used to gain information about poly-alphabetic substitution ciphers. Friedman's work on the index of coincidence was one of his most important contributions to cryptology, and it provided invaluable information to cryptanalysts during WWII, where poly-alphabetic ciphers played a major role.

Hill Cipher

As a final example of a substitution cipher, we consider the Hill cipher, which was introduced by mathematician Lester Hill in 1929 [67]. The Hill cipher is interesting since it is a pre-modern block cipher. The idea behind the Hill cipher is to create a substitution cipher with an extremely large "alphabet". Such a system is more resilient to cryptanalysis that relies on letter frequency counts and statistical analysis of the plaintext language. However, the cipher is linear which makes it vulnerable to a relatively straightforward known plaintext attack. The description of the Hill cipher requires some elementary linear algebra; see the Appendix for the necessary background information.

Suppose that Alice wants to send a message to Bob and they have decided to use the Hill cipher. First, the plaintext is divided into blocks $p_0, p_1, p_2, \ldots$, each consisting of n letters. Alice then chooses an $n \times n$ invertible matrix A, with the entries reduced modulo 26, which acts as the key. Encryption is accomplished by computing the ciphertext as $c_i = Ap_i \pmod{26}$ for each plaintext block p_i. Bob decrypts the message by computing $A^{-1}c_i \pmod{26}$, for each ciphertext block c_i, where A^{-1} is the inverse of A, modulo 26.

For example, suppose Alice wants to send the plaintext `MEETMEHERE`, using the encryption matrix

$$A = \begin{bmatrix} 22 & 13 \\ 11 & 5 \end{bmatrix}. \tag{1.5}$$

Converting letters to numbers, Alice finds

$$\texttt{MEETMEHERE} = (12, 4, 4, 19, 12, 4, 7, 4, 17, 4).$$

Next, she divides the plaintext into blocks of length two and then represents each block as a column vector, which yields

$$p_0 = \begin{bmatrix} 12 \\ 4 \end{bmatrix}, \ p_1 = \begin{bmatrix} 4 \\ 19 \end{bmatrix}, \ p_2 = \begin{bmatrix} 12 \\ 4 \end{bmatrix}, \ p_3 = \begin{bmatrix} 7 \\ 4 \end{bmatrix}, \ p_4 = \begin{bmatrix} 17 \\ 4 \end{bmatrix}.$$

To encrypt, Alice computes $c_i = Ap_i \pmod{26}$ for each column vector p_i. In this example, the resulting ciphertext is

$$c_0 = \begin{bmatrix} 4 \\ 22 \end{bmatrix}, \ c_1 = \begin{bmatrix} 23 \\ 9 \end{bmatrix}, \ c_2 = \begin{bmatrix} 4 \\ 22 \end{bmatrix}, \ c_3 = \begin{bmatrix} 24 \\ 19 \end{bmatrix}, \ c_4 = \begin{bmatrix} 10 \\ 25 \end{bmatrix}.$$

Converting into letters, we have

$$(4, 22, 23, 9, 4, 22, 24, 19, 10, 25) = \texttt{EWXJEWYTKZ},$$

which Alice sends to Bob. When Bob receives the ciphertext, he breaks it into blocks c_i of length two and treats these as column vectors. He then decrypts the message by computing $p_i = A^{-1}c_i \pmod{26}$ for each ciphertext block c_i.

The Hill cipher, with an invertible matrix $A \pmod{26}$ and block length n, can be viewed as a substitution cipher utilizing an alphabet of 26^n possible "letters" and the expected letter frequency distribution in the ciphertext is far more uniform than that of the plaintext. This makes a ciphertext only attack generally impractical. However, the Hill cipher is highly vulnerable to a known plaintext attack.

Suppose that Trudy suspects Alice of using a Hill cipher with an $n \times n$ encryption matrix A. Further, suppose that Trudy can obtain ciphertext blocks c_i, for $i = 0, 1, \ldots, n-1$, where each block is of length n, as well as the corresponding plaintext blocks, that is, p_i, for $i = 0, 1, \ldots, n-1$. Then Trudy may be able to recover the key A as follows: Let P and C be the $n \times n$ matrices whose columns are formed by the plaintext p_i and ciphertext c_i, respectively. Then $AP = C$ and if it is the case that $\gcd(\det(P), 26) = 1$, the matrix $P^{-1} \pmod{26}$ exists. If the inverse matrix exists, Trudy can compute P^{-1} and from P^{-1} she can determine A via $A = CP^{-1}$. Once Trudy finds A, the decryption matrix A^{-1} is easily calculated.

The Hill cipher is an example of a *linear cipher*. The linearity of the Hill cipher effectively creates a large number of substitutions, which is desirable. However, the linear structure can be exploited, since linear equations are easy to solve. The lesson here is that a cipher must have some nonlinear component. However, linear components are useful and, in fact, modern ciphers combine both linearity and nonlinearity. In Shannon's terminology [133], linearity provides an effective method to increase diffusion while nonlinearity is essential for confusion.

1.4.3 One-Time Pad

In 1917, Gilbert Vernam and Joseph Mauborgne invented a cipher system which would eventually become known as the *one-time pad*. When correctly used, this system is invulnerable to a ciphertext only attack. This is the only real-world cipher that is provably secure.

Suppose Alice wants to send a message to Bob and she wants to encrypt her message using a one-time pad. Alice first converts her plaintext message P into binary. She then generates a random binary key K of the same length as P. Encryption is accomplished by adding K to P, bit by bit, modulo 2, to obtain the ciphertext C. That is, $C = P \oplus K$, where "$\oplus$" is XOR.

To recover the plaintext P from the ciphertext C, Bob, knowing the key K, computes $C \oplus K = (P \oplus K) \oplus K = P$. For example, suppose $P = 01001100$ and $K = 11010110$. Then

$$C = P \oplus K = 01001100 \oplus 11010110 = 10011010$$

and P can be recovered from C via

$$P = C \oplus K = 10011010 \oplus 11010110 = 01001100.$$

The one-time pad is immune to a ciphertext only attack, since the ciphertext yields no information about the plaintext, other than its length. To see why this is true, consider the eight-letter alphabet in Table 1.8, with the given binary encodings.

Table 1.8: Abbreviated Alphabet

Letter	C	A	T	D	O	G	E	N
Binary	000	001	010	100	011	101	110	111

The plaintext message CAT is encoded as 000 001 010. Suppose the key $K = 110\ 100\ 001$ is used for encryption. Then the ciphertext is given by $C = 110\ 101\ 111$ which corresponds to EGN. Now, suppose Trudy intercepts C and she guesses the putative key $K' = 010\ 110\ 010$. Using K', Trudy computes the putative plaintext

$$P' = C \oplus K' = 110\ 101\ 111 \oplus 010\ 110\ 010 = 100\ 011\ 101$$

which corresponds to the message DOG. Based on the ciphertext and the putative plaintext, Trudy has no way to judge whether the message DOG is any more likely than the message CAT, or any other three letter message that can be spelled from the eight letters in Table 1.8. That is, "decrypting" C with any one of the possible $8^3 = 512$ keys gives one of the 512 possible plaintext messages and the ciphertext itself gives no hint as to which of these is correct.

The "one-time" in the one-time pad is crucial. If a key K is used more than once, then the one-time pad (which is, technically, no longer a one-time pad) is subject to an attack. Different messages encrypted with the same key are said to be in *depth*.

Suppose that plaintext messages P_0 and P_1 are in depth, that is, both are encrypted with a one-time pad using the same key K, yielding ciphertexts C_0 and C_1, respectively. Then if Trudy obtains both ciphertexts, she can compute

$$C_0 \oplus C_1 = (P_0 \oplus K) \oplus (P_1 \oplus K) = P_0 \oplus P_1,$$

that is, Trudy can obtain the XOR of the two plaintexts. It might then be possible for Trudy to "peel apart" these two messages, depending on the properties of the plaintext. The fundamental issue here is that the attacker can, in effect, use one of the messages as a check on any guess for the other message (or the key). Consequently, the ciphertext now provides information about the underlying plaintext and the security can no longer be assured. The problem only gets worse (or, from Trudy's perspective, better) the more the one-time pad is reused.

An obvious practical problem with the one-time pad is that a key having the same length as the plaintext must be securely transmitted to the recipient, and this key can only be used once. If the key can be securely distributed, why not send the message by the same means, in which case there is no need to encrypt?

However, it is important to note that there are some cases where a one-time pad is practical. In some situations it may be easy to send the key at a particular time, and then use it at a later time when it would be difficult or impossible to communicate securely. For example, in the 1930s and 1940s, the Soviet Union used a one-time pad cipher to transmit intelligence gathered from spies in the United States. Soviet agents would simply bring their one-time pads with them when entering the United States, then use these to encrypt sensitive messages as necessary. In fact, these one-time pads were often used more than once and, as a result, many of the messages were eventually broken by United States cryptanalysts. The famous Project VENONA [151] details this impressive cryptanalytic success. The VENONA decrypts provide tremendous insight into Soviet spying in general, and nuclear espionage in particular.

Modern *stream ciphers* are a generalization of the one-time pad, where provable security is traded for practicality. In a stream cipher a short secret key is "stretched" into a long pseudo-random string of bits, which is then used just like a one-time pad. The provable security is lost since the number of possible keys is much smaller than the number of possible messages. Stream ciphers are discussed in Chapter 3.

1.4.4 Codebook Ciphers

Finally, we discuss codebook ciphers, which are, literally, books filled with "codes". In a classic codebook cipher, there are two books, one of which has the plaintext words (or phrases) listed in alphabetical order, each of which is adjacent to its corresponding codeword. A particular word or phrase is encrypted by looking it up in the codebook and replacing it with the appropriate codeword. A corresponding codebook indexed by codewords is used to decrypt. For example, Table 1.9 contains an excerpt from a famous (encryption) codebook of World War I. In fact, this excerpt is from the codebook that was used to encrypt the infamous Zimmermann Telegram [149]. In this particular codebook, the plaintext consists of German words and the ciphertext consists of 5-digit numbers. The inverse codebook, where the words are indexed by the corresponding 5-digit codewords, would be used to decrypt.

Table 1.9: Excerpt from World War I German Codebook

plaintext	ciphertext
`Februar`	13605
`fest`	13732
`finanzielle`	13850
`folgender`	13918
`Frieden`	17142
`Friedenschluss`	17149
⋮	⋮

The security of a classic codebook cipher depends heavily on the physical security of the book itself. That is, the book must be protected from capture by the enemy. In addition, statistical attacks such as those described above for the simple substitution cipher apply equally to codebooks, although the amount of data required to attack a codebook would be much larger. This is due to the fact that the size of the "alphabet" is larger for a codebook, and consequently much more data must be collected before the statistical information can rise above the noise.

As late as World War II, codebooks were in widespread use. Cryptographers realized that these ciphers were subject to statistical attack, so codebooks were regularly replaced with new codebooks. Since this was an expensive and risky process, it was necessary to extend the life of a codebook as much as possible. To this end, an *additive* book was generally used.

Suppose that for a particular codebook cipher, the codewords are all 5-digit numbers. Then the additive book would consist of a long list of randomly generated 5-digit numbers. After a plaintext message had been converted to

a series of 5-digit codewords, a random starting point in the additive book would be selected, and the subsequent 5-digit additives would be added to the codewords to create the ciphertext. For a codebook with 5-digit codewords, the addition would be taken modulo 100,000. In this case, the ith ciphertext word would be

$$C_i = F(P_i) + A_j \pmod{100{,}000},$$

where $F(X)$ is the result of looking up plaintext word X in the codebook, A_j is the additive and P_i is the plaintext. To decrypt,

$$P_i = F^{-1}(C_i - A_j \pmod{100{,}000}),$$

where $F^{-1}(Y)$ is the plaintext word that corresponds to codeword Y. Note that the additive book is required to encrypt or decrypt a message.

Often, the starting point in the additive book was selected at random by the sender and sent in the clear (or in a slightly obfuscated form) at the start of the transmission. The additive information was part of the *message indicator* (MI). In general, an MI includes any information (other than the key) needed by the recipient to decrypt the message correctly. More examples of MIs appear in the next chapter, where we discuss World War II cipher machines.

Note that if the additive material were only used once, the resulting cipher would be a one-time pad and therefore, provably secure. However, in practice, the additive was reused multiple times and, therefore, any messages sent with overlapping additives would have their codewords "encrypted" with the same additives. Therefore, any messages with overlapping additive sequences could be used to gather the statistical information needed to attack the underlying codebook. In effect, the additive book simply increased the amount of ciphertext required to mount a statistical attack on the codebook, which is precisely the effect the cryptographers hoped to achieve.

Modern *block ciphers* are, in a sense, the descendants of classic codebook ciphers. In addition, the concept of an additive also lives on, in the form of a so-called *initialization vector* (IV), which is often used with block ciphers (and sometimes with stream ciphers as well). The use of IVs in block ciphers is discussed in detail in Chapter 4.

1.5 Summary

In this chapter, we introduced the basic terminology used in the remaining chapters, and we gave an overview of a few selected classical cryptosystems. These classic systems illustrate many of the important concepts seen in later chapters where we analyze modern ciphers.

We also considered various aspects of elementary cryptanalysis. Specifically, we mentioned attacks based on each of the following:

- Exhaustive key search.
- Divide and conquer.
- Statistical weaknesses.
- Linearity of the underlying cipher.

These same cryptanalytic principles appear in various forms throughout the subsequent chapters of this book.

The remaining chapters are primarily focused on case studies illustrating the cryptanlysis of specific real-world ciphers. In the next chapter we discuss the cryptanalysis of the three most famous cipher machines from World War II. Then we turn our attention to modern ciphers, including examples of stream ciphers, block ciphers, hash functions and public key systems. All of these attacks are "applied" in the sense that they are realistic attacks that can be used to break the security of real ciphers.

1.6 Problems

1. Many companies use proprietary cryptosystems. Google to find a specific example of a company that has violated Kerckhoffs' Principle.
2. Edgar Allan Poe's 1843 short story, "The Gold Bug," features a cryptanalytic attack. What type of cipher is broken and how?
3. Solve the following congruence: $19x = 3 \pmod{26}$.
4. Fill in the missing steps in the derivation of the formula for the index of coincidence in (1.2).
5. Consider the ciphertext `QJKES REOGH GXXRE OXEO`, which was generated using an affine cipher. Recover the decryption function and decipher the message. Hint: Plaintext `T` encrypts to ciphertext `H` and plaintext `O` encrypts to ciphertext `E`.
6. Decrypt the ciphertext

   ```
   TNFOS FOZSW PZLOC GQAOZ WAGQR PJZPN ABCZP QDOGR AMTHA
   RAXTB AGZJO GMTHA RAVAP ZW.
   ```

   Hint: This is from a simple substitution cipher and the word "liberty" appears in the plaintext.
7. Cryptanalyze the following message, which is from a Vigenère cipher with a 3-letter English keyword:

   ```
   CTMYR DOIBS RESRR RIJYR EBYLD IYMLC CYQXS RRMLQ FSDXF
   OWFKT CYJRR IQZSM X.
   ```

8. Write a computer program to calculate the index of coincidence for an English ciphertext. Compute the index of coincidence for the ciphertext in (1.1).

9. Using the result of Problem 8, compute k in (1.4) for the ciphertext in (1.1).

10. Write a computer program to approximate the key length for a Vigenère ciphertext. Verify your program on the ciphertext in (1.1).

11. The following ciphertext is from a columnar transposition cipher:

    TSEHVAIESSRYIYQ.

    Find the corresponding plaintext.

12. The following ciphertext is from a double transposition cipher, where the encryption matrix is 10×11:

    TNOSSKAIMAGAEITMHETHTSRHXXIHEUXDX
    NUEIDSATDTDDSARAHHENTTTDSOUIOEART
    FHDAOMWYWFERTNEONFDYAHSEIMEDGRWTA
    TISURUARTHJ.

    Find the corresponding plaintext.

13. Verify the derivation of (1.4), which can be used to find the number of letters in the keyword of a Vigenère cipher.

14. Consider the Hill cipher with matrix A as given in (1.5).

    a. Find $A^{-1} \pmod{26}$.

    b. Using the result of part a, decrypt the ciphertext EWXJEWYTKZ and verify that the corresponding plaintext is MEETMEHERE.

    c. Using the same A matrix as in part a, decrypt the ciphertext

    QCNDVUHLKGANIYVUWEGMWTNHHXXD.

15. Consider a one-time pad using the letter encodings in Table 1.8. Suppose that Trudy intercepts $C = 110\ 101\ 111$.

    a. Find a putative key K' such that the corresponding putative plaintext P' yields the word GET.

    b. Find another putative key K'' such that the corresponding putative plaintext is TAG.

16. Using the codebook excerpt in Table 1.9 and the additive sequence

$$A_0 = 88{,}900,\ A_1 = 92{,}331,\ A_2 = 23{,}546$$

encrypt and decrypt the plaintext message

`folgender Frieden Februar.`

Assume that the additive arithmetic is taken modulo 100,000. Show all intermediate steps.

17. Consider two ciphers, Cipher A and Cipher B, and suppose that Cipher A has a 64-bit key, while Cipher B has a 128-bit key. Alice prefers Cipher A, while Bob wants the additional security provided by a 128-bit key, so he insists on Cipher B. As a compromise, Alice proposes that they use Cipher A, but they encrypt each message twice, using two independent 64-bit keys. Assuming that no shortcut attack is available for either cipher, is Alice's approach sound?