Chapter 1

An Introduction to Cryptography

1.1 Simple substitution ciphers

As Julius Caesar surveys the unfolding battle from his hilltop outpost, an exhausted and disheveled courier bursts into his presence and hands him a sheet of parchment containing gibberish:

`j s j r d k f q q n s l g f h p g w j f p y m w t z l m n r r n s j s y q z h n z x`

Within moments, Julius sends an order for a reserve unit of charioteers to speed around the left flank and exploit a momentary gap in the opponent's formation.

How did this string of seemingly random letters convey such important information? The trick is easy, once it is explained. Simply take each letter in the message and shift it five letters up the alphabet. Thus `j` in the *ciphertext* becomes `e` in the *plaintext*,[1] because `e` is followed in the alphabet by `f`,`g`,`h`,`i`,`j`. Applying this procedure to the entire ciphertext yields

```
j s j r d k f q q n s l g f h p g w j f p y m w t z l m n r r n s j s y q z h n z x
-----------------------------------------------------------------------------------
e n e m y f a l l i n g b a c k b r e a k t h r o u g h i m m i n e n t l u c i u s
```

The second line is the decrypted plaintext, and breaking it into words and supplying the appropriate punctuation, Julius reads the message

`Enemy falling back. Breakthrough imminent. Lucius.`

There remains one minor quirk that must be addressed. What happens when Julius finds a letter such as `d`? There is no letter appearing five letters before `d`

[1]The *plaintext* is the original message in readable form and the *ciphertext* is the encrypted message.

in the alphabet. The answer is that he must wrap around to the end of the alphabet. Thus d is replaced by y, since y is followed by z,a,b,c,d.

This wrap-around effect may be conveniently visualized by placing the alphabet abcd...xyz around a circle, rather than in a line. If a second alphabet circle is then placed within the first circle and the inner circle is rotated five letters, as illustrated in Figure 1.1, the resulting arrangement can be used to easily encrypt and decrypt Caesar's messages. To decrypt a letter, simply find it on the inner wheel and read the corresponding plaintext letter from the outer wheel. To encrypt, reverse this process: find the plaintext letter on the outer wheel and read off the ciphertext letter from the inner wheel. And note that if you build a cipherwheel whose inner wheel spins, then you are no longer restricted to always shifting by exactly five letters. Cipher wheels of this sort have been used for centuries.[2]

Although the details of the preceding scene are entirely fictional, and in any case it is unlikely that a message to a Roman general would have been written in modern English(!), there is evidence that Caesar employed this early method of cryptography, which is sometimes called the *Caesar cipher* in his honor. It is also sometimes referred to as a *shift cipher*, since each letter in the alphabet is shifted up or down. *Cryptography*, the methodology of concealing the content of messages, comes from the Greek root words kryptos, meaning hidden,[3] and graphikos, meaning writing. The modern scientific study of cryptography is sometimes referred to as *cryptology*.

In the Caesar cipher, each letter is replaced by one specific substitute letter. However, if Bob encrypts a message for Alice[4] using a Caesar cipher and allows the encrypted message to fall into Eve's hands, it will take Eve very little time to decrypt it. All she needs to do is try each of the 26 possible shifts.

Bob can make his message harder to attack by using a more complicated replacement scheme. For example, he could replace every occurrence of a by z and every occurrence of z by a, every occurrence of b by y and every occurrence of y by b, and so on, exchanging each pair of letters c $\leftrightarrow$ x,..., m $\leftrightarrow$ n.

This is an example of a *simple substitution cipher*, that is, a cipher in which each letter is replaced by another letter (or some other type of symbol). The Caesar cipher is an example of a simple substitution cipher, but there are many simple substitution ciphers other than the Caesar cipher. In fact, a

[2]A cipher wheel with mixed up alphabets and with encryption performed using different offsets for different parts of the message is featured in a 15th century monograph by Leon Batista Alberti [58].

[3]The word cryptic, meaning hidden or occult, appears in 1638, while crypto- as a prefix for concealed or secret makes its appearance in 1760. The term cryptogram appears much later, first occurring in 1880.

[4]In cryptography, it is traditional for Bob and Alice to exchange confidential messages and for their adversary Eve, the eavesdropper, to intercept and attempt to read their messages. This makes the field of cryptography much more personal than other areas of mathematics and computer science, whose denizens are often X and Y!

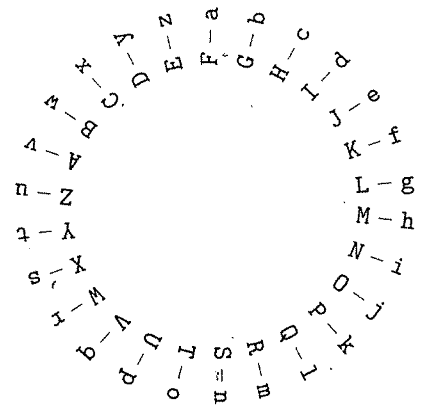

Figure 1.1: A cipher wheel with an offset of five letters

simple substitution cipher may be viewed as a rule or function

$$\{\mathtt{a,b,c,d,e,\ldots,x,y,z}\} \longrightarrow \{\mathtt{A,B,C,D,E,\ldots,X,Y,Z}\}$$

assigning each plaintext letter in the domain a different ciphertext letter in the range. (To make it easier to distinguish the plaintext from the ciphertext, we write the plaintext using lowercase letters and the ciphertext using uppercase letters.) Note that in order for decryption to work, the encryption function must have the property that no two plaintext letters go to the same ciphertext letter. A function with this property is said to be *one-to-one* or *injective*.

A convenient way to describe the encryption function is to create a table by writing the plaintext alphabet in the top row and putting each ciphertext letter below the corresponding plaintext letter.

Example 1.1. A simple substitution encryption table is given in Table 1.1. The ciphertext alphabet (the uppercase letters in the bottom row) is a randomly chosen permutation of the 26 letters in the alphabet. In order to encrypt the plaintext message

`Four score and seven years ago,`

we run the words together, look up each plaintext letter in the encryption table, and write the corresponding ciphertext letter below.

f	o	u	r	s	c	o	r	e	a	n	d	s	e	v	e	n	y	e	a	r	s	a	g	o
N	U	R	B	K	S	U	B	V	C	G	Q	K	V	E	V	G	Z	V	C	B	K	C	F	U

It is then customary to write the ciphertext in five-letter blocks:

`NURBK SUBVC GQKVE VGZVC BKCFU`

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

Table 1.1: Simple substitution encryption table

j	r	a	x	v	g	n	p	b	z	s	t	l	f	h	q	d	u	c	m	o	e	i	k	w	y
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

Table 1.2: Simple substitution decryption table

Decryption is a similar process. Suppose that we receive the message

```
GVVQG VYKCM CQQBV KKWGF SCVKV B
```

and that we know that it was encrypted using Table 1.1. We can reverse the encryption process by finding each ciphertext letter in the second row of Table 1.1 and writing down the corresponding letter from the top row. However, since the letters in the second row of Table 1.1 are all mixed up, this is a somewhat inefficient process. It is better to make a decryption table in which the ciphertext letters in the lower row are listed in alphabetical order and the corresponding plaintext letters in the upper row are mixed up. We have done this in Table 1.2. Using this table, we easily decrypt the message.

G	V	V	Q	G	V	Y	K	C	M	C	Q	Q	B	V	K	K	W	G	F	S	C	V	K	V	B
n	e	e	d	n	e	w	s	a	l	a	d	d	r	e	s	s	i	n	g	c	a	e	s	e	r

Putting in the appropriate word breaks and some punctuation reveals an urgent request!

```
Need new salad dressing. -Caesar
```

1.1.1 Cryptanalysis of simple substitution ciphers

How many different simple substitution ciphers exist? We can count them by enumerating the possible ciphertext values for each plaintext letter. First we assign the plaintext letter a to one of the 26 possible ciphertext letters A–Z. So there are 26 possibilities for a. Next, since we are not allowed to assign b to the same letter as a, we may assign b to any one of the remaining 25 ciphertext letters. So there are $26 \cdot 25 = 650$ possible ways to assign a and b. We have now used up two of the ciphertext letters, so we may assign c to any one of the remaining 24 ciphertext letters. And so on.... Thus the total number of ways to assign the 26 plaintext letters to the 26 ciphertext letters, using each ciphertext letter only once, is

$$26 \cdot 25 \cdot 24 \cdots 4 \cdot 3 \cdot 2 \cdot 1 = 26! = 403291461126605635584000000.$$

There are thus more than 10^{26} different simple substitution ciphers. Each associated encryption table is known as a *key*.

Suppose that Eve intercepts one of Bob's messages and that she attempts to decrypt it by trying every possible simple substitution cipher. The process of decrypting a message without knowing the underlying key is called *cryptanalysis*. If Eve (or her computer) is able to check one million cipher alphabets per second, it would still take her more than 10^{13} years to try them all.[5] But the age of the universe is estimated to be on the order of 10^{10} years. Thus Eve has almost no chance of decrypting Bob's message, which means that Bob's message is secure and he has nothing to worry about![6] Or does he?

It is time for an important lesson in the practical side of the science of cryptography:

> Your opponent always uses her best strategy to defeat you, not the strategy that you want her to use. Thus the security of an encryption system depends on the best known method to break it. As new and improved methods are developed, the level of security can only get worse, never better.

Despite the large number of possible simple substitution ciphers, they are actually quite easy to break, and indeed many newspapers and magazines feature them as a companion to the daily crossword puzzle. The reason that Eve can easily cryptanalyze a simple substitution cipher is that the letters in the English language (or any other human language) are not random. To take an extreme example, the letter q in English is virtually always followed by the letter u. More useful is the fact that certain letters such as e and t appear far more frequently than other letters such as f and c. Table 1.3 lists the letters with their typical frequencies in English text. As you can see, the most frequent letter is e, followed by t, a, o, and n.

Thus if Eve counts the letters in Bob's encrypted message and makes a frequency table, it is likely that the most frequent letter will represent e, and that t, a, o, and n will appear among the next most frequent letters. In this way, Eve can try various possibilities and, after a certain amount of trial and error, decrypt Bob's message.

In the remainder of this section we illustrate how to cryptanalyze a simple substitution cipher by decrypting the message given in Table 1.4. Of course the end result of defeating a simple substitution cipher is not our main goal here. Our key point is to introduce the idea of statistical analysis, which will prove to

[5]Do you see how we got 10^{13} years? There are $60 \cdot 60 \cdot 24 \cdot 365$ seconds in a year, and 26! divided by $10^6 \cdot 60 \cdot 60 \cdot 24 \cdot 365$ is approximately $10^{13.107}$.

[6]The assertion that a large number of possible keys, in and of itself, makes a cryptosystem secure, has appeared many times in history and has equally often been shown to be fallacious.

By decreasing frequency			
E	13.11%	M	2.54%
T	10.47%	U	2.46%
A	8.15%	G	1.99%
O	8.00%	Y	1.98%
N	7.10%	P	1.98%
R	6.83%	W	1.54%
I	6.35%	B	1.44%
S	6.10%	V	0.92%
H	5.26%	K	0.42%
D	3.79%	X	0.17%
L	3.39%	J	0.13%
F	2.92%	Q	0.12%
C	2.76%	Z	0.08%

In alphabetical order			
A	8.15%	N	7.10%
B	1.44%	O	8.00%
C	2.76%	P	1.98%
D	3.79%	Q	0.12%
E	13.11%	R	6.83%
F	2.92%	S	6.10%
G	1.99%	T	10.47%
H	5.26%	U	2.46%
I	6.35%	V	0.92%
J	0.13%	W	1.54%
K	0.42%	X	0.17%
L	3.39%	Y	1.98%
M	2.54%	Z	0.08%

Table 1.3: Frequency of letters in English text

```
LOJUM YLJME PDYVJ QXTDV SVJNL DMTJZ WMJGG YSNDL UYLEO SKDVC
GEPJS MDIPD NEJSK DNJTJ LSKDL OSVDV DNGYN VSGLL OSCIO LGOYG
ESNEP CGYSN GUJMJ DGYNK DPPYX PJDGG SVDNT WMSWS GYLYS NGSKJ
CEPYQ GSGLD MLPYN IUSCP QOYGM JGCPL GDWWJ DMLSL OJCNY NYLYD
LJQLO DLCNL YPLOJ TPJDM NJQLO JWMSE JGGJG XTUOY EOOJO DQDMM
YBJQD LLOJV LOJTV YIOLU JPPES NGYQJ MOYVD GDNJE MSVDN EJM
```

Table 1.4: A simple substitution cipher to cryptanalyze

have many applications throughout cryptography. Although for completeness we provide full details, the reader may wish to skim this material.

There are 298 letters in the ciphertext. The first step is to make a frequency table listing how often each ciphertext letter appears.

	J	L	D	G	Y	S	O	N	M	P	E	V	Q	C	T	W	U	K	I	X	Z	B	A	F	R	H
Freq	32	28	27	24	23	22	19	18	17	15	12	12	8	8	7	6	6	5	4	3	1	1	0	0	0	0
%	11	9	9	8	8	7	6	6	6	5	4	4	3	3	2	2	2	2	1	1	0	0	0	0	0	0

Table 1.5: Frequency table for Table 1.4—Ciphertext length: 298

The ciphertext letter J appears most frequently, so we make the provisional guess that it corresponds to the plaintext letter e. The next most frequent ciphertext letters are L (28 times) and D (27 times), so we might guess from Table 1.3 that they represent t and a. However, the letter frequencies in a short message are unlikely to exactly match the percentages in Table 1.3. All that we can say is that among the ciphertext letters L, D, G, Y, and S are likely to appear several of the plaintext letters t, a, o, n, and r.

th	he	an	re	er	in	on	at	nd	st	es	en	of	te	ed
168	132	92	91	88	86	71	68	61	53	52	51	49	46	46

(a) Most common English bigrams (frequency per 1000 words)

LO	OJ	GY	DN	VD	YL	DL	DM	SN	KD	LY	NG	OY	JD	SK	EP	JG	SV	JM	JQ
9	7	6 each		5 each					4 each										

(b) Most common bigrams appearing in the ciphertext in Table 1.4

Table 1.6: Bigram frequencies

There are several ways to proceed. One method is to look at *bigrams*, which are pairs of consecutive letters. Table 1.6(a) lists the bigrams that most frequently appear in English, and Table 1.6(b) lists the ciphertext bigrams that appear most frequently in our message. The ciphertext bigrams LO and OJ appear frequently. We have already guessed that J = e, and based on its frequency we suspect that L is likely to represent one of the letters t, a, o, n, or r. Since the two most frequent English bigrams are th and he, we make the tentative identifications

$$\text{LO} = \text{th} \qquad \text{and} \qquad \text{OJ} = \text{he}.$$

We substitute the guesses J = e, L = t, and O = h, into the ciphertext, writing the putative plaintext letter below the corresponding ciphertext letter.

```
LOJUM YLJME PDYVJ QXTDV SVJNL DMTJZ WMJGG YSNDL UYLEO SKDVC
the-- -te-- ----e ----- --e-t ---e- --e-- ----t --t-h -----
-----------------------------------------------------------
GEPJS MDIPD NEJSK DNJTJ LSKDL OSVDV DNGYN VSGLL OSCIO LGOYG
---e- ----- --e-- --e-e t---t h---- ----- ---tt h---h t-h--
-----------------------------------------------------------
ESNEP CGYSN GUJMJ DGYNK DPPYX PJDGG SVDNT WMSWS GYLYS NGSKJ
----- ----- --e-e ----- ----- -e--- ----- ----- --t-- ----e
-----------------------------------------------------------
CEPYQ GSGLD MLPYN IUSCP QOYGM JGCPL GDWWJ DMLSL OJCNY NYLYD
----- ---t- -t--- ----- -h--- e---t ----e --t-t he--- --t--
-----------------------------------------------------------
LJQLO DLCNL YPLOJ TPJDM NJQLO JWMSE JGGJG XTUOY EOOJO DQDMM
te-th -t--t --the --e-- -e-th e---- e--e- ---h- -hheh -----
-----------------------------------------------------------
YBJQD LLOJV LOJTV YIOLU JPPES NGYQJ MOYVD GDNJE MSVDN EJM
--e-- tthe- the-- --ht- e---- ----e -h--- ---e- ----- -e-
```

At this point, we can look at the fragments of plaintext and attempt to guess some common English words. For example, in the second line we see the three blocks

```
VSGLL OSCIO LGOYG,
---tt h---h t-h--.
```

Looking at the fragment th---ht, we might guess that this is the word thought, which gives three more equivalences,

$$\mathtt{S} = \mathtt{o}, \qquad \mathtt{C} = \mathtt{u}, \qquad \mathtt{I} = \mathtt{g}.$$

This yields

```
LOJUM YLJME PDYVJ QXTDV SVJNL DMTJZ WMJGG YSNDL UYLEO SKDVC
the-- -te-- ----e ----- o-e-t ---e- --e-- -o--t --t-h o---u
-----------------------------------------------------------
GEPJS MDIPD NEJSK DNJTJ LSKDL OSVDV DNGYN VSGLL OSCIO LGOYG
---eo --g-- --eo- --e-e to--t ho--- ----- -o-tt hough t-h--
-----------------------------------------------------------
ESNEP CGYSN GUJMJ DGYNK DPPYX PJDGG SVDNT WMSWS GYLYS NGSKJ
-o--- u--o- --e-e ----- ----- -e--- o---- --o-o --t-o --o-e
-----------------------------------------------------------
CEPYQ GSGLD MLPYN IUSCP QOYGM JGCPL GDWWJ DMLSL OJCNY NYLYD
u---- -o-t- -t--- g-ou- -h--- e-u-t ----e --tot heu-- --t--
-----------------------------------------------------------
LJQLO DLCNL YPLOJ TPJDM NJQLO JWMSE JGGJG XTUOY EOOJO DQDMM
te-th -tu-t --the --e-- -e-th e--o- e--e- ---h- -hheh -----
-----------------------------------------------------------
YBJQD LLOJV LOJTV YIOLU JPPES NGYQJ MOYVD GDNJE MSVDN EJM
--e-- tthe- the-- -ght- e---o ----e -h--- ---e- -o--- -e-
```

Now look at the three letters ght in the last line. They must be preceded by a vowel, and the only vowels left are a and i, so we guess that Y = i. Then we find the letters itio in the third line, and we guess that they are followed by an n, which gives N = n. (There is no reason that a letter cannot represent itself, although this is often forbidden in the puzzle ciphers that appear in newspapers.) We now have

```
LOJUM YLJME PDYVJ QXTDV SVJNL DMTJZ WMJGG YSNDL UYLEO SKDVC
the-- ite-- --i-e ----- o-ent ---e- --e-- ion-t -it-h o---u
-----------------------------------------------------------
GEPJS MDIPD NEJSK DNJTJ LSKDL OSVDV DNGYN VSGLL OSCIO LGOYG
---eo --g-- n-eo- -ne-e to--t ho--- -n-in -o-tt hough t-hi-
-----------------------------------------------------------
ESNEP CGYSN GUJMJ DGYNK DPPYX PJDGG SVDNT WMSWS GYLYS NGSKJ
-on-- u-ion --e-e --in- ---i- -e--- o--n- --o-o -itio n-o-e
-----------------------------------------------------------
CEPYQ GSGLD MLPYN IUSCP QOYGM JGCPL GDWWJ DMLSL OJCNY NYLYD
u--i- -o-t- -t-in g-ou- -hi-- e-u-t ----e --tot heuni niti-
-----------------------------------------------------------
LJQLO DLCNL YPLOJ TPJDM NJQLO JWMSE JGGJG XTUOY EOOJO DQDMM
te-th -tunt i-the --e-- ne-th e--o- e--e- ---hi -hheh -----
-----------------------------------------------------------
YBJQD LLOJV LOJTV YIOLU JPPES NGYQJ MOYVD GDNJE MSVDN EJM
i-e-- tthe- the-- ight- e---o n-i-e -hi-- --ne- -o--n -e-
```

So far, we have reconstructed the following plaintext/ciphertext pairs:

	J	L	D	G	Y	S	O	N	M	P	E	V	Q	C	T	W	U	K	I	X	Z	B	A	F	R	H
	e	t	-	-	i	o	h	n	-	-	-	-	-	u	-	-	-	-	g	-	-	-	-	-	-	-
Freq	32	28	27	24	23	22	19	18	17	15	12	12	8	8	7	6	6	5	4	3	1	1	0	0	0	0

Recall that the most common letters in English (Table 1.3) are, in order of decreasing frequency,

$$\mathtt{e},\ \mathtt{t},\ \mathtt{a},\ \mathtt{o},\ \mathtt{n},\ \mathtt{r},\ \mathtt{i},\ \mathtt{s},\ \mathtt{h}.$$

We have already assigned ciphertext values to e, t, o, n, i, h, so we guess that D and G represent two of the three letters a, r, s. In the third line we notice that GYLYSN gives -ition, so clearly G must be s. Similarly, on the fifth line we have LJQLO DLCNL equal to te-th -tunt, so D must be a, not r. Substituting these new pairs G = s and D = a gives

```
LOJUM YLJME PDYVJ QXTDV SVJNL DMTJZ WMJGG YSNDL UYLEO SKDVC
the-- ite-- -ai-e ---a- o-ent a--e- --ess ionat -it-h o-a-u
-----------------------------------------------------------
GEPJS MDIPD NEJSK DNJTJ LSKDL OSVDV DNGYN VSGLL OSCIO LGOYG
s--eo -ag-a n-eo- ane-e to-at ho-a- ansin -ostt hough tshis
-----------------------------------------------------------
ESNEP CGYSN GUJMJ DGYNK DPPYX PJDGG SVDNT WMSWS GYLYS NGSKJ
-on-- usion s-e-e asin- a--i- -eass o-an- --o-o sitio nso-e
-----------------------------------------------------------
CEPYQ GSGLD MLPYN IUSCP QOYGM JGCPL GDWWJ DMLSL OJCNY NYLYD
u--i- sosta -t-in g-ou- -his- esu-t sa--e a-tot heuni nitia
-----------------------------------------------------------
LJQLO DLCNL YPLOJ TPJDM NJQLO JWMSE JGGJG XTUOY EOOJO DQDMM
te-th atunt i-the --ea- ne-th e--o- esses ---hi -hheh a-a--
-----------------------------------------------------------
YBJQD LLOJV LOJTV YIOLU JPPES NGYQJ MOYVD GDNJE MSVDN EJM
i-e-a tthe- the-- ight- e---o nsi-e -hi-a sane- -o-an -e-
```

It is now easy to fill in additional pairs by inspection. For example, the missing letter in the fragment atunt i-the on the fifth line must be l, which gives P = l, and the missing letter in the fragment -osition on the third line must be p, which gives W = p. Substituting these in, we find the fragment e-p-ession on the first line, which gives Z = x and M = r, and the fragment -on-lusion on the third line, which gives E = c. Then consi-er on the last line gives Q = d and the initial words the-riterclai-e- must be the phrase "the writer claimed," yielding U = w and V = m. This gives

```
LOJUM YLJME PDYVJ QXTDV SVJNL DMTJZ WMJGG YSNDL UYLEO SKDVC
thewr iterc laime d--am oment ar-ex press ionat witch o-amu
-----------------------------------------------------------
GEPJS MDIPD NEJSK DNJTJ LSKDL OSVDV DNGYN VSGLL OSCIO LGOYG
scleo ragla nceo- ane-e to-at homam ansin mostt hough tshis
-----------------------------------------------------------
ESNEP CGYSN GUJMJ DGYNK DPPYX PJDGG SVDNT WMSWS GYLYS NGSKJ
concl usion swere asin- alli- leass oman- propo sitio nso-e
-----------------------------------------------------------
CEPYQ GSGLD MLPYN IUSCP QOYGM JGCPL GDWWJ DMLSL OJCNY NYLYD
uclid sosta rtlin gwoul dhisr esult sappe artot heuni nitia
-----------------------------------------------------------
LJQLO DLCNL YPLOJ TPJDM NJQLO JWMSE JGGJG XTUOY EOOJO DQDMM
tedth atunt ilthe -lear nedth eproc esses --whi chheh adarr
-----------------------------------------------------------
YBJQD LLOJV LOJTV YIOLU JPPES NGYQJ MOYVD GDNJE MSVDN EJM
i-eda tthem the-m ightw ellco nside rhima sanec roman cer
```

It is now a simple matter to fill in the few remaining letters and put in the appropriate word breaks, capitalization, and punctuation to recover the plaintext:

> The writer claimed by a momentary expression, a twitch of a muscle or a glance of an eye, to fathom a man's inmost thoughts. His

conclusions were as infallible as so many propositions of Euclid. So startling would his results appear to the uninitiated that until they learned the processes by which he had arrived at them they might well consider him as a necromancer.[7]

1.2 Divisibility and greatest common divisors

Much of modern cryptography is built on the foundations of algebra and number theory. So before we explore the subject of cryptography, we need to develop some important tools. In the next four sections we begin this development by describing and proving fundamental results from algebra and number theory. If you have already studied number theory in another course, a brief review of this material will suffice. But if this material is new to you, then it is vital to study it closely and to work out the exercises provided at the end of the chapter.

At the most basic level, *Number Theory* is the study of the natural numbers

$$1, 2, 3, 4, 5, 6, \ldots,$$

or slightly more generally, the study of the integers

$$\ldots, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, \ldots.$$

The set of integers is denoted by the symbol $\mathbb{Z}$. Integers can be added, subtracted, and multiplied in the usual way, and they satisfy all the usual rules of arithmetic (commutative law, associative law, distributive law, etc.). The set of integers with their addition and multiplication rules are an example of a *ring*. See Section 2.10.1 for more about the theory of rings.

If a and b are integers, then we can add them $a + b$, subtract them $a - b$, and multiply them $a \cdot b$. In each case, we get an integer as the result. This property of staying inside of our original set after applying operations to a pair of elements is characteristic of a ring.

But if we want to stay within the integers, then we are not always able to divide one integer by another. For example, we cannot divide 3 by 2, since there is no integer that is equal to $\frac{3}{2}$. This leads to the fundamental concept of divisibility.

Definition. Let a and b be integers with $b \neq 0$. We say that b *divides* a, or that a *is divisible by* b, if there is an integer c such that

$$a = bc.$$

We write $b \mid a$ to indicate that b divides a. If b does not divide a, then we write $b \nmid a$.

[7]*A Study in Scarlet* (Chapter 2), Sir Arthur Conan Doyle.

Remark 1.33. If p is large, then the finite field $\mathbb{F}_p$ has quite a few primitive roots. The precise formula says that $\mathbb{F}_p$ has exactly $\phi(p-1)$ primitive roots, where ϕ is Euler's phi function (see page 22). For example, you can check that the following is a complete list of the primitive roots for $\mathbb{F}_{29}$:

$$\{2, 3, 8, 10, 11, 14, 15, 18, 19, 21, 26, 27\}.$$

This agrees with the value $\phi(28) = 12$. More generally, if k divides $p-1$, then there are exactly $\phi(k)$ elements of $\mathbb{F}_p^*$ having order k.

1.6 Cryptography before the computer age

We pause for a short foray into the history of precomputer cryptography. Our hope is that these brief notes will whet your appetite for further reading on this fascinating subject, in which political intrigue, daring adventure, and romantic episodes play an equal role with technical achievements.

The origins of cryptography are lost in the mists of time, but presumably secret writing arose shortly after people started using some form of written communication, since one imagines that the notion of confidential information must date back to the dawn of civilization. There are early recorded descriptions of ciphers being used in Roman times, including Julius Caesar's shift cipher from Section 1.1, and certainly from that time onward, many civilizations have used both substitution ciphers, in which each letter is replaced by another letter or symbol, and transposition ciphers, in which the order of the letters is rearranged.

The invention of cryptanalysis, that is, the art of decrypting messages without previous knowledge of the key, is more recent. The oldest surviving texts, which include references to earlier lost volumes, are by Arab scholars from the 14th and 15th centuries. These books describe not only simple substitution and transposition ciphers, but also the first recorded instance of a homophonic substitution cipher, which is a cipher in which a single plaintext letter may be represented by any one of several possible ciphertext letters. More importantly, they contain the first description of serious methods of cryptanalysis, including the use of letter frequency counts and the likelihood that certain pairs of letters will appear adjacent to one another. Unfortunately, most of this knowledge seems to have disappeared by the 17th century.

Meanwhile, as Europe emerged from the Middle Ages, political states in Italy and elsewhere required secure communications, and both cryptography and cryptanalysis began to develop. The earliest known European homophonic substitution cipher dates from 1401. The use of such a cipher suggests contemporary knowledge of cryptanalysis via frequency analysis, since the only reason to use a homophonic system is to make such cryptanalysis more difficult.

In the 15th and 16th centuries there arose a variety of what are known as polyalphabetic ciphers. (We will see an example of a polyalphabetic cipher, called the Vigenère cipher, in Section 4.2.) The basic idea is that each letter of the plaintext is enciphered using a different simple substitution cipher. The name "polyalphabetic" refers to the use of many different cipher alphabets, which were used according to some sort of key. If the key is reasonably long, then it takes a long time for the any given cipher alphabet to be used a second time. It wasn't until the 19th century that statistical methods were developed to reliably solve such systems, although there are earlier recorded instances of cryptanalysis via special tricks or lucky guesses of part of the message or the key. Jumping forward several centuries, we note that the machine ciphers that played a large role in World War II were, in essence, extremely complicated polyalphabetic ciphers.

Ciphers and codes[13] for both political and military purposes become increasingly widespread during the 18th, 19th, and early 20th centuries, as did cryptanalytic methods, although the level of sophistication varied widely from generation to generation and from country to country. For example, as the United States prepared to enter World War I in 1917, the U.S. Army was using ciphers, inferior to those invented in Italy in the 1600s, that any trained cryptanalyst of the time would have been able to break in a few hours!

The invention and widespread deployment of long-range communication methods, especially the telegraph, opened the need for political, military, and commercial ciphers, and there are many fascinating stories of intercepted and decrypted telegraph messages playing a role in historical events. One example, the infamous Zimmerman telegram, will suffice. With the United States maintaining neutrality in 1917 as Germany battled France and Britain on the Western Front, the Germans decided that their best hope for victory was to tighten their blockade of Britain by commencing unrestricted submarine warfare in the Atlantic. This policy, which meant sinking ships from neutral countries, was likely to bring the United States into the war, so Germany decided to offer an alliance to Mexico. In return for Mexico invading the United States, and thus distracting it from the ground war in Europe, Germany proposed giving Mexico, at the conclusion of the war, much of present-day Texas, New Mexico, and Arizona. The British secret service intercepted this communication, and despite the fact that it was encrypted using one of Germany's most secure cryptosystems, they were able to decipher the cable and pass its contents on to the United States, thereby helping to propel the United States into World War I.

The invention and development of radio communications around 1900 caused an even more striking change in the cryptographic landscape, especially in urgent military and political situations. A general could now

[13]In classical terminology, a code is a system in which each word of the plaintext is replaced with a code word. This requires sender and receiver to share a large dictionary in which plaintext words are paired with their ciphertext equivalents. Ciphers operate on the individual letters of the plaintext, either by substitution, transposition, or some combination. This distinction between the words "code" and "cipher" seems to have been largely abandoned in today's literature.

instantaneously communicate with all of his troops, but unfortunately the enemy could listen in on all of his broadcasts. The need for secure and efficient ciphers became paramount and led to the invention of machine ciphers, such as Germany's Enigma machine. This was a device containing a number of rotors, each of which had many wires running through its center. Before a letter was encrypted, the rotors would spin in a predetermined way, thereby altering the paths of the wires and the resultant output. This created an immensely complicated polyalphabetic cipher in which the number of cipher alphabets was enormous. Further, the rotors could be removed and replaced in a vast number of different starting configurations, so breaking the system involved knowing both the circuits through the rotors and figuring out that day's initial rotor configuration.

Despite these difficulties, during World War II the British managed to decipher a large number of messages encrypted on Enigma machines. They were aided in this endeavor by Polish cryptographers who, just before hostilities commenced, shared with Britain and France the methods that they had developed for attacking Enigma. But determining daily rotor configurations and analyzing rotor replacements was still an immensely difficult task, especially after Germany introduced an improved Enigama machine having an extra rotor. The existence of Britain's ULTRA project to decrypt Enigma remained secret until 1974, but there are now several popular accounts. Military intelligence derived from ULTRA was of vital importance in the Allied war effort.

Another WWII cryptanalytic success was obtained by United States cryptographers against a Japanese cipher machine that they code-named Purple. This machine used switches, rather than rotors, but again the effect was to create an incredibly complicated polyalphabetic cipher. A team of cryptographers, led by William Friedman, managed to reconstruct the design of the Purple machine purely by analyzing intercepted encrypted messages. They then built their own machine and proceeded to decrypt many important diplomatic messages.

In this section we have barely touched the surface of the history of cryptography from antiquity through the middle of the 20th century. Good starting points for further reading include Simon Singh's light introduction [128] and David Kahn's massive and comprehensive, but fascinating and quite readable, book *The Codebreakers* [58].

1.7 Symmetric and asymmetric ciphers

We have now seen several different examples of ciphers, all of which have a number of features in common. Bob wants to send a secret message to Alice. He uses a secret key k to scramble his plaintext message m and turn it into a ciphertext c. Alice, upon receiving c, uses the secret key k to unscramble c and reconstitute m. If this procedure is to work properly, then both Alice and Bob

· general $\boxed{M} \to x \to \boxed{M^{-1}}$

Symmetric: "$M = M^{-1}$" — e.g. XOR

important b/c not mathematically costly

must possess copies of the secret key k, and if the system is to provide security, then their adversary Eve must not know k, must not be able to guess k, and must not be able to recover m from c without knowing k.

In this section we formulate the notion of a cryptosystem in abstract mathematical terms. There are many reasons why this is desirable. In particular, it allows us to highlight similarities and differences between different systems, while also providing a framework within which we can rigorously analyze the security of a cryptosystem against various types of attacks.

1.7.1 Symmetric ciphers

Returning to Bob and Alice, we observe that they must share knowledge of the secret key k. Using that secret key, they can both encrypt and decrypt messages, so Bob and Alice have equal (or symmetric) knowledge and abilities. For this reason, ciphers of this sort are known as *symmetric ciphers*. Mathematically, a symmetric cipher uses a key k chosen from a space (i.e., a set) of possible keys $\mathcal{K}$ to encrypt a plaintext message m chosen from a space of possible messages $\mathcal{M}$, and the result of the encryption process is a ciphertext c belonging to a space of possible ciphertexts $\mathcal{C}$.

Thus encryption may be viewed as a function

$$e : \mathcal{K} \times \mathcal{M} \to \mathcal{C}$$

whose domain $\mathcal{K} \times \mathcal{M}$ is the set of pairs (k, m) consisting of a key k and a plaintext m and whose range is the space of ciphertexts $\mathcal{C}$. Similarly, decryption is a function

$$d : \mathcal{K} \times \mathcal{C} \to \mathcal{M}.$$

Of course, we want the decryption function to "undo" the results of the encryption function. Mathematically, this is expressed by the formula

$$d(k, e(k, m)) = m \qquad \text{for all } k \in \mathcal{K} \text{ and all } m \in \mathcal{M}.$$

It is sometimes convenient to write the dependence on k as a subscript. Then for each key k, we get a pair of functions

$$e_k : \mathcal{M} \longrightarrow \mathcal{C} \qquad \text{and} \qquad d_k : \mathcal{C} \longrightarrow \mathcal{M}$$

satisfying the decryption property

$$d_k\big(e_k(m)\big) = m \qquad \text{for all } m \in \mathcal{M}.$$

In other words, for every key k, the function d_k is the inverse function of the function e_k. In particular, this means that e_k must be one-to-one, since if $e_k(m) = e_k(m')$, then

$$m = d_k\big(e_k(m)\big) = d_k\big(e_k(m')\big) = m'.$$

Example of direct encoding scheme: shift (like ex. 1.41) or substitution or binary XOR, etc.

It is safest for Alice and Bob to assume that Eve knows the encryption method that is being employed. In mathematical terms, this means that Eve knows the functions e and d. What Eve does not know is the particular key k that Alice and Bob are using. For example, if Alice and Bob use a simple substitution cipher, they should assume that Eve is aware of this fact. This illustrates a basic premise of modern cryptography called *Kerckhoff's principle*, which says that the security of a cryptosystem should depend only on the secrecy of the key, and not on the secrecy of the encryption algorithm itself.

If $(\mathcal{K}, \mathcal{M}, \mathcal{C}, e, d)$ is to be a successful cipher, it must have the following properties:

1. For any key $k \in \mathcal{K}$ and plaintext $m \in \mathcal{M}$, it must be easy to compute the ciphertext $e_k(m)$.

2. For any key $k \in \mathcal{K}$ and ciphertext $c \in \mathcal{C}$, it must be easy to compute the plaintext $d_k(c)$.

3. Given one or more ciphertexts $c_1, c_2, \ldots, c_n \in \mathcal{C}$ encrypted using the key $k \in \mathcal{K}$, it must be very difficult to compute any of the corresponding plaintexts $d_k(c_1), \ldots, d_k(c_n)$ without knowledge of k.

There is a fourth property that is desirable, although it is more difficult to achieve.

4. Given one or more pairs of plaintexts and their corresponding ciphertexts, $(m_1, c_1), (m_2, c_2), \ldots, (m_n, c_n)$, it must be difficult to decrypt any ciphertext c that is not in the given list without knowing k. This is known as security against a *chosen plaintext attack.*

Example 1.34. The simple substitution cipher does not have Property 4, since even a single plaintext/ciphertext pair (m, c) reveals most of the encryption table. Thus simple substitution ciphers are vulnerable to chosen plaintext attacks. See Exercise 1.41 for a further example.

In our list of four desirable properties for a cryptosystem, we have left open the question of what exactly is meant by the words "easy" and "hard." We defer a formal discussion of this profound question to Section 4.7 (see also Sections 2.1 and 2.6). For now, we informally take "easy" to mean computable in less than a second on a typical desktop computer and "hard" to mean that all of the computing power in the world would require several years (at least) to perform the computation.

1.7.2 Encoding schemes

It is convenient to view keys, plaintexts, and ciphertexts as numbers and to write those numbers in binary form. For example, we could take strings of

ASCII as intermediate step (encoding scheme)

	32	00100000
(	40	00101000
)	41	00101001
,	44	00101100
.	46	00101110

A	65	01000001
B	66	01000010
C	67	01000011
D	68	01000100
⋮	⋮	⋮
X	88	01011000
Y	89	01011001
Z	90	01011010

a	97	01100001
b	98	01100010
c	99	01100011
d	100	01100100
⋮	⋮	⋮
x	120	01111000
y	121	01111001
z	122	01111010

Table 1.10: The ASCII encoding scheme

eight bits,[14] which give numbers from 0 to 255, and use them to represent the letters of the alphabet via

$$\mathtt{a} = 00000000, \quad \mathtt{b} = 00000001, \quad \mathtt{c} = 00000010, \quad \ldots, \quad \mathtt{z} = 00011001.$$

To distinguish lowercase from uppercase, we could let A = 00011011, B = 00011100, and so on. This encoding method allows up to 256 distinct symbols to be translated into binary form.

Your computer may use a method of this type, called the ASCII code,[15] to store data, although for historical reasons the alphabetic characters are not assigned the lowest binary values. Part of the ASCII code is listed in Table 1.10. For example, the phrase "Bed bug." (including spacing and punctuation) is encoded in ASCII as

B	e	d	␣	b	u	g	.
66	101	100	32	98	117	103	46
01000010	01100101	01100100	00100000	01100010	01110101	01100111	00101110

Thus where you see the phrase "Bed bug.", your computer sees the list of bits

$$0100001001100101011001000010000001100010011101010110011100101110.$$

Definition. An encoding scheme is a method of converting one sort of data into another sort of data, for example, converting text into numbers. The distinction between an encoding scheme and an encryption scheme is one of intent. An encoding scheme is assumed to be entirely public knowledge and used by everyone for the same purposes. An encryption scheme is designed to hide information from anyone who does not possess the secret key. Thus an encoding scheme, like an encryption scheme, consists of an encoding function and its inverse decoding function, but for an encoding scheme, both functions are public knowledge and should be fast and easy to compute.

[14]A *bit* is a 0 or a 1. The word "bit" is an abbreviation for *binary digit.*

[15]ASCII is an acronym for American Standard Code for Information Interchange.

With the use of an encoding scheme, a plaintext or ciphertext may be viewed as a sequence of binary blocks, where each block consists of eight bits, i.e., of a sequence of eight ones and zeros. A block of eight bits is called a *byte*. For human comprehension, a byte is often written as a decimal number between 0 and 255, or as a two-digit hexadecimal (base 16) number between 00 and FF. Computers often operate on more than one byte at a time. For example, a 64-bit processor operates on eight bytes at a time.

1.7.3 Symmetric encryption of encoded blocks

In using an encoding scheme as described in Section 1.7.2, it is convenient to view the elements of the plaintext space $\mathcal{M}$ as consisting of bit strings of a fixed length B, i.e., strings of exactly B ones and zeros. We call B the *blocksize* of the cipher. A general plaintext message then consists of a list of message blocks chosen from $\mathcal{M}$, and the encryption function transforms the message blocks into a list of ciphertext blocks in $\mathcal{C}$, where each block is a sequence of B bits. If the plaintext ends with a block of fewer than B bits, we pad the end of the block with zeros. Keep in mind that this encoding process, which converts the original plaintext message into a sequence of blocks of bits in $\mathcal{M}$, is public knowledge.

Encryption and decryption are done one block at a time, so it suffices to study the process for a single plaintext block, i.e., for a single $m \in \mathcal{M}$. This, of course, is why it is convenient to break a message up into blocks. A message can be of arbitrary length, so it's nice to be able to focus the cryptographic process on a single piece of fixed length. The plaintext block m is a string of B bits, which for concreteness we identify with the corresponding number in binary form. In other words, we identify $\mathcal{M}$ with the set of integers m satisfying $0 \le m < 2^B$ via

$$\overbrace{m_{B-1}m_{B-2}\cdots m_2m_1m_0}^{\text{list of } B \text{ bits of } m} \longleftrightarrow \overbrace{m_{B-1}\cdot 2^{B-1} + \cdots + m_2\cdot 2^2 + m_1\cdot 2 + m_0}^{\text{integer between 0 and } 2^B - 1}.$$

Here $m_0, m_1, \ldots, m_{B-1}$ are each 0 or 1.

Similarly, we identify the key space $\mathcal{K}$ and the ciphertext space $\mathcal{C}$ with sets of integers corresponding to bit strings of a certain blocksize. For notational convenience, we denote the blocksizes for keys, plaintexts, and ciphertexts by B_k, B_m, and B_c. They need not be the same. Thus we have identified $\mathcal{K}$, $\mathcal{M}$, and $\mathcal{C}$ with sets of positive integers

$$\mathcal{K} = \{k \in \mathbb{Z} : 0 \le k < 2^{B_k}\},$$
$$\mathcal{M} = \{m \in \mathbb{Z} : 0 \le m < 2^{B_m}\},$$
$$\mathcal{C} = \{c \in \mathbb{Z} : 0 \le c < 2^{B_c}\}.$$

An important question immediately arises: how large should Alice and Bob make the set $\mathcal{K}$, or equivalently, how large should they choose the key blocksize B_k? If B_k is too small, then Eve can check every number from 0 to $2^{B_k} - 1$

until she finds Alice and Bob's key. More precisely, since Eve is assumed to know the decryption algorithm d (Kerckhoff's principle), she takes each $k \in \mathcal{K}$ and uses it to compute $d_k(c)$. Assuming that Eve is able to distinguish between valid and invalid plaintexts, eventually she will recover the message.

This attack is known as an *exhaustive search attack* (also sometimes referred to as a *brute-force attack*), since Eve exhaustively searches through the key space. With current technology, an exhaustive search is considered to be infeasible if the space has at least 2^{80} elements. Thus Bob and Alice should definitely choose $B_k \ge 80$.

For many cryptosystems, especially the public key cryptosystems that form the core of this book, there are refinements on the exhaustive search attack that effectively replace the size of the space with its square root. These methods are based on the principle that it is easier to find matching objects (collisions) in a set than it is to find a particular object in the set. We describe some of these *meet-in-the-middle* or *collision attacks* in Sections 2.7, 4.4, 4.5, 6.2, and 6.10. If meet-in-the-middle attacks are available, then Alice and Bob should choose $B_k \ge 160$.

1.7.4 Examples of symmetric ciphers

Before descending further into a morass of theory and notation, we pause to give a mathematical description of some elementary symmetric ciphers.

Let p be a large prime,[16] say $2^{159} < p < 2^{160}$. Alice and Bob take their key space $\mathcal{K}$, plaintext space $\mathcal{M}$, and ciphertext space $\mathcal{C}$ to be the same set,

$$\mathcal{K} = \mathcal{M} = \mathcal{C} = \{1, 2, 3, \ldots, p-1\}.$$

In fancier terminology, $\mathcal{K} = \mathcal{M} = \mathcal{C} = \mathbb{F}_p^*$ are all taken to be equal to the group of units in the finite field $\mathbb{F}_p$.

Alice and Bob randomly select a key $k \in \mathcal{K}$, i.e., they select an integer k satisfying $1 \le k < p$, and they decide to use the encryption function e_k defined by

$$e_k(m) \equiv k \cdot m \pmod{p}. \tag{1.9}$$

Here we mean that $e_k(m)$ is set equal to the unique positive integer between 1 and p that is congruent to $k \cdot m$ modulo p. The corresponding decryption function d_k is

$$d_k(c) \equiv k' \cdot c \pmod{p},$$

where k' is the inverse of k modulo p. It is important to note that although p is very large, the extended Euclidean algorithm (Remark 1.15) allows us to calculate k' in fewer than $2\log_2 p + 2$ steps. Thus finding k' from k counts as "easy" in the world of cryptography.

[16]There are in fact many primes in the interval $2^{159} < p < 2^{160}$. The prime number theorem implies that almost 1% of the numbers in this interval are prime. Of course, there is also the question of identifying a number as prime or composite. There are efficient tests that do this, even for very large numbers. See Section 3.4.

more bad example: multiplication (in $\mathbb{Z}$, $\mathbb{Z}_p$)

It is clear that Eve has a hard time guessing k, since there are approximately 2^{160} possibilities from which to choose. Is it also difficult for Eve to recover k if she knows the ciphertext c? The answer is yes, it is still difficult. Notice that the encryption function

$$e_k : \mathcal{M} \longrightarrow \mathcal{C}$$

is surjective (onto) for any choice of key k. This means that for every $c \in \mathcal{C}$ and any $k \in \mathcal{K}$ there exists an $m \in \mathcal{M}$ such that $e_k(m) = c$. Further, any given ciphertext may represent any plaintext, provided that the plaintext is encrypted by an appropriate key. Mathematically, this may be rephrased by saying that given any ciphertext $c \in \mathcal{C}$ and any plaintext $m \in \mathcal{M}$, there exists a key k such that $e_k(m) = c$. Specifically this is true for the key

$$k \equiv m^{-1} \cdot c \pmod{p}. \tag{1.10}$$

This shows that Alice and Bob's cipher has Properties 1, 2, and 3 as listed on page 38, since anyone who knows the key k can easily encrypt and decrypt, but it is hard to decrypt if you do not know the value of k. However, this cipher does not have Property 4, since even a single plaintext/ciphertext pair (m, c) allows Eve to recover the private key k using the formula (1.10).

It is also interesting to observe that if Alice and Bob define their encryption function to be simply multiplication of integers $e_k(m) = k \cdot m$ with no reduction modulo p, then their cipher still has Properties 1 and 2, but Property 3 fails. If Eve tries to decrypt a single ciphertext $c = k \cdot m$, she still faces the (moderately) difficult task of factoring a large number. However, if she manages to acquire several ciphertexts $c_1, c_2, \dots, c_n$, then there is a good chance that

$$\begin{aligned}\gcd(c_1, c_2, \dots, c_n) &= \gcd(k \cdot m_1, k \cdot m_2, \dots, k \cdot m_n)\\ &= k \cdot \gcd(m_1, m_2, \dots, m_n)\end{aligned}$$

equals k itself or a small multiple of k. Note that it is an easy task to compute the greatest common divisor.

This observation provides our first indication of how reduction modulo p has a wonderful "mixing" effect that destroys properties such as divisibility. However, reduction is not by itself the ultimate solution. Consider the vulnerability of the cipher (1.9) to a chosen plaintext attack. As noted above, if Eve can get her hands on both a ciphertext c and its corresponding plaintext m, then she easily recovers the key by computing

$$k \equiv m^{-1} \cdot c \pmod{p}.$$

Thus even a single plaintext/ciphertext pair suffices to reveal the key, so the encryption function e_k given by (1.9) does not have Property 4 on page 38.

There are many variants of this "multiplication-modulo-p" cipher. For example, since addition is more efficient than multiplication, there is an "addition-modulo-p" cipher given by

$$e_k(m) \equiv m + k \pmod{p} \qquad \text{and} \qquad d_k(c) \equiv c - k \pmod{p},$$

which is nothing other than the shift or Caesar cipher that we studied in Section 1.1. Another variant, called an *affine cipher*, is a combination of the shift cipher and the multiplication cipher. The key for an affine cipher consists of two integers $k = (k_1, k_2)$ and encryption and decryption are defined by

$$\begin{aligned} e_k(m) &= k_1 \cdot m + k_2 \pmod{p},\\ d_k(c) &= k_1' \cdot (c - k_2) \pmod{p}, \end{aligned} \tag{1.11}$$

where k_1' is the inverse of k_1 modulo p.

The affine cipher has a further generalization called the *Hill cipher*, in which the plaintext m, the ciphertext c, and the second part of the key k_2 are replaced by column vectors consisting of n numbers modulo p. The first part of the key k_1 is taken to be an n-by-n matrix with mod p integer entries. Encryption and decryption are again given by (1.11), but now multiplication $k_1 \cdot m$ is the product of a matrix and a vector, and k_1' is the inverse matrix of k_1 modulo p. Both the affine cipher and the Hill cipher are vulnerable to chosen plaintext attacks (see Exercises 1.41. and 1.42).

Example 1.35. As noted earlier, addition is generally faster than multiplication, but there is another basic computer operation that is even faster than addition. It is called *exclusive or* and is denoted by XOR or $\oplus$. At the lowest level, XOR takes two individual bits $\beta \in \{0, 1\}$ and $\beta' \in \{0, 1\}$ and yields

$$\beta \oplus \beta' = \begin{cases} 0 & \text{if } \beta \text{ and } \beta' \text{ are the same,}\\ 1 & \text{if } \beta \text{ and } \beta' \text{ are different.}\end{cases} \tag{1.12}$$

If you think of a bit as a number that is 0 or 1, then XOR is the same as addition modulo 2. More generally, the XOR of two bit strings is the result of performing XOR on each corresponding pair of bits. For example,

$$10110 \oplus 11010 = [1 \oplus 1]\,[0 \oplus 1]\,[1 \oplus 0]\,[1 \oplus 1]\,[0 \oplus 0] = 01100.$$

Using this new operation, Alice and Bob have at their disposal yet another basic cipher defined by

$$e_k(m) = k \oplus m \qquad \text{and} \qquad d_k(c) = k \oplus c.$$

Here $\mathcal{K}$, $\mathcal{M}$, and $\mathcal{C}$ are the sets of all binary strings of length B, or equivalently, the set of all numbers between 0 and $2^B - 1$.

This cipher has the advantage of being highly efficient and completely symmetric in the sense that e_k and d_k are the same function. If k is chosen randomly and is used only once, then this cipher is known as *Vernam's one-time pad*. In Section 4.56 we show that the one-time pad is provably secure. Unfortunately, it requires a key that is as long as the plaintext, which makes it too cumbersome for most practical applications. And if k is used to encrypt more than one plaintext, then Eve may be able to exploit the fact that

$$c \oplus c' = (k \oplus m) \oplus (k \oplus m') = m \oplus m'$$

to extract information about m or m'. It's not obvious how Eve would proceed to find k, m, or m', but simply the fact that the key k can be removed so easily, revealing the potentially less random quantity $m \oplus m'$, should make a cryptographer nervous. Further, this method is vulnerable in some situations to a chosen plaintext attack; see Exercise 1.46.

1.7.5 Random bit sequences and symmetric ciphers

We have arrived, at long last, at the fundamental question regarding the creation of secure and efficient symmetric ciphers. Is it possible to use a single relatively short key k (say consisting of 160 random bits) to securely and efficiently send arbitrarily long messages? Here is one possible construction. Suppose that we could construct a function

$$R : \mathcal{K} \times \mathbb{Z} \longrightarrow \{0,1\}.$$

with the following properties:

1. For all $k \in \mathcal{K}$ and all $j \in \mathbb{Z}$, it is easy to compute $R(k,j)$.
2. Given an arbitrarily long sequence of integers $j_1, j_2, \ldots, j_n$ and given all of the values $R(k,j_1), R(k,j_2), \ldots, R(k,j_n)$, it is hard to determine k.
3. Given any list of integers $j_1, j_2, \ldots, j_n$ and given all of the values

$$R(k,j_1), R(k,j_2), \ldots, R(k,j_n),$$

   it is hard to guess the value of $R(k,j)$ with better than a 50% chance of success for any value of j not already in the list.

If we could find a function R with these three properties, then we could use it to turn an initial key k into a sequence of bits

$$R(k,1), R(k,2), R(k,3), R(k,4), \ldots, \tag{1.13}$$

and then we could use this sequence of bits as the key for a one-time pad as described in Example 1.35.

The fundamental problem with this approach is that the sequence of bits (1.13) is not truly random, since it is generated by the function R. Instead, we say that the sequence of bits (1.13) is a *pseudorandom sequence* and we call R a *pseudorandom number generator.*

Do pseudorandom number generators exist? If so, they would provide examples of the one-way functions defined by Diffie and Hellman in their groundbreaking paper [36], but despite more than a quarter century of work, no one has yet proven the existence of even a single such function. We return to this fascinating subject in Sections 2.1 and 8.2. For now, we content ourselves with a few brief remarks.

Although no one has yet conclusively proven that pseudorandom number generators exist, many candidates have been suggested, and some of these proposals have withstood the test of time. There are two basic approaches to constructing candidates for R, and these two methods provide a good illustration of the fundamental conflict in cryptography between security and efficiency.

The first approach is to repeatedly apply an ad hoc collection of mixing operations that are well suited to efficient computation and that appear to be very hard to untangle. This method is, disconcertingly, the basis for most practical symmetric ciphers, including the Data Encryption Standard (DES) and the Advanced Encryption Standard (AES), which ~~are the two systems~~ most widely used today. See Section 8.10 for a brief description of these modern symmetric ciphers.

The second approach is to construct R using a function whose efficient inversion is a well-known mathematical problem that is believed to be difficult. This approach provides a far more satisfactory theoretical underpinning for a symmetric cipher, but unfortunately, all known constructions of this sort are far less efficient than the ad hoc constructions, and hence are less attractive for real-world applications.

1.7.6 Asymmetric ciphers make a first appearance

If Alice and Bob want to exchange messages using a symmetric cipher, they must first mutually agree on a secret key k. This is fine if they have the opportunity to meet in secret or if they are able to communicate once over a secure channel. But what if they do not have this opportunity and if every communication between them is monitored by their adversary Eve? Is it possible for Alice and Bob to exchange a secret key under these conditions?

Most people's first reaction is that it is not possible, since Eve sees every piece of information that Alice and Bob exchange. It was the brilliant insight of Diffie and Hellman[17] that under certain hypotheses, it is possible. The search for efficient (and provable) solutions to this problem, which is called *public key* (or *asymmetric*) *cryptography*, forms one of the most interesting parts of mathematical cryptography and is the principal focus of this book.

We start by describing a nonmathematical way to visualize public key cryptography. Alice buys a safe with a narrow slot in the top and puts her safe in a public location. Everyone in the world is allowed to examine the safe and see that it is securely made. Bob writes his message to Alice on a piece of paper and slips it through the slot in the top of the safe. Now only a person with the key to the safe, which presumably means only Alice, can retrieve and read Bob's message. In this scenario, Alice's public key is the safe, the

[17]The history is actually somewhat more complicated than this; see our brief discussion in Section 2.1 and the references listed there for further reading.

encryption algorithm is the process of putting the message in the slot, and the decryption algorithm is the process of opening the safe with the key. Note that this setup is not far-fetched; it is used in the real world. For example, the night deposit slot at a bank has this form, although in practice the "slot" must be well protected to prevent someone from inserting a long thin pair of tongs and extracting other people's deposits!

A useful feature of our "safe-with-a-slot" cryptosystem, which it shares with actual public key cryptosystems, is that Alice needs to put only one safe in a public location, and then everyone in the world can use it repeatedly to send encrypted messages to Alice. There is no need for Alice to provide a separate safe for each of her correspondents. And there is also no need for Alice to open the safe and remove Bob's message before someone else such as Carl or Dave uses it to send Alice a message.

We are now ready to give a mathematical formulation of an asymmetric cipher. As usual, there are spaces of keys $\mathcal{K}$, plaintexts $\mathcal{M}$, and ciphertexts $\mathcal{C}$. However, an element k of the key space is really a pair of keys,

$$k = (k_{\mathsf{priv}}, k_{\mathsf{pub}}),$$

called the *private key* and the *public key*, respectively. For each public key k_{pub} there is a corresponding encryption function

$$e_{k_{\mathsf{pub}}} : \mathcal{M} \longrightarrow \mathcal{C},$$

and for each private key k_{priv} there is a corresponding decryption function

$$d_{k_{\mathsf{priv}}} : \mathcal{C} \longrightarrow \mathcal{M}.$$

These have the property that if the pair $(k_{\mathsf{priv}}, k_{\mathsf{pub}})$ is in the key space $\mathcal{K}$, then

$$d_{k_{\mathsf{priv}}}\bigl(e_{k_{\mathsf{pub}}}(m)\bigr) = m \qquad \text{for all } m \in \mathcal{M}.$$

If an asymmetric cipher is to be secure, it must be difficult for Eve to compute the decryption function $d_{k_{\mathsf{priv}}}(c)$, even if she knows the public key k_{pub}. Notice that under this assumption, Alice can send k_{pub} to Bob using an insecure communication channel, and Bob can send back the ciphertext $e_{k_{\mathsf{pub}}}(m)$, without worrying that Eve will be able to decrypt the message. To easily decrypt, it is necessary to know the private key k_{priv}, and presumably Alice is the only person with that information. The private key is sometimes called Alice's *trapdoor information*, because it provides a trapdoor (i.e., a shortcut) for computing the inverse function of $e_{k_{\mathsf{pub}}}$. The fact that the encryption and decryption keys k_{pub} and k_{priv} are different makes the cipher asymmetric, whence its moniker.

It is quite intriguing that Diffie and Hellman created this concept without finding a candidate for an actual pair of functions, although they did propose a similar method by which Alice and Bob can securely exchange a random piece of data whose value is not known initially to either one. We describe Diffie

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

Table 1.11: Simple substitution encryption table for exercise 1.3

and Hellman's key exchange method in Section 2.3 and then go on to discuss a number of asymmetric ciphers, including ElGamal (Section 2.4), RSA (Section 3.2), ECC (Section 5.4), and NTRU (Section 6.10), whose security relies on the presumed difficulty of a variety of different mathematical problems.

Exercises

Section 1.1. Simple substitution ciphers

1.1. Build a cipher wheel as illustrated in Figure 1.1, but with an inner wheel that rotates, and use it to complete the following tasks. (For your convenience, there is a cipher wheel that you can print and cut out at `www.math.brown.edu/~jhs/MathCrypto/CipherWheel.pdf`.)

(a) Encrypt the following plaintext using a rotation of 11 clockwise.

"A page of history is worth a volume of logic."

(b) Decrypt the following message, which was encrypted with a rotation of 7 clockwise.

`AOLYLHYLUVZLJYLAZILAALYAOHUAOLZLJYLALZAOHALCLYFIVKFNBLZZLZ`

(c) Decrypt the following message, which was encrypted by rotating 1 clockwise for the first letter, then 2 clockwise for the second letter, etc.

`XJHRFTNZHMZGAHIUETXZJNBWNUTRHEPOMDNBJMAUGORFAOIZOCC`

1.2. Decrypt each of the following Caesar encryptions by trying the various possible shifts until you obtain readable text.

(a) `LWKLQNWKDWLVKDOOQHYHUVHHDELOOERDUGORYHOBDVDWUHH`

(b) `UXENRBWXCUXENFQRLQJUCNABFQNWRCJUCNAJCRXWORWMB`

(c) `BGUTBMBGZTFHNLXMKTIPBMAVAXXLXTEPTRLEXTOXKHHFYHKMAXFHNLX`

1.3. For this exercise, use the simple substitution table given in Table 1.11.

(a) Encrypt the plaintext message

`The gold is hidden in the garden.`

(b) Make a decryption table, that is, make a table in which the ciphertext alphabet is in order from `A` to `Z` and the plaintext alphabet is mixed up.

(c) Use your decryption table from (b) to decrypt the following message.

`IBXLX JVXIZ SLLDE VAQLL DEVAU QLB`

1.4. Each of the following messages has been encrypted using a simple substitution cipher. Decrypt them. For your convenience, we have given you a frequency table and a list of the most common bigrams that appear in the ciphertext. (If you do not want to recopy the ciphertexts by hand, they can be downloaded or printed from the web site listed in the preface.)

(a) "A Piratical Treasure"

```
JNRZR BNIGI BJRGZ IZLQR OTDNJ GRIHT USDKR ZZWLG OIBTM NRGJN
IJTZJ LZISJ NRSBL QVRSI ORIQT QDEKJ JNRQW GLOFN IJTZX QLFQL
WBIMJ ITQXT HHTBL KUHQL JZKMM LZRNT OBIMI EURLW BLQZJ GKBJT
QDIQS LWJNR OLGRI EZJGK ZRBGS MJLDG IMNZT OIHRK MOSOT QHIJL
QBRJN IJJNT ZFIZL WIZTO MURZM RBTRZ ZKBNN LFRVR GIZFL KUHIM
MRIGJ LJNRB GKHRT QJRUU RBJLW JNRZI TULGI EZLUK JRUST QZLUK
EURFT JNLKJ JNRXR S
```

The ciphertext contains 316 letters. Here is a frequency table:

	R	J	I	L	Z	T	N	Q	B	G	K	U	M	O	S	H	W	F	E	D	X	V
Freq	33	30	27	25	24	20	19	16	15	15	13	12	12	10	9	8	7	6	5	5	3	2

The most frequent bigrams are: JN (11 times), NR (8 times), TQ (6 times), and LW, RB, RZ, and JL (5 times each).

(b) "A Botanical Code"

```
KZRNK GJKIP ZBOOB XLCRG BXFAU GJBNG RIXRU XAFGJ BXRME MNKNG
BURIX KJRXR SBUER ISATB UIBNN RTBUM NBIGK EBIGR OCUBR GLUBN
JBGRL SJGLN GJBOR ISLRS BAFFO AZBUN RFAUS AGGBI NGLXM IAZRX
RMNVL GEANG CJRUE KISRM BOOAZ GLOKW FAUKI NGRIC BEBRI NJAWB
OBNNO ATBZJ KOBRC JKIRR NGBUE BRINK XKBAF QBROA LNMRG MALUF
BBG
```

The ciphertext contains 253 letters. Here is a frequency table:

	B	R	G	N	A	I	U	K	O	J	L	X	M	F	S	E	Z	C	T	W	P	V	Q
Freq	32	28	22	20	16	16	14	13	12	11	10	10	8	8	7	7	6	5	3	2	1	1	1

The most frequent bigrams are: NG and RI (7 times each), BU (6 times), and BR (5 times).

(c) In order to make this one a bit more challenging, we have removed all occurrences of the word "the" from the plaintext.

"A Brilliant Detective"

```
GSZES GNUBE SZGUG SNKGX CSUUE QNZOQ EOVJN VXKNG XGAHS AWSZZ
BOVUE SIXCQ NQESX NGEUG AHZQA QHNSP CIPQA OIDLV JXGAK CGJCG
SASUB FVQAV CIAWN VWOVP SNSXV JGPCV NODIX GJQAE VOOXC SXXCG
OGOVA XGNVU BAVKX QZVQD LVJXQ EXCQO VKCQG AMVAX VWXCG OOBOX
VZCSO SPPSN VAXUB DVVAX QJQAJ VSUXC SXXCV OVJCS NSJXV NOJQA
MVBSZ VOOSH VSAWX QHGMV GWVSX CSXXC VBSNV ZVNVN SAWQZ ORVXJ
CVOQE JCGUW NVA
```

The ciphertext contains 313 letters. Here is a frequency table:

	V	S	X	G	A	O	Q	C	N	J	U	Z	E	W	B	P	I	H	K	D	M	L	R	F
Freq	39	29	29	22	21	21	20	20	19	13	11	11	10	8	8	6	5	5	5	4	3	2	1	1

The most frequent bigrams are: XC (10 times), NV (7 times), and CS, OV, QA, and SX (6 times each).

1.5. Suppose that you have an alphabet of 26 letters.

(a) How many possible simple substitution ciphers are there?

(b) A letter in the alphabet is said to be *fixed* if the encryption of the letter is the letter itself. How many simple substitution ciphers are there that leave:

(i) no letters fixed?

(ii) at least one letter fixed?

(iii) exactly one letter fixed?

(iv) at least two letters fixed?

(Part (b) is quite challenging! You might try doing the problem first with an alphabet of four or five letters to get an idea of what is going on.)

Chapter 2

Discrete Logarithms and Diffie–Hellman

2.1 The birth of public key cryptography

In 1976, Whitfield Diffie and Martin Hellman published their now famous paper [36] entitled "New Directions in Cryptography." In this paper they formulated the concept of a public key encryption system and made several groundbreaking contributions to this new field. A short time earlier, Ralph Merkle had independently isolated one of the fundamental problems and invented a public key construction for an undergraduate project in a computer science class at Berkeley, but this was little understood at the time. Merkle's work "Secure communication over insecure channels" appeared in 1982 [74].

However, it turns out that the concept of public key encryption was originally discovered by James Ellis while working at the British Government Communications Headquarters (GCHQ). Ellis's discoveries in 1969 were classified as secret material by the British government and were not declassified and released until 1997, after his death. It is now known that two other researchers at GCHQ, Malcolm Williamson and Clifford Cocks, discovered the Diffie–Hellman key exchange algorithm and the RSA public key encryption system, respectively, before their rediscovery and public dissemination by Diffie, Hellman, Rivest, Shamir, and Adleman. To learn more about the fascinating history of public key cryptography, see for example [35, 39, 58, 128].

The Diffie–Hellman publication was an extremely important event—it set forth the basic definitions and goals of a new field of mathematics/computer science, a field whose existence was dependent on the then emerging age of the digital computer. Indeed, their paper begins with a call to arms:

We stand today on the brink of a revolution in cryptography.

An original or breakthrough scientific idea is often called revolutionary, but in this instance, as the authors were fully aware, the term revolutionary was relevant in another sense. Prior to the publication of "New Directions...," encryption research in the United States was the domain of the National Security Agency, and all information in this area was classified. Indeed, until the mid-1990s, the United States government treated cryptographic algorithms as munitions, which meant that their export was prosecutable as a treasonable offense. Eventually, the government realized the futility of trying to prevent free and open discussion about abstract cryptographic algorithms and the dubious legality of restricting domestic use of strong cryptographic methods. However, in order to maintain some control, the government continued to restrict export of high security cryptographic algorithms if they were "machine readable." Their object, to prevent widespread global dissemination of sophisticated cryptography programs to potential enemies of the United States, was laudable,[1] but there were two difficulties that rendered the government's policy unworkable.

First, the existence of optical scanners creates a very blurry line between "machine readable" and "human text." To protest the government's policy, people wrote a three line version of the RSA algorithm in a programming language called perl and printed it on tee shirts and soda cans, thereby making these products into munitions. In principle, wearing an "RSA enabled" tee shirt on a flight from New York to Europe subjected the wearer to a large fine and a ten year jail term. Even more amusing (or frightening, depending on your viewpoint), tattoos of the RSA perl code made people's bodies into non-exportable munitions!

Second, although these and other more serious protests and legal challenges had some effect, the government's policy was ultimately rendered moot by a simple reality. Public key algorithms are quite simple, and although it requires a certain expertise to implement them in a secure fashion, the world is full of excellent mathematicians and computer scientists and engineers. Thus government restrictions on the export of "strong crypto" simply encouraged the creation of cryptographic industries in other parts of the world. The government was able to slow the adoption of strong crypto for a few years, but it is now possible for anyone to purchase for a nominal sum cryptographic software that allows completely secure communications.[2]

The first important contribution of Diffie and Hellman in [36] was the definition of a *Public Key Cryptosystem* (PKC) and its associated components—

[1]It is surely laudable to keep potential weapons out of the hands of one's enemies, but many have argued, with considerable justification, that the government also had the less benign objective of preventing other governments from using communication methods secure from United States prying.

[2]Of course, one never knows what cryptanalytic breakthroughs have been made by the scientists at the National Security Agency, since virtually all of their research is classified. The NSA is reputed to be the world's largest single employer of Ph.D.s in mathematics. However, in contrast to the situation before the 1970s, there are now far more cryptographers employed in academia and in the business world than there are in government agencies.

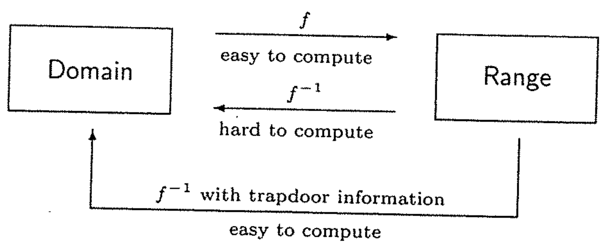

Figure 2.1: Illustration of a one-way trapdoor function

one-way functions and trapdoor information. A *one-way function* is an invertible function that is easy to compute, but whose inverse is difficult to compute. What does it mean to be "difficult to compute"? Intuitively, a function is difficult to compute if any algorithm that attempts to compute the inverse in a "reasonable" amount of time, e.g., less than the age of the universe, will almost certainly fail, where the phrase "almost certainly" must be defined probabilistically. (For a more rigorous definition of "hardness," see Section 2.6.)

Secure PKCs are built using one-way functions that have a *trapdoor*. The trapdoor is a piece of auxiliary information that allows the inverse to be easily computed. This idea is illustrated in Figure 2.1, although it must be stressed that there is a vast chasm separating the abstract idea of a one-way trapdoor function and the actual construction of such a function.

As described in Section 1.7.6, the key for a public key (or asymmetric) cryptosystem consists of two pieces, a private key k_{priv} and a public key k_{pub}, where in practice k_{pub} is computed by applying some key-creation algorithm to k_{priv}. For each public/private key pair $(k_{\mathsf{priv}}, k_{\mathsf{pub}})$ there is an encryption algorithm $e_{k_{\mathsf{pub}}}$ and a corresponding decryption algorithm $d_{k_{\mathsf{priv}}}$. The encryption algorithm $e_{k_{\mathsf{pub}}}$ corresponding to k_{pub} is public knowledge and easy to compute. Similarly, the decryption algorithm $d_{k_{\mathsf{priv}}}$ must be easily computable by someone who knows the private key k_{priv}, but it should be very difficult to compute for someone who knows only the public key k_{pub}.

One says that the private key k_{priv} is *trapdoor information* for the function $e_{k_{\mathsf{pub}}}$, because without the trapdoor information it is very hard to compute the inverse function to $e_{k_{\mathsf{pub}}}$, but with the trapdoor information it is easy to compute the inverse. Notice that in particular, the function that is used to create k_{pub} from k_{priv} must be difficult to invert, since k_{pub} is public knowledge and k_{priv} allows efficient decryption.

It may come as a surprise to learn that despite years of research, it is still not known whether one-way functions exist. In fact, a proof of the existence of one-way functions would simultaneously solve the famous $\mathcal{P} = \mathcal{NP}$

probem in complexity theory.[3] Various candidates for one-way functions have been proposed, and some of them are used by modern public key encryption algorithms. But it must be stressed that the security of these cryptosystems rests on the *assumption* that inverting the underlying function (or finding the private key from the public one) is a hard problem.

The situation is somewhat analogous to theories in physics that gain credibility over time, as they fail to be disproved and continue to explain or generate interesting phenomena. Diffie and Hellman made several suggestions in [36] for one-way functions, including knapsack problems and exponentiation mod q, but they did not produce an example of a PKC, mainly for lack of finding the right trapdoor information. They did, however, describe a public key method by which certain material could be securely shared over an insecure channel. Their method, which is now called Diffie–Hellman key exchange, is based on the assumption that the discrete logarithm problem (DLP) is difficult to solve. We discuss the DLP in Section 2.2, and then describe Diffie–Hellman key exchange in Section 2.3. In their paper, Diffie and Hellman also defined a variety of cryptanalytic attacks and introduced the important concepts of digital signatures and one-way authentication, which we discuss in Chapter 7 and Section 8.5.

With the publication of [36] in 1976, the race was on to invent a practical public key cryptosystem. Within two years, two major papers describing public key cryptosystems were published: the RSA scheme of Rivest, Shamir, and Adleman [100] and the knapsack scheme of Merkle and Hellman [75]. Of these two, only RSA has withstood the test of time, in the sense that its underlying hard problem of integer factorization is still sufficiently computationally difficult to allow RSA to operate efficiently. By way of contrast, the knapsack system of Merkle and Hellman was shown to be insecure at practical computational levels [114]. However, the cryptanalysis of knapsack systems introduces important links to hard computational problems in the theory of integer lattices that we explore in Chapter 6.

2.2 The discrete logarithm problem

The discrete logarithm problem is a mathematical problem that arises in many settings, including the mod p version described in this section and the elliptic curve version that will be studied later, in Chapter 5. The first published public key construction, due to Diffie and Hellman [36], is based on the discrete logarithm problem in a finite field $\mathbb{F}_p$, where recall that $\mathbb{F}_p$ is a field with a prime number of elements. (See Section 1.4.) For convenience, we interchangeably use the notations $\mathbb{F}_p$ and $\mathbb{Z}/p\mathbb{Z}$ for this field, and we use equality notation for elements of $\mathbb{F}_p$ and congruence notation for elements of $\mathbb{Z}/p\mathbb{Z}$ (cf. Remark 1.24).

[3]The $\mathcal{P} = \mathcal{NP}$ problem is one of the so-called Millennium Prizes, each of which has a \$1,000,000 prize attached. See Section 4.7 for more on $\mathcal{P}$ versus $\mathcal{NP}$.