Chapter 2

Perfectly Secret Encryption

In the previous chapter we presented some historical encryption schemes and showed that they can be broken easily. In this chapter, we look at the other extreme and study encryption schemes that are *provably* secure even against an adversary with unbounded computational power. Such schemes are called *perfectly secret*. We rigorously define this notion, and explore conditions under which perfect secrecy can be achieved.

The material in this chapter belongs, in some sense, more to the world of "classical" cryptography than to the world of "modern" cryptography. Besides the fact that all the material introduced here was developed before the revolution in cryptography that took place in the mid-1970s and 1980s, the constructions we study in this chapter rely only on the first and third principles outlined in Section 1.4. That is, precise mathematical definitions are used and rigorous proofs are given, but it will not be necessary to rely on any unproven computational assumptions. It is clearly advantageous to avoid such assumptions; we will see, however, that doing so has inherent limitations. Thus, in addition to serving as a good basis for understanding the principles underlying modern cryptography, the results of this chapter also justify our later adoption of all three of the aforementioned principles.

Beginning with this chapter, we will define security and analyze schemes using probabilistic experiments involving randomized algorithms. (We assume familiarity with basic probability theory. The relevant notions are reviewed in Appendix A.3.) A simple example is given by the "experiment" in which the parties who wish to communicate using a private-key encryption scheme generate a random key. Since randomness is so essential, we briefly discuss the issue of generating randomness suitable for cryptographic applications before returning to a discussion of cryptography *per se*.

Generating randomness. Throughout the book, we will assume for simplicity that parties have access to an unlimited supply of independent, unbiased (i.e., uniform) bits. Where do these random bits come from? Since classical computation is deterministic, it is not at all clear how computers can be used to generate random bits. In principle, one could generate a small number of uniform bits by hand, e.g., by flipping a fair coin. But that approach is not very convenient, nor does it scale.

Modern *random-number generation* proceeds in two steps. First, a "pool" of high-entropy data is collected. (For our purposes a formal definition of

entropy is not needed, and it suffices to think of entropy as a measure of unpredictability.) Next, this high-entropy data is processed to yield a sequence of nearly independent and unbiased bits. This second step is necessary since high-entropy data is not necessarily uniform.

For the first step, some source of unpredictable data is needed. This can come from external inputs, for example, delays between network events, hard-disk access times, keystrokes or mouse movements made by the user, and so on. More sophisticated approaches—which, by design, incorporate random-number generation more tightly into the system at the hardware level—can also be used. These rely on physical phenomena such as thermal/shot noise or radioactive decay; for example, certain Intel processors use thermal noise to generate high-entropy data on-chip. Hardware random-number generators of this sort generally produce high-entropy data at a faster rate than techniques relying on external sources.

The processing needed to "smooth" the high-entropy data to obtain (nearly) independent and uniform bits is non-trivial, and is discussed briefly in Section 6.6.4. Here, we consider a simple example to give an idea of what can be done. Imagine that our high-entropy pool contains a sequence of *biased* bits, where 1 occurs with probability p and 0 occurs with probability $1-p$. (We do assume, however, that the bits are all *independent*. In practice this assumption is typically not valid and so more-complex processing must be done.) Thousands of such bits have lots of entropy, but are not close to uniform. We can obtain a uniform sequence of bits by taking the original bits in pairs: if we see a 1 followed by a 0 then we output 0, and if we see a 0 followed by a 1 then we output 1. (If we see two 0s or two 1s in a row we output nothing, and simply move on to the next pair.) The probability that any pair results in a 0 is $p \cdot (1-p)$, which is exactly equal to the probability that any pair results in a 1. (Note that we do not even need to know the value of p!) We thus obtain a uniformly distributed output from our initial high-entropy pool.

Care must be taken in how random bits are produced, and using poor random-number generators can often leave a good cryptosystem vulnerable to attack. One should use a random-number generator that is *designed for cryptographic use*, rather than a "general-purpose" random-number generator that is generally not suitable for cryptographic applications. In particular, the `rand()` function in the C stdlib.h library is *not* cryptographically secure, and using it in cryptographic settings can have disastrous consequences.

2.1 Definitions

We begin by recalling and expanding upon the syntax of encryption, as introduced in the previous chapter. An encryption scheme is defined by three

algorithms Gen, Enc, and Dec, as well as a specification of a *message space* $\mathcal{M}$ with $|\mathcal{M}| > 1$.[1] The key-generation algorithm Gen is a probabilistic algorithm that outputs a key k chosen according to some distribution. We denote by $\mathcal{K}$ the (finite) *key space*, i.e., the set of all possible keys that can be output by Gen. The encryption algorithm Enc takes as input a key $k \in \mathcal{K}$ and a message $m \in \mathcal{M}$, and outputs a ciphertext c. We now explicitly allow the encryption algorithm to be probabilistic (so $\mathsf{Enc}_k(m)$ might output a different ciphertext when run multiple times), and we write $c \leftarrow \mathsf{Enc}_k(m)$ to denote the possibly probabilistic process by which message m is encrypted using key k to give ciphertext c. (Looking ahead, we also sometimes use the notation $x \leftarrow S$ to denote uniform selection of x from a set S. In case Enc is deterministic, we may emphasize this by writing $c := \mathsf{Enc}_k(m)$.) We let $\mathcal{C}$ denote the set of all possible ciphertexts that can be output by $\mathsf{Enc}_k(m)$, for all possible choices of $k \in \mathcal{K}$ and $m \in \mathcal{M}$ (and for all random choices of Enc in case it is randomized). The decryption algorithm Dec takes as input a key $k \in \mathcal{K}$ and a ciphertext $c \in \mathcal{C}$ and outputs a message $m \in \mathcal{M}$. We assume *perfect correctness*, meaning that for all $k \in \mathcal{K}$, $m \in \mathcal{M}$, and any ciphertext c output by $\mathsf{Enc}_k(m)$, it holds that $\mathsf{Dec}_k(c) = m$ with probability 1. Perfect correctness implies that we may assume Dec is deterministic without loss of generality, since $\mathsf{Dec}_k(c)$ must give the same output every time it is run. We will thus write $m := \mathsf{Dec}_k(c)$ to denote the (deterministic) process of decrypting ciphertext c using key k to yield the message m.

In the definitions and theorems below, we refer to probability distributions over $\mathcal{K}$, $\mathcal{M}$, and $\mathcal{C}$. The distribution over $\mathcal{K}$ is the one defined by running Gen and taking the output. (It is almost always the case that Gen chooses a key uniformly from $\mathcal{K}$ and, in fact, we may assume this without loss of generality; see Exercise 2.1.) We let K be the random variable denoting the value of the key output by Gen; thus, for any $k \in \mathcal{K}$, $\Pr[K = k]$ denotes the probability that the key output by Gen is equal to k. Similarly, we let M be the random variable denoting the message being encrypted, so $\Pr[M = m]$ denotes the probability that the message takes on the value $m \in \mathcal{M}$. The probability distribution of the message is not determined by the encryption scheme itself, but instead reflects the likelihood of different messages being sent by the parties using the scheme, as well as an adversary's uncertainty about what will be sent. As an example, an adversary may know that the message will either be `attack today` or `don't attack`. The adversary may even know (by other means) that with probability 0.7 the message will be a command to attack and with probability 0.3 the message will be a command not to attack. In this case, we have $\Pr[M = \texttt{attack today}] = 0.7$ and $\Pr[M = \texttt{don't attack}] = 0.3$.

K and M are required to be independent, i.e., what is being communicated by the parties must be independent of the key they share. This makes sense,

[1] If $|\mathcal{M}| = 1$ there is only one message and no point in communicating, let alone encrypting.

among other reasons, because the distribution over $\mathcal{K}$ is determined by the encryption scheme itself (since it is defined by Gen), while the distribution over $\mathcal{M}$ depends on the context in which the encryption scheme is being used.

Fixing an encryption scheme and a distribution over $\mathcal{M}$ determines a distribution over the space of ciphertexts $\mathcal{C}$ given by choosing a key $k \in \mathcal{K}$ (according to Gen) and a message $m \in \mathcal{M}$ (according to the given distribution), and then computing the ciphertext $c \leftarrow \mathsf{Enc}_k(m)$. We let C be the random variable denoting the resulting ciphertext and so, for $c \in \mathcal{C}$, write $\Pr[C = c]$ to denote the probability that the ciphertext is equal to the fixed value c.

Example 2.1

We work through a simple example for the shift cipher (cf. Section 1.3). Here, by definition, we have $\mathcal{K} = \{0, \ldots, 25\}$ with $\Pr[K = k] = 1/26$ for each $k \in \mathcal{K}$.

Say we are given the following distribution over $\mathcal{M}$:

$$\Pr[M = \mathtt{a}] = 0.7 \quad \text{and} \quad \Pr[M = \mathtt{z}] = 0.3.$$

What is the probability that the ciphertext is $\mathtt{B}$? There are only two ways this can occur: either $M = \mathtt{a}$ and $K = 1$, or $M = \mathtt{z}$ and $K = 2$. By independence of M and K, we have

$$\begin{aligned}\Pr[M = \mathtt{a} \wedge K = 1] &= \Pr[M = \mathtt{a}] \cdot \Pr[K = 1] \\ &= 0.7 \cdot \left(\frac{1}{26}\right).\end{aligned}$$

Similarly, $\Pr[M = \mathtt{z} \wedge K = 2] = 0.3 \cdot \left(\frac{1}{26}\right)$. Therefore,

$$\begin{aligned}\Pr[C = \mathtt{B}] &= \Pr[M = \mathtt{a} \wedge K = 1] + \Pr[M = \mathtt{z} \wedge K = 2] \\ &= 0.7 \cdot \left(\frac{1}{26}\right) + 0.3 \cdot \left(\frac{1}{26}\right) \;=\; 1/26.\end{aligned}$$

We can calculate conditional probabilities as well. For example, what is the probability that the message $\mathtt{a}$ was encrypted, given that we observe ciphertext $\mathtt{B}$? Using Bayes' Theorem (Theorem A.8) we have

$$\begin{aligned}\Pr[M = \mathtt{a} \mid C = \mathtt{B}] &= \frac{\Pr[C = \mathtt{B} \mid M = \mathtt{a}] \cdot \Pr[M = \mathtt{a}]}{\Pr[C = \mathtt{B}]} \\ &= \frac{\Pr[C = \mathtt{B} \mid M = \mathtt{a}] \cdot 0.7}{1/26}.\end{aligned}$$

Note that $\Pr[C = \mathtt{B} \mid M = \mathtt{a}] = 1/26$, since if $M = \mathtt{a}$ then the only way $C = \mathtt{B}$ can occur is if $K = 1$ (which occurs with probability $1/26$). We conclude that $\Pr[M = \mathtt{a} \mid C = \mathtt{B}] = 0.7$. ♢

Example 2.2

Consider the shift cipher again, but with the following distribution over $\mathcal{M}$:

$$\Pr[M = \mathtt{kim}] = 0.5, \; \Pr[M = \mathtt{ann}] = 0.2, \; \Pr[M = \mathtt{boo}] = 0.3.$$

What is the probability that $C = \texttt{DQQ}$? The only way this ciphertext can occur is if $M = \texttt{ann}$ and $K = 3$, or $M = \texttt{boo}$ and $K = 2$, which happens with probability $0.2 \cdot 1/26 + 0.3 \cdot 1/26 = 1/52$.

We can also compute the probability that `ann` was encrypted, conditioned on observing the ciphertext `DQQ`? A calculation as above using Bayes' Theorem gives $\Pr[M = \texttt{ann} \mid C = \texttt{DQQ}] = 0.4$. ♢

Perfect secrecy. We are now ready to define the notion of *perfect secrecy*. We imagine an adversary who knows the probability distribution of M; that is, the adversary knows the likelihood that different messages will be sent. The adversary also knows the encryption scheme being used. The only thing unknown to the adversary is the key shared by the parties. A message is chosen by one of the honest parties and encrypted, and the resulting ciphertext is transmitted to the other party. The adversary can *eavesdrop* on the parties' communication, and thus observe this ciphertext. (That is, this is a ciphertext-only attack, where the attacker sees only a single ciphertext.) For a scheme to be perfectly secret, observing this ciphertext should have *no effect* on the adversary's knowledge regarding the actual message that was sent; in other words, the *a posteriori* probability that some message $m \in \mathcal{M}$ was sent, conditioned on the ciphertext that was observed, should be no different from the *a priori* probability that m would be sent. This means that the ciphertext reveals nothing about the underlying plaintext, and the adversary learns absolutely nothing about the plaintext that was encrypted. Formally:

DEFINITION 2.3 *An encryption scheme* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *with message space* $\mathcal{M}$ *is* perfectly secret *if for every probability distribution for* M*, every message* $m \in \mathcal{M}$*, and every ciphertext* $c \in \mathcal{C}$ *for which* $\Pr[C = c] > 0$*:*

$$\Pr[M = m \mid C = c] = \Pr[M = m].$$

(The requirement that $\Pr[C = c] > 0$ is a technical one needed to prevent conditioning on a zero-probability event.)

Example 2.4

We show that the shift cipher is *not* perfectly secret when used with the message space $\mathcal{M}$ consisting of all two-character plaintexts. To do so, we work with Definition 2.3, and show a probability distribution over $\mathcal{M}$ for which, for some message m and ciphertext c,

$$\Pr[M = m \mid C = c] \neq \Pr[M = m].$$

Many such distributions are possible, but we pick a simple one: say the message is either `aa` or `ab`, each with half probability. Set $m = \texttt{ab}$ and $c = \texttt{XX}$. Then clearly $\Pr[M = \texttt{ab} \mid C = \texttt{XX}] = 0$, as there is no way that `XX` can ever result from the encryption of `ab`. But $\Pr[M = \texttt{ab}] = 1/2$. ♢

We now give an equivalent formulation of perfect secrecy. This formulation defines perfect secrecy by requiring that the distribution of the ciphertext does not depend on the plaintext, i.e., for any two messages $m, m' \in \mathcal{M}$ the distribution of the ciphertext when m is encrypted should be identical to the distribution of the ciphertext when m' is encrypted. That is, for every $m, m' \in \mathcal{M}$, and every $c \in \mathcal{C}$, we have

$$\Pr[\mathsf{Enc}_K(m) = c] = \Pr[\mathsf{Enc}_K(m') = c] \tag{2.1}$$

(where the probabilities are over choice of K and any randomness of Enc). Note that the above probabilities depend *only* on the encryption scheme, and make no reference to any underlying distribution on $\mathcal{M}$. The above condition implies that a ciphertext contains no information about the plaintext, and that it is impossible to distinguish an encryption of m from an encryption of m', since the distributions of the ciphertext are the same in each case.

LEMMA 2.5 *An encryption scheme* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *with message space* $\mathcal{M}$ *is perfectly secret if and only if Equation (2.1) holds for every* $m, m' \in \mathcal{M}$ *and every* $c \in \mathcal{C}$.

PROOF The proof is straightforward, but we go through it in detail. The key observation is that for any scheme, any distribution on $\mathcal{M}$, any $m \in \mathcal{M}$ for which $\Pr[M = m] > 0$, and any $c \in \mathcal{C}$, we have

$$\begin{aligned} \Pr[C = c \mid M = m] &= \Pr[\mathsf{Enc}_K(M) = c \mid M = m] \\ &= \Pr[\mathsf{Enc}_K(m) = c \mid M = m] \\ &= \Pr[\mathsf{Enc}_K(m) = c], \end{aligned} \tag{2.2}$$

where the first equality is by definition of the random variable C, the second is because we are conditioning on the event that $M = m$, and the third is because K is independent of M. We also use the fact that for any $c \in \mathcal{C}$ with $\Pr[C = c] > 0$, we have

$$\Pr[M = m \mid C = c] \cdot \Pr[C = c] = \Pr[C = c \mid M = m] \cdot \Pr[M = m]. \tag{2.3}$$

Take the uniform distribution over $\mathcal{M}$. If the scheme is perfectly secret then $\Pr[M = m \mid C = c] = \Pr[M = m]$, and so Equation (2.3) implies that $\Pr[C = c \mid M = m] = \Pr[C = c]$. Since m and c were arbitrary, this shows that for every $m, m' \in \mathcal{M}$ and every $c \in \mathcal{C}$,

$$\begin{aligned} \Pr[\mathsf{Enc}_K(m) = c] &= \Pr[C = c \mid M = m] \\ &= \Pr[C = c] \\ &= \Pr[C = c \mid M = m'] = \Pr[\mathsf{Enc}_K(m') = c] \end{aligned}$$

(using Equation (2.2)), proving one direction of the lemma.

Conversely, say Equation (2.1) holds for every $m, m' \in \mathcal{M}$ and every $c \in \mathcal{C}$. Fix some distribution over $\mathcal{M}$, a message $m \in \mathcal{M}$, and a ciphertext $c \in \mathcal{C}$ with $\Pr[C = c] > 0$. If $\Pr[M = m] = 0$ then we trivially have

$$\Pr[M = m \mid C = c] = \Pr[M = m] = 0.$$

So, assume $\Pr[M = m] > 0$. For $c \in \mathcal{C}$, define $p_c \stackrel{\text{def}}{=} \Pr[\mathsf{Enc}_K(m) = c]$. Equations (2.1) and (2.2) imply that $\Pr[C = c \mid M = m'] = p_c$ for every $m' \in \mathcal{M}$. So,

$$\begin{aligned}\Pr[C = c] &= \sum_{m' \in \mathcal{M}} \Pr[C = c \mid M = m'] \cdot \Pr[M = m'] \\ &= \sum_{m' \in \mathcal{M}} p_c \cdot \Pr[M = m'] = p_c = \Pr[C = c \mid M = m],\end{aligned}$$

where the sum is over m' with $\Pr[M = m'] > 0$. Equation (2.3) implies that $\Pr[M = m \mid C = c] = \Pr[M = m]$, so the scheme is perfectly secret. ■

Perfect (adversarial) indistinguishability. We conclude this section by presenting another equivalent definition of perfect secrecy. This definition is based on an *experiment* involving an adversary passively observing a ciphertext and then trying to guess which of two possible messages was encrypted. We introduce this notion since it will serve as our starting point for defining computational security in the next chapter; throughout the rest of the book we will often use experiments like this one to define security.

In the present context, we consider the following experiment: An adversary $\mathcal{A}$ first specifies two arbitrary messages $m_0, m_1 \in \mathcal{M}$. Next, a key k is generated using Gen. Then, one of the two messages specified by $\mathcal{A}$ is chosen (each with probability 1/2) and encrypted using k; the resulting ciphertext is given to $\mathcal{A}$. Finally, $\mathcal{A}$ outputs a "guess" as to which of the two messages was encrypted; $\mathcal{A}$ *succeeds* if it guesses correctly. An encryption scheme is *perfectly indistinguishable* if *no* adversary $\mathcal{A}$ can succeed with probability better than 1/2. (Note that, for any encryption scheme, $\mathcal{A}$ can succeed with probability 1/2 by outputting a uniform guess; the requirement is simply that no attacker can do any better than this.) We stress that no limitations are placed on the computational power of $\mathcal{A}$.

Formally, let $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be an encryption scheme with message space $\mathcal{M}$. Let $\mathcal{A}$ be an adversary, which is formally just a (stateful) algorithm that we may assume is deterministic without loss of generality. We define an experiment $\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},\Pi}$, based, on $\mathcal{A}$ and Π, as follows:

The adversarial indistinguishability experiment $\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},\Pi}$:

1. *The adversary $\mathcal{A}$ outputs a pair of messages $m_0, m_1 \in \mathcal{M}$.*
2. *A key k is generated using Gen, and a uniform bit $b \in \{0,1\}$ is chosen. Ciphertext $c \leftarrow \mathsf{Enc}_k(m_b)$ is computed and given to $\mathcal{A}$. We refer to c as the* challenge ciphertext.

3. *$\mathcal{A}$ outputs a bit b'.*
4. *The output of the experiment is defined to be 1 if $b' = b$, and 0 otherwise. We write $\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},\Pi} = 1$ if the output of the experiment is 1 and in this case we say that $\mathcal{A}$ succeeds.*

As noted earlier, it is trivial for $\mathcal{A}$ to succeed with probability 1/2 by outputting a random guess. Perfect indistinguishability requires that it is impossible for any $\mathcal{A}$ to do better.

DEFINITION 2.6 *Encryption scheme $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ with message space $\mathcal{M}$ is* perfectly indistinguishable *if for every $\mathcal{A}$ it holds that*

$$\Pr\left[\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},\Pi} = 1\right] = \frac{1}{2}.$$

The following lemma states that Definition 2.6 is equivalent to Definition 2.3. We leave the proof of the lemma as Exercise 2.6.

LEMMA 2.7 *Encryption scheme Π is perfectly secret if and only if it is perfectly indistinguishable.*

Example 2.8
We show that the Vigenère cipher is *not* perfectly indistinguishable, at least for certain parameters. Concretely, let Π denote the Vigenère cipher for the message space of two-character strings, and where the period is chosen uniformly in $\{1, 2\}$. To show that Π is not perfectly indistinguishable, we exhibit an adversary $\mathcal{A}$ for which $\Pr\left[\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},\Pi} = 1\right] > \frac{1}{2}$.

Adversary $\mathcal{A}$ does:

1. Output $m_0 = \mathtt{aa}$ and $m_1 = \mathtt{ab}$.
2. Upon receiving the challenge ciphertext $c = c_1c_2$, do the following: if $c_1 = c_2$ output 0; else output 1.

Computation of $\Pr\left[\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},\Pi} = 1\right]$ is tedious but straightforward.

$$\begin{aligned}
&\Pr\left[\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},\Pi} = 1\right] \\
&= \frac{1}{2} \cdot \Pr\left[\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},\Pi} = 1 \mid b = 0\right] + \frac{1}{2} \cdot \Pr\left[\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},\Pi} = 1 \mid b = 1\right] \\
&= \frac{1}{2} \cdot \Pr[\mathcal{A} \text{ outputs } 0 \mid b = 0] + \frac{1}{2} \cdot \Pr[\mathcal{A} \text{ outputs } 1 \mid b = 1], \qquad (2.4)
\end{aligned}$$

where b is the uniform bit determining which message gets encrypted. As defined, $\mathcal{A}$ outputs 0 if and only if the two characters of the ciphertext $c = c_1c_2$ are equal. When $b = 0$ (so $m_0 = \mathtt{aa}$ is encrypted) then $c_1 = c_2$ if either (1) a

key of period 1 is chosen, or (2) a key of period 2 is chosen and both characters of the key are equal. The former occurs with probability $\frac{1}{2}$, and the latter occurs with probability $\frac{1}{2} \cdot \frac{1}{26}$. So

$$\Pr[\mathcal{A} \text{ outputs } 0 \mid b = 0] = \frac{1}{2} + \frac{1}{2} \cdot \frac{1}{26} \approx 0.52.$$

When $b = 1$ then $c_1 = c_2$ only if a key of period 2 is chosen and the first character of the key is one more than the second character of the key, which happens with probability $\frac{1}{2} \cdot \frac{1}{26}$. So

$$\Pr[\mathcal{A} \text{ outputs } 1 \mid b = 1] = 1 - \Pr[\mathcal{A} \text{ outputs } 0 \mid b = 1] = 1 - \frac{1}{2} \cdot \frac{1}{26} \approx 0.98.$$

Plugging into Equation (2.4) then gives

$$\Pr\left[\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},\Pi} = 1\right] = \frac{1}{2} \cdot \left(\frac{1}{2} + \frac{1}{2} \cdot \frac{1}{26} + 1 - \frac{1}{2} \cdot \frac{1}{26}\right) = 0.75 > \frac{1}{2},$$

and the scheme is not perfectly indistinguishable. ♢

2.2 The One-Time Pad

In 1917, Vernam patented a perfectly secret encryption scheme now called the *one-time pad*. At the time Vernam proposed the scheme, there was no proof that it was perfectly secret; in fact, the notion of perfect secrecy was not yet defined. Approximately 25 years later, however, Shannon introduced the definition of perfect secrecy and demonstrated that the one-time pad satisfied that definition.

In describing the scheme we let $a \oplus b$ denote the *bitwise exclusive-or* (XOR) of two equal-length binary strings a and b. (i.e., if $a = a_1 \cdots a_\ell$ and $b = b_1 \cdots b_\ell$ are ℓ-bit strings, then $a \oplus b$ is the ℓ-bit string given by $(a_1 \oplus b_1) \cdots (a_\ell \oplus b_\ell)$.) In the one-time pad encryption scheme the key is a uniform string of the same length as the message, and the ciphertext is computed by simply XORing the key and the message; a formal definition is given as Construction 2.9. Before discussing security, we first verify correctness: For every key k and every message m it holds that $\mathsf{Dec}_k(\mathsf{Enc}_k(m)) = k \oplus k \oplus m = m$, and so the one-time pad constitutes a valid encryption scheme.

One can easily prove perfect secrecy of the one-time pad using Lemma 2.5 because the ciphertext is uniformly distributed regardless of what message is encrypted. We give a direct proof based on the original definition.

THEOREM 2.10 *The one-time pad encryption scheme is perfectly secret.*

CONSTRUCTION 2.9

Fix an integer $\ell > 0$. The message space $\mathcal{M}$, key space $\mathcal{K}$, and ciphertext space $\mathcal{C}$ are all equal to $\{0,1\}^\ell$ (the set of all binary strings of length ℓ).

- Gen: the key-generation algorithm chooses a key from $\mathcal{K} = \{0,1\}^\ell$ according to the uniform distribution (i.e., each of the 2^ℓ strings in the space is chosen as the key with probability exactly $2^{-\ell}$).
- Enc: given a key $k \in \{0,1\}^\ell$ and a message $m \in \{0,1\}^\ell$, the encryption algorithm outputs the ciphertext $c := k \oplus m$.
- Dec: given a key $k \in \{0,1\}^\ell$ and a ciphertext $c \in \{0,1\}^\ell$, the decryption algorithm outputs the message $m := k \oplus c$.

The one-time pad encryption scheme.

PROOF We first compute $\Pr[C = c \mid M = m]$ for arbitrary $c \in \mathcal{C}$ and $m \in \mathcal{M}$ with $\Pr[M = m] > 0$. For the one-time pad, we have

$$\begin{aligned}\Pr[C = c \mid M = m] &= \Pr[K \oplus m = c \mid M = m] \\ &= \Pr[K = m \oplus c \mid M = m] \\ &= 2^{-\ell},\end{aligned}$$

where the first equality is by definition of the scheme and the fact that we condition on the event $M = m$, and the final equality holds because the key K is a uniform ℓ-bit string that is independent of M. Fix any distribution over $\mathcal{M}$. Using the above result, we see that for any $c \in \mathcal{C}$ we have

$$\begin{aligned}\Pr[C = c] &= \sum_{m \in \mathcal{M}} \Pr[C = c \mid M = m] \cdot \Pr[M = m] \\ &= 2^{-\ell} \cdot \sum_{m \in \mathcal{M}} \Pr[M = m] \\ &= 2^{-\ell},\end{aligned}$$

where the sum is over $m \in \mathcal{M}$ with $\Pr[M = m] \neq 0$. Bayes' Theorem gives:

$$\begin{aligned}\Pr[M = m \mid C = c] &= \frac{\Pr[C = c \mid M = m] \cdot \Pr[M = m]}{\Pr[C = c]} \\ &= \frac{2^{-\ell} \cdot \Pr[M = m]}{2^{-\ell}} \\ &= \Pr[M = m].\end{aligned}$$

We conclude that the one-time pad is perfectly secret. ■

The one-time pad was used by several national-intelligence agencies in the mid-20th century to encrypt sensitive traffic. Perhaps most famously, the "red phone" linking the White House and the Kremlin during the Cold War

was protected using one-time pad encryption, where the governments of the US and the USSR would exchange extremely long keys using trusted couriers carrying briefcases of paper on which random characters were written.

Notwithstanding the above, one-time pad encryption is rarely used nowadays because it has a number of drawbacks. Most prominent is that *the key is as long as the message*.[2] This limits the usefulness of the scheme for sending very long messages (as it may be difficult to securely share and store a very long key), and is problematic when the parties cannot predict in advance (an upper bound on) how long the message will be.

Moreover, the one-time pad—as the name indicates—*is only secure if used once* (with a given key). Although we did not yet define a notion of secrecy when multiple messages are encrypted, it is easy to see that encrypting more than one message with the same key leaks a lot of information. In particular, say two messages m, m' are encrypted using the same (unknown) key k. An adversary who obtains $c = m \oplus k$ and $c' = m' \oplus k$ can compute

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

and thus learn the XOR of the two messages or, equivalently, exactly where the two messages differ. This attack extends to more than two messages as well, where it enables the attacker to learn the XOR of all pairs of messages. While this may not seem very significant, it is enough to rule out any claims of perfect secrecy for encrypting more than one message using the same key. Moreover, if the messages correspond to natural-language text, then given the XOR of sufficiently many pairs of messages—or even two sufficiently long messages—it is possible to perform frequency analysis (as in the previous chapter, though more complex) and recover the messages themselves. (See Exercise 2.16 for an example.) An interesting historical example of this is given by the *VENONA project*, as part of which the US and UK were able to decrypt ciphertexts sent by the Soviet Union that were mistakenly encrypted with repeated portions of a one-time pad over several decades.

2.3 Limitations of Perfect Secrecy

We ended the previous section by noting some drawbacks of the one-time pad encryption scheme. Here, we show that these drawbacks are not specific to that scheme, but are instead *inherent* limitations of perfect secrecy. Specifically, we prove that *any* perfectly secret encryption scheme must have a key space that is at least as large as the message space. If all keys are the same

[2] This does not make the one-time pad useless, since it may be easier for two parties to share a key at some point in time before the message to be communicated is known.

length, and the message space consists of all strings of some fixed length, this implies that the key is at least as long as the message. In particular, the key length of the one-time pad is optimal. (The other limitation—namely, that a key can be used only once—is also inherent; see Exercise 2.19.)

THEOREM 2.11 *If* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is a perfectly secret encryption scheme with message space* $\mathcal{M}$ *and key space* $\mathcal{K}$*, then* $|\mathcal{K}| \geq |\mathcal{M}|$*.*

PROOF We show that if $|\mathcal{K}| < |\mathcal{M}|$ then the scheme cannot be perfectly secret. Assume $|\mathcal{K}| < |\mathcal{M}|$. Consider the uniform distribution over $\mathcal{M}$ and let $c \in \mathcal{C}$ be a ciphertext that occurs with nonzero probability. Let $\mathcal{M}(c)$ be the set of all possible messages that are possible decryptions of c; that is

$$\mathcal{M}(c) \stackrel{\text{def}}{=} \{m \mid m = \mathsf{Dec}_k(c) \text{ for some } k \in \mathcal{K}\}.$$

Clearly $|\mathcal{M}(c)| \leq |\mathcal{K}|$. (Recall that we may assume Dec is deterministic.) If $|\mathcal{K}| < |\mathcal{M}|$, there is some $m' \in \mathcal{M}$ such that $m' \notin \mathcal{M}(c)$. But then

$$\Pr[M = m' \mid C = c] = 0 \neq \Pr[M = m'],$$

and so the scheme is not perfectly secret. ■

Perfect secrecy with shorter keys? The above theorem shows an inherent limitation of schemes that achieve perfect secrecy. Even so, individuals occasionally claim they have developed a radically new encryption scheme that is "unbreakable" and achieves the security of the one-time pad without using keys as long as what is being encrypted. The above proof demonstrates that such claims *cannot* be true; anyone making such claims either knows very little about cryptography or is blatantly lying.

2.4 *Shannon's Theorem

In his work on perfect secrecy, Shannon also provided a characterization of perfectly secret encryption schemes. This characterization says that, under certain conditions, the key-generation algorithm Gen must choose the key *uniformly* from the set of all possible keys (as in the one-time pad); moreover, for every message m and ciphertext c there is a *unique* key mapping m to c (again, as in the one-time pad). Beyond being interesting in its own right, this theorem is a useful tool for proving (or disproving) perfect secrecy of schemes. We discuss this further after the proof.

The theorem as stated here assumes $|\mathcal{M}| = |\mathcal{K}| = |\mathcal{C}|$, meaning that the sets of plaintexts, keys, and ciphertexts all have the same size. We have already

seen that for perfect secrecy we must have $|\mathcal{K}| \geq |\mathcal{M}|$. It is easy to see that correct decryption requires $|\mathcal{C}| \geq |\mathcal{M}|$. Therefore, in some sense, perfectly secret encryption schemes with $|\mathcal{M}| = |\mathcal{K}| = |\mathcal{C}|$ are "optimal."

THEOREM 2.12 (Shannon's theorem) *Let* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *be an encryption scheme with message space* $\mathcal{M}$*, for which* $|\mathcal{M}| = |\mathcal{K}| = |\mathcal{C}|$*. The scheme is perfectly secret if and only if:*

1. *Every key* $k \in \mathcal{K}$ *is chosen with (equal) probability* $1/|\mathcal{K}|$ *by* Gen*.*
2. *For every* $m \in \mathcal{M}$ *and every* $c \in \mathcal{C}$*, there is a unique key* $k \in \mathcal{K}$ *such that* $\mathsf{Enc}_k(m)$ *outputs* c*.*

PROOF The intuition behind the proof is as follows. To see that the stated conditions imply perfect secrecy, note that condition 2 means that any ciphertext c could be the result of encrypting any possible plaintext m, because there is some key k mapping m to c. Since there is a *unique* such key, and each key is chosen with equal probability, perfect secrecy follows as for the one-time pad. For the other direction, perfect secrecy immediately implies that for every m and c there is at least one key mapping m to c. The fact that $|\mathcal{M}| = |\mathcal{K}| = |\mathcal{C}|$ means, moreover, that for every m and c there is exactly *one* such key. Given this, each key must be chosen with equal probability or else perfect secrecy would fail to hold. A formal proof follows.

We assume for simplicity that Enc is deterministic. (One can show that this is without loss of generality here.) We first prove that if the encryption scheme satisfies conditions 1 and 2, then it is perfectly secret. The proof is essentially the same as the proof of perfect secrecy for the one-time pad, so we will be relatively brief. Fix arbitrary $c \in \mathcal{C}$ and $m \in \mathcal{M}$. Let k be the unique key, guaranteed by condition 2, for which $\mathsf{Enc}_k(m) = c$. Then,

$$\Pr[\mathsf{Enc}_K(m) = c] = \Pr[K = k] = 1/|\mathcal{K}|\,,$$

where the final equality holds by condition 1. Since this holds for arbitrary m and c, Lemma 2.5 implies that the scheme is perfectly secret.

For the second direction, assume the encryption scheme is perfectly secret; we show that conditions 1 and 2 hold. Fix arbitrary $c \in \mathcal{C}$. There must be some message m^* for which $\Pr[\mathsf{Enc}_K(m^*) = c] \neq 0$. Lemma 2.5 then implies that $\Pr[\mathsf{Enc}_K(m) = c] \neq 0$ for every $m \in \mathcal{M}$. In other words, if we let $\mathcal{M} = \{m_1, m_2, \ldots\}$, then for each $m_i \in \mathcal{M}$ we have a nonempty set of keys $\mathcal{K}_i \subset \mathcal{K}$ such that $\mathsf{Enc}_k(m_i) = c$ if and only if $k \in \mathcal{K}_i$. Moreover, when $i \neq j$ then $\mathcal{K}_i$ and $\mathcal{K}_j$ must be disjoint or else correctness fails to hold. Since $|\mathcal{K}| = |\mathcal{M}|$, we see that each $\mathcal{K}_i$ contains only a single key k_i, as required by condition 2. Now, Lemma 2.5 shows that for any $m_i, m_j \in \mathcal{M}$ we have

$$\Pr[K = k_i] = \Pr[\mathsf{Enc}_K(m_i) = c] = \Pr[\mathsf{Enc}_K(m_j) = c] = \Pr[K = k_j].$$

Since this holds for all $1 \leq i, j \leq |\mathcal{M}| = |\mathcal{K}|$, and $k_i \neq k_j$ for $i \neq j$, this means each key is chosen with probability $1/|\mathcal{K}|$, as required by condition 1. ■

Shannon's theorem is useful for deciding whether a given scheme is perfectly secret. Condition 1 is easy to check, and condition 2 can be demonstrated (or contradicted) without having to compute any probabilities (in contrast to working with Definition 2.3 directly). As an example, perfect secrecy of the one-time pad is trivial to prove using Shannon's theorem. We stress, however, that the theorem only applies when $|\mathcal{M}| = |\mathcal{K}| = |\mathcal{C}|$.

References and Additional Reading

The one-time pad is popularly credited to Vernam [200], who filed a patent on it, but recent historical research [28] shows that it was invented some 35 years earlier. Analysis of the one-time pad had to await the groundbreaking work of Shannon [177], who introduced the notion of perfect secrecy.

In this chapter we studied perfectly secret *encryption*. Some other cryptographic problems can also be solved with "perfect" security. A notable example is the problem of message authentication where the aim is to prevent an adversary from (undetectably) modifying a message sent from one party to another. We study this problem in depth in Chapter 4, discussing "perfectly secure" message authentication in Section 4.6.

Exercises

2.1 Prove that, by redefining the key space, we may assume that the key-generation algorithm Gen chooses a uniform key from the key space, without changing $\Pr[C = c \mid M = m]$ for any m, c.

> **Hint:** Define the key space to be the set of all possible random bits used by the randomized algorithm Gen.

2.2 Prove that, by redefining the key space as well as the encryption algorithm, we may assume that encryption is deterministic without changing $\Pr[C = c \mid M = m]$ for any m, c.

2.3 Prove or refute: An encryption scheme with message space $\mathcal{M}$ is perfectly secret if and only if for every probability distribution on $\mathcal{M}$ and every $c_0, c_1 \in \mathcal{C}$ we have $\Pr[C = c_0] = \Pr[C = c_1]$.