Chapter 2

A Look Back

In this chapter we take a quick look at some classical encryption techniques, illustrating their weakness and using these examples to initiate questions about how to define privacy. We then discuss Shannon's notion of perfect security.

2.1 Substitution ciphers

One of the earliest approaches to symmetric encryption is what is called a substitution cipher. Say the plaintext is English text. We can view this as a sequence of symbols, each symbol being either a letter, a blank or a punctuation mark. Encryption substitutes each symbol σ with another symbol $\pi(\sigma)$. The function π is the key, and has to be a permutation (meaning, one-to-one and onto) so that decryption is possible.

Encryption of this form is quite natural and well known, and, indeed, to many people it defines how encryption is done. We will later see many other (and better) ways to encrypt, but it is worth beginning by exploring this one.

Let's begin by specifying the scheme a little more mathematically. It may be valuable at this time to review the box in the Introduction that recalls the vocabulary of formal languages; we will be talking of things like alphabets, symbols, and strings.

Let Σ be a finite alphabet, whose members are called symbols. (In our examples, Σ would contain the 26 letters of the English alphabet, the blank symbol $\sqcup$, and punctuation symbols. Let us refer to this henceforth as the *English alphabet.*) If x is a string over Σ then we denote by $x[i]$ its i-th symbol.

Recall that if x is a string then $|x|$ denotes the length of x, meaning the number of symbols in it. Let us also adopt the convention that if X is a set then $|X|$ denotes its size. The double use of the "$|\cdot|$" notation should not cause much problem since the type of object to which it is applied, namely a set or a string, will usually be quite clear.

A *permutation* on a set S is a map π: $S \to S$ that is one-to-one and onto. Such a map is invertible, and we denote its inverse by π^{-1}. The inverse is also a permutation, and the map and its inverse are related by the fact that $\pi^{-1}(\pi(x)) = x$ and $\pi(\pi^{-1}(y)) = y$ for all $x, y \in S$. We let $\mathsf{Perm}(S)$ denote the set of all permutations on set S. Note that this set has size $|S|!$.

In the introduction, we had discussed symmetric encryption schemes, and said that any such scheme is specified as a triple $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ consisting of a key-generation algorithm, an encryption

algorithm, and a decryption algorithm. A *substitution cipher over alphabet* Σ is a special kind of symmetric encryption scheme in which the output of the key-generation algorithm $\mathcal{K}$ is always a permutation over Σ and the encryption and decryption algorithms are as follows:

Algorithm $\mathcal{E}_\pi(M)$	Algorithm $\mathcal{D}_\pi(C)$
For $i = 1, \ldots, \lvert M\rvert$ do	For $i = 1, \ldots, \lvert C\rvert$ do
$C[i] \leftarrow \pi(M[i])$	$M[i] \leftarrow \pi^{-1}(C[i])$
Return C	Return M

Above, the plaintext M is a string over Σ, as is the ciphertext C. The key is denoted π and is a permutation over Σ. We will let $\mathsf{Keys}(\mathcal{SE})$ denote the set of all keys that might be output by $\mathcal{K}$.

There are many possible substitution ciphers over Σ, depending on the set $\mathsf{Keys}(\mathcal{SE})$. In the simplest case, this is the set of all permutations over Σ, and $\mathcal{K}$ is picking a permutation at random. But one might consider schemes in which permutations are chosen from a much smaller set.

In our examples, unless otherwise indicated, the alphabet will be the English one defined above, namely Σ contains the 26 English letters, the blank symbol $\sqcup$, and punctuation symbols. We will, for simplicity, restrict attention to substitution ciphers that are *punctuation respecting*. By this we mean that any key (permutation) $\pi \in \mathsf{Keys}(\mathcal{SE})$ leaves blanks and punctuation marks unchanged. In specifying such a key, we need only say how it transforms each of the 26 English letters.

Example 2.1 This is an example of how encryption is performed with a (punctuation respecting) substitution cipher. An example key (permutation) π is depicted below:

σ	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
$\pi(\sigma)$	D	B	U	P	W	I	Z	L	A	F	N	S	G	K	H	T	J	X	C	M	Y	O	V	E	Q	R

Note every English letter appears once and exactly once in the second row of the table. That's why π is called a permutation. The inverse π^{-1} permutation is obtained by reading the table backwards. Thus $\pi^{-1}(\texttt{D}) = \texttt{A}$ and so on. The encryption of the plaintext

$$M \;=\; \texttt{HI THERE}$$

is

$$C \;=\; \pi(\texttt{H})\pi(\texttt{H})\pi(\texttt{I})\pi(\sqcup)\pi(\texttt{T})\pi(\texttt{H})\pi(\texttt{E})\pi(\texttt{R})\pi(\texttt{E}) \;=\; \texttt{LA MLWXW} \quad \blacksquare$$

Now let $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an arbitrary substitution cipher. We are interested in its security. To assess this we think about what the adversary has and what it might want to do.

The adversary begins with the disadvantage of not being given the key π. It is assumed however to come in possession of a ciphertext C. The most basic goal that we can consider for it is that it wants to recover the plaintext $M = \mathcal{D}(\pi, C)$ underlying C.

The adversary is always assumed to know the "rules of the game." Meaning, it knows the algorithms $\mathcal{K}, \mathcal{E}, \mathcal{D}$. It knows that a substitution cipher is being used, and that it is punctuation respecting in our case. The *only* thing it does not know a priori is the key, for that is assumed to have been shared secretly and privately between the sender and receiver.

So the adversary seems some gibberish, such as the text `LA MLWXW`. One might imagine that in the absence of the key π it would have a tough time figuring out that the message was `HI THERE`. But in fact, substitution ciphers are not so hard to cryptanalyze. Indeed, breaking a substitution cipher is a popular exercise in a Sunday newspaper or magazine, and many of you may have done it. The adversary can use its knowledge of the structure of English text to its advantage. Often a

τ	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
$\pi^{-1}(\tau)$		R	T												H					A				E		
$\pi^{-1}(\tau)$		R	T	I					N						H	C				A		W		E		
$\pi^{-1}(\tau)$	L	R	T	I			M	F	N		O				H	C		S		A		W		E		
$\pi^{-1}(\tau)$	L	R	T	I			M	F	N		O			P	H	C	U	S		A	D	W		E		

Figure 2.1: Cryptanalysis of Example 2.2.

good way to begin is by making what is called a *frequency table*. This table shows, for ever letter τ, how often τ occurs in the ciphertext. Now it turns out that the most common letter in English text is typically E. The next most common are the group T, A, O, I, N, S, H, R. (These letters have roughly the same frequency, somewhat lower than that of E, but higher than other letters.) So if X is the most frequent ciphertext symbol, a good guess would be that it represents E. (The guess is not necessarily true, but one attempts to validate or refute it in further stages.) Another thing to do is look for words that have few letters. Thus, if the letter T occurs by itself in the ciphertext, we conclude that it must represent A or I. Two letter words give similar information. And so on, it is remarkable how quickly you actually (usually) can figure out the key.

Example 2.2 Let us try to decrypt the following ciphertext:

```
COXBX TBX CVK CDGXR DI T GTI'R ADHX VOXI OX ROKQAU IKC RNXPQATCX: VOXI OX
PTI'C THHKBU DC, TIU VOXI OX PTI.
```

Here is our frequency table:

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
3	3	7	4	0	0	2	3	9	0	4	0	0	1	8	3	2	4	0	8	3	4	0	13	0	0

The most common symbol being X, we guess that $\pi^{-1}(\mathtt{X}) = \mathtt{E}$. Now we see the word OX, and, assuming X represents E, O must represent one of B, H, M, W. We also note that O has a pretty high frequency count, namely 8. So my guess is that O falls in the second group of letters mentioned above. But of the letters B, H, M and W, only H is in this group, so let's guess that $\pi^{-1}(\mathtt{O}) = \mathtt{H}$. Now, consider the first word in the ciphertext, namely COXBX. We read it as $*\mathtt{HE}*\mathtt{E}$. This could be THERE or THESE. I will guess that $\pi^{-1}(\mathtt{C}) = \mathtt{T}$, keeping in mind that $\pi^{-1}(\mathtt{B})$ should be either R or S. The letter T occurs on its own in the ciphertext, so must represent A or I. But the second ciphertext word can now be read as $*\mathtt{RE}$ or $*\mathtt{SE}$, depending on our guess for B discussed above. We know that the $*$ (which stands for the letter T in the ciphertext) decodes to either A or I. Even though a few choices yield English words, my bet is the word is ARE, so I will guess $\pi^{-1}(\mathtt{T}) = \mathtt{A}$ and $\pi^{-1}(\mathtt{B}) = \mathtt{R}$. The second row of the table in Fig. 2.1 shows where we are. Now let us write the ciphertext again, this time indicating above different letters what we believe them to represent:

```
THERE ARE T   T  E     A  A '     E  HE  HE  H       T   E   ATE:  HE  HE
COXBX TBX CVK CDGXR DI T GTI'R ADHX VOXI OX ROKQAU IKC RNXPQATCX: VOXI OX

 A 'T A   R   T, A    HE  HE  A .
PTI'C THHKBU DC, TIU VOXI OX PTI.
```

Since the last letter of the ciphertext word DC represents T, the first letter must represent A or I. But we already have something representing A, so we guess that $\pi^{-1}(\mathtt{D}) = \mathtt{I}$. From the ciphertext word DI it follows that I must be either N, T or S. It can't be T because C already represents T. But I is also the last letter of the ciphertext word VOXI, and ∗HEN is a more likely ending than ∗HES so I will guess $\pi^{-1}(\mathtt{I}) = \mathtt{N}$. To make sense of the ciphertext word VOXI, I then guess that $\pi^{-1}(\mathtt{V}) = \mathtt{W}$. The ciphertext word PTI'C is now ∗AN'T and so surely $\pi^{-1}(\mathtt{P}) = \mathtt{C}$. The second row of the table of Fig. 2.1 shows where we are now, and our text looks like:

```
THERE ARE TW  TI E  IN A  AN'   I E WHEN HE  H     N T   EC  ATE: WHEN HE
COXBX TBX CVK CDGXR DI T GTI'R ADHX VOXI OX ROKQAU IKC RNXPQATCX: VOXI OX

CAN'T A   R  IT, AN  WHEN HE CAN.
PTI'C THHKBU DC, TIU VOXI OX PTI.
```

At this point I can decrypt the first 8 words of the ciphertext pretty easily: THERE ARE TWO TIMES IN A MAN'S LIFE. The third row of the table of Fig. 2.1 shows where we are after I put in the corresponding guesses. Applying them, our status is:

```
THERE ARE TWO TIMES IN A MAN'S LIFE WHEN HE SHO L  NOT S EC LATE: WHEN HE
COXBX TBX CVK CDGXR DI T GTI'R ADHX VOXI OX ROKQAU IKC RNXPQATCX: VOXI OX

CAN'T AFFOR  IT, AN  WHEN HE CAN.
PTI'C THHKBU DC, TIU VOXI OX PTI.
```

The rest is easy. The decryption is:

```
THERE ARE TWO TIMES IN A MAN'S LIFE WHEN HE SHOULD NOT SPECULATE: WHEN HE
COXBX TBX CVK CDGXR DI T GTI'R ADHX VOXI OX ROKQAU IKC RNXPQATCX: VOXI OX

CAN'T AFFORD IT, AND WHEN HE CAN.
PTI'C THHKBU DC, TIU VOXI OX PTI.
```

The third row of the table of Fig. 2.1 shows our final knowledge of the key π. The text, by the way, is a quotation from Mark Twain. ▌

Some people argue that this type of cryptanalysis is not possible if the ciphertext is short, and thus that substitution ciphers work fine if, say, one changes the key quite frequently. Other people argue for other kinds of variants and extensions. And in fact, these types of systems have been the basis for encryption under relatively modern times. We could spend a lot of time on this subject, and many books do, but we won't. The reason is that, as we will explain, the idea of a substitution cipher is flawed at a quite fundamental level, and the flaw remains in the various variations and enhancements proposed. It will take some quite different ideas to get systems that deliver quality privacy.

To illustrate why the idea of a substitution cipher is flawed at a fundamental level, consider the following example usage of the scheme. A polling station has a list of voters, call them $V_1, V_2, \ldots, V_n$. Each voter casts a (secret) ballot which is a choice between two values. You could think of them as YES or NO, being votes on some Proposition, or BUSH and KERRY. In any case, we represent them as letters: the two choices are Y and N. At the end of the day, the polling station has a list $v_1, \ldots, v_n$ of n votes, where v_i is V_i's vote. Each vote being a letter, either Y or N, we can think of the list of votes as a string $v = v_1 \ldots v_n$ over the alphabet of English letters. The polling station wants to transmit this string to a tally center, encrypted in order to preserve anonymity of votes. The polling station and tally center have agreed on a key π for a substitution cipher. The polling station encrypts the message string v to get a ciphertext string $c = \pi(v_1) \ldots \pi(v_n)$ and transmits this to the tally center. Our question is, is this secure?

It quickly becomes apparent that it is not. There are only two letters in v, namely Y and N. This means that c also contains only two letters. Let's give them names, say A and B. One of these is $\pi(\mathtt{Y})$ and the other is $\pi(\mathtt{N})$. If the adversary knew which is which, it would, from the ciphertext, know the votes of all voters. But we claim that it is quite easy for the adversary to know which is which. Consider for example that the adversary is one of the voters, say V_1. So it knows its own vote v_1. Say this is Y. It now looks at the first symbol in the ciphertext. If this is A, then it knows that $A = \pi(\mathtt{Y})$ and thus that $B = \mathtt{N}$, and can now immediately recover all of $v_2, \ldots, v_n$ from the ciphertext. (If the first symbol is B, it is the other way around, but again it recovers all votes.)

This attack works even when the ciphertext is short (that is, when n is small). The weakness is exhibits is in the very nature of the cipher, namely that a particular letter is always encrypted in the same way, and thus repetitions can be detected.

Pinpointing this weakness illustrates something of the types of mode of thought we need to develop in cryptography. We need to be able to think about usage of application scenarios in which a scheme that otherwise seems good will be seen to be bad. For example, above, we considered not only that the encrypted text is votes, but that the adversary could be one of the voters. We need to always ask, "what if?"

We want symmetric encryption schemes that are not subject to these types of attacks, and in particular would provide security in an application like the voting one described above. Towards this end let us consider another scheme.

The One-Time-Pad (OTP) scheme with key-length m is the symmetric encryption scheme $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ whose algorithms are as follows. The code for the key-generation algorithm is $K \xleftarrow{\$} \{0,1\}^m$; **return** K, meaning a key is a random m-bit string. The encryption algorithm is defined by $\mathcal{E}_K(M) = K \oplus M$, where the message M is an m-bit binary string and $\oplus$ denotes bitwise XOR. The decryption algorithm is defined by $\mathcal{D}_K(C) = K \oplus C$, where $C \in \{0,1\}^m$. The correctness condition is met because $\mathcal{D}_K(\mathcal{E}_K(M)) = K \oplus (K \oplus M) = M$ for all $M \in \{0,1\}^m$.

Let us go back to our voting example. Represent Y by 1 and N by 0, and now encrypt the vote string $v = v_1 \ldots v_n$ with the OTP scheme with key-length n. Thus the ciphertext is $C = K \oplus v$. This time, weaknesses such as those we saw above are not present. Say the adversary has the ciphertext $C = C_1 \ldots C_n$, where C_i is the i-th bit of C. Say the adversary knows that $v_1 = 1$. It can deduce that K_1, the first bit of K, is $1 \oplus C_1$. But having K_1 tells it nothing about the other bits of K, and hence $v_2, \ldots, v_n$ remain hidden.

It turns out that the higher quality of privacy we see here is not confined to this one application setting. The scheme has a property called perfect security, which we will define below, and effectively provides the best possible security. We will also see that substitution ciphers do not have this property, providing a more formal interpretation of the concrete weaknesses we have seen in them.

Before going on, we remark that the perfect security of the OTP with key-length m relies crucially on our encrypting only a single message of length m. Were we to encrypt two or more messages, privacy would be lost. To see this, suppose we encrypt M_1, then M_2. The adversary obtains $C_1 = K \oplus M_1$ and $C_2 = K \oplus M_2$. XORing them together, it obtains $M_1 \oplus M_2$. This however provides partial information about the data, and, in particular, if the adversary would now happen to learn M_1, it could deduce M_2, which is not desirable.

The idea behind perfect security is to consider that one of two messages M_1 or M_2 is being encrypted. The adversary is aware of this, and all it wants to know is which of the two it is. It has in hand the ciphertext C, and now asks itself whether, given C, it can tell whether it was M_1 or M_2 that gave rise to it. Perfect security says that the adversary cannot tell. It asks that the probability

that C arises as the ciphertext is the same whether M_1 or M_2 was chosen to be encrypted.

Definition 2.3 (Perfect Security of a Symmetric Encryption Scheme) Let $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a symmetric encryption scheme, and assume we use it to encrypt just one message, drawn from a set $\mathsf{Plaintexts}$. We say that $\mathcal{SE}$ is *perfectly secure* if for any two messages $M_1, M_2 \in \mathsf{Plaintexts}$ and any C

$$\Pr[\mathcal{E}_K(M_1) = C] \;=\; \Pr[\mathcal{E}_K(M_2) = C] \;. \tag{2.1}$$

In both cases, the probability is over the random choice $K \xleftarrow{\$} \mathcal{K}$ and over the coins tossed by $\mathcal{E}$ if any. ▮

Let us now show that a substitution cipher fails to have this property, even if the ciphertext encrypted is very short, say three letters.

Claim 2.4 Let $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a substitution cipher over the alphabet Σ consisting of the 26 English letters. Assume that $\mathcal{K}$ picks a random permutation over Σ as the key. (That is, its code is $\pi \xleftarrow{\$} \mathsf{Perm}(\Sigma)$; **return** π.) Let $\mathsf{Plaintexts}$ be the set of all three letter English words. Assume we use $\mathcal{SE}$ to encrypt a single message from $\mathsf{Plaintexts}$. Then $\mathcal{SE}$ is *not* perfectly secure. ▮

Intuitively, this is due to the weakness we saw above, namely that if a letter appears twice in a plaintext, it is encrypted the same way in the ciphertext, and thus repetitions can be detected. If M_1, M_2 are messages such that M_1 contains repeated letters but M_2 does not, then, if the adversary sees a ciphertext with repeated letters it knows that M_1 was encrypted. This means that this particular ciphertext has different probabilities of showing up in the two cases. We now make all this formal.

Proof of Claim 2.4: We are asked to show that the condition of Definition 2.3 does not hold, so the first thing to do is refer to the definition and right down what it means for the condition to not hold. It is important here to be careful with the logic. The contrapositive of "for all M_1, M_2, C some condition holds" is "there exist M_1, M_2, C such that the condition does not hold." Accordingly, we need to show there exist $M_1, M_2 \in \mathsf{Plaintexts}$, and there exists C, such that

$$\Pr[\mathcal{E}_\pi(M_1) = C] \neq \Pr[\mathcal{E}_\pi(M_2) = C] \;. \tag{2.2}$$

We have replaced K with π because the key here is a permutation.

We establish the above by picking M_1, M_2, C in a clever way. Namely we set M_1 to some three letter word that contains a repeated letter; specifically, let us set it to $\mathtt{FEE}$. We set M_2 to a three letter word that does not contain any repeated letter; specifically, let us set it to $\mathtt{FAR}$. We set C to $\mathtt{XYY}$, a ciphertext that has the same "pattern" as $\mathtt{FEE}$ in the sense that the last two letters are the same and the first is different from these. Now we evaluate the probabilities in question:

$$\begin{aligned}
\Pr[\mathcal{E}_\pi(M_1) = C] &= \Pr[\mathcal{E}_\pi(\mathtt{FEE}) = \mathtt{XYY}] \\
&= \frac{|\{\, \pi \in \mathsf{Perm}(\Sigma) \;:\; \mathcal{E}_\pi(\mathtt{FEE}) = \mathtt{XYY} \,\}|}{|\mathsf{Perm}(\Sigma)|} \\
&= \frac{|\{\, \pi \in \mathsf{Perm}(\Sigma) \;:\; \pi(\mathtt{F})\pi(\mathtt{E})\pi(\mathtt{E}) = \mathtt{XYY} \,\}|}{|\mathsf{Perm}(\Sigma)|} \\
&= \frac{24!}{26!} \\
&= \frac{1}{650} \;.
\end{aligned}$$

Recall that the probability is over the choice of key, here π, which is chosen at random from $\mathsf{Perm}(\Sigma)$, and over the coins of $\mathcal{E}$, if any. In this case, $\mathcal{E}$ does not toss coins, so the probability is over π alone. The probability can be expressed as the ratio of the number of choices of π for which the stated event, namely that $\mathcal{E}_\pi(\mathtt{FEE}) = \mathtt{XYY}$, is true, divided by the total number of possible choices of π, namely the size of the set $\mathsf{Perm}(\Sigma)$ from which π is drawn. The second term is 26!. For the first, we note that the condition means that $\pi(\mathtt{F}) = \mathtt{X}$ and $\pi(\mathtt{E}) = \mathtt{Y}$, but the value of π on any of the other 24 input letters may still be any value other than $\mathtt{X}$ or $\mathtt{Y}$. There are 24! different ways to assign distinct values to the remaining 24 inputs to π, so this is the numerator above. Now, we proceed similarly for M_2:

$$\begin{aligned}
\Pr\left[\mathcal{E}_\pi(M_2) = C\right] &= \Pr\left[\mathcal{E}_\pi(\mathtt{FAR}) = \mathtt{XYY}\right] \\
&= \frac{\left|\{\ \pi \in \mathsf{Perm}(\Sigma)\ :\ \mathcal{E}_\pi(\mathtt{FAR}) = \mathtt{XYY}\ \}\right|}{\left|\mathsf{Perm}(\Sigma)\right|} \\
&= \frac{\left|\{\ \pi \in \mathsf{Perm}(\Sigma)\ :\ \pi(\mathtt{F})\pi(\mathtt{A})\pi(\mathtt{R}) = \mathtt{XYY}\ \}\right|}{\left|\mathsf{Perm}(\Sigma)\right|} \\
&= \frac{0}{26!} \\
&= 0\ .
\end{aligned}$$

In this case, the numerator asks us to count the number of permutations π with the property that $\pi(\mathtt{F}) = \mathtt{X}$, $\pi(\mathtt{A}) = \mathtt{Y}$ and $\pi(\mathtt{R}) = \mathtt{Y}$. But no permutation can have the same output $\mathtt{Y}$ on two different inputs. So the number of permutations meeting this condition is zero.

In conclusion, we have Equation (2.2) because the two probabilities we computed above are different. ▮

Let us now show that the OTP scheme with key-length m does have the perfect security property. Intuitively, the reason is as follows. Say $m = 3$, and consider two messages, say $M_1 = 010$ and $M_2 = 001$. Say the adversary receives the ciphertext $C = 101$. It asks itself whether it was M_1 or M_2 that was encrypted. Well, it reasons, if it was M_1, then the key must have been $K = M_1 \oplus C = 111$, while if M_2 was encrypted then the key must have been $K = M_2 \oplus C = 100$. But either of these two was equally likely as the key, so how do I know which of the two it was? Here now is the formal statement and proof.

Claim 2.5 Let $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be the OTP scheme with key-length $m \geq 1$. Assume we use it to encrypt a single message drawn from $\{0,1\}^m$. Then $\mathcal{SE}$ is perfectly secure. ▮

Proof of Claim 2.5: As per Definition 2.3, for any $M_1, M_2 \in \{0,1\}^m$ and any C we need to show that Equation (2.1) is true. So let M_1, M_2 be any m-bit strings. We can assume C is also an m-bit string, since otherwise both sides of Equation (2.1) are zero and thus equal. Now

$$\begin{aligned}
\Pr\left[\mathcal{E}_K(M_1) = C\right] &= \Pr\left[K \oplus M_1 = C\right] \\
&= \frac{\left|\{\ K \in \{0,1\}^m\ :\ K \oplus M_1 = C\ \}\right|}{\left|\{0,1\}^m\right|} \\
&= \frac{1}{2^m}\ .
\end{aligned}$$

Above, the probability is over the random choice of K from $\{0,1\}^m$, with M_1, C fixed. We write the probability as the ratio of two terms: the first is the number of keys K for which $K \oplus M = C$, and the second is the total possible number of keys. The first term is one, because K can only be the string $M_1 \oplus C$, while the second term is 2^m. Similarly we have

$$\begin{aligned} \Pr\left[\mathcal{E}_K(M_2) = C\right] &= \Pr\left[K \oplus M_2 = C\right] \\ &= \frac{|\{\, K \in \{0,1\}^m \;:\; K \oplus M_2 = C \,\}|}{|\{0,1\}^m|} \\ &= \frac{1}{2^m} \,. \end{aligned}$$

In this case the numerator of the fraction is one because only the key $K = M_2 \oplus C$ has the property that $K \oplus M_2 = C$. Now, since the two probabilities we have computed above are equal, Equation (2.1) is true, and thus our proof is complete. ∎

Perfect security is great in terms of security, but comes at a hefty price. It turns out that in any perfectly secure scheme, the length of the key must be at least the length of the (single) message encrypted. This means that in practice a perfectly secure scheme (like the OTP) is prohibitively expensive, requiring parties to exchange very long keys before they can communicate securely.

In practice we want parties to be able to hold a short key, for example 128 bits, and then be able to securely encrypt essentially any amount of data. To achieve this, we need to make a switch regarding what kinds of security attributes we seek. As we discussed in the Introduction, we will ask for security that is not perfect but good enough, the latter interpreted in a computational sense. Visualizing an adversary as someone running programs to break our system, we will say something like, yes, in principle you can break my scheme, but it would take more than 100 years running on the world's fastest computers to break it with a probability greater than 2^{-60}. In practice, this is good enough.

To get schemes like that we need some tools, and this is what we turn to next.