Crypto 1

Note: this material is not intended to replace the live lecture for students.

Contents

(Gen , Enc , Dec)

1.1 Introduction to Cryptography

The art and science of keeping messages secure is **cryptography**, and it is practiced by **cryptographers**. **Cryptanalysts** are practitioners of **cryptanalysis**, the art and science of breaking ciphertext; that is, seeing through the disguise. The branch of mathematics encompassing both cryptography and cryptanalysis is **cryptology** and its practitioners are **cryptologists**. Modern cryptologists are generally trained in theoretical mathematics **they have to be**.

[Schneier15, Chapter 1, page 1]

If I take a letter, lock it in a safe, hide the safe somewhere in New York, then tell you to read the letter, that's not security. That's obscurity. On the other hand, if I take a letter and lock it in a safe, and then give you the safe along with the design specifications of the safe and a hundred identical safes with their combinations so that you and the world's best safecrackers can study the locking mechanism and you still can't open the safe and read the letter-that's security.

[Schneier15, Preface, page xxi]

NOTE 1.1.1

Important ideas in Cryptography were developed before the computer era. But along this course we are interested in computer related problems:

1) Communications between computers separated by space.

2) Communications between computers separated by time, e.g., to cipher a hard disk.

We are going to use **python**.

Here a link with **python** basics : https://medium.com/free-code-camp/learning-python-from-zero-to-hero-120ea540b567

Actually at the end of the course it should be possible to read and try to understand documents of the type seen here: NIST-SP

Exercise 1.1.2

Download Hacking Secret Ciphers with **python** and take a look to Chapter 1.

1.2 Basics

Figure 1.2.1: Encipher & Decipher: symmetric

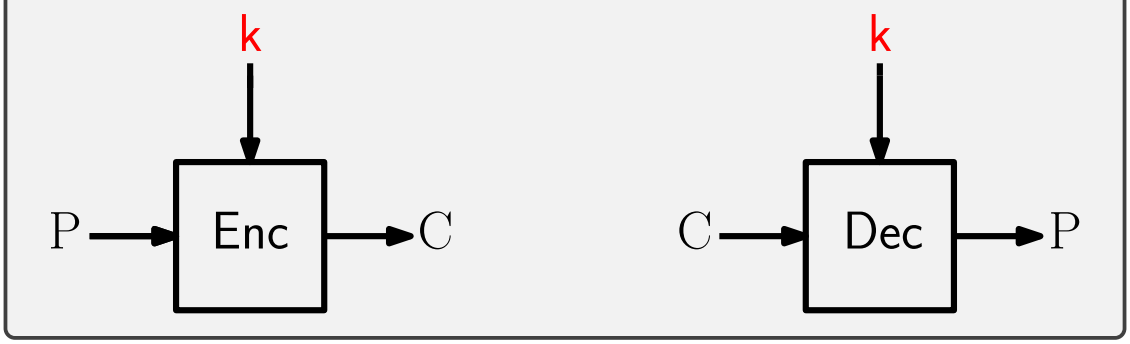
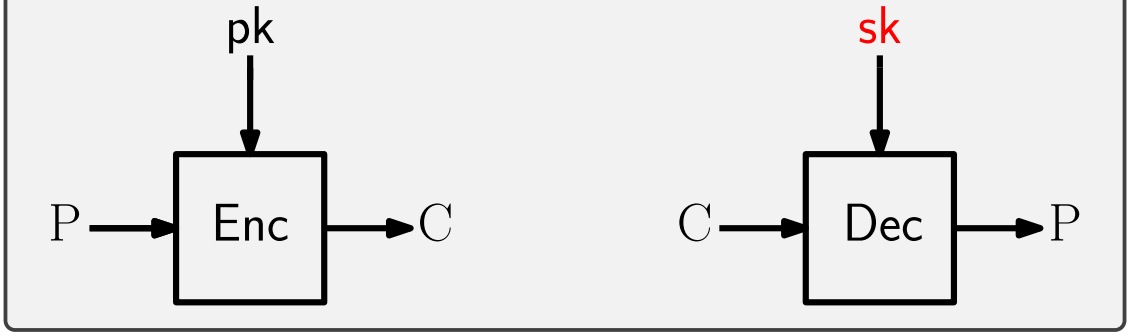


Figure 1.2.2: Encipher & Decipher: Asymmetric



P = plaintext, cleartext, message. Sometimes also indicated as M.

k = key, pk public key , sk secret key.

Gen = Key generation algorithm .

Enc = Encipher algorithm (encryption, ISO 7498-2 "encipher").

C = ciphertext, cryptogram .

Dec = Decipher algorithm (decryption, ISO 7498-2 "decipher").

Cryptanalyst = enemy, eavesdropper.

attack & break .

spaces: e.g. key space $\mathcal{K}$ or message space $\mathcal{M}$.

forze brutte attack = to check all $k \in \mathcal{K}$ or all $M \in \mathcal{M}$, etc

NOTE 1.2.3

To use crypto all parties must agree on all the elements defining the crypto system, that is, the **domain parameters** of the scheme.

Kerckhoffs principle

The enemy knows the system (Shannon's Maxim). This means that security should just depend on the secrecy of the key.

We will call a task *computationally infeasible* if its cost as measured by either the amount of memory used or the runtime is finite but impossible large.

[DH76, page 646]

To get a clue of the meaning of *computationally infeasible* imagine you are looking for a key k of m bits:

$$k \in \{0,1\}^m$$

1.2.4 Brute Force: 30 bits

```
from timeit import default_timer as timer

#loop with 2**m rounds
m=30
start = timer()
j=0
while j < 2**m:
        # try with key k_j
        print(j)
        j+=1
end = timer()

#print the total time employed to check all keys
print(m, (end - start)/60 ,"minutes") # Time in minutes.
```

Exercise 1.2.5

How long it takes for m=64 bits ?

Take a look to [Paar10, page 12, Table 1.2] e [Schneier15, page 18, Table 1.1]

NOTE 1.2.6

Bits in **python**. Here are two of them:

```
'{0:b}'.format(22)

int('01001', 2)
```

https://docs.python.org/3/library/string.html#formatspec
https://docs.python.org/3/tutorial/datastructures.html

1.3 Three baby examples of symmetric ciphers

Alice have messages constructed by using plaintexts of 5 bits $\mathrm{P} \in \{0,1\}^5$ e.g. $\mathrm{P} = [11100]$ and wants to send them to Bob. Oscar is a eavesdropper.

NOTE 1.3.1

The motivation of using 5 bits in these examples is the Baudot code for telegraphy invented by Emile Baudot in the 1870s. Baudot's code is the predecessor of the ITA-2 code which was used until the advent of ASCII.

Baudot developed his first multiplexed telegraph in 1872 and patented it in 1874. In 1876, he changed from a six-bit code to a five-bit code, as suggested by Carl Friedrich Gauss and Wilhelm Weber in 1834. The code itself was not patented (only the machine) because French patent law does not allow concepts to be patented.

1.3.1 The Vernam cipher or $\oplus$-cipher

Alice and Bob meet somewhere and agree in using $\mathsf{k} = [01010]$ as their secret key.

Alice		**Bob**
$\mathrm{C} = \mathsf{Enc}_{\mathsf{k}}(\mathrm{P}) = \mathrm{P} \oplus \mathsf{k}$		
	$\xrightarrow{\qquad\qquad\mathrm{C}\qquad\qquad}$	
		$\mathrm{P} = \mathsf{Dec}_{\mathsf{k}}(\mathrm{C}) = \mathrm{C} \oplus \mathsf{k}$

Notice that the key k is the binary representation of the number $10 = (01010)_2$. Any plaintext $\mathrm{P} \in \{0,1\}^5$ can be regarded as a number $0 \leq \mathrm{P} < 2^5 = 32$.

By using the code

0	1	2	3	$\cdots$	25	26	27	28	29	30	31
A	B	C	D	$\cdots$	Z	.	’	(	)	:	

Alice encipher a message M and send to Bob the following ciphertext:

11000||01110||01110||10101||10010||00100||11110||10101||11001||00100||00110
00100||11011||11011||00100||11100||10000||01010||00001||00010||01000||01110

Exercise 1.3.2

After some days Alice and Bob meet again and decide to change the key to a new one $\mathsf{k} \in \{0,1\}^5$. Then Alice, by using the new key, encrypt M and send the following ciphertext to Bob:

$$00100\|10010\|10010\|01001\|01110\|11000\|00010\|01001\|00101\|11000\|11010$$
$$11000\|00111\|00111\|11000\|00000\|01100\|10110\|11101\|11110\|10100\|10010$$

Can you find the new key k ?

1.3.2 The Caesar cipher or $\boxplus$-cipher

Here instead the $\oplus$ we use the $\boxplus$ operation which means the sum with the carry bit up to 5 bits. For example

$$11011 \boxplus 10101 = 10000$$

The $\boxplus$ operation is easy to see by using base 10 representation and arithmetic modulo $2^5 = 32$:

$$27 \boxplus 21 = 48 = 16 \pmod{32}$$

Alice		**Bob**
$\mathrm{C} = \mathsf{Enc}_{\mathsf{k}}(\mathrm{P}) = \mathrm{P} \boxplus \mathsf{k}$		
	$\xrightarrow{\qquad\qquad\mathrm{C}\qquad\qquad}$	
		$\mathrm{P} = \mathsf{Dec}_{\mathsf{k}}(\mathrm{C}) = \mathrm{C} \boxplus (-\mathsf{k})$

Here an example with key $\mathsf{k} = 3$:

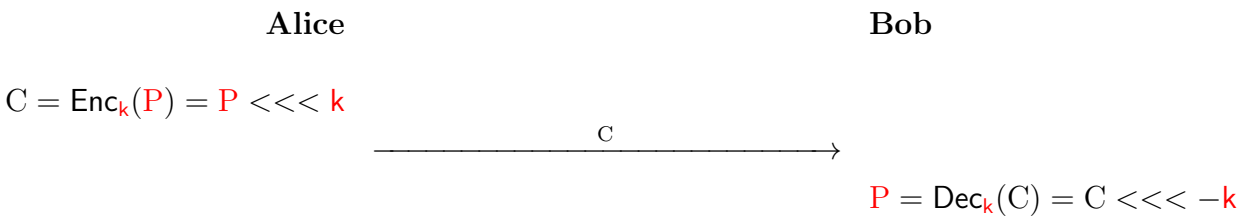
M = TOMORROW I CAN NOT ATTEND.ALICE
P = 19‖14‖12‖14‖17‖17‖14‖22‖31‖8‖31‖2‖0‖13‖31‖13‖14‖19‖31‖0‖19‖19‖4‖13‖3‖26‖0‖11‖8‖2‖4
C = 22‖17‖15‖17‖20‖20‖17‖25‖2‖11‖2‖5‖3‖16‖2‖16‖17‖22‖2‖3‖22‖22‖7‖16‖6‖29‖3‖14‖11‖5‖7

1.3.3 The rot cipher or circular shift <<< r

In this cipher instead of $\oplus$ or $\boxplus$ we use the operation $<<< r$. For example

$$[110111] <<< 3 = [111110]$$

Namely the bits are shifted toward left in a circular way.



For this cipher the key k used by Alice and Bob is an integer.

Exercise 1.3.3

Alice encrypt M using <<< k and send the following ciphertext C to Bob:

M = TOMORROW I CAN NOT ATTEND.ALICE
P = 19‖14‖12‖14‖17‖17‖14‖22‖31‖8‖31‖2‖0‖13‖31‖13‖14‖19‖31‖0‖19‖19‖4‖13‖3‖26‖0‖11‖8‖2‖4‖
C = 14‖25‖17‖25‖6‖6‖25‖26‖31‖1‖31‖8‖0‖21‖31‖21‖25‖14‖31‖0‖14‖14‖16‖21‖12‖11‖0‖13‖1‖8‖16‖

Can you find the key k ?

1.4 Basics of Modular Arithmetic

Modular Arithmetic deals with the study of the finite **ring**[1]:

$$\mathbb{Z}_n = \mathbb{Z}/n \cdot \mathbb{Z} = \{0, 1, \cdots, (n-1)\}$$

To get the remainder $r \in \mathbb{Z}_n$ of the integer $a \in \mathbb{Z}$ modulo n in **python**: `r = a%n`.
In mathematics such operation is written as

$$r = a \pmod{n}$$

whose meaning is that

$$a = n \cdot q + r$$

where r belongs to $\{0, 1, \cdots, (n-1)\}$ and $q \in \mathbb{Z}$.

NOTE 1.4.1

If p is a prime number the ring $\mathbb{Z}_p$ is also denoted as $\mathbb{F}_p$ or $\mathrm{GF}(p)$. Actually, GF means Galois Field. A field is a ring but with 4 operations: +,-,×, /.

The notation

$$a \equiv b \pmod{n}$$

means that

$$a \pmod{n} = b \pmod{n}$$

or in **python**

`a%n` = `b%n`

[1] Roughly speaking a ring is a set with 3 operations: +, - , ×.

Casting out nines (Prova del nove)

Perhaps one of the earliest contacts with modular arithmetics is represented by the so called "casting out nines" check, (prova del nove in italian). Indeed, to check the correctness of a plain arithmetic operation you $\pmod 9$ and check its validity $\pmod 9$ i.e. in the ring $\mathbb{Z}_9$. Usually the test is explained by using a diagram which made it easy to learn and memorize to children. Here an example: somebody claim that

$$832 \times 714 = 594048 \ .$$

Without doing the multiplication you can check it $\pmod 9$:

$$(832 \times 714) \pmod 9 = 594048 \pmod 9$$

so casting out nines:

$$4 \times 3 = 48 = 3 \pmod 9$$

hence the result pass the check since $12 = 3 \pmod 9$.
https://en.wikipedia.org/wiki/Casting_out_nines
https://mathworld.wolfram.com/CastingOutNines.html

Exercise 1.4.2

Show that $a \equiv b \pmod n$ if and only if n divides $a - b$.

1.4.1 The operation ⊠

For given $a, b \in \{0,1\}^5$ the operation ⊠ is the multiplication in base 2 disregarding bits of positions > 5 . For example, if $a = [01010], b = [10011]$ then we can compute $a \boxtimes b$ by hand as:

$$\begin{array}{cccccccccc}
 & & & & & 0 & 1 & 0 & 1 & 0 \\
 & & & & \boxtimes & 1 & 0 & 0 & 1 & 1 \\
\hline
 & & & & & 0 & 1 & 0 & 1 & 0 \\
 & & & & 0 & 1 & 0 & 1 & 0 & \\
 & & & 0 & 1 & 0 & & & & \\
\hline
 & & & & & 1 & 1 & 1 & 1 & 0
\end{array}$$

Actually, if a, b are regarded as integers in base 2 , i.e. $a = 10, b = 19$, then

$$10 \boxtimes 19 = 30$$

Notice that $10 \boxtimes 19 = 30$ because $10 \cdot 19 = 190 = 30 \pmod{32}$ or in **python** `(10*19)%32` gives 30.

Exercise 1.4.3

Show that $a \boxtimes b = (a \cdot b) \pmod{2^5}$.

Exercise 1.4.4

Find $x \in \{0,1\}^5$ such that $19 \boxtimes x = 1$.

1.4.2 a baby example of asymmetric cipher

Bob's pair $(\mathsf{sk}, \mathsf{pk})$: the public key is $\mathsf{pk} = 19$ and the secret key sk is the x such that $19 \boxtimes x = 1$. Here is how Alice use Bob's public key to encipher $\mathrm{P} \in \{0,1\}^5$:

Alice		**Bob**
$\mathrm{C} = \mathsf{Enc}_{\mathsf{pk}}(\mathrm{P}) = \mathrm{P} \boxtimes 19$	$\xrightarrow{\quad \mathrm{C} \quad}$	
		$\mathrm{P} = \mathsf{Dec}_{\mathsf{k}}(\mathrm{C}) = \mathrm{C} \boxtimes \mathsf{sk}$

Exercise 1.4.5

Can you find Bob's secret key sk?

1.5 A baby Galois ⊗-cipher

Following Evariste Galois we introduce a new multiplication ⊗ between strings $\mathrm{a}, \mathrm{b} \in \{0,1\}^5$:

$$\mathrm{a} \otimes \mathrm{b} = \mathrm{c}\,.$$

To compute c we regard the strings $\mathrm{a} = [a_4 a_3 a_2 a_1 a_0]$, $\mathrm{b} = [b_4 b_3 b_2 b_1 b_0]$ as polynomials

$$\mathrm{a}(x) = a_4 x^4 + a_3 x^3 + a_2 x^2 + a_1 x + a_0$$

$$\mathrm{b}(x) = b_4 x^4 + b_3 x^3 + b_2 x^2 + b_1 x + b_0$$

1.5.1 Galois multiplication

$\mathrm{c}(x)$ **is the remainder of** $\mathrm{a}(x) \times \mathrm{b}(x)$ **when divided by** $G(x) = 1 + x^2 + x^5$

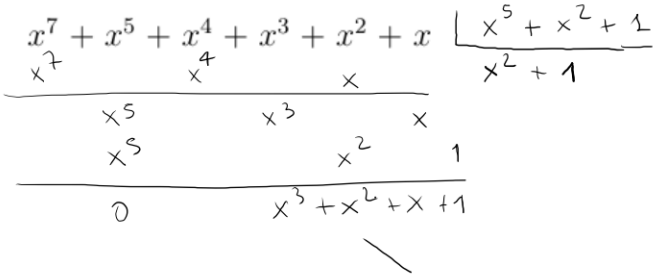
As an explicit example let us compute $[01010] \otimes [10011]$: First of all we compute

$$(0x^4 + 1x^3 + 0x^2 + 1x + 0) \times (1x^4 + 0x^3 + 0x^2 + 1x + 1)\,.$$

Namely:

				$0x^4$	$1x^3$	$0x^2$	$1x$	0
			$\times$	$1x^4$	$0x^3$	$0x^2$	$1x$	1
				$0x^4$	$1x^3$	$0x^2$	$1x$	0
			$0x^5$	$1x^4$	$0x^3$	$1x^2$	$0x$	
$0x^8$	$1x^7$	$0x^6$	$1x^5$	$0x^4$				
$0x^8$	$1x^7$	$0x^6$	$1x^5$	$1x^4$	$1x^3$	$1x^2$	$1x$	0

and we get $x^7 + x^5 + x^4 + x^3 + x^2 + x$. Now its remainder when divided by $x^5 + x^2 + 1$.



So

$$[01010] \otimes [10011] = [01111]\ ,\ \text{or}\ \ 10 \otimes 19 = 15$$

Bob's pair (sk, pk): the public key is pk = 19 and the secret key sk is the x such that $19 \otimes x = 1$. Here is how Alice use Bob's public key to encipher $\mathrm{P} \in \{0,1\}^5$:

Alice		**Bob**
$\mathrm{C} = \mathsf{Enc}_{\mathsf{pk}}(\mathrm{P}) = \mathrm{P} \otimes 19$	$\xrightarrow{\quad\mathrm{C}\quad}$	
		$\mathrm{P} = \mathsf{Dec}_{\mathsf{k}}(\mathrm{C}) = \mathrm{C} \otimes \mathsf{sk}$

Exercise 1.5.2

Can you find Bob's secret key sk?

NOTE 1.5.3

Here you can download Python libreries to handle polynomials with coefficients in $\mathbb{Z}_2$ https://github.com/popcornell/pyGF2

1.6 Symmetric Cryptography

Symmetric or Private Key Cryptosystem

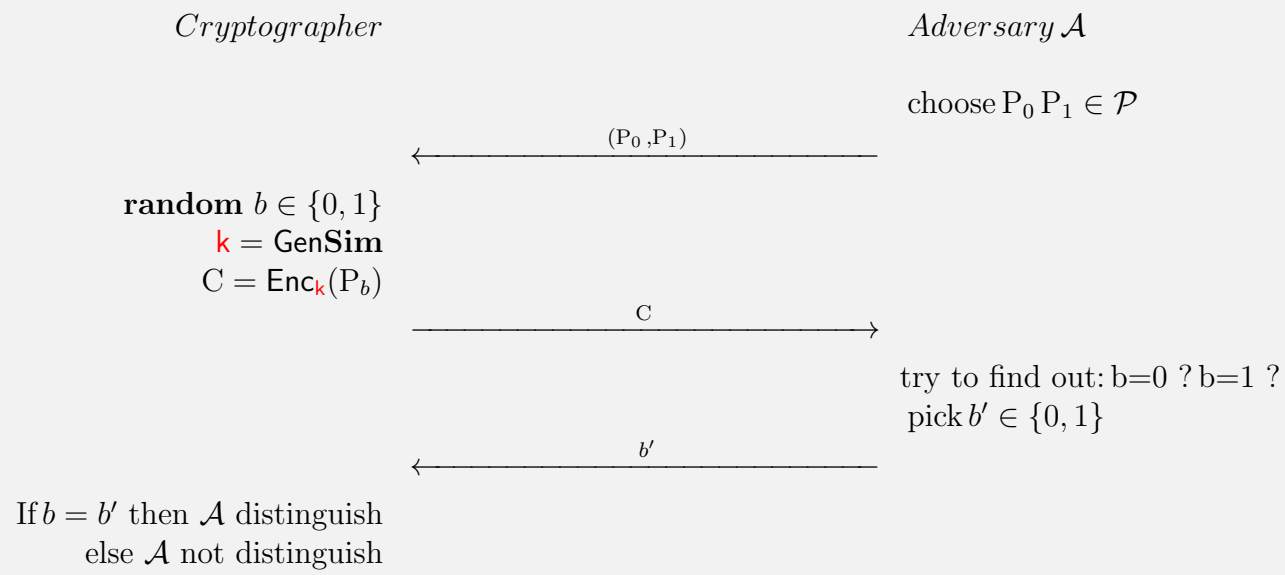
Such crypto system Π consists of three algorithms

$$(\mathsf{Gen}\,,\ \mathsf{Enc}\,,\ \mathsf{Dec})$$

Gen generate a key k.

The algorithms Gen, Enc e Dec must be computationally feasible,

1.6.1 IND Indistinguishability experiment



1.6.2 IND secure

The symmetric crypto system $\Pi = (\mathsf{Gen}\,,\ \mathsf{Enc}\,,\ \mathsf{Dec})$ is **IND**-secure if no adversary $\mathcal{A}$ can distinguish with probability better than $1/2$.

1.7 The Key Distribution Problem

> The goal is for two users, A and B, to securely exchange a key over an insecure channel. This key is then used by both users in a normal cryptosystem for both enciphering and deciphering.

[DH76, page 648]

Key-Exchange Protocol

is a protocol Π i.e. a set of instructions for the parties say Alice and Bob, that starting from a security parameter n allows them to compute keys k_A and k_B.
The correctness requirement is that

$$\mathsf{k}_A = \mathsf{k}_B$$

so we can speak simply of *the* key $\mathsf{k} = \mathsf{k}_A = \mathsf{k}_B$ as the exchanged key.

The security of the exchange protocol Π is related to the following:

The key-exchange experiment $\mathsf{KE}^{\mathsf{eav}}_{\mathcal{A},\Pi}(n)$:

1. *Two parties holding 1^n execute protocol Π. This execution of the protocol results in a transcript* trans *containing all the messages sent by the parties, and a key k that is output by each of the parties.*
2. *A random bit $b \leftarrow \{0,1\}$ is chosen. If $b = 0$ then choose $\hat{k} \leftarrow \{0,1\}^n$ uniformly at random, and if $b = 1$ set $\hat{k} := k$.*
3. *$\mathcal{A}$ is given* trans *and $\hat{k}$, and outputs a bit b'.*
4. *The output of the experiment is defined to be 1 if $b' = b$, and 0 otherwise. (In case $\mathsf{KE}^{\mathsf{eav}}_{\mathcal{A},\Pi}(n) = 1$, we say that $\mathcal{A}$* succeeds.*)*

1.7.1 Π is secure

if no adversary $\mathcal{A}$ can succeed with probability better than $1/2$.

Active adversaries. So far we have considered only the case of an eavesdropping adversary. Although eavesdropping attacks are by far the most common (as they are so easy to carry out), they are by no means the only possible attack. *Active* attacks, in which the adversary sends messages of its own to one or both of the parties are also a concern, and any protocol used in practice must be resilient to active attacks as well. When considering active attacks, it is useful to distinguish, informally, between *impersonation* attacks where only one of the honest parties is executing the protocol and the adversary impersonates the other party, and *man-in-the-middle* attacks where both honest parties are executing the protocol and the adversary is intercepting and modifying messages being sent from one party to the other.

We will not define security against either class of attacks, as such a definition is rather involved and also cannot be achieved without the parties sharing *some* information in advance. Nevertheless, it is worth remarking that the Diffie-Hellman protocol is *completely insecure* against man-in-the-middle attacks. In fact, a man-in-the-middle adversary can act in such a way that Alice and Bob terminate the protocol with different keys k_A and k_B that are both known to the adversary, yet neither Alice nor Bob can detect that any attack was carried out. We leave the details of this attack as an exercise.

The fact that the Diffie-Hellman protocol is not resilient to man-in-the-middle attacks does not detract in any way from its importance. The Diffie-Hellman protocol served as the first demonstration that asymmetric techniques (and number-theoretic problems) could be used to alleviate the problems of key distribution in cryptography. Furthermore, extensions of the Diffie-Hellman protocol can be shown to prevent man-in-the-middle attacks, and such protocols are widely used today.

1.8 Public Key Cryptography

Public Key Cryptosystem (PKC)

Such cryptosystem consists of three algorithms

$$(\mathsf{Gen}\,,\ \mathsf{Enc}\,,\ \mathsf{Dec})$$

Gen generate a pair (sk,pk) of secret key sk and public key pk.

1) The algorithms Gen, Enc e Dec must be computationally feasible,
2) The secret key sk should be computationally infeasible to compute from the public key pk.

Property 2) allows the publication of pk in public web pages.

The security of the public key cryptosystem Π is related to the following:

The eavesdropping indistinguishability experiment $\mathsf{PubK}^{\mathsf{eav}}_{\mathcal{A},\Pi}(n)$**:**

1. $\mathsf{Gen}(1^n)$ *is run to obtain keys* (pk, sk).
2. *Adversary* $\mathcal{A}$ *is given* pk, *and outputs a pair of messages* m_0, m_1 *of the same length. (These messages must be in the plaintext space associated with* pk.*)*
3. *A random bit* $b \leftarrow \{0,1\}$ *is chosen; and then a ciphertext* $c \leftarrow \mathsf{Enc}_{pk}(m_b)$ *is computed and given to* $\mathcal{A}$. *We call* c *the* challenge ciphertext.
4. $\mathcal{A}$ *outputs a bit* b'.
5. *The output of the experiment is defined to be 1 if* $b' = b$, *and 0 otherwise.*

1.8.1 Π is secure

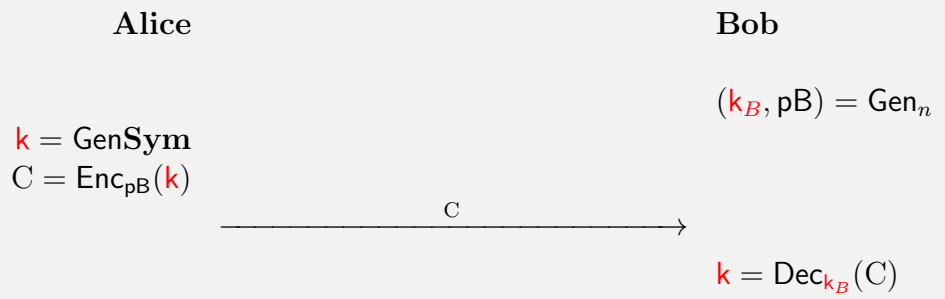
if no adversary $\mathcal{A}$ can succeed with probability better than $1/2$.

1.8.1 Using a PKC to get a exchange protocol Π

Here is how to use a **PKC** to exchange a key k.

1.8.2 PKC ⇒ PKDS

Alice and Bob are users of a **PKC**=(Gen, Enc, Dec).
They want to use a symmetric cipher **Sym**, to encipher big plaintexts, by using the key k which was generated by Alice.



NOTE 1.8.3

Something similar do the software PGP (Pretty Good Privacy) by Phil Zimmermann developed in 1991. In PGP, the **PKC** was RSA and the symmetric cipher **Sym** was IDEA.

Here is a nice plug-in to cipher gmails: https://flowcrypt.com/

1.9 Attacks!!

There are four general types of cryptanalytic attacks. Of course, each of them assumes that the cryptanalyst has complete knowledge of the encryption algorithm used:

1. **Ciphertext-only attack**. The cryptanalyst has the ciphertext of several messages, all of which have been encrypted using the same encryption algorithm. The cryptanalyst's job is to recover the plaintext of as many messages as possible, or better yet to deduce the key (or keys) used to encrypt the messages, in order to decrypt other messages encrypted with the same keys.

   Given: $C_1 = E_k(P_1), C_2 = E_k(P_2), \ldots C_i = E_k(P_i)$
   Deduce: Either $P_1, P_2, \ldots P_i$; k; or an algorithm to infer P_{i+1} from $C_{i+1} = E_k(P_{i+1})$

2. **Known-plaintext attack**. The cryptanalyst has access not only to the ciphertext of several messages, but also to the plaintext of those messages. His job is to deduce the key (or keys) used to encrypt the messages or an algorithm to decrypt any new messages encrypted with the same key (or keys).

   Given: $P_1, C_1 = E_k(P_1), P_2, C_2 = E_k(P_2), \ldots P_i, C_i = E_k(P_i)$
   Deduce: Either k, or an algorithm to infer P_{i+1} from $C_{i+1} = E_k(P_{i+1})$

3. **Chosen-plaintext attack**. The cryptanalyst not only has access to the ciphertext and associated plaintext for several messages, but he also chooses the plaintext that gets encrypted. This is more powerful than a known-plaintext attack, because the cryptanalyst can choose specific plaintext blocks to encrypt, ones that might yield more information about the key. His job is to deduce the key (or keys) used to encrypt the messages or an algorithm to decrypt any new messages encrypted with the same key (or keys).

   Given: $P_1, C_1 = E_k(P_1), P_2, C_2 = E_k(P_2), \ldots P_i, C_i = E_k(P_i)$, where the cryptanalyst gets to choose $P_1, P_2, \ldots P_i$
   Deduce: Either k, or an algorithm to infer P_{i+1} from $C_{i+1} = E_k(P_{i+1})$

4. **Adaptive-chosen-plaintext attack**. This is a special case of a chosen-plaintext attack. Not only can the cryptanalyst choose the plaintext that is encrypted, but he can also modify his choice based on the results of previous encryption. In a chosen-plaintext attack, a cryptanalyst might just be able to choose one large block of plaintext to be encrypted; in an adaptive-chosen-plaintext attack he can choose a smaller block of plaintext and then choose another based on the results of the first, and so forth.

[Schneier15, page 6]

5. **Chosen-ciphertext attack**. The cryptanalyst can choose different ciphertexts to be decrypted and has access to the decrypted plaintext. For example, the cryptanalyst has access to a tamperproof box that does automatic decryption. His job is to deduce the key.

   Given: C_1, $P_1 = D_k(C_1)$, C_2, $P_2 = D_k(C_2)$, . . . C_n, $P_n = D_k(C_n)$
   Deduce: k

   This attack is primarily applicable to public-key algorithms and will be discussed in Section 19.3. A chosen-ciphertext attack is sometimes effective against a symmetric algorithm as well. (Sometimes a chosen-plaintext attack and a chosen-ciphertext attack are together known as a **chosen-text attack**.)

6. **Chosen-key attack**. This attack doesn't mean that the cryptanalyst can choose the key; it means that he has some knowledge about the relationship between different keys. It's strange and obscure, not very practical, and discussed in Section 12.4.

7. **Rubber-hose cryptanalysis**. The cryptanalyst threatens, blackmails, or tortures someone until they give him the key. Bribery is sometimes referred to as a **purchase-key attack**. These are all very powerful attacks and often the best way to break an algorithm.

[Schneier15, page 7]

1.9.1 Security Level

Security level

An encryption algorithm has a security level of n bits if the best known attack requires $\mathcal{O}(2^n)$ steps. This allows us to compare algorithms and is useful when we combine several primitives in a hybrid cryptosystem to understand any weaknesses. The security level is related to a **security parameter** λ or κ which is usually written in unary 1^n.

In most cryptographic functions, the key length is an important security parameter.

1.10 CPA-IND and Probabilistic Encryption (Non deterministic)

Different ciphertexts for the same cleartext M encrypted with a key k.

The CPA indistinguishability experiment $\mathsf{PrivK}^{\mathsf{cpa}}_{\mathcal{A},\Pi}(n)$**:**

1. *A key k is generated by running* $\mathsf{Gen}(1^n)$.
2. *The adversary $\mathcal{A}$ is given input 1^n and oracle access to* $\mathsf{Enc}_k(\cdot)$, *and outputs a pair of messages m_0, m_1 of the same length.*
3. *A random bit $b \leftarrow \{0,1\}$ is chosen, and then a ciphertext $c \leftarrow \mathsf{Enc}_k(m_b)$ is computed and given to $\mathcal{A}$. We call c the* challenge ciphertext.
4. *The adversary $\mathcal{A}$ continues to have oracle access to* $\mathsf{Enc}_k(\cdot)$, *and outputs a bit b'.*
5. *The output of the experiment is defined to be 1 if $b' = b$, and 0 otherwise. (In case $\mathsf{PrivK}^{\mathsf{cpa}}_{\mathcal{A},\Pi}(n) = 1$, we say that $\mathcal{A}$* succeeded.)

1.10.1 CPA-IND secure

The symmetric crypto system $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is **CPA-IND**-secure (or just CPA-secure) if no adversary $\mathcal{A}$ can succeed with probability better than $1/2$.

Exercise 1.10.2

Modify the CPA indistinguishability experiment for COA indistinguishability i.e. Ciphertext Only Attack. Then give a definition of **COA-IND**-secure.

Exercise 1.10.3

Are COA-IND secure the baby ciphers of the previous section?

Exercise 1.10.4

Are CPA-IND secure the baby ciphers of the previous section?

1.11 Appendixes:

1.11.1 More about Security

In order for cryptography to actually do anything, it has to be embedded in a protocol, written in a programming language, embedded in software, run on an operating system and computer attached to a network, and used by living people. All of those things add vulnerabilities and-more importantly-they're more conventionally balanced.

[Schneier15, Introduction, page xiv]

Figure 1.11.1: Security is hard.

1.11.2 Security Notions and Goals

Security Notions:

- ♣ Perfect Secrecy and One Time Pad (OTP).
- ♣ Computational Security.
- ♣ Provable Security.

Security Goals

I've informally defined the goal of security as "nothing can be learned about the cipher's behavior." To turn this idea into a rigorous mathematical definition, cryptographers define two main security goals that correspond to different ideas of what it means to learn something about a cipher's behavior:

Indistinguishability (IND) Ciphertexts should be indistinguishable from random strings. This is usually illustrated with this hypothetical game: if an attacker picks two plaintexts and then receives a ciphertext of one of the two (chosen at random), they shouldn't be able to tell which plaintext was encrypted, even by performing encryption queries with the two plaintexts (and decryption queries, if the model is CCA rather than CPA).

Non-malleability (NM) Given a ciphertext $C_1 = \mathbf{E}(K, P_1)$, it should be impossible to create another ciphertext, C_2, whose corresponding plaintext, P_2, is related to P_1 in a meaningful way (for example, to create a P_2 that is equal to $P_1 \oplus 1$ or to $P_1 \oplus X$ for some known value X). Surprisingly, the one-time pad is malleable: given a ciphertext $C_1 = P_1 \oplus K$, you can define $C_2 = C_1 \oplus 1$, which is a valid ciphertext of $P_2 = P_1 \oplus 1$ under the same key K. Oops, so much for our perfect cipher.

Next, I'll discuss these security goals in the context of different attack models.

[Aumasson18, Chapter 1]

1.11.3 Some pre computer ciphers

Classical ciphers or pre-computers cipher were constructed by using two ideas

Substitution and Permutations

Exercise 1.11.2

Check that $26! = 403291461126605635584000000$. So $\log_2(26!) \approx 88$.

Monoalphabetics: Caesar and ROT13 ciphers.

Exercise 1.11.3

Try to read the following ciphertext by using caesarCipher.py

zc mvif irggfikf kir gvejzvif vu vjjviv efe glf' vjjviv tyv hlvjkf: c'vjjviv v' zc jfxxvkkf, zc gvejzvif v' zc givuztrkf. zc gvejzvif ulehlv uvizmr urcc'vjjviv, dr efe c'vjjviv urc gvejzvif. c'vjjviv v' ur jv jkvjjf v gvi fgvir uz jv jkvjjf, c'vjjviv mzvev urkf jfckrekf gvi fgvir uvcc'vjjviv, c'vjjviv yr zc jlf wfeurdvekf ze jv jkvjjf, gvity jfckrekf c'vjjviv v' jvejf, irxzfev, evtvjjzkr', mvizkr', ze sivmv v' klkkf ze klkkf. c'vjjviv v', gvity zc efe vjjviv, tzfv' zc elccr, v' rjjliuf

cr mvizkr' efe v' rckif tyv cr kfkrczkr' uvccr mzkr v uvcc'vjjviv ldref . c'lfdf jzexfcf, tfejzuvirkf ze jv jkvjjf, efe irttyzluv c'vjjveqr uvcc'lfdf ze j, e ze hlrekf vjjviv dfircv, e ze hlrekf vjjviv gvejrekv. c'vjjveqr uvcc'lfdf v' tfekvelkr jfckrekf evccr tfdlezfev, evcc'lezkr' uvcc'lfdf tfe c'lfdf: vu v' krcv lezkr' tyv jz rggfxxzr jlccr ivrckr' uvccr uzwwviveqr kir c'zf v zc kl. cr jfczkluzev v' wzezkvqqr v czdzkrkvqqr; cr tfdlezfev v' czsvikr' v zewzezkluzev. c'lfdf tfejzuvirkf gvi jv jkvjjf v' lfdf evc jvejf rszklrcv uvccr grifcr; c'lfdf tfe c'lfdf, fjjzr c'lezkr' uvcc'zf v uvc kl, v' uzf . cr mvir uzrcvkkztr efe v' le dfefcfxf uvc gvejrkfiv jfczkrizf tfe jv jkvjjf, dr le uzrcfxf kir c'zf v zc kl

Sveqr givxzluzqzf uvccr uzxezkr' v uvccr rlkfefdzr kvfiztr, reqz ze gifwfeuf rttfiuf tfe vjjr, cr elfmr wzcfjfwzr yr vjjveqzrcdvekv ler kveuveqr girkztr evc gzl' rckf jvejf uvccr grifcr: vjjr jlsvekir rc gfjkf uvccr ivczxzfev, zdgcztr ze jv jkvjjr c'vjjveqr uvccr ivczxzfev, v' vjjr jkvjjr mvirdvekv ivczxzfev

Polyalphabetics:
Alberti's cipher
Macchina a rotori di Hebern

Vigenère Cipher
Hill's cipher.

Figure 1.11.4: Trithemius Tabula recta

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

1.11.5 Example of: Permutation table

1	18	11	5	22	28
25	15	8	2	19	12
6	23	29	26	16	9
3	20	13	7	24	30
27	17	10	4	21	14

The table gives a permutation of the set $\{1, 2, \cdots, 30\}$. Here in standard notation:

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & \cdots & 30 \\ 1 & 10 & 19 & 28 & 4 & 13 & 22 & 9 & \cdots & 24 \end{bmatrix}$$

Exercise 1.11.6

Complete:

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & \cdots \\ 1 & 10 & 19 & 28 & 4 & 13 & 22 & 9 & ? & ? & \cdots \end{bmatrix}$$

Exercise 1.11.7

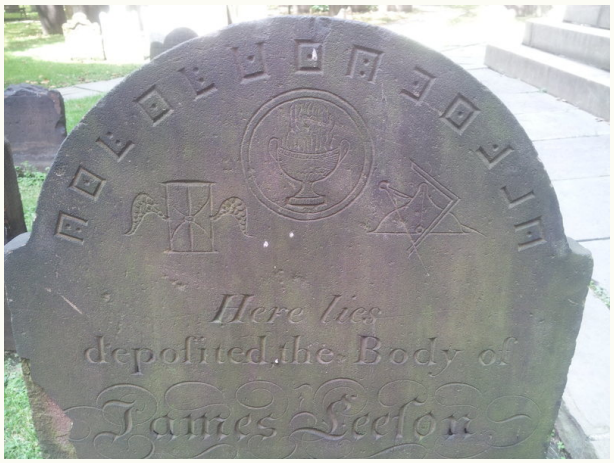
Study the "knight's tour cipher" [Gaines56, page 10].

Francis Bacon's cipher

1.11.4 Cryptanalysis of Classical Ciphers

Exercise 1.11.8

Deciphering An Ominous Cryptogram on a Manhattan Tomb (Trinity Churchyard in Lower Manhattan):



Hint: https://www.youtube.com/watch?v=bRVQ4vGOMfg?autoplay=1 or https://fontstruct.com/fontstructions/show/1544432/james-leeson-pigpen-cipher

Brute Force attack

Frequency letter Attack (Al-Kindi)

Kasiski/Babagge attack to Vigenère.

Kerckhoffs superimposition attack.

Exercise 1.11.9

Show that classical ciphers are insecure under KPA, CPA or CCA attacks.

Exercise 1.11.10

In [Sacco14, pagina 92] the italian General Luigi Sacco explains the cryptanalysis of a permutation cipher:

76. — Soluzione comune a tutti i sistemi di trasposizione.

Abbiamo detto al N. 66 che la base del lavoro di decrittazione dei sistemi di trasposizione è costituita da lettere a sequenze obbligate, che in ogni lingua sono ben note. Tali lettere, che abbiamo chiamato pilote, devono quindi essere subito rilevate nei crittogrammi di trasposizione: in loro mancanza le lettere che presentano le maggiori dissimetrie, nelle tavole dei bigrammi della lingua, potranno sostituirle.

Ciò fatto, una soluzione generale, indipendente cioè dal tipo di chiave adottata, semplice, doppia o di qualsiasi specie, viene suggerita dalla osservazione che, in qualunque sistema di trasposizione, la posizione che una determinata lettera del testo chiaro va ad occupare nel testo cifrato, dipende solo dal numero delle lettere del testo stesso.

In altre parole, se un testo chiaro comprende, ad es., 45 lettere, e, per effetto della trasposizione adottata, la prima lettera del testo si trova, a trasposizione ultimata, al 15° posto nel testo cifrato, qualunque altro testo di 45 lettere, cifrato con la stessa trasposizione, manderà la prima lettera del chiaro ad occupare il 15° posto nel testo cifrato; e così ogni altra lettera che occupa un determinato posto nel testo chiaro andrà sempre ad occupare un altro determinato posto nel crittogramma, purché questo sia sempre di 45 lettere.

Ne segue che quando si avessero vari crittogrammi dello stesso numero di lettere, ottenuti con la stessa chiave, scrivendoli in linee sovrapposte si avrà la certezza che la regola che permette di riordinare le lettere di una linea, riordinerà anche le corrispondenti lettere delle altre linee.

Per trovare questa regola si tratterà di esaminare nelle varie linee se esistano delle lettere pilote o a sequenze obbligate o quasi, cioè lettere che nei testi chiari della lingua considerata devono certamente, o quasi, trovarsi vicine. Se queste esistono si tratterà di verificare se la riunione di tali lettere in una linea dà luogo, nelle altre linee, a delle sequenze possibili nella lingua. Scartate le combinazioni impossibili o improbabili, si potranno trattenere quelle che si presentano più probabili e tentare di proseguirle per formare delle parole probabili, confermando ogni ipotesi fatta in una linea col risultato ottenuto nelle altre linee.

Supponiamo, ad es., di avere intercettato quattro radi oteie -grammi di 30 lettere ciascuno che, per avere la stessa data e provenienza, possiamo ritenere fatti con la stessa chiave. Essendo risultato dal computo delle frequenze che i quattro r. t. g. sono di semplice trasposizione, si dispongano in quattro linee sovrapposte, in modo da formare 30 colonne di quattro lettere come segue:

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
1°RTG	—	R	I	O	T	F	G	E	G	N	O	N	N	U	E	I	N	N	I	U	Q	T	O	E	S	Z	A	A	R	R	T
2°RTG	—	C	E	I	U	E	N	R	S	N	S	I	U	E	I	I	O	O	H	M	S	B	D	I	U	S	I	N	E	Z	O
3°RTG	—	C	E	T	L	B	A	R	T	F	I	M	U	T	O	F	U	U	A	T	O	L	I	T	A	E	T	Q	O	O	N
4°RTG	—	M	O	E	N	S	I	R	C	I	A	S	P	T	S	E	R	M	I	M	N	I	I	U	O	N	A	O	O	T	D

La lingua usata essendo l'italiana, si sa che, in questa, la lettera Q è sempre seguita da U, che la lettera H è quasi sempre preceduta da C, che la lettera Z precede quasi sempre I od A. Si dovrà dunque cercare se esistono di tali lettere nelle varie linee e vedere se la loro riunione produce sequenze possibili o meno nelle altre linee. Nella prima linea vi è una Q al 20° posto, e due U, una al 13° e l'altra al 19°: proviamo le due combinazioni 20-13 e 20-19:

20	13		20	19	
Q	U		Q	U	
S	E	(91)	S	M	(11)
O	T	(30)	O	T	(30)
N	T	(124)	N	M	(5)
		(245)			(46)

Le cifre segnate accanto ai bigrammi sono le rispettive frequenze medie, dedotte dalle tabelle dei bigrammi normali (tav. 2), cifre quindi proporzionali alle rispettive probabilità d'impiego nella lingua. Si vede

Can you find the key and read the four "radiotelegrammi" ?

1.11.5 Motivation

With the general expansion of electronic data processing in the 1960s, the discipline of cryptography experienced a sort of quantum leap. Modern hardware and software made it possible to implement complex, sophisticated mathematical algorithms that allowed previously unparalleled levels of security to be achieved. Moreover, this new technology was available to everyone, in contrast to the previous situation in which cryptography was a covert science in the private reserve of the military and secret services. With these modern cryptographic procedures, the strength of the security mechanisms in electronic data-processing systems could be mathematically calculated. It was no longer necessary to rely on a highly subjective assessment of conventional techniques, whose security essentially rests on the secrecy of the procedures used.

[WRWE03, Introduction, page 4]

Authentication, Integrity, and Nonrepudiation

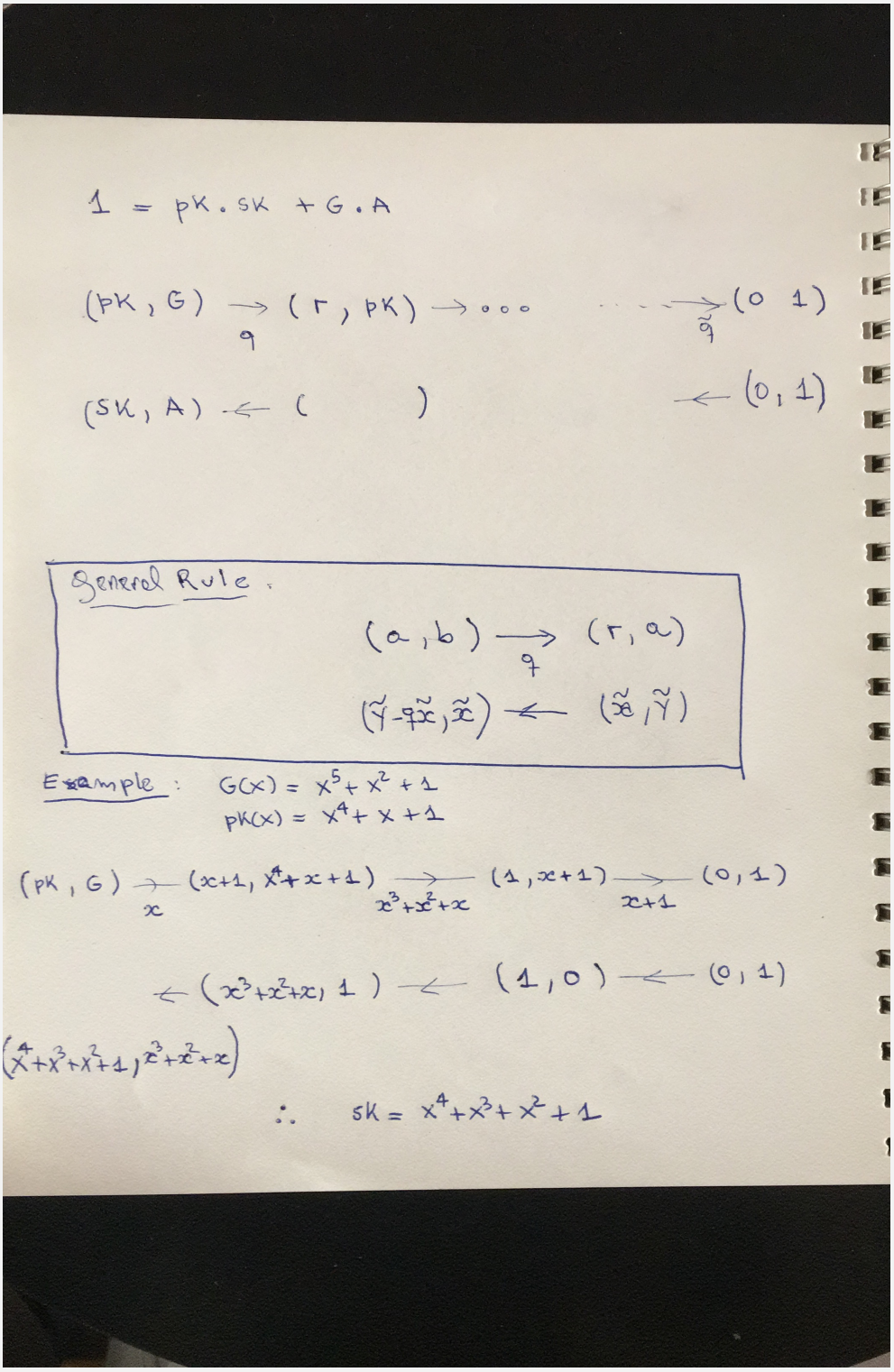
In addition to providing confidentiality, cryptography is often asked to do other jobs:

- **Authentication**. It should be possible for the receiver of a message to ascertain its origin; an intruder should not be able to masquerade as someone else.
- **Integrity**. It should be possible for the receiver of a message to verify that it has not been modified in transit; an intruder should not be able to substitute a false message for a legitimate one.
- **Nonrepudiation**. A sender should not be able to falsely deny later that he sent a message.

[Schneier15, page 2]

1.11.6 Euclid and Galois

1.11.11 Solution Exercise: 1.5.2



1.12 Bibliography

Books I used to prepare this note:

[Aumasson18] Jean-Philippe Aumasson, *Serious Cryptography: A Practical Introduction to Modern Encryption*, No Starch Press, 2018.

[Elia18] Michele Elia, *An Introduction to Classic Cryptography; With an exposition of the mathematics of private and public key ciphers*, Aracne, 2018.

[Gaines56] Gaines, Helen Fouchè , *Cryptanalysis - a study of ciphers and their solution*, Dover, 1956.

[KatLin15] Jonathan Katz; Yehuda Lindell, *Introduction to Modern Cryptography* Second Edition, Chapman & Hall/CRC, Taylor & Francis Group, 2015.

[Paar10] Paar, Christof, Pelzl, Jan, *Understanding Cryptography, A Textbook for Students and Practitioners*, Springer-Verlag, 2010.

[Sacco14] Gen. Luigi Sacco *Manuale di Crittografia*, Quarta edizione ampliata a cura di Paolo Bonavoglia. youcanprint, (2014), http://luigi.sacco.crittologia.eu/manuale_crittografia.html.

[Schneier15] Bruce Schneier, *Applied Cryptography: Protocols, Algorithms and Source Code in C* ,Wiley; 20th Anniversary edition,2015.

[Sweigart13] Al Sweigart, *Hacking Secret Ciphers with Python*, http://inventwithpython.com/hackingciphers.pdf

[WRWE03] Wolfgang Rankl, Wolfgang Effing, *Smart Card Handbook, Third Edition*,Wiley, 2003.

Here a list of papers:

[DH76] Diffie, W.; Hellman, M. *New directions in cryptography*, (1976). IEEE Transactions on Information Theory. 22 (6): 644-654.

[DH79] Diffie, W.; Hellman, M. *Privacy and Authentication: An Introduction to Cryptography*, (1979). Proceedings of the IEEE, vol. 67, no.3, March 1979: 397-427.

[Friedman20] William F. Friedman, *The Index of Coincidence and Its Applications in Cryptography*, Riverbank Publication No. 22, Riverbank Labs, 1920. Reprinted by Aegean ParkPress, 1987.

[GM82] Goldwasser, S. and Micali, S.; *Probabilistic Encryption & How To Play Mental Poker Keeping Secret All Partial Information*, Proc. 14th Symposium on Theory of Computing: 365-377. (1982)

[O'Donnell02] O'Donnell, M. *Identification Protocols in Cryptography*, The ITB Journal, Vol. 3, Issue 1, Article 3, pages 12-47, (2002). https://arrow.dit.ie/cgi/viewcontent.cgi?article=1031&context=itbj

[Shannon49] C.E. Shannon, *Communication Theory of Secrecy Systems*, Bell System Technical Journal, vol. 28 -4. October 1949 pp 656-715.

and some interesting links:

Paar's Lectures

https://people.csail.mit.edu/rivest/Rivest-Cryptography.pdf

http://people.csail.mit.edu/rivest/crypto-security.html#Books

https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=2ahUKEwjpqsfO7_ffAhXgShUIHXOkB_UQFjAAegQIChAC&url=http%3A%2F%2Fcgi.di.uoa.gr%2F~secprot%2Fnotes%2F04-cryptography.doc&usg=AOvVaw0abs9ZVwsXStvcU7ogCrFs

Arne Beurling

Karl Stein

Museo by Klaus Pommerening

Black Chambers

La Crittografia da Atbash a RSA

What are the eras of cryptography?

ASIC