

A Practical Lecture on IT Security

Jean-François COUCHOT

`couchot [at] femto-st [dot] fr`

September 27, 2021

Contents

1	Introduction to Security	2
2	Integrity and Hash Functions	3
2.1	Using Hash Function for Data Integrity	3
2.2	Cryptographic Hash Function	4
2.3	Resistance to collisions	5
2.4	Pre-image Resistance of MD5 and of SHA1	6
2.5	Conclusion	6
3	Confidentiality and Cryptography	7
3.1	Introduction	7
3.2	Symmetric encryption: AES	8
3.2.1	Operations of AES	8
3.2.2	Obtaining 128-bit blocks	9
3.3	Asymmetric Encryption: RSA	10
3.3.1	Arithmetic Reminders	10
3.3.2	Greatest common divisor	10
3.3.3	Euclidean Algorithm	10
3.3.4	RSA algorithm	12
3.3.5	The key points of the algorithm	12
3.4	Digital Signature by asymmetric encryption	13
3.4.1	General Principle	13
4	Authenticity and Watermarking	15
4.1	Introduction	15
4.2	Applications of watermarking	15
4.2.1	Proof of media ownership	15
4.2.2	Copy control	15
4.2.3	Content integrity	16
4.3	Implementing a robust image watermarking scheme	16
4.3.1	Naive watermarking by LSB replacement	16
4.3.2	Quantization based Watermarking	16
4.4	Reversible watermarking	17
5	Data Privacy	18
5.1	From a meaningful wish to law	18
5.2	Pseudonymization as a first coarse idea	18
5.3	k -anonymity	19
5.4	l -diversity	20

Chapter 1

Introduction to Security

DEFINITION 1.1. Security of systems concerns the whole set of technical, organisational, juridic and human based ways that are used to keep, to establish to guarantee that approaches aiming at collecting, saving, treating and publishing data are safe.

Particularly, it is interesting to preserve the following properties: *integrity, confidentiality, availability, authenticity, tracking, privacy*, ... of information are the core of this field. This chapter is inspired from [AJOP15, Ghe13] and is a an short introuction into data security.

Integrity The integrity of an information is established if one can prove that it has not been modified, intentionally, or not. For instance, during a file exchange, it is essential that the received file is the one that has been sent: there is no difference between them. *Hash functions* (detailed in chapter 2) or *message authentication code* (MAC) (precised in chapter 3) are used in such an objective.

Confidentiality In information security, *confidentiality* is the property, that information is not made available or disclosed to unauthorized individuals, entities, or processes. Such a property may be governed by laws in some cases: medical records, banking data, personal data. . . It is often the first property we have in mind when we think about security. Tools that allow to obtain such a property are cryptosystems. Examples of such tools are detailed in chapter 3.

Availability System availability is evaluated thanks to a metric that expresses the percent of time during the system is able to exactly do the awaited task. This is usually measured in terms of nines: for instance, five-9's (99.999%) means less than 5 minutes when the system is not operating correctly over the span of one year. We won't study this property.

Authenticity Authenticity is assurance that a message, transaction, or other exchange of information is from the source it claims to be from. Authenticity involves proof of identity of the origin. For instance, one can add an indelible mark inside the information. This method further denoted as *watermarking* is presented in chapter 4.

Authentication is a particular property of authenticity: in this case, one have to ensure that identifiers are authentic. This property is often proven thanks to digital signature which are based on cryptosystems. This approach is presented in chapter 3.

Data Privacy Certain data, typically medical, location, political orientations, etc. are in the private domain and should not be accessible, and even less published, without the consent of their owners. That's the privacy protection. When consent is obtained, the owners of these data must also be guaranteed that they will not be easily identifiable following the publication of summaries on them. This approach is presented in chapter 5.

Exercise 1.1. *You have to work on a project aiming at storing and further working on medical items in a database. Such database should be accessible to medical staff of an hospital. Can you explain which security concepts are linked to this project?*

Chapter 2

Integrity and Hash Functions

In broad terms, data integrity refers to the state of data which, when processed, stored or transmitted, are not altered or destroyed, whether intentionally or accidentally, and are in a suitable format for use. A hash function is a special function, which calculates a digital fingerprint from an input data to quickly identify the initial data, just like a signature to identify a person. Hash functions are used in computer science and cryptography to quickly recognize files or passwords, for example.

In this chapter we will intuitively introduce what a hash function is (Section 2.1). We will then describe cryptographically secure hash functions (section 2.2). Finally, we will show on examples why MD5 and SHA1 are not (section 2.4).

In its french version this chapter is inspired from [Ver15, TIP].

2.1 Using Hash Function for Data Integrity

A hash function H is an algorithm aiming at characterizing an information. From each input data m , it produces a digest $H(m)$ which allows to identify this data m . Moreover, the approach is reproducible. What follows gives an example of executing the sha256 hash function on two close messages.

```
$ python xplhash.py
Please give your text 1: ISIFC security
31dfb5202e384fa22709a1435be4819509bfcd4089d2989f8cf1898cef0ce0a5
Please give your text 2: ISIFC seurity
1fb6bd2385da6b21578dcb459119491ec3c540bf5c4d145914f2aaf1bd18f7f
```

We first can see that the digest has fixed length: (64 hexadecimal numbers, *i.e.*, $64 * 4 = 256$ bits). Next, it is obvious to see that a small modification of message (“c” is replace by “b”) produces large variations. Such a function is relevant when data integrity is necessary.

Practical Work 2.1. *The hashlib python module contains a family of hash functions: md5, sha1, sha224, sha256, sha384 and sha512.*

1. Explain the following code

```
# coding: utf-8
import hashlib
txt1 = input('Please give your text 1: ')
hash_object_1 = hashlib.sha256(txt1.encode('utf-8'))
hex_dig_1 = hash_object_1.hexdigest()
print(hex_dig_1)
```

2. For each hash method of the hashlib module, calculate the fingerprint of the message:

```
C'est un trou de verdure où chante une rivière,
Accrochant follement aux herbes des haillons
D'argent ; où le soleil, de la montagne fière,
Luit : c'est un petit val qui mousse de rayons.
```

3. Develop the program that:

(a) Asks the user to enter a text;

- (b) Memorizes the fingerprint of this text by using sha256;
- (c) Asks the user to re-enter his/her text;
- (d) If the fingerprints are identical, the program congratulates the user; in the opposite case it outputs a message stating that 2 texts are not the same.

Practical Work 2.2. Develop the program that

1. asks the user to enter a password until the user enters the word "end".
2. for each entered password,
 - (a) calculates its `script` fingerprint
 - (b) saves this fingerprint on a new line of a file (`myscriptpsswords.txt`) which, in the end, will therefore contains one line per fingerprint.

In what follows, we consider the following passwords:

```
Security
SecuriTy
security00
security
isifc
easyFC
1slF(
```

2.2 Cryptographic Hash Function

A cryptographic hash function H has to ensure the following constraints:

- *Avalanche effect*: when an input is slightly changed (for example, flipping a single bit) the output changes significantly; in the previous example, you can notice that “c” is changed to “b”. Only one bit is changed. And the result is completely different.
- *Pre-image resistance*. Given a hash value h , it should be very difficult to find any message m such that would lead such a digest (i.e., $h = H(m)$);
- *Second pre-image resistance*. Given an input m_1 , it should be difficult to find different input m_2 that would have the same digest than m_1 (i.e., s. t. $H(m_1) = H(m_2)$). If this property is not established, it is possible to generate a false data that would be valid for the user.
- *Collision resistance*. It should be difficult to find two different messages m_1 and m_2 such that $H(m_1) = H(m_2)$. Such a pair is called a cryptographic hash collision.

Exercise 2.1. Congruence modulo 1024. Consider the hash function which, given a number, returns its remainder in the division by 1024.

1. Formalize this function $H : \{0, 1\}^* \rightarrow \{0, 1\}^{10}$.
2. Compute the fingerprints of 156; 1224; 10240.
3. Is it collision resistant? Justify.
4. Is it resistant to the first preimage ? Justify.
5. Is it resistant to the second pre-image? Justify.

2.3 Resistance to collisions

One can show that the resistance to collision does not imply resistance to the pre-image.

Example 2.1. Let $g : \{0, 1\}^* \rightarrow \{0, 1\}^n$ be a function which is resistant to collisions. We build $h : \{0, 1\}^* \rightarrow \{0, 1\}^{n+1}$ defined by

$$h(x) = \begin{cases} 0x & \text{if } x \text{ is of length } n. \\ 1g(x) & \text{otherwise} \end{cases}$$

Let us show that h is resistant to collisions. Let x and x' two sequences of bits such as $h(x) = h(x')$. The two strings are not of size n : in this case we would have $0x = 0x'$, i.e., $x = x'$. Thus if $h(x) = h(x')$, then $g(x) = g(x')$ and x would collide with x' for g , which is resistant by assumption.

Show that h is not resistant to the pre-image. If the fingerprint is $0h_1h_2 \dots h_n$ (i.e., starts with 0), we know immediately how to find a data message which admits such a mark: $h_1h_2 \dots h_n$.

We know today that the MD5 hash function is not collision resistant.

When trying to achieve a collision, arises the question of how many fingerprints it is sufficient to calculate to detect a collision. If the fingerprint is a word of n bits, taking $2^n + 1$ different data, will necessarily lead to a collision. However, this is not practically possible to perform if n is large: for example, let $n = 64$ bits, it would take $2^{64} + 1$ operations knowing that since the creation of the universe, about 2^{70} ms have passed.

The next section shows how to reduce the number of experiments to perform.

PROPOSITION 2.1 (BIRTHDAY ATTACK). If k different data are selected in $\{0, 1\}^*$, with

$$k \geq \frac{1 + \sqrt{1 + 2^{n+3} \ln 2}}{2} \quad (2.1)$$

then the probability that two fingerprints are equal is superior to $\frac{1}{2}$.

It is assumed that there are m possible fingerprints. In the case of a fingerprint of n bits, it leads to no more than $m = 2^n$. Let p_k the probability that two data among k yield the same fingerprint. Let $q_k = 1 - p_k$ the probability that the k data generate all a different fingerprint.

For:

- $q_1 = 1$;
- $q_2 = 1(1 - \frac{1}{m})$;
- $q_3 = 1(1 - \frac{1}{m})(1 - \frac{2}{m})$;
- ...
- $q_k = \prod_{i=1}^{k-1} (1 - \frac{i}{m})$

Since $1 + x \leq e^x$ for each real number x . The previous formula gives

$$q_k \leq \prod_{i=1}^{k-1} e^{-\frac{i}{m}} = e^{-\sum_{i=1}^{k-1} \frac{i}{m}} = e^{-\frac{k(k-1)}{2m}}.$$

Thus, we successively have

$$\begin{aligned} 1 - e^{-\frac{k(k-1)}{2m}} &\geq \frac{1}{2} \\ \Leftrightarrow e^{-\frac{k(k-1)}{2m}} &\leq \frac{1}{2} \\ \Leftrightarrow -\frac{k(k-1)}{2m} &\leq -\ln 2 \\ \Leftrightarrow k^2 - k - 2m \ln 2 &\geq 0 \\ \Leftrightarrow k &\geq \frac{1 + \sqrt{1 + 8m \ln 2}}{2} \\ \Leftrightarrow k &\geq f(n) = \frac{1 + \sqrt{1 + 2^{n+3} \ln 2}}{2} \end{aligned}$$

For a given number n of bits, if we generate all the words of length $\log_2(f(n))$ bits, the probability to obtain a collision is larger than 1/2. The following array gives the value of $\log_2(f(n))$ for some n .

n	50	100	150	200
$\log_2(f(n))$	25, 24	50, 24	75, 24	100, 24

We can conclude that if we build $2^{\frac{n}{2}}$ fingerprints, we have about one chance over two to find a collision.

2.4 Pre-image Resistance of MD5 and of SHA1

hashcat¹ is a tool allowing to crack lot of MD5/SHA1 fingerprint.

Exemple of usage:

```
hashcat -m 0 -a 0 example0.hash example.dict -r d3ad0ne.rule --show
```

On the collection of fingerprints stored into `example0.hash`, the approach uses a dictionary (`-a 0`), namely `example.dict` and some concatenations, upercase/lowercase rules specified by `-r d3ad0ne.rule`, suposing the hash function is MD5 (`-m 0`).

```
hashcat -m 0 -a 0 mymd5psswords.txt example.dict -r best64.rule
hashcat (v4.0.1) starting...
```

```
e91e6348157868de9dd8b25c81aebfb9:security
2fae32629d4ef4fc6341f1751b405e45:Security
91416a1d7c091eb634bc398f5d9c4a26:security00
```

```
hashcat -m 0 -a 0 mymd5psswords.txt example.dict -r d3ad0ne.rule
hashcat (v4.0.1) starting...
```

```
9cd8762be7dead2bded747dea91e602e:easyFC
4f65be536206f10c8f33a8a3ed1e0dfd:SecuriTy
```

On the collection of fingerprints stored into `mymd5psswords.txt` produced aboved, the approach uses a dictionary and rules to find message corresponding to these fingerprint. 5 over the 7 passwords are founds in a couple of seconds.

```
hashcat -m 0 -a 3 hashMD5.txt
```

On the collection of fingerprints stored into `hashMD5.txt`, the approach uses a brute force approach to find message corresponding to these fingerprint, even if there are no in the dictionary.

```
hashcat -m 0 -a 3 mymd5psswords.txt
hashcat (v4.0.1) starting...
```

```
769201f42547567f8f4da4383b2cf783:isifc
```

One password is found by brute force in less than 6 seconds.

2.5 Conclusion

Today, it is forbidden to use either MD5 or SHA1. In order to check integrity, hash functions that can be used are:

- SHA2 (224, 256, 384)
- SHA3

In order to memorise passwords, hash functions that can be used are:

- scrypt
- argon2

¹<https://hashcat.net/hashcat/>

Chapter 3

Confidentiality and Cryptography

In its french version, this document is mainly inspired from [RSA09, Ver15]. The presentation of the digital signature principle is principally extracted https://en.wikipedia.org/wiki/Digital_signature.

3.1 Introduction

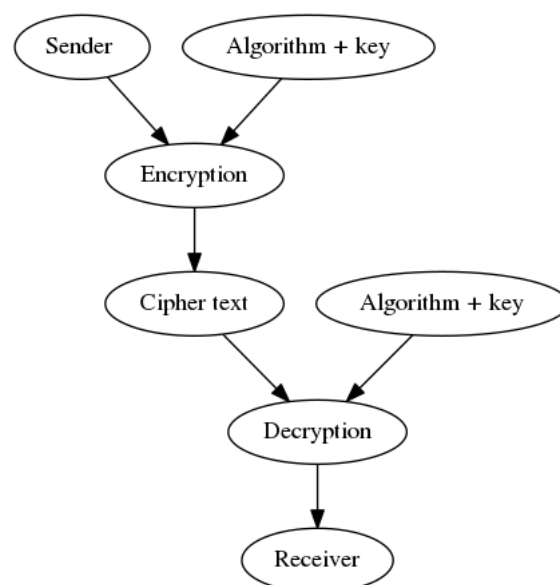


Figure 3.1: General outline of a method of encryption/decryption

Cryptography, which is the art of writing with a key, appeared at the same time as writing. Once information must be transmitted in a safe manner, the message must be protected from interception: it is encrypted by the transmitter and decrypted by the receiver. In the case where one uses an encryption key, we have the diagram shown in Figure 3.1. However in this figure, nothing indicates that the encryption key is the same as that of decryption.

A method that relies on a single key to encrypt and to decrypt a message is called *symmetric cryptography*. The problem of confidentiality of the key and the implementation of this property arises immediately when the number of recipients is large: one key is required for each recipient. To solve this key exchange problem, *asymmetric cryptography* has been developed over the years 1970. It is based on the principle of a public key that is broadcast and a private key that is guarded by the initiator of the exchange. However, if this solves the key exchange problem, it has a cost: slowness. Also, this type of cryptography is mainly used for the exchange of keys and electronic signature. TLS, a protocol used by browsers whenever a connection must be secured, exploits the two families of cryptography, symmetric and asymmetric.

3.2 Symmetric encryption: AES

Advanced Encryption Standard (AES) was born in 2001. It is the most symmetric cryptographic algorithm used and the safest today. It is a 128-bit block encryption algorithm. The keys may be variable size (128, 192 or 256 bits).

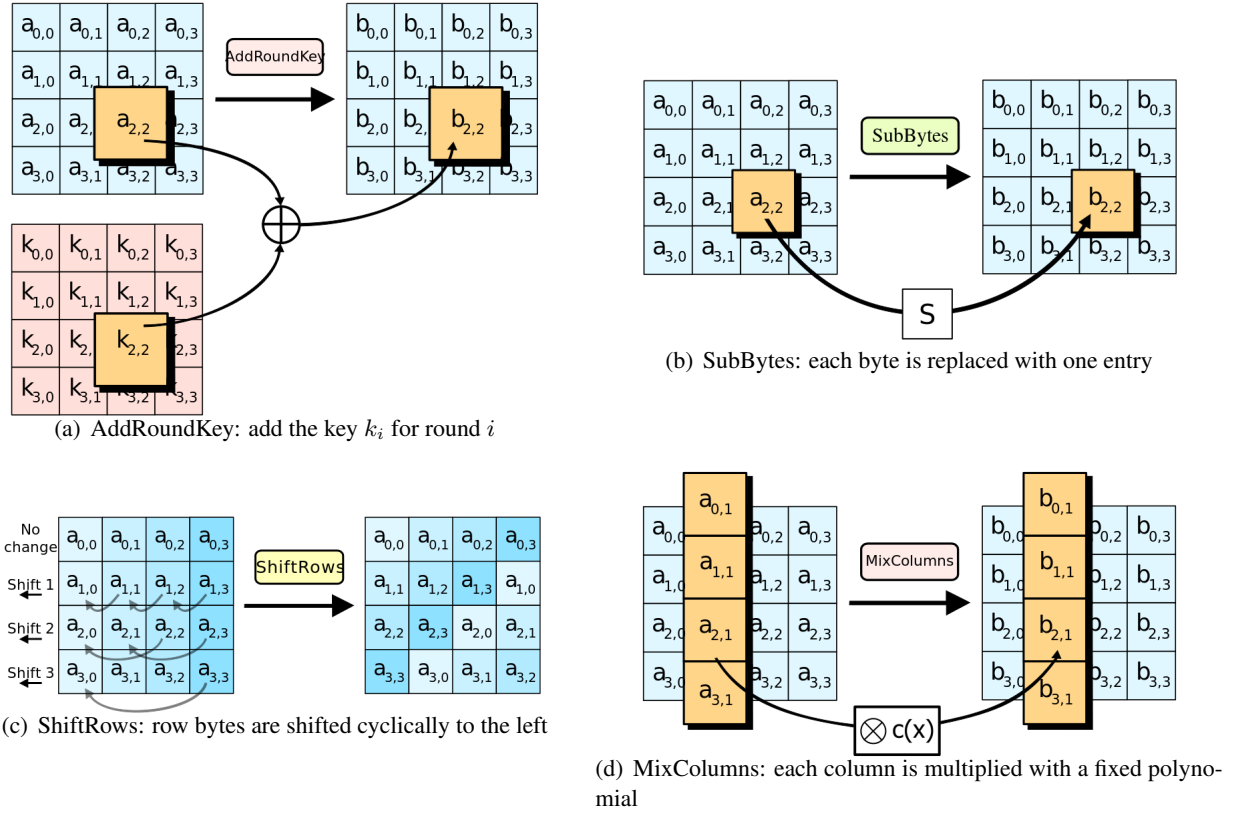


Figure 3.2: High-level description of AES (Wikipedia)

3.2.1 Operations of AES

In the following, let be given a 128-bit key k and we have to encrypt a message block that has 128 bits as well. Coded in bytes, each block contains 16 elements organized in the form of a 4×4 array. The encryption algorithm AES therefore takes as input two arrays a and k of size 4×4 and has mainly 4 major steps:

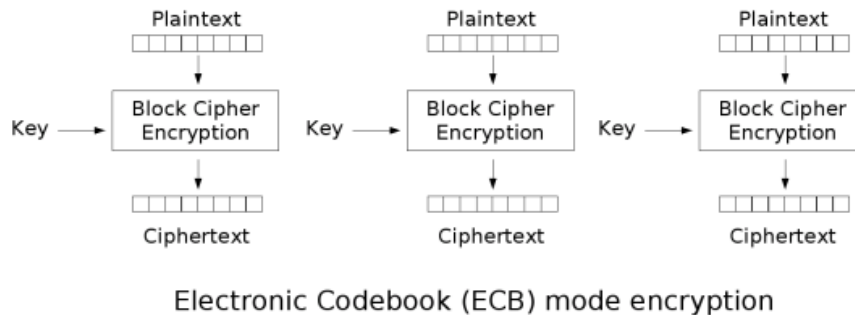
1. Generation of the 11 keys k^0, k^1, \dots, k^{10} from k . Each of them is a 4×4 size array of bytes.
2. Initial stage. For each line i and column j , $0 \leq i, j \leq 3$, a exclusive OR (XOR) with elements $a_{i,j}$ and $k_{i,j}^0$ is executed. This step is denoted as *AddRoundKey* and is shown in Figure 3.2(a).
3. A set of 9 repetitions. For $t = 1, \dots, 9, \dots$
 - (a) *Substitution of bytes* (SubBytes) by applying a function S to each of the 16 array elements. This is shown in Figure 3.2(b).
 - (b) *Line shift* (ShiftRows) to the left. The offset is 3 for the lowest line, 2 for the one above and 1 for the second line. This is shown in Figure 3.2(c).
 - (c) *Mixing Columns* (MixColumns) by applying a linear transformation on each column. This is shown in Figure 3.2(d).
 - (d) *Adding round key* (already shown in Figure 3.2(a)), but with the key k^t here.
4. The last repetition is the same except that it does not contain the mixture of the columns. It is therefore made of :
 - (a) Substitution of bytes
 - (b) Line Offset

(c) Add round key

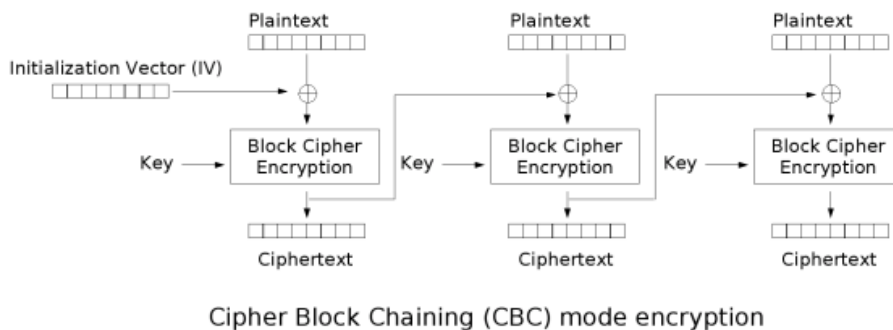
The decryption is similar but takes exactly the inverse functions.

3.2.2 Obtaining 128-bit blocks

It remains to propose methods for obtaining blocks of 128 bits. Two are presented: ECB and CBC.



(a) Electronic codebook (ECB)



(b) Cipher Block Chaining (CBC)

Figure 3.3: Block cipher mode of operation (Wikipedia)

Electronic codebook (ECB). In this procedure, the message is simply split into several blocks of 128 bits which are encrypted separately one after the other. This is represented at Figure 3.3(a). Two blocks with the same content will be encrypted in the same way! An attacker can therefore deduce information from the encrypted text by searching for identical sequences and obtaining a "codebook" establishing the correspondence between the plain text and the encrypted message.

Cipher Block Chaining (CBC). In this procedure, each block executes an "exclusive OR" with encryption of the previous block before it is itself encrypted (see Figure 3.3(b)). The process is initialized with an Initialization Vector (IV).

Exercise 3.1. 1. Explain the following program.

2. Explain differences between Parts 1 and 2.

3. Run the program.

```
from Crypto.Cipher import AES

message = "The answer is 1.The answer is 2.The answer is 1.".encode("utf-8")

#Part 1
aes_ecb_e = AES.new('This is a key123'.encode("utf-8"), AES.MODE_ECB)
ciphertext_aes_ecb = aes_ecb_e.encrypt(message)
```

```

print("ciphertext with ECB: " + str([x for x in ciphertext_aes_ecb]))

aes_ecb_d = AES.new('This is a key123'.encode("utf-8"), AES.MODE_ECB)
res=aes_ecb_d.decrypt(ciphertext_aes_ecb)
print(res.decode('utf-8'))

#Part 2
aes_cbc_e = AES.new('This is a key123'.encode("utf-8"), AES.MODE_CBC, 'This is an IV456'.encode("utf-8"))
ciphertext_aes_cbc = aes_cbc_e.encrypt(message)
print("ciphertext with CBC: " + str([x for x in ciphertext_aes_cbc]))

aes_cbc_d = AES.new('This is a key123'.encode("utf-8"), AES.MODE_CBC, 'This is an IV456'.encode("utf-8"))
res=aes_cbc_d.decrypt(ciphertext_aes_cbc)
print(res.decode('utf-8'))

```

Practical Work 3.1. 1. *Encrypt and thus decrypt a string of your choice with AES and ECB, with CBC.*

2. *Encrypt an image with AES and ECB. Display the encrypted image. What do you notice? You can use the following example code to manipulate images:*

```

from Crypto.Cipher import AES
from PIL import Image

im = Image.open("imgISIFC.png")
#im.show()
message = im.tobytes()
aes_ecb_e = AES.new('This is a key123'.encode("utf-8"), AES.MODE_ECB)
ciphertext_aes_ecb = aes_ecb_e.encrypt(message)

#output the encrypted image
imb= Image.frombytes(im.mode,im.size,ciphertext_aes_ecb)
imb.show()

```

3. *Encrypt an image with AES and CBC. Display the encrypted image. What do you notice?*

3.3 Asymetric Encryption: RSA

First, some elements of arithmetic are recalled.

3.3.1 Arithmetic Reminders

Given two integers a and b in \mathbb{Z} . We say that a divides b (which is denoted $a|b$) if there is an integer $q \in \mathbb{Z}$ such that $b = aq$.

3.3.2 Greatest common divisor

The greatest common divisor (GCD) of a and b denoted as $\gcd(a, b)$ is the natural number that verifies:

- $\gcd(a, b)|a$ and $\gcd(a, b)|b$;
- If $d|a$ and $d|b$, thus $d|\gcd(a, b)$.

Exercise 3.2. *Compute $\gcd(550, 1540)$.*

3.3.3 Euclide Algorithm

By definition, the GCD of a non-zero a with 0 is a (reasonable definition, because 0 is divisible by any non-zero number a , which is also divisible by a). Finally the GCD of 0 and 0 is not defined. Here we limit ourselves to the case of two integers a and b strictly positive. For example, assume $a > b$.

1. The Euclidean division of a by b can be written as $a = bq + r$ with $0 \leq r < b$.
2. Let us show that “ d is a common divisor of a and b ” is equivalent to “ d is a common divisor of b and r ”.
 - Let b be of a common divisor of a and b , which can be written $a = da'$ and $b = db'$. The equality $a = bq + r$ thus becomes $da' = db'q + r$ or $r = d(a' - b'q)$. Also d is a common divisor of b and r .

- Conversely, let d be a common divisor of b and r , which may be written as $b = db'$ and $r = dr'$. Equality $a = bq + r$ becomes $a = d(b'q + r')$. So d is a common divisor of a and b .

Thus, the sets of common divisors of a and b on the one hand and b and r on the other hand are identical. Particularly, $\gcd(a, b) = \gcd(b, r)$.

3. If $r = 0$ we have $\gcd(a, b) = \gcd(b, 0)$ which is equal to b .
4. Otherwise, r is different from 0, and one can therefore perform the Euclidean division of b by r , which gives a remainder r_1 , $0 \leq r_1 < r$ and $\gcd(b, r) = \gcd(r, r_1)$.
5. This algorithm is iterated until a null remainder is obtained, which necessarily occurs because we consider natural numbers and that the sequence of constructed remainders is strictly decreasing. The GCD is then the pre-last remainder (the last non-zero).

Exercise 3.3. Determine $\gcd(154, 35)$ by the Euclidean algorithm.

Exercise 3.4. Give the code of a program that takes two integers a and b s.t. $a > b \geq 0$ and returns $\gcd(a, b)$.

PROPOSITION 3.1 (BEZOUT IDENTITY). Consider two positive integers a and b . It exists a pair of integers x and y such that $ax + by = d$, where $d = \gcd(a, b)$.

PROOF. In the proof of the preceding proposition, we had successively:

$$a = b \times q_1 + r_1 \quad (3.1)$$

$$b = r_1 \times q_2 + r_2$$

$$r_1 = r_2 \times q_3 + r_3$$

$$\vdots$$

$$r_{n-4} = r_{n-3} \times q_{n-2} + r_{n-2} \quad (3.2)$$

$$r_{n-3} = r_{n-2} \times q_{n-1} + r_{n-1} \quad (3.3)$$

$$r_{n-2} = r_{n-1} \times q_n + r_n \quad (3.4)$$

$$r_{n-1} = r_n \times q_{n+1} + 0$$

We know that $\gcd(a, b)$ is r_n , i.e. the last non-zero remainder. We go up the equations one by one starting from Eq. (3.4).

$$\begin{aligned} r_n &= r_{n-2} - r_{n-1} \times q_n \\ &= r_{n-2} - (r_{n-3} - r_{n-2} \times q_{n-1}) \times q_n \quad (r_{n-1} \text{ is replaced by its expression derived from (3.3)}) \\ &= r_{n-2} \cdot (1 + q_{n-1} \cdot q_n) - r_{n-3} \cdot q_n \quad (\text{factoring}) \\ &= (r_{n-4} - r_{n-3} \times q_{n-2}) \cdot (1 + q_{n-1} \cdot q_n) - r_{n-3} \cdot q_n \quad (r_{n-2} \text{ is replaced by its expression derived from (3.2)}) \\ &\vdots \\ &= \dots \quad (r_1 \text{ is replaced by its expression derived from (3.1)}) \\ &= ax + by \end{aligned}$$

Exercise 3.5. Show that there exist x and y such that $29x + 72y = 1$ then find a value for x and y .

DEFINITION 3.1 (COPRIME INTEGERS). Two integers a and b are said to be coprime if the only positive integer that divides both of them is 1, i.e., $\gcd(a, b) = 1$.

Exercise 3.6. Show that 55 and 21 are coprime.

3.3.4 RSA algorithm

In an asymmetric cryptographic system, two keys are generated: the one that is public and the one that is private. The public key is broadcast while the one that is private is not.

When transmitting information from a transmitter to a receiver, the receiver generates the keys and disseminates the public key. Each transmitter can encrypt its message and only the receiver will be able to decrypt it.

When it comes to authenticating a sender, it is the sender who generates the keys and who broadcasts the public key. Each receiver will be able to decipher the signature and will thus have the guarantee that it is authentic.

In the following, we place ourselves in the first frame, *i.e.*, the receiver generates the keys to obtain data of anyone.

Step 1: choosing two prime numbers p and q . The receiver chooses two large prime numbers p and q and computes $n = pq$. Then he/she calculates $\varphi(n)$, where $\varphi : \mathbb{N}^* \rightarrow \mathbb{N}^*$ is the *Euler function*. that is the number of integers in the set $\{1, 2, \dots, n - 1\}$ that are coprime with n .

Exercise 3.7. *The recipient chooses $p = 7$, $q = 13$. Construct the set of integers that are coprime with $n = pq$ and deduce that $\varphi(91) = 72$.*

Sep 2: choice of the public key A natural number $e \in \{1, \dots, \varphi(n) - 1\}$ coprime with $\varphi(n)$ is selected. The public key is the pair (e, n) . Each sender will use it to encrypt his/her message to the receiver. Encryption is detailed in the fourth step below.

Exercise 3.8. *Show that $(29, 91)$ is a acceptable public key candidate.*

Step 3: Build the private key. The receiver calculates the integer $d \in \{1, \dots, \varphi(n) - 1\}$ such that the remainder in the Euclidean division of ed by $\varphi(n)$ is 1. This is also denoted as $ed \equiv 1[\varphi(n)]$. The pair (d, n) is the private decryption key. It is secret and allows the receiver to decrypt all received messages that are encrypted with (e, n) .

Exercise 3.9. *Find the associated private key.*

Step 4: message encryption. The sender may encrypt any written message in the form of a number m belonging to $\{1, \dots, n - 1\}$ and which is coprime with n . The coded message is the remainder a of dividing by n of m^e . We thus have $m^e \equiv a[n]$, where $a \in \{1, \dots, n - 1\}$.

Exercise 3.10. 1. *Show that the sender can encrypt the message $m = 59$.*

2. *Build the encrypted message a using the public key.*

Step 5: decryption of the message. The receiver has a and its private key (d, n) . To decrypt, he/she calculates the remainder when divided by n of a^d (*i.e.*, $a^d[n]$). If no calculation error has been made, it is the m initial message.

Exercise 3.11. *Decrypt the message using the private key.*

3.3.5 The key points of the algorithm

The RSA algorithm relies on several key points encountered successively:

- the generation of two large prime numbers p and q ;
- modular arithmetic;
- the Euclidean algorithm of generation of GCD and its corollary of Bézout;
- factorization, which as long as it is not feasible on large numbers, guarantees the security of the encryption process.

Exercise 3.12. 1. *Explain the following program.*

2. *With your neighbor:*

(a) *generate the public and private keys each;*

- (b) *transmit to him/her your public key;*
- (c) *encrypt a message with his/her public key and transmit the encrypted message;*
- (d) *decrypt his/her message.*

```

from Crypto.PublicKey import RSA
from Crypto.Cipher import PKCS1_OAEP

# 1
rsa_obj = RSA.generate(2048)
rsa_pub = rsa_obj.publickey()

#2
f = open('mPubKey.pem', 'w')
st = rsa_pub.exportKey('PEM')
f.write(st.decode('utf-8'))
f.close()

#3
f = open('mPrivKey.pem', 'w')
st = rsa_obj.exportKey('PEM')
f.write(st.decode('utf-8'))
f.close()

#4
pubKey = RSA.importKey(open('mPubKey.pem').read())
encryptor = PKCS1_OAEP.new(pubKey)
message_to_encrypt = 'To be encrypted'
ciphertext = encryptor.encrypt(message_to_encrypt.encode('utf-8'))

#5
privKey = RSA.importKey(open('mPrivKey.pem').read())
decryptor = PKCS1_OAEP.new(privKey)
decrypted_message = decryptor.decrypt(ciphertext)

print(decrypted_message.decode('utf-8'))

```

3.4 Digital Signature by asymmetric encryption

Digital signature (or electronic signature) is a mechanism for ensuring the *integrity* of an electronic document and to *authenticate* the author, by analogy with the handwritten signature of a paper document. It combines the fingerprint of the document by hash function and asymmetric encryption of the document to guarantee respectively these two properties.

3.4.1 General Principle

Suppose Alice wants to digitally sign a message m .

Setting up.

1. She chooses
 - An asymmetric encryption algorithm (RSA for example), consisting of an encryption function C and a decryption function D ;
 - A hash function H (SHA2 for example).

and informs all persons concerned of her choice.

2. She generates a private key K_{pr} and public key K_{pu} for asymmetric encryption algorithm.
3. She distributes the public key K_{pu} through an unsecured channel and keeps the secrecy of K_{pr} .

Generating the signed message.

1. She produces fingerprint $H(m)$ of the message m with the selected hash function H .
2. She encrypts this fingerprint with encryption function C using her private key K_{pr} . The digital signature of this message is $S = C(K_{pr}, H(m))$.
3. The signed message is the pair (S, m) that can be transmitted.

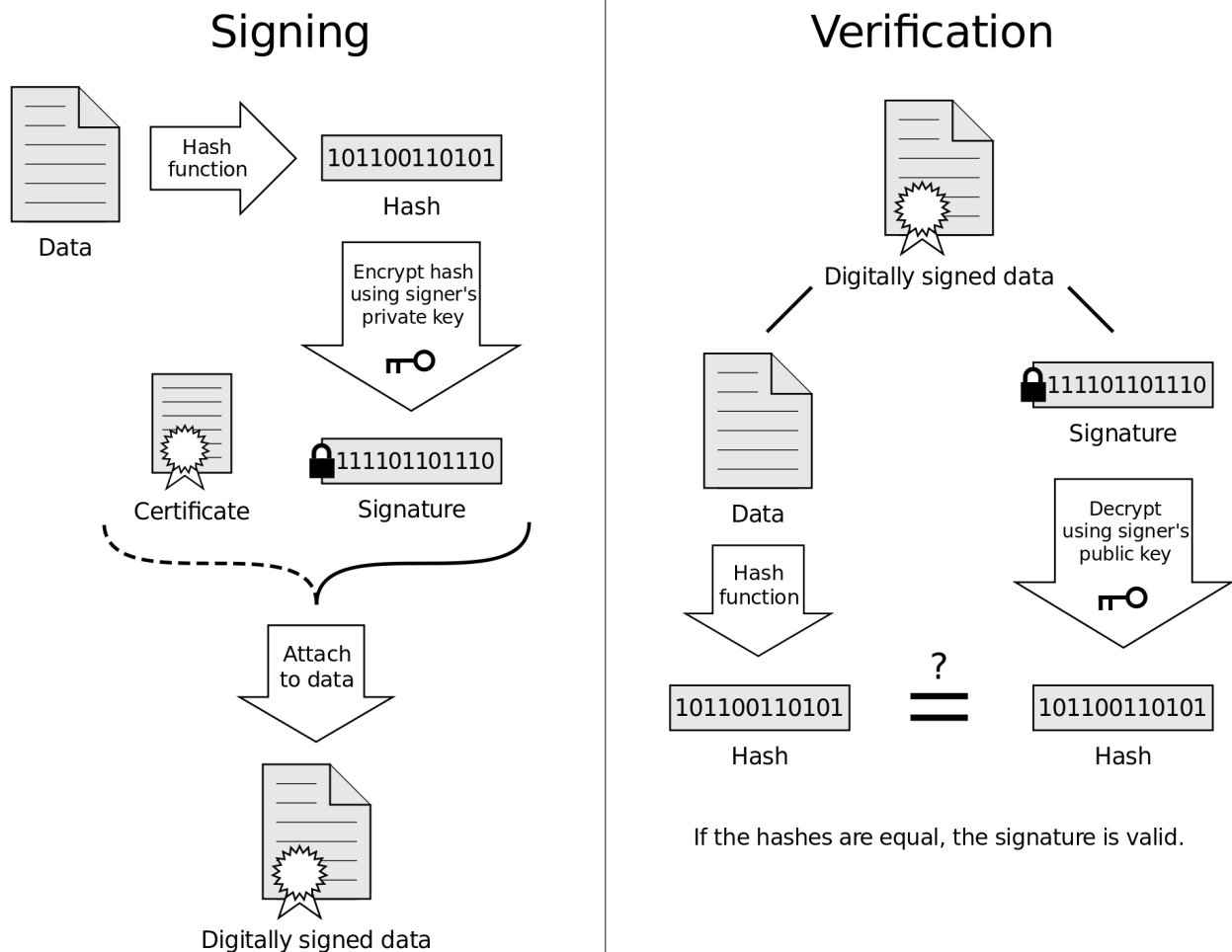


Figure 3.4: General scheme of digital signature and verification thereof (Wikipedia)

Verification of the signed message. Upon receipt of a signed message (S, m) , to check its authenticity and integrity:

1. Bob builds a fingerprint $H(m)$ of the message m using the agreed hash function H .
2. Then he decrypts the signature S using the decryption function D with the public key K_{pu} . Let $S'' = D(K_{pu}, S)$.
3. Finally he compares S with $H(m)$: if the two fingerprints are the same, the signature is authentic, the message is authenticated.

Figure 3.4 summarizes this.

Chapter 4

Authenticity and Watermarking

This chapter is inspired by [CMB⁺07, KSMM14]

4.1 Introduction

Digital watermarking is a computational method for inserting information in a digital document. This method should address some of the following properties:

- *Imperceptibility*: in this case, it must be difficult to distinguish the original document with document containing the mark;
- *Fragility*: in this case, a tiny change in the marked paper will make it impossible to extract the mark;
- *Robustness*: in this case, on the contrary, the mark is still present even if the document has been modified;
- *Reversibility*: in this case, it is possible to restore the document in its original form by replacing bits containing the mark with those of the original document.

4.2 Applications of watermarking

4.2.1 Proof of media ownership

Often the photo agencies add a visible mark to freely available versions of their photographs. These mark make it unusable in practice these photos in a professional context. These marked photos are just appeal products the high resolution pictures.

These high-resolution images usually embed an identifier property, but invisibly (by modifying only a few bits). Also, if the watermarking is robust, the agency can next justify to be the owner of the image if this one is traded without his consent, or if it is a slightly modified version that is found (partial, having undergone transformation. . .). The best-known example in omission of watermarking and support property loss is that of Lena Sjööblom.

4.2.2 Copy control

The watermarking can be used to identify the person to whom a digital document is given. The document proposer adds the person's identification as a mark inside it. This can be traced back to that person if the document should not be released, but it is.

A known example where this has been implemented in the early 2000s. Hollywood studios had found that a vast majority of illegal downloading movies available came from Hollywood industry itself. From that date, each copy distributed to an employee has been marked with a unique identifier. The Carmine Caridi actor received digital versions of films (marked) in order to vote in the Oscars 2004. Part of his films ("Mystic River," "The Last Samurai"...) -containing its watermark- were found on the download networks. This made possible to identify him as one of the authors of this share.

4.2.3 Content integrity

The watermarking can be used to ensure that the document has not been altered. In this context one can think of building the fingerprint of the document (via a hash function for example). If this document is accessed again and if the fingerprint is the same, there is a very high probability that the document has not been changed.

This raises the issue of storage of this fingerprint. One idea may be to build it from the significant parts of the document and only then insert into its non-significant parts.

4.3 Implementing a robust image watermarking scheme

4.3.1 Naive watermarking by LSB replacement

LSB means *Least Significant Bit*. The mark is inserted into the lower bits. This simple watermarking algorithm holds in a single line:

```
y = [int(bin(x[i])[:-1]+str(mark[i]),2) for i in range(len(x))]
```

In this line,

- `mark` is the list of message bit: `mark=[0,1,0,0...]`;
- `x` is the data list; it is the medium in which the watermark is embedded; its size is the same as that of the mark;
- For each index `i` browsing the entire list `x`, for example `i=2`;
 - Converting `x[i]` in binary: there is thus a string 45 becomes "101101" for example;
 - When writing `[:-1]`, all of this chain is preserved except for the last bit, "10110" for example;
 - The last bit is added the `i`th value of the mark, 0 in this example;
 - When writing `int('101100',2)`, the bit string is converted into integer, here 44.

Exercise 4.1. Explain this code except the line that has already been presented below.

```
import random as rd
from PIL import Image

support = Image.open("lena512.png")
support.show()

x = list(support.getdata())
mark=[rd.randint(0,1) for _ in range(len(x))]
y = [int(bin(x[i])[:-1]+str(mark[i]),2) for i in range(len(x))]

watermarked_image = Image.new("L",support.size)
watermarked_image.putdata(y)
watermarked_image.show()
```

Practical Work 4.1. 1. Implement the code seen in the previous example.

2. Watermark the image of your choice with a simple message using the LSB replacement scheme. Extract the mark of the watermarked image.
3. Implement an attack that changes at random every non-significant bits. Is it possible to recover the mark after the attack.

4.3.2 Quantization based Watermarking

Quantization is the process which allows to approach a continuous signal by the values of a finite small set. The underlying idea of the watermarking by quantization is that a mark even in a a watermarked but attacked document can be restored if its value lies in its quantization area. STDM (Spread Transform Dither Modulation) is an algorithm of this family. In a basic version, the watermarking function is parameterized by:

- a variable Δ used in the quantization: the larger the value is, the more robust is the approach, but it may become perceptible;
- A projection vector (p_1, \dots, p_n) that is used as a key.

Exercise 4.2. Explain each line of the following code

```
#####
N=10                # size of the mark
size_x=5000         # size of the support
size_p=int(size_x/N)# taille du vecteur de projection
Delta = 20

x=[50*rd.random() for _ in range(size_x)]
p=[rd.random() for _ in range(size_p)]
normep = math.sqrt(sum([e**2 for e in p]))
p = [pi/normep for pi in p]
m=[rd.randint(0,1) for _ in range(N)]

y      = emb_stdm(x,p,m,Delta)
mp      = dec_stdm(y,p,Delta)
yp      = [yi + rd.gauss(0,1) for yi in y]
mpp     = dec_stdm(yp,p,Delta)

print(y[:15])
print(yp[:15])

print(mp)
print(mpp)
```

Practical Work 4.2. 1. Get the `stdm_etu.py` file and run the code from the previous example.

2. Watermark the image of your choice with a simple message with STDm. Notice that in a RAW image, each pixel is defined by an integer value while `stdm` returns float values. Use of functions provided in the file.
3. Vary Δ . Does it affect the visual quality of the marked image?
4. Extract the mark of the watermarked image.
5. Implement an attack that randomly changes every non-significant bits. Show the watermarked attacked image. Can you use in practice? Is it possible to recover the mark after the attack?
6. Implement an attack that adds a Gaussian noise with zero mean and a σ standard deviation. Vary σ until this image can be usable.

4.4 Reversible watermarking

When the processed data is sensitive (military, medical diagnostic ...) end users can not to allow the information that analysis is not the original. For a radiologist who seeks to detect pathology with a radio image, it is intolerable that the diagnosis is wrong for the patient's health and for the radiologist himself. In this case, the irreversible watermarking (which irreversibly alters the original signal) is not an acceptable solution.

To solve this problem, the reversible watermarking was created: the original content can be restored from the watermarked document. The assessment may well be on the original signal. Thus watermarking step has a positive impact on the initial document as it includes additional data to it. It is important to note that the visual impact of the mark embedding is not as important that in the classic watermarking, since the original document can be restored. We can refer also to [KSMM14] for a comprehensive review of recent algorithms reversible watermarking

Chapter 5

Data Privacy

This chapter is based on the reports [Ger20, NC20].

5.1 From a meaningful wish to law

Protecting sensitive individual characteristics when storing or disseminating data for analysis was an issue long before the *open data* boom and the machine learning revolution. In 1977, the Swedish statistician Tore Dalenius [Dal77] precisely defined a guarantee of confidentiality: by accessing a database, the opponent should not be able to obtain information about an individual that he would not have been able to learn without such access. Cynthia Dwork demonstrated [DMNS06] that this Sangraal remains inaccessible, notably because of potential auxiliary information. This will motivate the development of other less rigid concepts.

Since 1978, the french law "Informatique et libertés" specifies what such data is, which makes it possible to know what anonymized data is:

“Constitue une donnée à caractère personnel toute information relative à une personne physique identifiée ou qui peut être identifiée, directement ou indirectement, par référence à un numéro d’identification ou à un ou plusieurs éléments qui lui sont propres. Pour déterminer si une personne est identifiable, il convient de considérer l’ensemble des moyens en vue de permettre son identification dont dispose ou auxquels peut avoir accès le responsable du traitement ou toute autre personne.”

At the beginning, French legislation was very rigid concerning anonymous data, since the definition was related to an obligation of result. More recently in 2018, this obligation of result has been transformed into an obligation of resources (GDPR).

“Pour déterminer si une personne physique est identifiable, il convient de prendre en considération l’ensemble des moyens raisonnablement susceptibles d’être utilisés par le responsable du traitement ou par toute autre personne pour identifier la personne physique directement ou indirectement, [...], c’est à dire] l’ensemble des facteurs objectifs, tels que le coût de l’identification et le temps nécessaire à celle-ci, en tenant compte des technologies disponibles au moment du traitement et de l’évolution de celles-ci.”

The definition of anonymization is therefore based on a probability of re-identification. Pseudonymization is a first attempt to reduce this probability of re-identification, but it is not at all sufficient, as we will see in the following section.

5.2 Pseudonymization as a first coarse idea

To allow statistics (on the citizens of a state for example), it is quite tempting to publish a database in which random numbers have been substituted for real names (for example by “hashing” these last ones). This approach, known as *pseudonymization*, proves incapable of guaranteeing anonymity and it often remains possible to re-identify individuals registered in the database because it is often possible to retrieve some of this data from other databases and re-associate it with an individual.

These n -uplets of fields present in the records of a database and which characterize the users in a unique way, at least with a high probability, are called *quasi-identifier*. For example, in a table of hospitalized patients, where the pathology treated is sensitive information, the data (approximate age; approximate weight; ethnic origin; marital status) makes it possible to identify patients in a quasi-unique way [DN03].

For instance¹, a team of researchers from the University of Melbourne have re-identified seven prominent Australians in an open medical dataset using publicly known information. The published dataset containing historic longitudinal medical billing records of one-tenth of all Australians, approximately 2.9 million people whose identifiers are encrypted, has been found to be re-identifiable thanks to additional information such as child births and professional sportspeople undergoing surgery to fix injuries often made public.

This and many other examples show that pseudonymization is insufficient.

Exercise 5.1. *Pseudonymization*² The dataset from a social network given in Table 5.1 has been pseudonymized in the strongest possible way, i.e. where name has been removed. However, there is additional information at your disposal in Table 5.2. in this exercise, we use this information to investigate how privacy can be leaked by cleverly linking data. Assume for all candidates that their are present in both databases.

Name	Gender	Age	City of birth	Favorite TV Series	Relationship Status
*	male	19-25	Saarbrücken	Game of Thrones	single
*	female	16-18	Trier	Game of Thrones	in relationship
*	male	12-15	München	Friends!	in relationship
*	female	19-25	Berlin	Big Bang Theory	in relationship
*	female	19-25	Hamburg	Big Bang Theory	single
*	female	19-25	Saarbrücken	Game of Thrones	single
*	male	16-18	Trier	Game of Thrones	single
*	female	12-15	München	Game of Thrones	in relationship
*	male	19-25	Berlin	Big Bang Theory	single

Table 5.1: “sanitized” dataset of a social network

Name	Email	TV Show	Rating (1=bad, 5=great)
Alice	alice1995@email.com	Friends!	1
Bob	bobbybob@email.com	Friends!	4
Charlie	s9charchar@email.com	Friends!	2
Eve	evelyn@myhighschool.com	Friends!	1
Bob	bobbybob@email.com	Game of Thrones	1
Alice	alice1995@email.com	Game of Thrones	5
Charlie	s9charchar@email.com	Game of Thrones	5
Bob	bobbybob@email.com	Big Bang Theory	3
Charlie	s9charchar@email.com	Big Bang Theory	5
Alice	alice1995@email.com	Big Bang Theory	2
Eve	evelyn@myhighschool.com	Big Bang Theory	5

Table 5.2: Additional information

1. Where is Alice most likely born and what is most likely her relationship status?
2. Can you learn any personal information about Charlie as well?
3. Can you learn any personal information about Bob as well?

5.3 k -anonymity

Here is a first response to re-identification, based on the aggregation of data introduced by *Latanya Sweeney* [Swe02]. The k -anonymity, $k \in \mathbb{N}^*$ is the guarantee that every n -upload of any quasi-identifiers is associated with k records in the database at a minimum, so that an individual is protected within a group of k or more. The probability of individual identification becomes $\frac{1}{k}$ in the worst case.

Concretely, one must first identify the groupings of non-sensitive attributes likely to identify a line of the base. Then, we reduce the level of detail in the values of certain fields, until we can guarantee that at least k records are

¹<https://pursuit.unimelb.edu.au/articles/the-simple-process-of-re-identifying-patients-in-public-health-data>

²<http://www.infsec.cs.uni-saarland.de/teaching/16WS/Cybersecurity/exercises/exercise-11.pdf>

ID	Age	Gender	Fav.Show
1	12-15	female	Friends!
2	19-25	male	Friends!
3	19-25	male	Friends!
4	12-15	female	Friends!
5	19-25	male	G.o.T.
6	19-25	male	G.o.T.
7	19-25	male	G.o.T.

ID	Age	Gender	Fav.Show
1	19-25	female	Grey's A.
2	19-25	female	Simpsons
3	19-25	female	Futurama
4	19-25	female	Friends!
5	19-25	male	G.o.T.
6	19-25	male	C.Minds
7	19-25	male	Br.Ba.

ID	Age	Gender	Fav.Show
1	19	male	Friends!
2	19	male	Friends!
3	19	male	Friends!
4	19	female	Friends!
5	20	male	G.o.T.
6	20	male	G.o.T.
7	20	male	G.o.T.

Figure 5.1: 3 small generalized datasets

associated with each quasi-identifier: we can group numerical data into classes and aggregate categories. This is the generalization phase. Let's take an example with $k = 2$:

<i>Name</i>	<i>Age</i>	<i>CP</i>	<i>Treatment</i>
...	19	25 000 _B	Asthma
...	43	25 200 _M	Cancer
...	73	25 170 _B	Leukemia
...	27	25 480 _B	HIV
...	35	25 120 _M	HIV
...	61	25 000 _B	Leukemia

⇒

<i>Age</i>	<i>Round.</i>	<i>Treatment</i>
18-30	Besançon	Asthma
18-30	Besançon	HIV
35-45	Montbéliard	Cancer
35-45	Montbéliard	HIV
60+	Besançon	Leukemia
60+	Besançon	Leukemia

Exercise 5.2 (K-anonymity on a short example). *This exercise is again inspired from².*

1. Does Dataset 1 from Figure 5.1 satisfy k -anonymity? If so: what is the maximal k for which it satisfies k -anonymity?
2. Same question for dataset 2.
3. Same question for dataset 3.

One of the major interests of k -anonymity is that it is understandable by a non-specialist. But this concept is not always simple to implement, to automate, and flaws may remain. It is difficult to provide formal guarantees.

From a quasi-identifier, it is often possible to know that some values are excluded, for a sensitive field (asthma impossible for “35-45, Montbéliard”), or that others are more probable than in the global population. And in case of identical values on sensitive data for all members of an anonymous group, we even find this information in a certain way (e.g. leukemia, for “60+, Besançon”). To avoid this situation, the concept of ℓ -diversity is introduced.

5.4 ℓ -diversity

Publishing k -anonymized information may be critical if they are not varied enough, then this information can be misused and led to a positive or a negative disclosure. For example, if all the outings that took place on a given date involved heart ailments and if we know that a person was rescued by firemen on that day, we deduce that they had a heart attack.

Zip	Age	Nationality	Disease
13053	28	Russian	Heart
13068	29	American	Heart
13068	21	Japanese	Viral
13053	23	American	Viral
14853	50	Indian	Cancer
14853	55	Russian	Heart
14850	47	American	Viral
14850	59	American	Viral
13053	31	American	Cancer
13053	37	Indian	Cancer
13068	36	Japanese	Cancer
13068	32	American	Cancer

Table 5.3: Dataset for diseases w.r.t nationality, age, and ZIP codes

Intuitively, a group of records (bloc, equivalent class) is said to be l -diverse if there are at least l “well-represented” values for the sensitive attributes (which may be a single sensitive attribute, a pair of sensitive attributes, ...). The dataset is said to be l -diverse if each group of records is l -diverse. The notion of “well-represented” is intentionally ambiguous. The fact that l separates values is not sufficient for this definition. A potential refinement could be that the current values are distributed according to a law approaching uniform distribution.

Exercise 5.3. Applying l -diversity Table 5.3 presents a virtual dataset of hospital patients where SSN have been removed. To generalize, some details can be removed:

- Zip codes: remove digit, one by one, $R \rightarrow L$: XXXX*, XXX**, XX***, X****, remove
- Age: group by cluster whose range is: **10**, 20, 40, remove
- Gender: remove
- Nationality: group by continent, remove

1. How many generalization schemes can be done?
2. Apply the generalization in bold above. What can be deduced if you know someone in the database that is 38 years old? What can you deduce from this anonymous dataset?
3. Apply the generalization with choices underlined above. Does it solve the previous problem?

Practical Work 5.1. ARX.³ All this practical work will be executed with ARX software which is a java package that can be seen as a first attempt to anonymize a dataset.

1. Download the jar file⁴ that corresponds to your OS.
2. Execute `java -jar arx-XXX.jar` to launch the tool. In ARX, import the dataset `original-data.csv` that is provided. This data set was obtained from the UC Irvine Machine Learning Repository⁵ and contains weighted census data extracted from the 1994 and 1995 Current Population Surveys conducted by the U.S. Census Bureau⁶.

³<https://arx.deidentifier.org/>

⁴<https://arx.deidentifier.org/downloads/>

⁵[http://archive.ics.uci.edu/ml/datasets/Census-Income+\(KDD\)](http://archive.ics.uci.edu/ml/datasets/Census-Income+(KDD))

⁶<https://www.census.gov/en.html>

- (a) For each column name, precise in table 5.4 whether it is a Quasi Identifier, sensitive, or other. You can use the website ⁷
3. Select the “age” column in ARX and precise it is a Quasi Identifier. Create the generalization hierarchy (by using the aforementioned button) that corresponds to the following tree $[0; 25[, [25; 50[, [50; 75[, [75; 100[\subset [0; 50[, [50; 100[\subset [0; 100[$.
 4. Select the “sex” column in ARX and precise it is a Quasi Identifier. Create the generalization hierarchy that aggregates the two values in a set.
 5. For the “state_of_previous” residence define a generalization strategy with 1 level.
 6. For the “contry_of_birth_self” define a generalization strategy with 1 level.
 7. Building a 2-anonymity model.
 - (a) Define the 2-anonymity model, and apply it. Thanks to the “Explore results” tabs, can you explain which generalization parameters have been used. What does it mean?
 - (b) Export resulting data as `out_k_2.csv`.
 - (c) Open this file in libreoffice. You have heard about a person, 52 years old, and who is a widow. Can you infer knowledge on this person in this anonymized output data?
 8. Building a 12-anonymity model and answer the same questions. Memorize the generalization parameters.
 9. In the last scenario, one can notice that the capital gain is almost equal to 0 for women in $[0,50]$, which is too much homogeneous.
 - (a) Define that “capital_gain” is a sensitive attribute.
 - (b) Add the privacy model of 5 diversity, and ask ARX to anonymize the dataset.
 - (c) compare the generalization strategy with the previous one.
 - (d) Export resulting data as `out_k_12_1_5.csv` and check for person in $[0,50]$ years old.
 - (e) give a conclusion

To conclude this chapter:

- Pseudonymization is not a method of data anonymization.
- k -anonymity coupled with l -diversity is a first approach, but it is not always sufficient.
- Only differential privacy (DP) provides mathematical guarantees on the disclosed information. However, it is a little more complex to implement.

⁷https://www2.1010data.com/documentationcenter/beta/Tutorials/MachineLearningExamples/CensusIncomeDataSet.html#reference_vn5_sxt_z4__input_variables

Column Name	Description	Type	Column Name	Description	Type
age	Age of the worker		class_worker	Class of worker	
det_ind_code	Industry code	?	det_occ_code	Occupation code	?
education	Level of education		wage_per_hour	Wage per hour	
hs_college	Enrolled in educational instt. last week		marital_stat	Marital status	
major_ind_code	Major industry code		major_occ_code	Major occupation code	
race	Race		hisp_origin	Hispanic origin	
sex	Sex		union_member	Member of a labor union	
unemp_reason	Reason for unemployment		full_or_part_emp	Full- or part-time employment status	
capital_gains	Capital gains		capital_losses	Capital losses	
stock_dividends	Dividends from stocks		tax_filer_stat	Tax filer status	
region_prev_res	Region of previous residence		state_prev_res	State of previous residence	
det_hh_fam_stat	Detailed household and family status		det_hh_summ	Detailed household summary	
mig_chg_msa	Migration code - change in MSA		mig_chg_reg	Migration code - change in region	
mig_move_reg	Migration code - move within region		mig_same	Live in this house one year ago	
mig_prev_sunbelt	Migration - previous residence in sunbelt		num_emp	# of persons that worked	
fam_under_18	Family members under 18		country_father	Country of birth father	
country_mother	Country of birth mother		country_self	Country of birth	
citizenship	Citizenship		own_or_self	Own business or self-employed	
vet_question	Fill included question. for Veterans Admin.		vet_benefits	Veterans benefits	
weeks_worked	Weeks worked in the year		year	Year of survey	
income_50k	Income less than or greater than \$50,000		edu_year	Number of years of education	

Table 5.4: Types of indicators

Bibliography

- [AJOP15] Gildas Avoine, Pascal Junod, Philippe Oechslin, and Sylvain Pasini. *Sécurité informatique, cours et exercices corrigés*. Vuibert, 2015.
- [CMB⁺07] Ingemar Cox, Matthew Miller, Jeffrey Bloom, Jessica Fridrich, and Ton Kalker. *Digital watermarking and steganography*. Morgan kaufmann, 2007.
- [Dal77] Tore Dalenius. Towards a methodology for statistical disclosure control. *statistik Tidskrift*, 15(429-444):2–1, 1977.
- [DMNS06] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.
- [DN03] Irit Dinur and Kobbi Nissim. Revealing information while preserving privacy. In *Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 202–210, 2003.
- [Ger20] Jean-Marc Gervais. Deep learning et respect des informations individuelles sensibles. Master’s thesis, Université de Franche-Comté, 2020. available at <https://www.researchgate.net/project/Deep-Learning-Differential-Privacy>.
- [Ghe13] S. Ghernaouti. *Sécurité informatique et réseaux - 3e éd.: Cours avec plus de 100 exercices corrigés*. Informatique. Dunod, 2013.
- [KSMM14] Asifullah Khan, Ayesha Siddiq, Summuyya Munib, and Sana Ambreen Malik. A recent survey of reversible watermarking techniques. *Information sciences*, 279:251–272, 2014.
- [NC20] Benjamin Nguyen and Claude Castelluccia. Techniques d’anonymisation tabulaire: concepts et mise en oeuvre. *arXiv preprint arXiv:2001.02650*, 2020.
- [RSA09] Christiane Rousseau and Yvan Saint-Aubin. La cryptographie à cle publique: le code rsa (1978). *Mathématiques et Technologie*, pages 213–244, 2009.
- [Swe02] Latanya Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 2002.
- [TIP] TIPE. Travaux ’initiative personnelle encadrés (tipe), les fonctions de hachage. on line https://www.scei-concours.fr/tipe/TIPE_2008/sujets_2008/informatique_2008.pdf.
- [Ver15] Damien Vergnaud. *Exercices et problèmes de cryptographie-2e éd*. Dunod, 2015.