Cybersécurité - Cours 5: La fonction de hachage

SIO - Bloc 3 - SMDSI

M. SPINA

Sommaire

1) Histoire de l'encodage
2) Le code ASCII
3) Les extensions du ASCII
    a) Unicode
    b) UTF-8
4) Rappel sur le chiffrement
5) Définition de fonction de hachage
    a) Exemples
    b) Mise en pratique
6) Mise en application du hachage
7) Différence entre le chiffrement et le hachage
8) Références

Histoire de l’encodage

<u>1er encodage historique :</u> ASCII (American Standard Code for Information Interchange): norme américaine, standardisée en 1963

→ **Son but est d'organiser l’univers informatique à l’échelle nationale**

Comme il a été principalement conçu pour représenter des textes en anglais, il n’y avait donc pas de caractère accentué.

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

Histoire de l’encodage: le code ASCII

A l’époque, les caractères considérés comme essentiels à une communication sont les suivants:

- les 10 chiffres
- les 26 lettres minuscules
- les 26 lettres majuscules
- 32 symboles (@, <, >, espace, etc)
- 33 symboles de mise en page (passage à la ligne, saut de page, etc.)

Soit 127 caractères

Histoire de l’encodage: le code ASCII

On code donc sur 1 octet (8 bits), le premier bit étant toujours 0 et sert au contrôle de parité (pour éviter des erreurs).

128	64	32	16	8	4	2	1

- Les caractères de numéro 0 à 31 correspondent à des commandes de contrôle de terminal informatique.
- Le caractère numéro 127 est la commande pour effacer.
- Les chiffres sont codés par les nombres de 48 à 57
- Les lettres majuscules par les nombres de 65 à 90
- Les minuscules par les nombres de 97 à 122

Histoire de l’encodage: les limites du ASCII

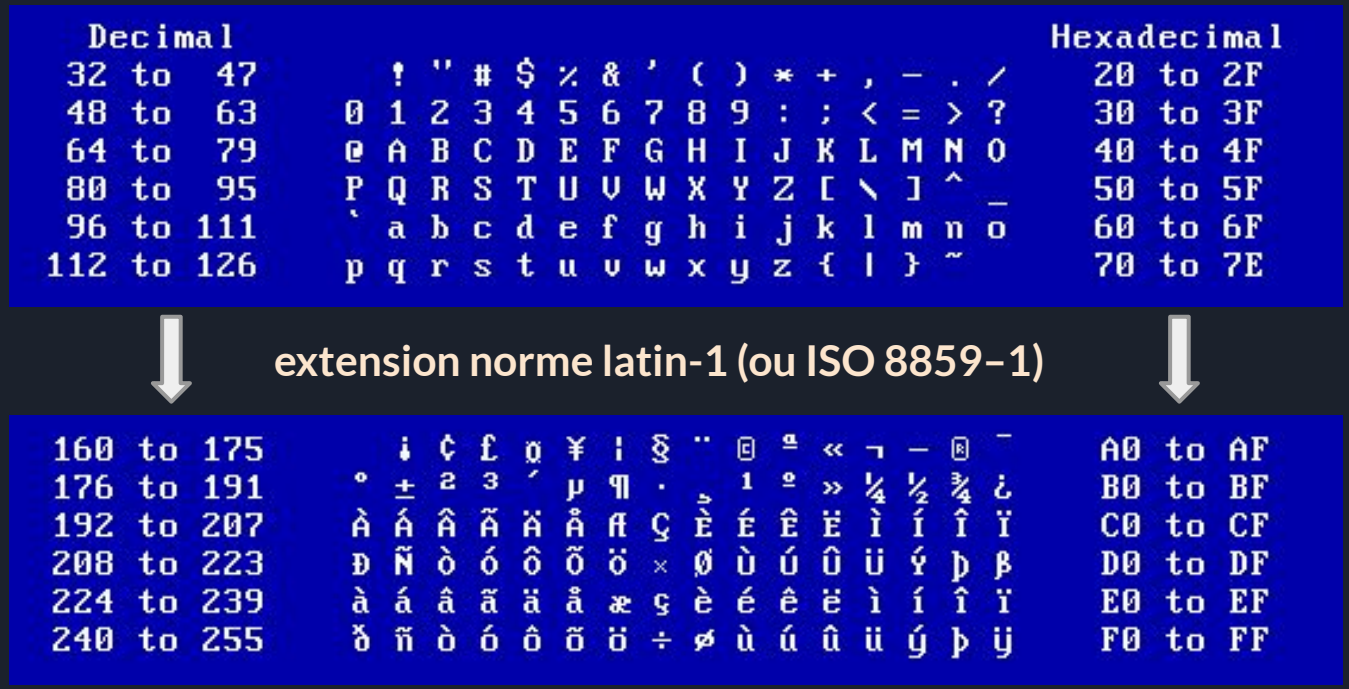
Simple et efficace, ce code a rapidement montré ses limites:

→ pas d’accent, pas de caractère spécial pour les langues latines

C’est pourquoi, plusieurs pays ont proposé des extensions en utilisant le 1er bit, les 8 bits sont alors utilisés.

→ les possibilités sont alors plus nombreuses, **combien ?**

Histoire de l’encodage: les extensions du ASCII

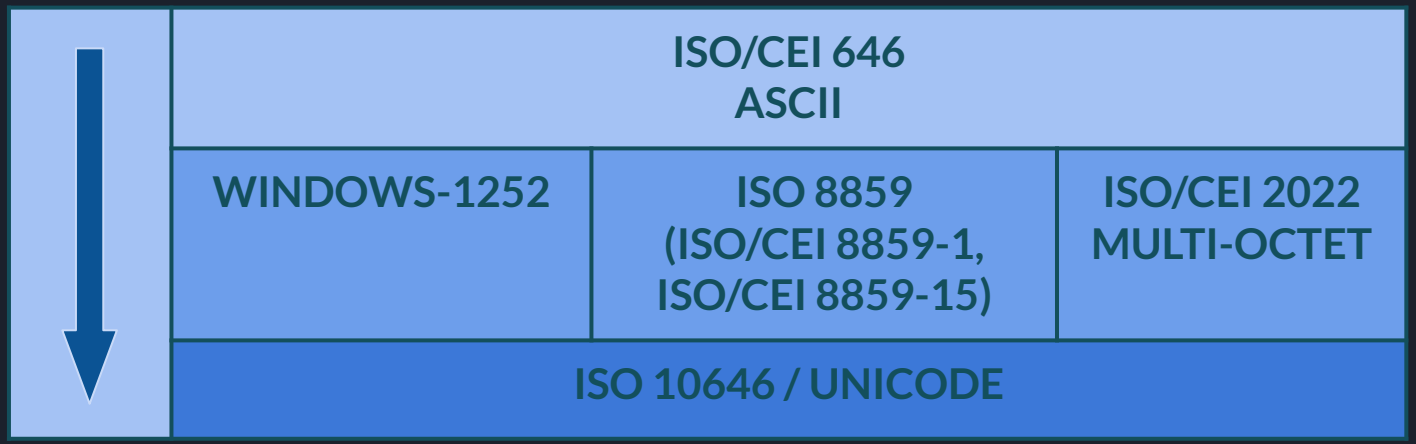


Decimal		Hexadecimal
32 to 47	! " # $ % & ' ( ) * + , - . /	20 to 2F
48 to 63	0 1 2 3 4 5 6 7 8 9 : ; < = > ?	30 to 3F
64 to 79	@ A B C D E F G H I J K L M N O	40 to 4F
80 to 95	P Q R S T U V W X Y Z [\ ] ^ _	50 to 5F
96 to 111	` a b c d e f g h i j k l m n o	60 to 6F
112 to 126	p q r s t u v w x y z { | } ~	70 to 7E

extension norme latin-1 (ou ISO 8859–1)

160 to 175	¡ ¢ £ ¤ ¥ ¦ § ¨ © ª « ¬ ® ¯	A0 to AF
176 to 191	° ± ² ³ ´ µ ¶ · ¸ ¹ º » ¼ ½ ¾ ¿	B0 to BF
192 to 207	À Á Â Ã Ä Å Æ Ç È É Ê Ë Ì Í Î Ï	C0 to CF
208 to 223	Ð Ñ Ò Ó Ô Õ Ö × Ø Ù Ú Û Ü Ý Þ ß	D0 to DF
224 to 239	à á â ã ä å æ ç è é ê ë ì í î ï	E0 to EF
240 to 255	ð ñ ò ó ô õ ö ÷ ø ù ú û ü ý þ ÿ	F0 to FF

Histoire de l’encodage: les extensions du ASCII



Histoire de l’encodage: ISO 10646 / UNICODE

1990: création de la solution ultime ISO 10646

→ Jeu Universel de Caractères (Universal Character Set en anglais)

Créé pour pouvoir accueillir n’importe quel caractère existant de n’importe quelle langue du monde.

1991: surcouche Unicode

Gère, par exemple, les différents sens de lecture

les deux normes sont développées parallèlement et synchronisées en permanence

Histoire de l’encodage: UTF-8

En théorie, **Unicode** est **plutôt efficace** mais en **pratique** ?

→ un caractère prend 2 octets : 2 fois plus qu’en ASCII

→ dans un texte Français, les caractères utilisent principalement l’ASCII

C’est alors que **l’UTF-8 fait son apparition**.

Un texte en UTF-8 est simple: **il est codé entièrement en ASCII** et dès qu’on a besoin d’un caractère appartenant à l’Unicode, **on utilise un caractère spécial** indiquant qu’il est en Unicode

Histoire de l’encodage: UTF-8

L’UTF-8 est un des formats de codage **les plus courants qui gère tous les caractères** Unicode dans les systèmes ASCII préexistants.

Exemple de texte codé en latin-1

“Bonjour les Ã©tudiants”

Exemple de texte codé en UTF-8

“Bonjour les étudiants”

Histoire de l’encodage: UTF-8

Et dans les pages web ?

La plupart des navigateurs **supportent l'UTF-8 et le détectent automatiquement**

On renseigne dans l’entête du document HTML **le type d’encodage utilisé**

Exemple:

```
<meta http-equiv="Content-type" content="text/html; charset=UTF-8">
```

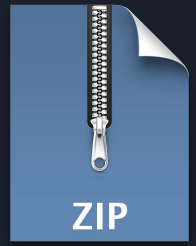
Histoire de l’encodage: en définitive

Qu’est-ce que l’encodage ?

Encodage désigne l'action de transcrire des données vers un format ou un protocole donné.

Exemples :

- Encodage d'une vidéo avec un codec plus performant
- Compression d'un document (zip)
- Conversion d’une chanson vers un autre format



Rappel: le chiffrement

Qu’est-ce que le chiffrement ?

Permet de rendre incompréhensible un document sauf si on possède la clé de (dé)chiffrement

Exemples :

- Chiffrer un message pour établir une communication privée
- Rendre illisible un document confidentiel
- Authentifier des transactions (exemple: cryptomonnaie)

Le hachage: définition

Qu’est-ce que le hachage ?

C’est un algorithme permettant de modifier un texte (appelé message) en valeur de longueur fixe (appelé hash).

Le hachage: exemples

Deux algorithmes de hachage:

1991: md5 est une fonction de hachage qui retourne toujours 32 caractères

→ md5('test'): 098f6bcd4621d373cade4e832627b4f6

1995: sha1 est une fonction de hachage qui retourne toujours 40 caractères

→ sha1('test'): a94a8fe5ccb19ba61c4c0873d391e987982fbbd3

Le hachage: précision

A l'origine, le but de cette technique était de créer une fonction à sens unique.

Le hash d'un message clair ne pouvait pas être dé-hashé.

Entrée  *Sortie*

Evidemment, les failles de sécurité apparaissent au fur et à mesure de l'avancé technologique des méthodes de crackage et des divers types d'attaque.

Le hachage: évolution des algorithmes

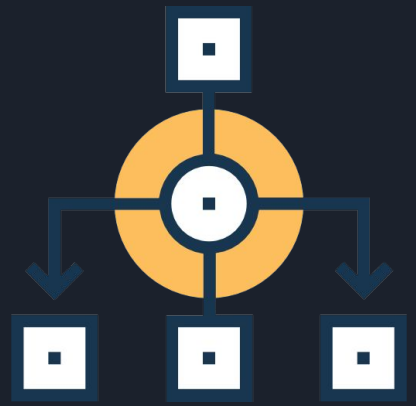
Des logiciels existent pour cracker des hashs MD5 et SHA1.

C’est pourquoi, les algorithmes de hachage évoluent aussi.

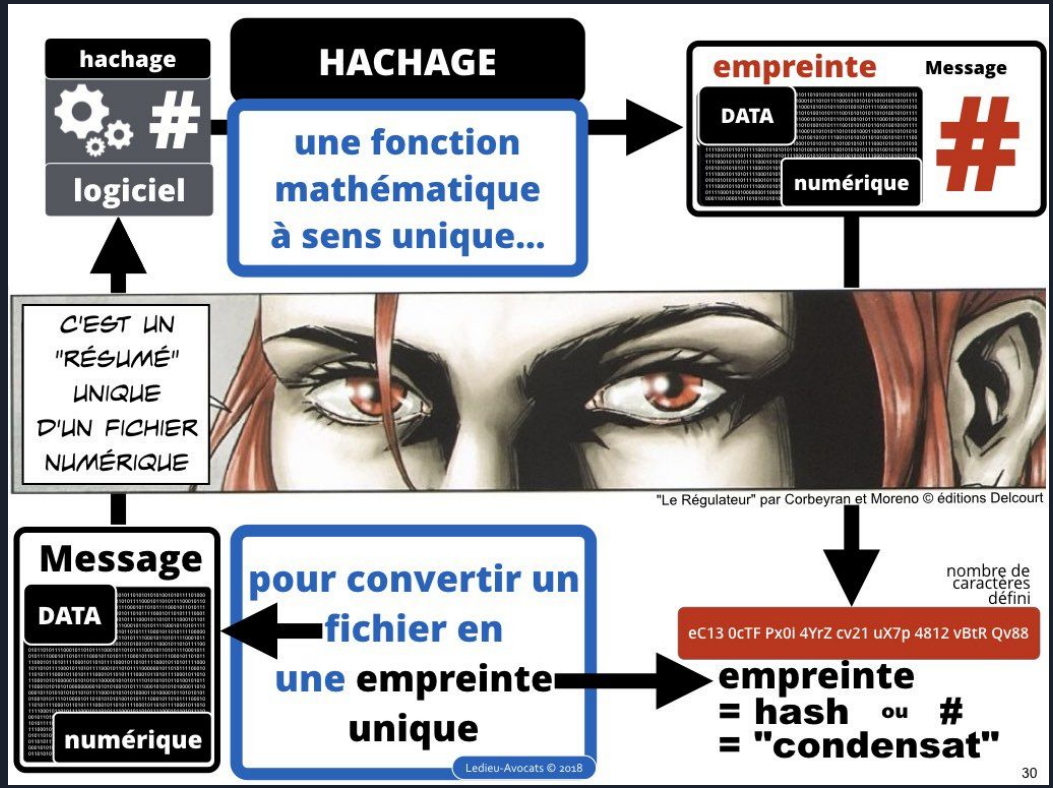
2008: MD5 → MD6

2015: SHA-1 devient SHA-2 puis SHA-3

Et plein d’autres algorithmes existent.



Le hachage: mise en pratique



Le hachage: mise en pratique

On peut utiliser une fonction de hachage lorsque **l'on souhaite comparer une valeur sans pouvoir stocker sa représentation simple** (pour diverses raisons)

Une des utilisations les plus courantes:

→ **stocker des mots de passe** dans une base de données

Il est interdit de stocker un MDP sous sa forme brut.

En cas d'oubli de mot de passe, on ne vous l'envoie pas, **on vous demande de le réinitialiser.**

Le chiffrement: mise en pratique

On peut utiliser le chiffrement lorsque l'on souhaite **stocker des données sensibles pour pouvoir y avoir accès ponctuellement**.

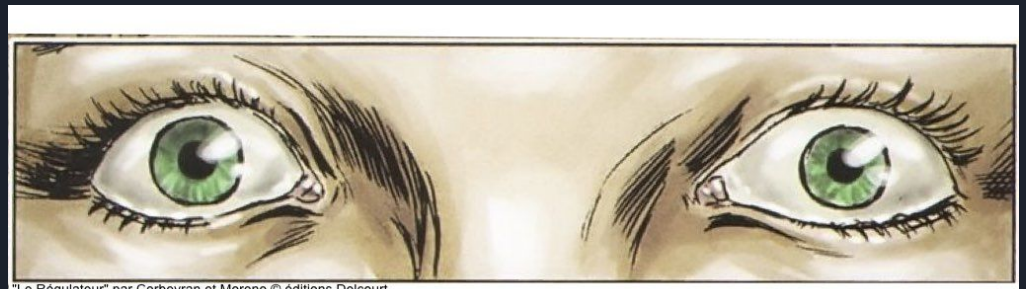
Une des utilisations les plus courantes:

→ **stocker les numéros d'une carte de crédit** dans une base de données

Nous avons besoin d'accéder aux numéros, c'est pourquoi une fonction à sens unique ne correspondait pas.

Le chiffrement permet de réaliser cette fonctionnalité.

Le chiffrement et le hachage: différence



"Le Régulateur" par Corbeyran et Moreno © éditions Delcourt

CHIFFREMENT

le chiffrement est
- par principe -
une fonction réversible

Ledieu-Avocats © 2018

déchiffrement
avec la clé secrète
symétrique
Asymétrique

SANS la clé secrète
décryptage
cryptanalyse

HACHAGE

le hachage
N'EST
- par principe -
PAS
réversible

Références

http://sdz.tdct.org/sdz/comprendre-les-encodages.html

https://www.difrance.com/informations/tout-savoir-sur-l-encodage

https://www.ipgirl.com/1662/difference-fondamentale-entre-les-algorithmes-de-hachage-et-de-chiffrement.html

https://waytolearnx.com/2018/07/difference-entre-cryptage-et-hachage.html

https://notes-de-cours.com/web/blogue/30/differencier-l-encodage-le-chiffrement-et-le-hachage

https://isnenroute.fr/codage_caracteres.php

https://www.ledieu-avocats.fr/chiffrement-symetrique-et-hachage/