

Bloc 3



TD14

OWASP

Défaillances cryptographiques

Durée : 1h30

Auteur : NGO

Sommaire

I. PRESENTATION :	3
II. EXEMPLES DE SCENARIOS D'ATTAQUES.....	4
III. DONNEES A CARACTERE PERSONNEL ET DONNEES SENSIBLES	4
IV. ÉTATS DES DONNEES SENSIBLES.....	5
V. CONFIGURATIONS DE CHIFFREMENT	5
VI. DEFI N°1 : SCAN DES ALGORITHMES DE CHIFFREMENT D'UNE APPLICATION WEB	6
Outils	6
À vous de jouer.....	9
VII. DEFI N°2 : FORCE BRUTE D'UN MOT DE PASSE HACHE EN MD5.....	9
Outils et documents	9
VIII. BONNES PRATIQUES ET CONTRE-MESURES	12
Document 1 : Recommandations de sécurité relatives à TLS de l'ANSSI	14
Document 2 : Le protocole TLS-1.0 et les cartes de paiement.....	14
Document 3 : Chiffrement symétrique et asymétrique	14
Document 4 : Fonctions de hachage d'après la CNIL	15

Cette activité concerne les problématiques liées à l'absence ou à la mauvaise configuration des protocoles de chiffrement. Cette vulnérabilité est en deuxième position dans le dernier classement du top 10 du groupe OWASP.

Le TD est à réaliser individuellement. Vous devez rendre les réponses de ce TD sur un document libre office normalisé et rédigé à déposer sur le Drive de la section :

DEPOT /TD14/nom_prenom.odt

CF B3-OWSAP_PRESENTATION.pdf pour la mise en place du lab

I. PRESENTATION :

Les développeurs doivent prendre en compte la sécurisation des données sensibles traitées par leurs applications. Les technologies de chiffrement sont une réponse indispensable pour éviter une brèche de confidentialité sur ces données. Si ces technologies sont mal configurées ou absentes alors le risque est une exposition de ces données à des personnes non habilitées. La responsabilité juridique du développeur peut être engagée. Ainsi, la CNIL a condamné le site infogreffe à une amende de 250 000 Euro pour ne pas avoir mis en application l'article 321 du RGPD sur la sécurisation des données. Dans son délibéré, la CNIL fait référence à la transmission en clair de mots de passe.

« la transmission, en clair, d'un mot de passe qui n'est ni temporaire, ni à usage unique et dont le renouvellement n'est pas imposé, le rend aisément et immédiatement utilisable par un tiers qui aurait un accès indu au message qui le contient... »

Classée par le groupe d'experts en cybersécurité OWASP en deuxième position, en progression d'une place, cette vulnérabilité concerne les problématiques de configuration SSL. Les configurations de chiffrement peuvent être insuffisantes voire absentes. L'enjeu fondamental de cette défaillance est l'exposition des données sensibles. Plusieurs CWE (Common Weakness Enumeration) concernent les problématiques d'exposition des données sensibles. CWE est une liste des vulnérabilités que l'on peut rencontrer dans les logiciels. Cette liste est maintenue par l'organisme MITRE, le projet étant soutenu par la National Cyber Security Division et le département de la Sécurité intérieure des États-Unis

CWE-259: Use of Hard-coded Password (mot de passe présent en clair dans le code source) ;

CWE-327: Broken or Risky Crypto Algorithm (Algorithmes de chiffrements obsolètes) ;

CWE-331 Insufficient Entropy (systèmes prédictibles car non assez randomisés).

II. EXEMPLES DE SCENARIOS D'ATTAQUES

Scénario 1 : Une application chiffre des numéros de cartes de crédit dans une base de données utilisant un chiffrement en base automatique. Cependant, ces données sont automatiquement déchiffrées lorsqu'elles sont récupérées, permettant, à une injection SQL de récupérer des numéros de carte de crédit en clair.

Scénario 2 : Un site n'utilise pas ou ne force pas l'utilisation de TLS sur toutes les pages ou supporte des protocoles de chiffrement faibles. Un attaquant surveille le trafic réseau (par exemple sur un réseau sans fil non sécurisé), dégrade les connexions de HTTPS à HTTP, intercepte les requêtes et vole le cookie de session d'un utilisateur (voir un exemple dans cette activité : <https://www.reseaucerta.org/sites/default/files/owasp-activite3-v1.0.pdf>). L'attaquant réutilise alors ce cookie et détourne la session. de l'utilisateur (authentifié), pouvant ainsi accéder aux données privées de l'utilisateur ou les modifier. Un attaquant pourrait également modifier toutes les données en transit, par exemple le destinataire d'un virement d'argent.

Scénario 3 : La base de données contenant les mots de passe n'utilise pas de fonction de salage, ou alors de simples hachés pour stocker les mots de passe. Une faille de téléversement de fichier permet à un attaquant de récupérer la base de données des mots de passe. Tous les hachés non salés peuvent alors être révélés avec une table arc en ciel (rainbow table)² de hachés pré-calculés. Des hachés générés par des fonctions de hachage simples ou rapides peuvent être déchiffrés même avec des fonctions de salage.

III. DONNEES A CARACTERE PERSONNEL ET DONNEES SENSIBLES

D'après la CNIL :

Données sensibles : ce sont des informations qui révèlent la prétendue origine raciale ou ethnique, les opinions politiques, les convictions religieuses ou philosophiques ou l'appartenance syndicale, ainsi que le traitement des données génétiques, des données biométriques aux fins d'identifier une personne physique de manière unique.

Données à caractère personnel : toute information relative à une personne physique susceptible d'être identifiée, directement ou indirectement.

IV. ÉTATS DES DONNEES SENSIBLES

Les données peuvent passer par trois états :

- Les données au repos : il s'agit de données archivées ou stockées sans qu'il y ait de traitement.
- Les données en cours de traitement : il s'agit des données manipulées par un programme qui effectue des traitements (consultation, modification, ajout et suppression).
- Les données en transit : il s'agit des données qui circulent sur un réseau entre divers périphériques finaux et intermédiaires.

Une règle importante est de ne stocker que les données nécessaires à la bonne exécution du traitement et uniquement durant le temps nécessaire.

V. CONFIGURATIONS DE CHIFFREMENT

Pour assurer la sécurité des données manipulées, le développeur a recours à des technologies de chiffrement. Le chiffrement consiste à rendre les données illisibles pour les personnes qui ne possèdent pas la clé de déchiffrement. Or certains codes sources peuvent contenir des informations permettant d'identifier des personnes ou d'accéder à des configurations confidentielles.

L'exemple typique concerne la gestion des mots de passe. En effet, une étude de 2019 réalisée par l'université de Bonn³ a montré que les développeurs ont tendance à choisir une méthode simple et à écrire du code qui stocke les mots de passe de manière non sécurisée.

Ainsi, parmi les systèmes de stockage de mots de passe sécurisés que les développeurs ont choisi de mettre en œuvre pour cette étude, seuls les deux derniers, PBKDF2 et Bcrypt, sont considérés comme sécurisés. Les systèmes de stockage rencontrés avec leurs occurrences sont les suivants :

- 8 - Base64
- 10 - MD5
- 1 - SHA-1
- 3 - 3DES
- 3 - AES
- 5 - SHA-256
- 1 - HMAC / SHA1
- 5 - PBKDF2
- 7 - Bcrypt

Base64 ne fournit que l'encodage. MD5 est une fonction de hachage 'appuyant sur des algorithmes obsolètes.

Le PBKDF2 est une fonction de dérivation de clé, appartenant à la famille des normes Public Key Cryptographic Standards, plus précisément PKCS #5 v2.0. Cette norme a également été publiée dans la RFC 2898. Elle succède au PBKDF1, qui pouvait produire des clés n'allant que jusqu'à 160 bits.

Bcrypt est une fonction de hachage créée par Niels Provos et David Mazières. Elle est basée sur l'algorithme de chiffrement Blowfish.

Précision de vocabulaire :

L'encodage permet uniquement de traduire ou convertir des données pour les rendre plus facilement échangeables entre deux systèmes. Ce qui est encodé peut être décodé par la seule connaissance de la fonction de codage utilisée (base 64 par exemple). Le chiffrement rend illisible une donnée chiffrée par celui qui ne possède pas la clé de déchiffrement.

VI. DEFI N°1 : SCAN DES ALGORITHMES DE CHIFFREMENT D'UNE APPLICATION WEB

Le premier défi est le scan d'un serveur web afin de lister les technologies de chiffrement configurées afin d'apprécier leur pertinence dans le cadre d'un audit.

Outils

Documents à utiliser :

- document 1 : Recommandations de sécurité relatives à TLS de l'ANSSI ;
- document 2 : Le protocole TLS-1.0 et les cartes de paiement.

Le programme sslscan :

L'outil sslscan permet de détecter la configuration SSL/TLS d'un site et de connaître les dates de création et de péremption des certificats. Les configurations SSL/TLS sont détectées et affichées avec les algorithmes utilisés ainsi que leur version. _____

Installation sous Linux :

Il faut installer le paquet sslscan depuis les dépôts.

```
prof@host555:~$ sslscan -h

  sslscan

      2.0.7
      OpenSSL 1.1.1n  15 Mar 2022

Command:
  sslscan [options] [host:port | host]


Options:
  --targets=<file>      A file containing a list of hosts to check.
                        Hosts can be supplied with ports (host:port)
  --sni-name=<name>     Hostname for SNI
  --ipv4, -4            Only use IPv4
  --ipv6, -6            Only use IPv6
  --show-certificate    Show full certificate information
  --show-client-cas     Show trusted CAs for TLS client auth
  --no-check-certificate Don't warn about weak certificate algorithm or keys
  --ocsp                Request OCSP response from server
  --pk=<file>           A file containing the private key or a PKCS#12 file
                        containing a private key/certificate pair
  --pkpass=<password>   The password for the private key or PKCS#12 file
  --certs=<file>        A file containing PEM/ASN1 formatted client certificates
```

```
#sudo apt install sslscan
```

Utilisation en ligne :

L'outil peut s'utiliser en ligne depuis le lien suivant :

<https://www.ssllabs.com/ssltest/>

 **Qualys. SSL Labs**

HomeProjectsQualys Free TrialContact

You are here: [Home](#) > [Projects](#) > SSL Server Test

SSL Server Test

This free online service performs a deep analysis of the configuration of any SSL web server on the public Internet. **Please note that the information you submit here is used only to provide you the service. We don't use the domain names or the test results, and we never will.**

Hostname:

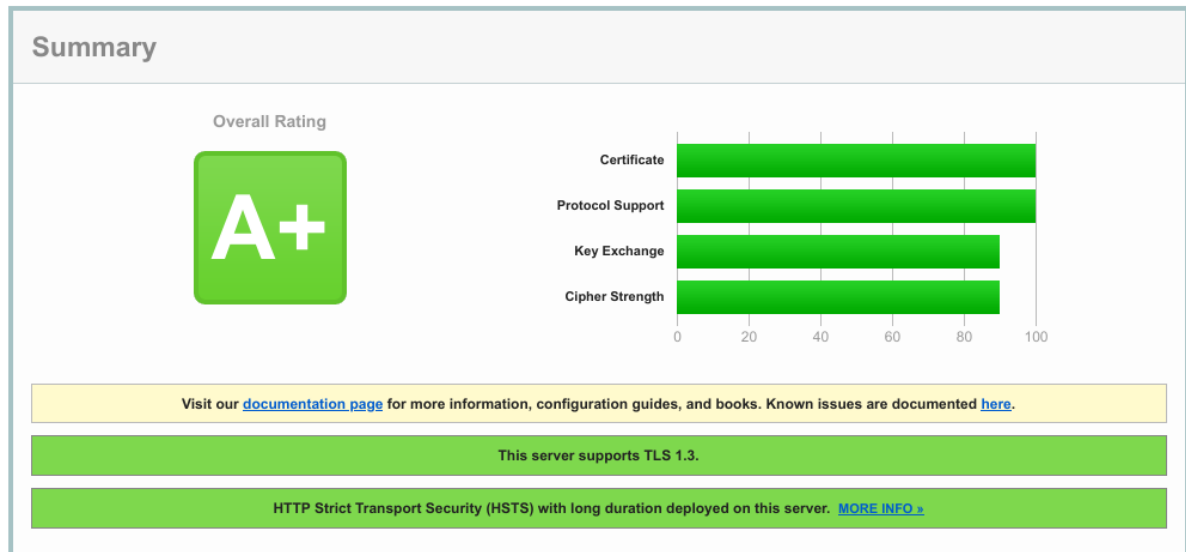
Submit

☐ Do not show the results on the boards

Cet outil teste les éléments suivants :

- Émetteur du certificat, validité, algorithme utilisé pour signer ;
- Détails du protocole, suites de chiffrement, simulation de prise de contact.

Les résultats des tests fournissent des informations techniques détaillées ainsi que des conseils à destination de l'administrateur système, de l'auditeur ou de l'ingénieur de sécurité Web pour connaître et corriger les paramètres faibles.



D'autres outils en ligne existent comme :

- SSL checker :

<https://www.thesslstore.com/ssltools/ssl-checker.php>

- Geekflare :

<https://geekflare.com/fr/tools/tls-test>

À vous de jouer

1. Définir le protocole TLS. Expliquer la différence entre TLSv1.2 et TLSv1.3 en terme de performance et de sécurité.
2. Qu'est ce qu'un certificat de sécurité SSL ? Quel est son rôle dans la sécurisation d'une application web ?
3. Qu'est ce qu'une autorité de certification ? Quelle différence entre une autorité privée et une autorité publique. Retrouver la liste des autorités publiques sur votre navigateur.
4. Tester la sécurité SSL/TLS des applications suivantes en utilisant les outils de scan :

- Google Gruyere

<https://google-gruyere.appspot.com/start>

-BWAP (noter le résultat du test de vulnérabilité Heartbleed)

<http://itsecgames.com/>

- Une application de votre choix développée en atelier de professionnalisation.

Reporter les résultats observés sur votre compte-rendu puis conclure.

5. A l'aide de vos recherches sur internet, expliquer ce qu'est la vulnérabilité Heartbleed. Quelles peuvent être les conséquences de cette vulnérabilité ?

VII. DEFI N°2 : FORCE BRUTE D'UN MOT DE PASSE HACHE EN MD5

Le deuxième défi est une attaque en force brute sur un mot de passe haché en MD5.

Outils et documents

Documents à utiliser :

- document 3 : Chiffrement symétrique et asymétrique ;

- document 4 : Fonctions de hachage d'après la CNIL. _____

Le programme hashcat :

Hashcat est un outil qui permet de retrouver des mots de passe à partir de plusieurs exemples de hachés.

1. Hashcat prend comme entrée un ensemble de mots à partir d'un dictionnaire ou par combinaisons ;
2. Hashcat calcule ensuite le haché pour chacun de ces mots puis compare le résultat avec le ou les hachés contenus dans un autre fichier qui stocke des hachés ciblés ;
3. Les coïncidences sont des mots de passe récupérés.

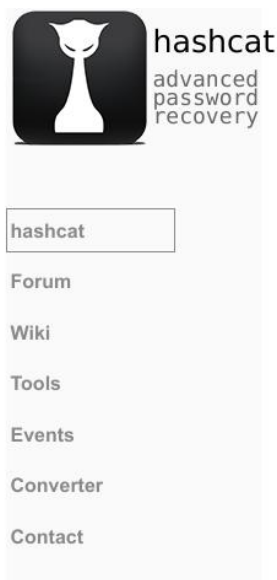
Cas d'usage :

- disponibilité de la base SAM⁴ sous Windows ou du fichier */etc/shadow*⁵ sous Linux ;
- récupération de hachés dans le code d'un programme ou dans une base de données.

Installation :

```
# sudo apt install hashcat
```

Le lien suivant donne le mode d'emploi de l'outil: <https://www.kali-linux.fr/hacking/comment-cracker-des-mots-de-passe-avec-hashcat>



```
hashcat (v6.2.1) starting...
CUDA API (CUDA 11.3)
=====
* Device #1: NVIDIA GeForce RTX 2080 Ti, 10137/11264 MB, 68MCU

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates

Optimizers applied:
* Optimized-Kernel
* Zero-Byte
* Precompute-Init
* Early-Skip
* Not-Iterated
* Prepend-Salt
* Single-Hash
* Single-Salt
* Brute-Force
* Raw-Hash

Watchdog: Temperature abort trigger set to 90c
Host memory required for this attack: 1100 MB
e983672a03adcc9767b24584338eb378:00:hashcat
```

Autres outils en ligne :

<https://crackstation.net/>

<https://hashes.com/en/decrypt/hash>

Free Password Hash Cracker

Enter up to 20 non-salted hashes, one per line:

42f749ade7f9e195bf475f37a44cafcb

Je ne suis pas un robot

reCAPTCHA

Confidentialité - Conditions

Crack Hashes

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1 sha1_bin), QubesV3.1BackupDefaults

Hash	Type	Result
42f749ade7f9e195bf475f37a44cafcb	md5	Password123

Le système détermine que le hachage MD5 de la chaîne Password123 a produit le haché fourni.

A vous de jouer

1. Rappeler la différence entre chiffrement symétrique et chiffrement asymétrique et proposer un schéma pour chacune de ces deux méthodes.
2. Recenser et décrire les principaux algorithmes de chiffrement symétrique et asymétrique.
3. Qu'appelle t-on une fonction de hachage, quel peut-être son rôle lors du développement d'une application. Lister les principales fonctions de hachage et leurs principales caractéristiques.
4. Expliquer ce que fait le code java suivant et quel est le problème de sécurité posé par ce code ?

```
@Test
public void givenFile_generatingChecksum_thenVerifying()
    throws NoSuchAlgorithmException, IOException {
    String filename = "src/test/resources/test_md5.txt";
    String checksum = "5EB63BBE01EEED093CB22BB8F5ACDC3";

    MessageDigest md = MessageDigest.getInstance("MD5");
    md.update(Files.readAllBytes(Paths.get(filename)));
    byte[] digest = md.digest();
    String myChecksum = DatatypeConverter
        .printHexBinary(digest).toUpperCase();

    assertThat(myChecksum.equals(checksum)).isTrue();
}
```

5. Dans le code Symfony suivant, expliquer le rôle de 'auto'. Conclure sur l'intérêt d'utiliser un framework lors du développement d'une application.

Source du code : <https://symfony.com/doc/current/security/passwords.html>

```
# config/packages/security.yaml
security:
# ...
password_hashers:
# auto hasher with default options for the User class (and children)
App\Entity\User: 'auto'
# auto hasher with custom options for all PasswordAuthenticatedUserInterface instances
Symfony\Component\Security\Core\User\PasswordAuthenticatedUserInterface:
algorithm: 'auto'
cost: 15
```

6. Un attaquant récupère le haché suivant dans une base de données.

4138dd8a9c4ac256fcf4a42b633f9f88 Trouver le mot de passe correspondant avec hashcat ou avec un autre outil en ligne.

VIII. BONNES PRATIQUES ET CONTRE-MESURES

Bonnes pratiques d'après le groupe OWASP :

Déterminer d'abord quelles données doivent bénéficier d'une protection chiffrée (mots de passe, données patients, numéros de cartes, données personnelles, etc.), lors de leur transfert ou leur stockage. Pour chacune de ces données :

- Les données circulent-elles en clair ? Ceci concerne les protocoles tels que HTTP, SMTP et FTP. Le trafic externe sur internet est particulièrement dangereux.
- Des algorithmes faibles ou désuets sont-ils utilisés, soit par défaut, soit dans le code source existant ?
- Est-ce que des clés de chiffrement par défaut sont utilisées ? Des clés de chiffrement faibles sont-elles générées ou réutilisées ? Leur gestion et rotation sont-elles prises en charge ?
- Les réponses transmises au navigateur incluent-elles les directives et en-têtes de sécurité adéquats ?
- Est-ce que le certificat serveur reçu est valide ?
- Est-ce que des fonctions de hachage dépréciées telles que MD5 ou SHA1 sont utilisées ou est-ce que des fonctions de hachage non cryptographiques sont utilisées là où des fonctions de hachage cryptographiques sont nécessaires ?
- Est-ce que des méthodes cryptographiques de remplissage dépréciées, comme PKCS 1 v1.5 sont utilisées ?

Contre-mesures : d'après OWASP.

On veillera au minimum à suivre les recommandations suivantes :

- Classifier les données traitées, stockées ou transmises par l'application. Identifier quelles données sont sensibles selon les lois concernant la protection de la vie privée, les exigences réglementaires ou les besoins métier.
- Ne pas stocker de données sensibles sans que cela ne soit nécessaire. Les rejeter ou utiliser une tokenisation⁶ conforme à la norme de sécurité de l'industrie des cartes de paiement (PCI DSS). Les données que l'on ne possède pas ne peuvent être volées.
- S'assurer de chiffrer toutes les données sensibles au repos.
- Choisir des algorithmes éprouvés et générer des clés robustes. S'assurer qu'une gestion des clés est en place.
- Chiffrer toutes les données transmises avec des protocoles sécurisés tels que TLS. Forcer le chiffrement en utilisant des directives comme HTTP Strict Transport Security (HSTS).
- Désactiver le cache pour les réponses contenant des données sensibles.
- Appliquer les contrôles de sécurité requis selon la classification de la donnée.
- Ne pas utiliser des protocoles obsolètes tels que FTP et SMTP pour échanger des données sensibles.
- Stocker les mots de passe en utilisant des fonctions de hachage avec salage et facteur de délai, telles que Argon2, scrypt, bcrypt ou PBKDF2.
- Les vecteurs d'initialisation doivent être choisis de façon appropriée au mode d'opération. Pour la plupart des modes, cela signifie utiliser un générateur de nombres pseudo-aléatoires cryptographiquement sécurisé (CSPRNG⁷). Dans tous les cas, un vecteur d'initialisation ne devrait pas être utilisé deux fois pour une clé fixe.
- S'assurer qu'une génération cryptographiquement aléatoire est utilisée là où c'est approprié, et qu'elle n'a pas une graine aléatoire prévisible ou avec une faible entropie.

Dossier documentaire

Document 1 : Recommandations de sécurité relatives à TLS de l'ANSSI

D'après ssi.gouv.fr.

Le protocole TLS est une des solutions les plus répandues pour la protection des flux réseau. Dans ce modèle client–serveur, les données applicatives sont encapsulées de manière à assurer la confidentialité, l'intégrité et empêcher leur rejeu. Le serveur est nécessairement authentifié, et des fonctions additionnelles permettent l'authentification du client si nécessaire.

Depuis l'apparition de son prédécesseur SSL en 1995, TLS a été adopté par de nombreux acteurs de l'Internet pour sécuriser le trafic lié aux sites web et à la messagerie électronique. Il s'agit par ailleurs d'une solution privilégiée pour la protection de flux d'infrastructure internes. Pour ces raisons, le protocole et ses implémentations font l'objet d'un travail de recherche conséquent. Au fil des années, plusieurs vulnérabilités ont été découvertes, motivant le développement de corrections et de contre-mesures pour prévenir la compromission des échanges.

Le déploiement TLS apportant le plus d'assurance en matière de sécurité repose donc sur l'utilisation de logiciels mis à jour, mais aussi sur l'ajustement des paramètres du protocole en fonction du contexte. Les explications apportées par le présent guide sont complétées par plusieurs recommandations visant à atteindre un niveau de sécurité conforme à l'état de l'art, notamment au sujet des suites cryptographiques à retenir.

Le document suivant détaille le problème posé par TLS v1.1 :

https://www.ssi.gouv.fr/uploads/2016/09/guide_tls_v1.1.pdf

Document 2 : Le protocole TLS-1.0 et les cartes de paiement

Le protocole TLS 1.0 est officiellement obsolète en raison de problèmes de sécurité. Il n'est pas considéré comme un système de cryptographie fort d'après la norme de sécurité de l'industrie des cartes de paiement PCI (Payment Card Industry).

Document 3 : Chiffrement symétrique et asymétrique

Chiffrement symétrique : une clé unique est utilisée pour chiffrer et déchiffrer le message. Avec cette méthode, le chiffrement est simple. Sa faiblesse réside dans la transmission de la clé à un correspondant.

Chiffrement asymétrique : deux clés (clé publique, clé privée) sont nécessaires pour chiffrer et déchiffrer le message. La clé publique peut être transmise à des tiers et la clé privée est confidentielle. Cette technique permet de transmettre la clé de chiffrement symétrique à un tiers de manière confidentielle.

Document 4 : Fonctions de hachage d'après la CNIL

D'après cnil.fr.

L'utilisation d'une fonction de hachage permet de ne pas stocker les mots de passe en clair dans la base mais uniquement de stocker une empreinte de ces derniers. Il est important d'utiliser un algorithme public réputé fort afin de calculer les dites empreintes. A ce jour, MD5 ne fait plus partie des algorithmes réputés forts.

De même, les fonctions de hachage publiques réputées fortes étant par nature à la disposition de tous, il est techniquement possible pour tout un chacun de calculer des empreintes. Aujourd'hui, on trouve facilement sur internet des dictionnaires immenses d'empreintes MD5 précalculées et, grâce à ces données, il est aisé de retrouver instantanément le mot de passe ayant été utilisé afin de générer ces empreintes. Afin de limiter ce risque, il est conseillé d'utiliser des fonctions spécialisées appelées « fonction de dérivation de clé », telles que scrypt ou Argon2 par exemple, qui sont conçues spécifiquement pour stocker des mots de passe.

1 *l'article 32 oblige en effet responsables et sous-traitants à « garantir un niveau de sécurité adapté au risque », en optant pour une série de techniques de protection comme la pseudonymisation et le chiffrement des données à caractère personnel, et « des moyens permettant de garantir la confidentialité, l'intégrité, la disponibilité et la résilience constantes des systèmes et des services de traitement ».*

2 *Rainbow table : une **rainbow table** (littéralement table arc-en-ciel) est, en cryptanalyse, une structure de données permettant de retrouver un mot de passe à partir de son empreinte.*

3 *<https://www.zdnet.fr/actualites/selon-une-etude-les-developpeurs-preferent-la-simplicité-a-la-securite-pour-les-mots-de-passe-39881839.htm>*

4 *Le gestionnaire de compte de sécurité (SAM) est une base de données présente sur les ordinateurs exécutant des systèmes d'exploitation Windows qui stocke les comptes d'utilisateur et les descripteurs de sécurité pour les utilisateurs sur l'ordinateur local.*

5 */etc/shadow est un fichier texte qui contient les informations sur les mots de passe des utilisateurs du système.*

6 *Procédé permettant de remplacer une donnée critique par un élément équivalent qui n'aura aucune valeur intrinsèque ou signification exploitable une fois sortie du système. Lorsqu'un jeton de session est utilisé, l'utilisateur bénéficie, pendant toute la durée de vie du jeton, d'un accès à l'application ou au site web pour lequel le jeton lui est accordé.*

7 *CSPRNG : Un générateur de nombres pseudo-aléatoires cryptographiquement sécurisé ou un générateur de nombres pseudo-aléatoires cryptographiques est un générateur de nombres pseudo-aléatoires avec des propriétés qui le rendent adapté à une utilisation en cryptographie.*

8 *PCI (Payment Card Industry) : La norme de sécurité de l'industrie des cartes de paiement est un standard de sécurité des données qui s'applique aux différents acteurs de la chaîne monétique. La norme PCI DSS est établie par les cinq principaux réseaux cartes et est gérée par le Conseil des normes de sécurité PCI.*