Cryptographie

ISI

3 - Cryptographie

Résumé du document

Dans ce document, nous allons explorer les principes fondamentaux de la cryptographie, qui est l'art de sécuriser les communications en transformant des données de manière à les rendre inintelligibles pour quiconque ne possède pas la clé de déchiffrement. Nous aborderons différentes techniques de chiffrement symétrique et asymétrique, ainsi que les méthodes de hashage, d'authentification et de signature numérique.

Table des matières

1. Vocabulaire

1.1. Cryptographie

- Chiffrer / Chiffrement
- Déchiffrer / Déchiffrement

Clé connue

1.2. Cryptanalyse

- Decrypter
- Decryptage

Clé inconnue

1.3. Autres

- Plaintext = texte en clair
- Ciphertext = texte chiffré
- HW = Hardware
- SW = Software

2. Chiffrement

2.1. Chiffre de César

Le chiffre de César est une méthode de chiffrement très simple qui consiste à décaler les lettres de l'alphabet d'un certain nombre de positions. Par exemple, avec un décalage de 3, A devient D, B devient E, etc.

2.2. Vernam Cipher (One-Time Pad)

Le chiffrement de Vernam est une méthode de chiffrement qui utilise une clé aléatoire de la même longueur que le message à chiffrer. La clé est utilisée une seule fois et n'est jamais réutilisée. C'est un chiffrement parfaitement sûr, mais il est difficile à mettre en place car il nécessite une clé de la même longueur que le message à chiffrer.

2.3. Cryptographie symétrique (clé secrète)

La cryptographie symétrique est une méthode de chiffrement qui utilise une seule clé pour chiffrer et déchiffrer un message. La clé doit être connue des deux parties pour que le message puisse être déchiffré. Dans ce type de cryptographie, la sécurité du message repose entièrement sur la sécurité de la clé.

- La même clé est requise pour le chiffrement et le déchiffrement.
- K est secrète : seuls, l'émetteur et le récepteur doivent la connaître. On partage un secret. La gestion des clés est critique.
- Comme pour tous les algorithmes cryptographiques, la sécurité repose sur la clé.
- L'émetteur et le récepteur doivent se faire confiance.
- Pas de notion d'authentification de l'origine (comme la signature).

2.4. Block Cipher vs Stream Cipher

- Block Cipher : Chiffrement par blocs (AES, DES, …)
  - Avantages : Facile à implémenter, sécurisé, adapté pour du SW
- Stream Cipher : Chiffrement par flux (RC4, …)
  - Avantages : Très haut débit, adapté pour du HW

	Block cipher	**Stream cipher**
Granularité	Large (blocs de 64/128 bits)	Fine (bit, octet)
Débit	Haut debit	Très haut débit
Taille	Beaucoup de place en silicium Adapté pour du SW	Très peu de place en silicium Adapté au HW
Sécurité	++ (confusion et diffusion)	+ (confusion)
Remarque	Nécessite un mode de chiffrement (et padding)	Nécessite un nonce (valeur à usage unique)

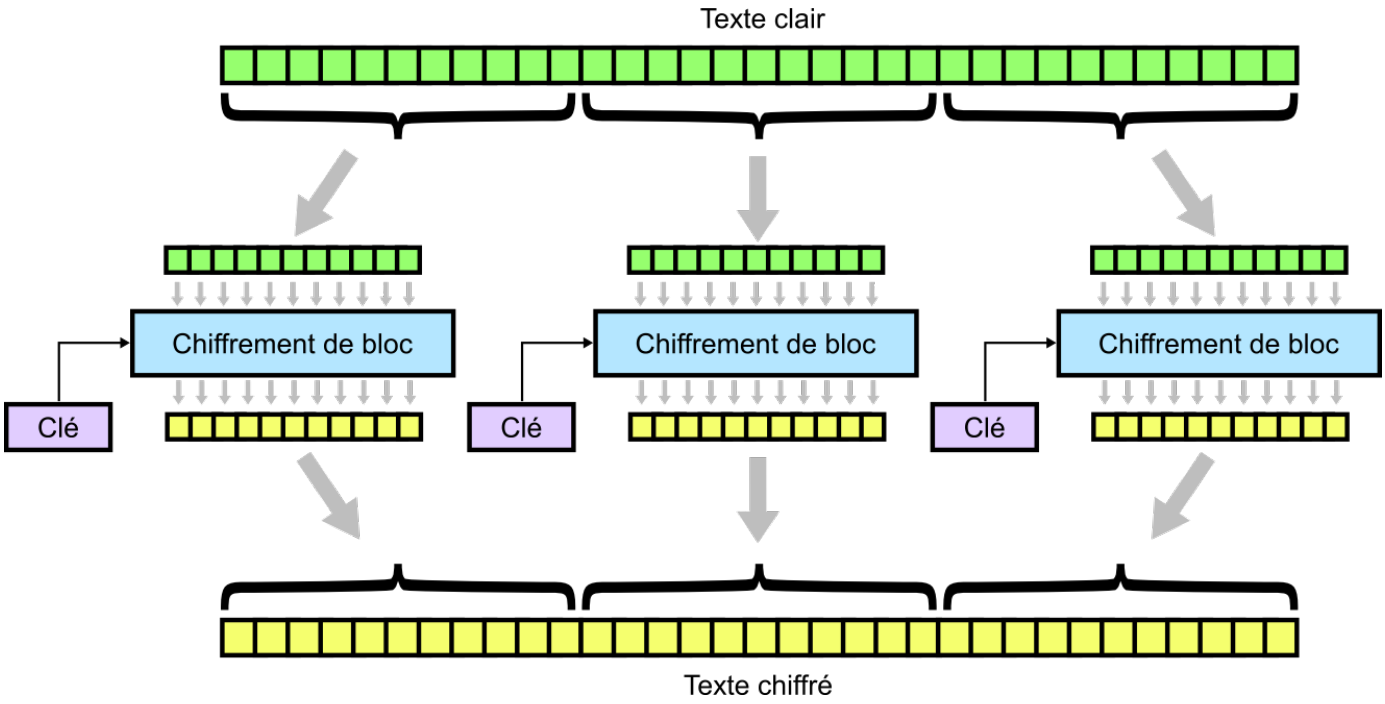
Confusion : complexité de la relation entre la clé de chiffrement et le texte chiffré.

Diffusion = l'inversion d'un seul bit en entrée doit changer chaque bit en sortie avec une probabilité de 0,5 (critère d'avalanche strict).

2.4.1. Exemple d'algorithme

2.4.1.1. Block Cipher

- DES (1976, FIPS 46)
- Triple-DES (1998, FIPS 46-3)
- AES (2001, FIPS 197)
- IDEA (1991, ETHZ)
- IDEA-NXT (2003, EPFL/Mediacrypt/Kudelski)



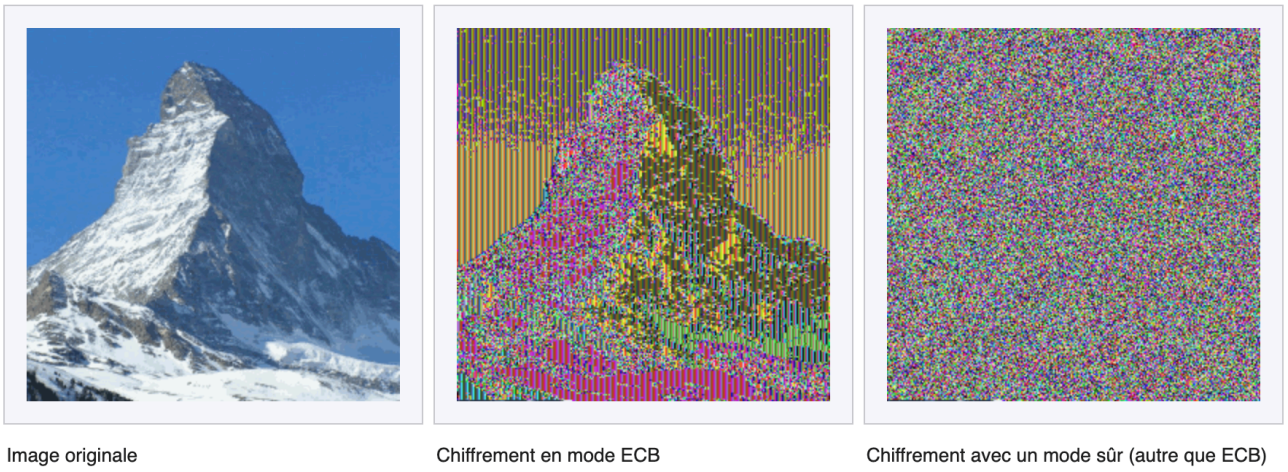


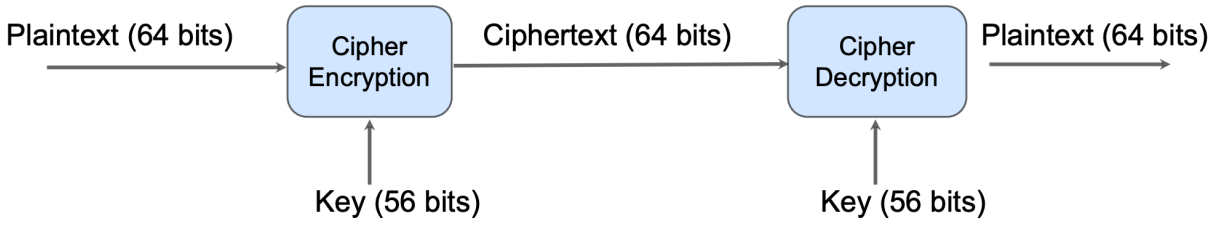
Image originale | Chiffrement en mode ECB | Chiffrement avec un mode sûr (autre que ECB)

2.4.1.2. Stream Cipher

- Chacha20 (2008)
- RC4 (WEP/WPA, 1987 sous licence, "public" 1994)
- A5/1 (GSM, 1987, "public" 1994-99)
- E0 (Bluetooth, 199x)

2.5. DES (Data Encryption Standard)

Le DES (Data Encryption Standard) est un algorithme de chiffrement par blocs qui utilise une clé de 56 bits. Il a été développé dans les années 1970 par IBM et est devenu un standard du gouvernement américain en 1977. Le DES a été remplacé par l'AES (Advanced Encryption Standard) en 2001 en raison de sa vulnérabilité aux attaques par brute force. Il est en partie dédié pour les implémentations en HW.



2.5.1. Faiblesses du DES

- Même en supposant DES sûr, la recherche exhaustive d'une clé est possible
  - 56 bits de clé : 256 ≅ 7 x 1016 clés possibles.
  - Hypothèse : 1 µs par essai (⟹ 10^6 essais / seconde)
  - Temps moyen : ≅ 1000 ans
- Copacobana (2006 - 2008)
  - Réseau de FPGAs
  - 48 milliards déchiffrements DES par seconde ($48 * 10^9$)
  - Temps requis moyen : ≅ 8 jours

2.6. Triple-DES

Le Triple-DES est une version améliorée du DES qui utilise trois clés de 56 bits pour un total de 168 bits. Il est plus sûr que le DES, mais il est plus lent et moins efficace en termes de performances. Il est en partie dédié pour les implémentations en HW.

- 3key-TDES
  - 3 clés de 56 bits, soit 168 bits !
  - Sécurité équivalente à 2key-TDES
- 2key-TDES (**le plus courant**)
  - 2 clés de 56 bits, soit 112 bits
  - Sécurité acceptable
  - Très déployé actuellement (surtout en embarqué, HW / SW)
- TDES est beaucoup trop lent (en SW)
- **AES est son successeur**

2.7. AES (Advanced Encryption Standard)

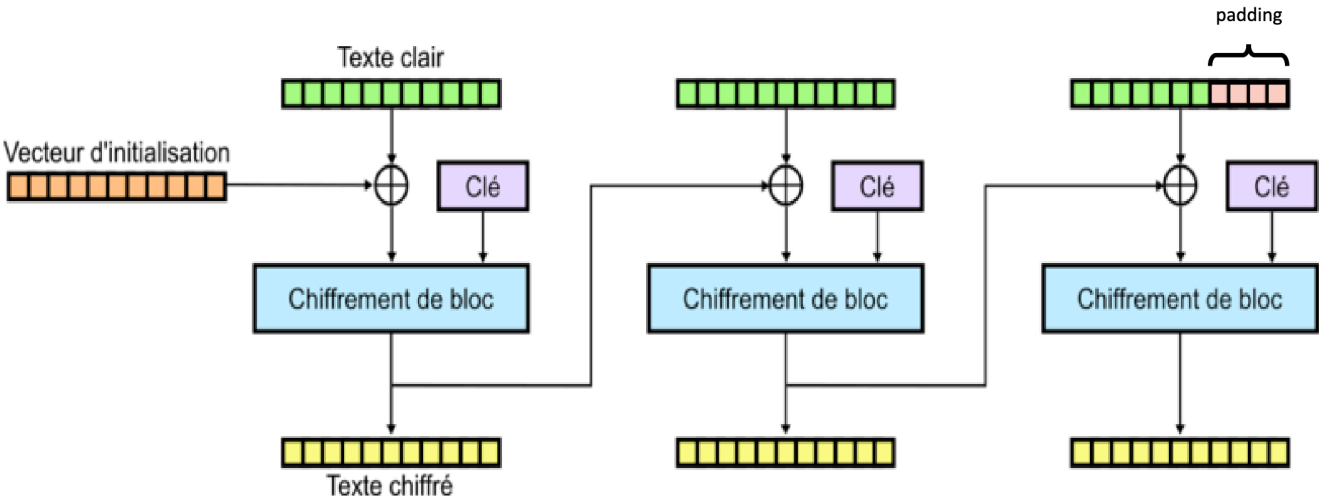
L'AES (Advanced Encryption Standard) est un algorithme de chiffrement par blocs qui utilise des clés de 128, 192 ou 256 bits. Il a été adopté par le gouvernement américain en 2001 pour remplacer le DES. L'AES est plus rapide et plus sûr que le DES, et il est devenu un standard mondial pour le chiffrement de données.

2.8. Modes de chiffrement

- ECB : Electronic Code Book (sans mode de chiffrement)
- CBC : Cipher Block Chaining
- CFB : Cipher Feedback
- OFB : Output Feedback
- CTR : CounTeR
- **GCM** : Galois/Counter Mode (chiffrement authentifié, très répandu en raison de son efficacité)
- CTS : CipherText Stealing (technique permettant d'éviter le padding)

2.8.1. CBC (Cipher Block Chaining)

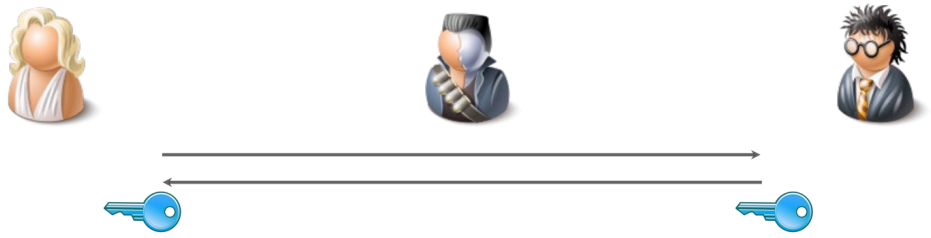
- Chaque bloc est chiffré avec la clé et le bloc précédent



3. Cryptographie asymétrique (clé publique)

3.1. Diffie-Hellman

- La distribution des clés symétriques secrètes a toujours été un problème.
  - Coûteux : acheminement, nombre nécessaire : n(n-1)/2
- Diffie et Hellman résolvent ce problème en inventant le concept suivant, **fondement de la cryptographie à clé publique** :



- **Sans secret** en commun a priori.
- Ce concept permet d'**échanger des clés ou messages secrets**

3.2. Protocole Diffie-Hellman

La distribution des clés symétriques secrètes a toujours été un problème. Diffie et Hellman résolvent ce problème en inventant le concept suivant, fondement de la cryptographie à clé publique.

- **Sans secret** en commun a priori.
- Ce concept permet d'**échanger des clés ou messages secrets**

Le protocole de Diffie-Hellman se base sur des fonctions mathématiques qui sont faciles à calculer dans un sens mais difficiles à inverser. Il permet à deux parties de s'échanger des clés secrètes.

1. Alice et Bob choisissent un nombre premier p et un générateur g.
2. Alice choisit un nombre secret a et calcule $A = g^a \bmod p$.
3. Bob choisit un nombre secret b et calcule $B = g^b \bmod p$.
4. Alice envoie A à Bob et Bob envoie B à Alice.
5. Alice calcule $K = B^a \bmod p$ et Bob calcule $K = A^b \bmod p$.
6. Alice et Bob ont maintenant une clé secrète K qu'ils peuvent utiliser pour chiffrer et déchiffrer leurs messages.

3.3. RSA

RSA est un algorithme de cryptographie asymétrique qui utilise une paire de clés publique/privée pour chiffrer et déchiffrer des messages. Il a été inventé en 1977 par Ron Rivest, Adi Shamir et Leonard Adleman et est devenu un standard mondial pour la cryptographie à clé publique.

4. Hashage

4.1. Hash

- Engagement / preuve
  - Caractérise un message de manière unique, sans le révéler
  - Application :
    - engagement qui est «binding» et «hiding» (voir exemple “pile ou face”)
    - mots de passe
- Extension de domaine
  - Transformer n’importe quelle longueur en une longueur fixe
  - Application :
    - signatures digitale avec le «hash-and-sign»
- Génération de nombres pseudo-aléatoires
  - Générer des nombres imprédictibles à partir d’une graine («seed»)
  - Application :
    - dérivation de clés cryptographiques à partir d’un «seed»

4.1.1. Exemples

- Schémas d’engagement
- Stockage (des témoins) de mots de passe
- Dérivation de clés cryptographiques
  - par exemple clé à partir d’une phrase de passe (« passphrase »)
- « File Integrity Assessment » (HIDS)

4.2. Fonction de hashage

- MD : «Message Digest»
  - Auteur : Ronald Rivest
  - MD2, 1989, hachés de 128 bits (RFC1319), obsolète.
  - MD4, 1990, hachés de 128 bits (RFC1320), cassé.
  - MD5, 1991, hachés de 128 bits (RFC1321), cassé.
- SHA : «Secure Hash Algorithm»
  - Auteur : NSA / NIST
  - FIPS 180-3 : Secure Hash Standard (SHS), 2008
  - SHA-0, 1993, hachés de 160 bits, obsolète.
  - SHA-1, 1995, hachés de 160 bits, cassé.
  - SHA-2, 2002, hachés de 224-512 bits, recommandé.
    - Famille : SHA-224, SHA-256, SHA-384, SHA-512
    - Codes hachés de 224, 256, 384, 512 bits
  - SHA-3, accepté en 2012, recommandé.

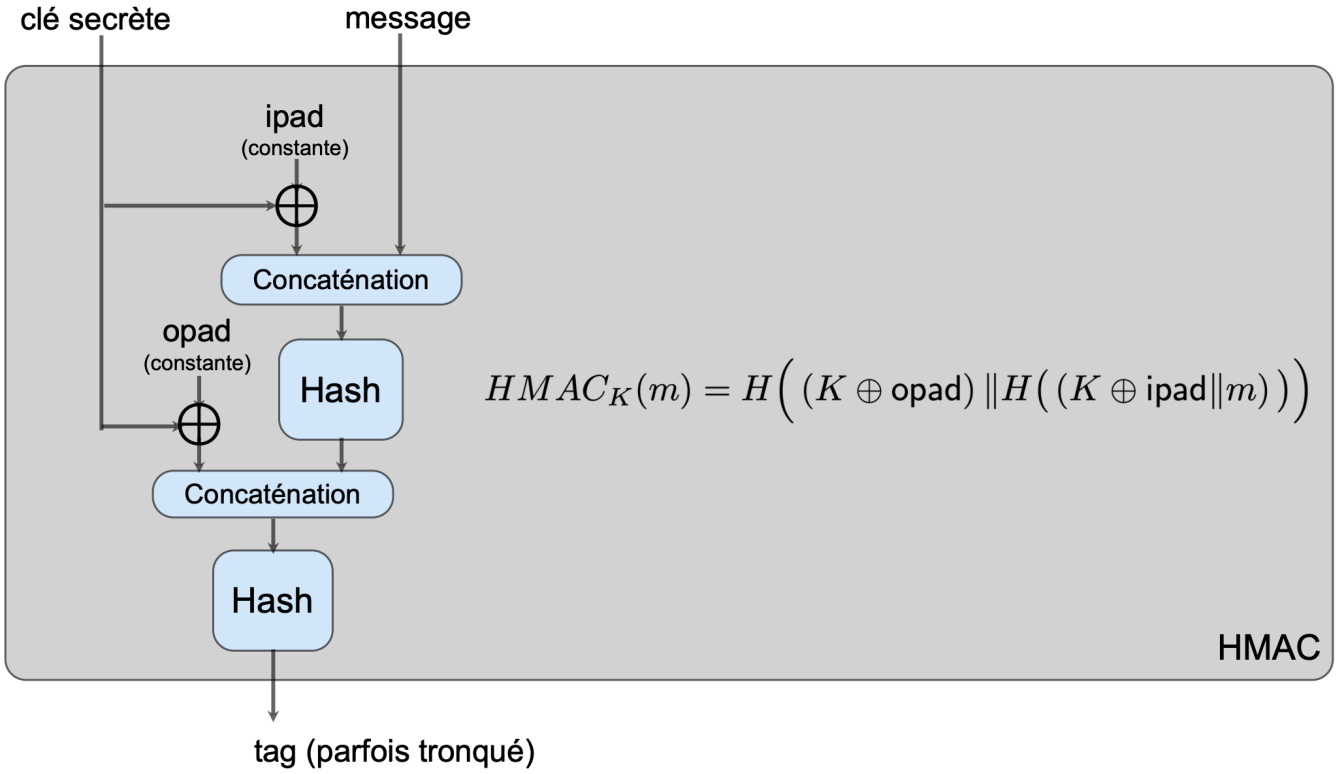
5. MAC

- Un MAC (Message Authentication Code)
  - code de taille fixe
  - envoyé avec un message afin de prouver
    - une « origine » (laquelle?)
    - et contrôler son intégrité
- But :
  - s'assurer de la provenance des données (authentification)
  - implique protection de l'intégrité des données
- Exemples :
  - authentifier les échanges d'un protocole (par exemple HTTP)
    - authentifier une mesure (dans le cas où la signature n'est pas envisageable)

5.1. Constructions

- Constructions basées sur des fonctions de hachage
  - Construction naïve H(K,m) **VULNÉRABLE**
  - HMAC **SÉCURISÉ**
- Constructions basées sur des algorithmes de chiffrement
  - CBC-MAC **VULNÉRABLE**
  - ISO/IEC 9797 **SÉCURISÉ**

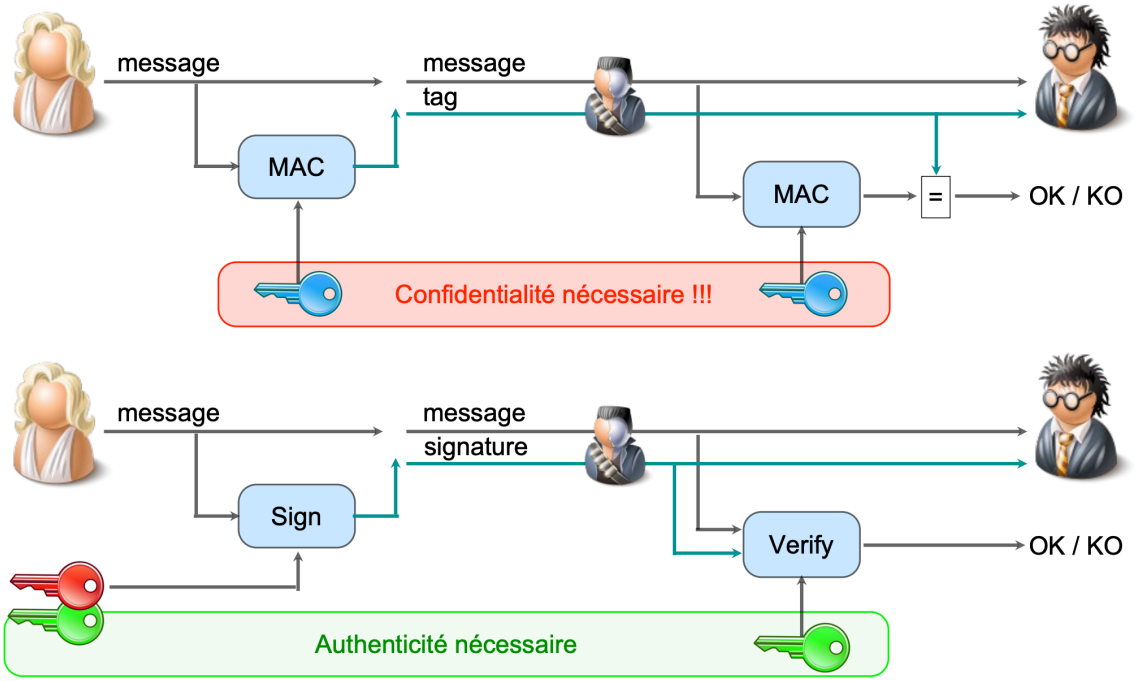
5.1.1. HMAC



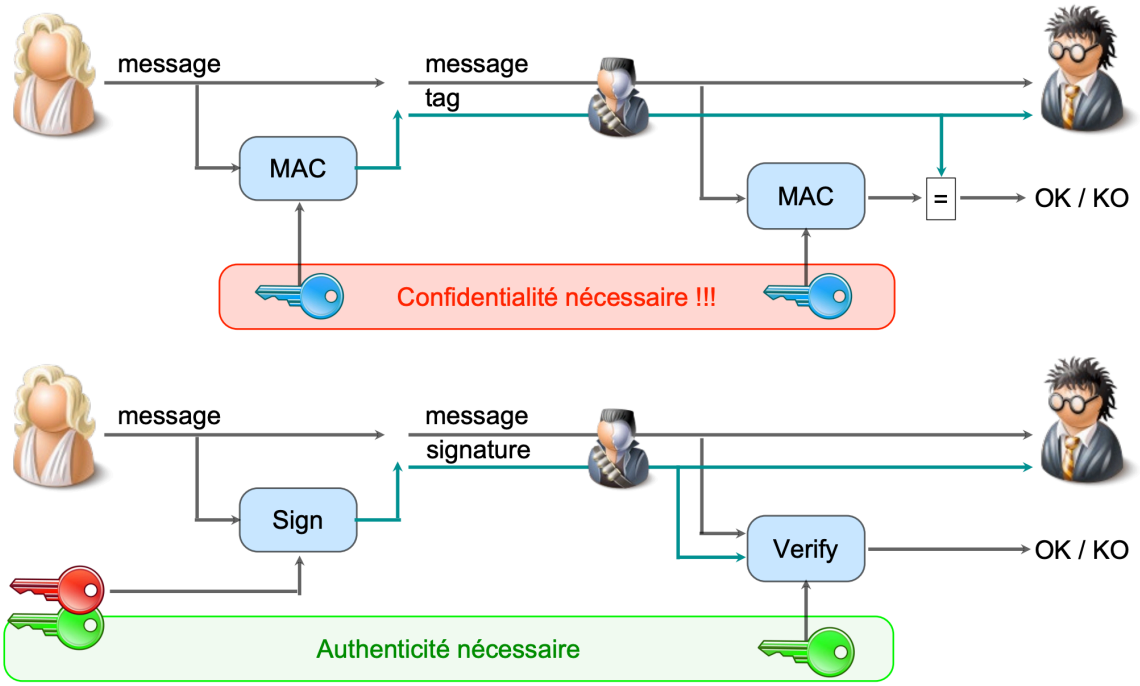
6. Signature

- Une signature est envoyée avec un message afin de prouver son «origine» et son intégrité au destinataire.
- De plus, la signature empêche le déni de la part du signataire. Une signature est une preuve non répudiable.
- But :
  - Authentique : s'assurer de la provenance des données
  - Non-répudiable : prouver la provenance des données
  - Inaltérable : implique protection de l'intégrité des données
  - Infalsifiable : la signature ne peut pas être falsifiée.
  - Non réutilisable: la signature est liée à un document et ne peut être déplacée sur un autre document.
- Exemple :
  - authentifier les messages d'un Diffie-Hellman
  - authentifier une mise-à-jour
  - contrats/documents, etc.

6.1. Signature vs. chiffrement



6.2. Signature vs. MAC



6.3. Non-répudiation

- La non-répudiation est la propriété d'un système qui garantit qu'une action ou un événement ne peut pas être nié par la partie qui l'a réalisé. On parle donc de la protection contre le déni d'implication.

Preuve d'origine

- fournit au récepteur la preuve de l'origine des données.
- empêche l'envoyeur de nier avoir généré les données qu'il a effectivement générées.

Preuve de livraison

- fournit à l'émetteur la preuve de livraison des données;
- empêche le récepteur de nier avoir reçu les données qu'il a effectivement reçues.

7. Authentification

L'authentification est le processus de vérification de l'identité d'une personne ou d'un système. Il permet de s'assurer que la personne ou le système est bien celui qu'il prétend être.

7.1. Facteurs d'authentification

- Authentification par mot de passe
- Authentification par carte à puce, SecureID, badge
- Authentification par empreinte digitale, reconnaissance faciale

7.1.1. Authentification forte

L'authentification forte est un processus d'authentification qui nécessite plus d'une méthode pour vérifier l'identité d'une personne. Elle doit combiner deux des types d'authentification cités précédemment.

7.2. Types d'authentification

- Authentification par jeton passif («constant token»)
  - Exemple : mot de passe
- Authentification par jeton actif («fresh token»)
- Authentification par question/réponse («challenge/response»)
- Authentification à clé publique
- Authentification biométrique
- Authentification par un autre canal (SMS, email, etc.)

7.3. Authetification par jeton actif

- Secret unique partagé
- Jeton frais, changeant dans le temps
- Équivalent dans les faits à un OTP (One Time Password).

Fonctionnement dit synchrone

7.4. Authentification par challenge/réponse

- Question/réponse avec utilisation d'un secret partagé
- Question avec nonce : «Number used ONCE»

Fonctionnement dit asynchrone

7.5. Authentification à clé publique

- Question/réponse avec signature électronique d'un nonce
- Nonce : « Number used ONCE » (aléatoire de taille suffisante ou compteur)
- Authentification utilisée dans SSH (par exemple avec les algorithmes : RSA et SHA2)

7.6. Risques associés

- « Man in the middle » :
  - Alice transmet sa clé publique à Bob avec un message disant que c'est la sienne.
  - Remplacement par une fausse clé publique ?
  - Comment Bob peut-il être sûr que le message et/ou la clé n'ont pas été modifiés ?

8. Infrastructure à clé publique (PKI)

La PKI (Public Key Infrastructure) est un ensemble de technologies, de politiques et de procédures qui permettent de gérer les clés publiques et privées, les certificats numériques et les autorités de certification.

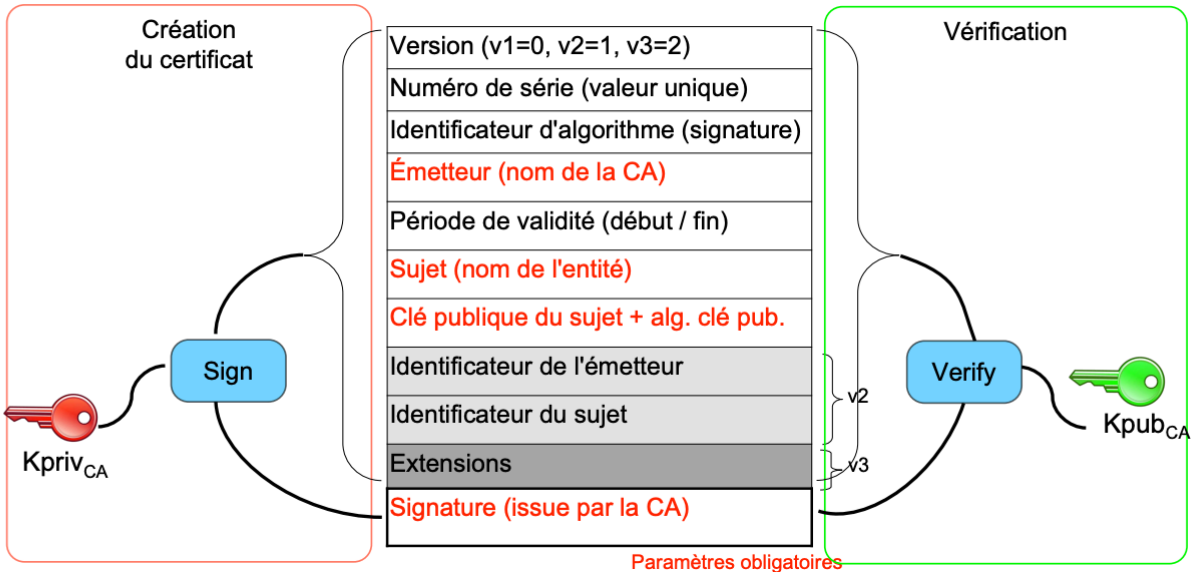
8.1. Certificat numérique

Un certificat numérique est un document électronique qui lie une clé publique à une entité (personne, organisation, site web, etc.). Il est signé par une autorité de certification (CA) et permet de vérifier l'identité de l'entité et de garantir l'intégrité des données.

- Ce lien est certifié par une autorité tierce :
  - Souvent l'autorité de certification (Certificate Authority ou CA)
  - Cela peut aussi être fait de manière ad-hoc (entre utilisateurs)
- Concrètement :
  - Un certificat est un fichier texte 1 KB.
  - Il contient surtout
    - le nom de l'entité,
    - la clé publique associée,
    - la signature de la CA (et donc sur la clé publique)

8.1.1. Norme X.509v3

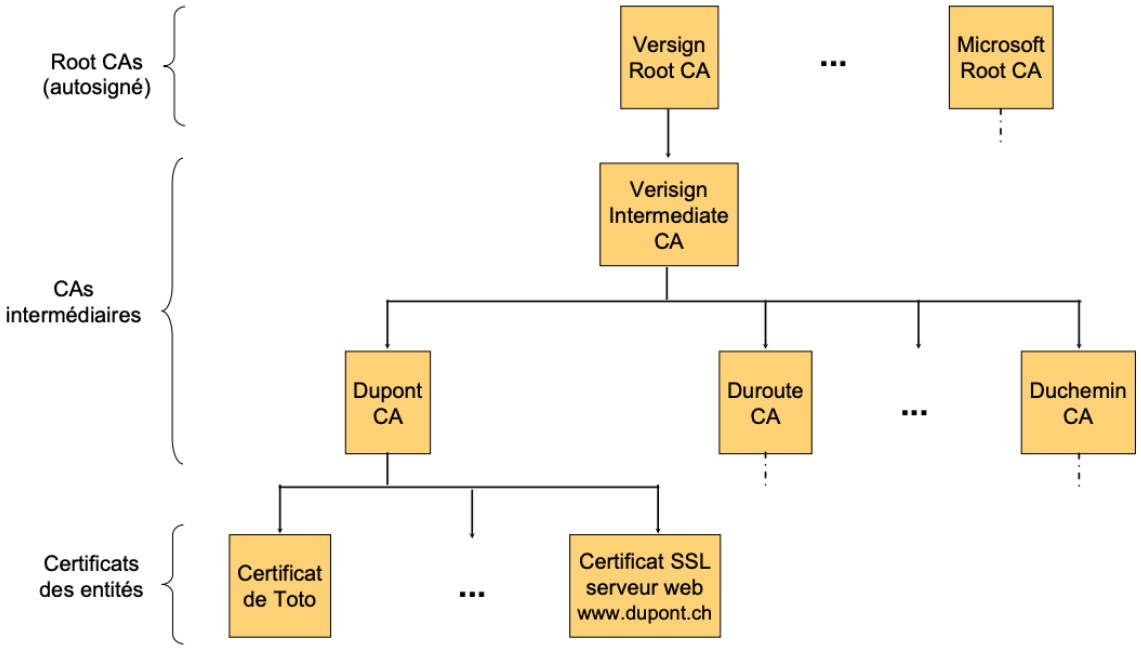
- Standard de l'ITU-T pour les certificats numériques



8.2. Utilisation des certificats

- TLS (SSL) :
  - authentification du serveur (Web, SMTP/StartTLS, IMAPS, etc.)
  - authentification du client (optionnel)
- Autres authentifications :
  - IPSec
  - Smartcard logon (PIN pour débloquer l'accès à la carte, après certificat)
- Signatures :
  - email (S/MIME, PGP, GPG),
  - code exécutable (Win SRP)

8.3. Chaines de certification



8.4. Trusted Root CAs

- Les navigateurs web et les systèmes d'exploitation contiennent une liste de CA racines de confiance.
- Cela évite aux utilisateurs de les télécharger
  - plus pratique
  - la sécurité devient « transparente » pour les utilisateurs
  - moins de risques de télécharger de faux certificats ou des certificats dangereux, mais les risques ne disparaissent pas (cf. l'affaire Diginotar)

8.5. PKI

8.5.1. Infrastructure à clé publique

- PKI : Public Key Infrastructure = infrastructure à clé publique
  - RFC 3280
  - Ensemble d'éléments matériels et logiciels, protocoles et services
  - Rôle : gestion des clés publiques à grande échelle
- Mécanismes :
  - Enregistrement et émission de certificats
  - Stockage et distribution (annuaire)
  - Révocation et vérification de statut
  - Éventuellement : sauvegarde et récupération

8.5.2. Durée de vie d'un certificat

- Expiration :
  - Chaque certificat contient une date d'expiration
- Révocation :
  - Un certificat peut être révoqué avant son expiration
  - Exemples :
    - clé privée d'une entité ou clé privée d'une CA compromise (heartbleed ou autre)
    - personne quittant l'entreprise
  - Liste de révocation des certificats (« annulés ») « Certificate Revocation List » (CRL)

8.5.3. Entités et services

- **CA : « Certificate Authority »**

- Émettre, renouveler, maintenir les certificats et CRLs.
- Sécurité essentielle (idéalement séparée physiquement, bunker)
- CA publiques, par exemple : Verisign, Microsoft, etc.
  - Dans les faits, définies par la liste des CAs contenue dans les navigateurs ou les systèmes d'exploitation
- CAs privées, par exemple : PKI interne d'entreprise

- **RA : « Registration Authority »**
  - Gérer les demandes : stockage, contrôle des données soumises, communication avec l'entité, vérification du respect de la politique de certification
  - Publier les certificats et CRLs Exemples de RAs : chambres de commerce, banques, poste, etc.
- **VA : « Validation Authority »**
  - Service de contrôle de certificats (signature, date, validité, etc.)
  - Annuaire (« directory ») : contient les certificats et CRLs.
  - Exemple de protocole de vérification :
    - OCSP (Online Certificate Status Protocol), RFC 2560.

8.5.4. Règles

- CPS : **« Certification Practice Statement »**
  - Déclaration des pratiques utilisées par une CA pour émettre les certificats.
  - Exemples sur la verification des identités :
    - Verisign CPS certificats de classe 1 :
      - nom + email (individus).
    - Verisign CPS certificats de classe 2 :
      - formulaire à remplir + vérification dans bases de données (individus).
    - Verisign CPS certificats de classe 3 :
      - présence physique (individus) ou documents officiels (organisations).
- CP : **« Certificate Policy »**
  - Règles sous lesquelles le certificat a été émis (cf. CPS) et types d'utilisations autorisées.
- Formats/syntaxes : série PKCS (Public-Key Cryptography Standards) par RSA Laboratories :
  - PKCS5 : chiffrement à partir d'un mot de passe (RFC2898).
  - PKCS10 : demande de certificat (RFC2986).
  - PKCS12 : stockage de la clé privée dans un fichier avec protection d'un PIN (ex: container IE ou Firefox).

8.5.5. Points critiques

- Génération des clés :
  - Idéalement aléatoire.
  - Utilisation de bons PRNG (cf. affaire Debian)
- Stockage de la clé privée.
  - Solutions : HSM (Hardware Security Module), clés USB, smartcards.