Outils et applications de la cryptographie web et mobile

Techniques de Compression et utilisation d'outils Cryptographiques web et mobile

Birahime DIOUF, Docteur en Télécoms et Réseaux Enseignant chercheur

Cryptographie dans les applications web et mobiles

Mise en œuvre de méthodes cryptographiques

- A l'ère de la cybercriminalité, la protection des données et systèmes devient un enjeu de taille pour les différents utilisateur d'Internet (entreprises, administrations ou particuliers).
  - Tandis que le web et les terminaux mobiles et sont de plus en plus intégrés à notre environnement de travail, la sécurité représente un défi toujours plus important.
  - La sécurité en développement web et mobile est l'une des questions qui préoccupent aussi bien les développeurs que les utilisateurs.
  - Applications web et mobiles sont la cible privilégiée des pirates. Elles peuvent leur permettre d'accéder à des infos sensibles de l'utilisateur, voire pénétrer dans le système informatique d'une entreprise.
  - Sécuriser une application commence par l'estimation du risque qu'elle peut faire courir.
- Pour cela la cryptographie est l'outils le plus utilisée et dans de nombreuses applications :
  - Sur Internet, elle permet d'assurer la confidentialité, l'intégrité et l'authentification de l'émetteur :
    - messageries électroniques ou
    - commerce électronique : les transmissions de code d'une carte bancaire pour achat en ligne.
  - Téléphonie : la cryptographie est utilisée pour assurer la confidentialité des communications.
  - *sécurité des échanges Bluetooth* ou *wifi*.

Cryptographie dans les applications web et mobiles

Mise en œuvre de méthodes cryptographiques

- Cryptographie : moyen de sécurité très efficaces, à la disposition des développeurs web et mobiles pour réduire l'impact et le coût d'une violation de données.
  - Le développeur web ou mobile doit savoir quel type de cryptographie convient à quel contexte, même s'il ne comprend pas les opérations mathématiques utilisées dans leur moindre détail.
  - Des méthodes de cryptographie mal appliquées n'apportent aucun avantage et peuvent même s'avérer dangereuses en procurant un faux sentiment de sécurité.

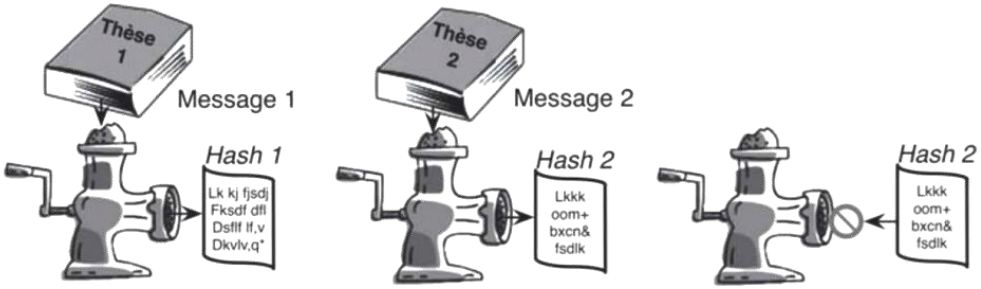
- Les crypto-systèmes permettant d'assurer la confidentialité et l'intégralité des données et d'authentifier des utilisateurs dans les applications web et mobiles sont basés sur :
  - hachage des données
  - signature électronique
  - certificats numérique

- Les protocoles de cryptographie, notamment en basés sur TLS, sont utilisés pour sécuriser les connexions et les applications web et mobiles.

Contrôle de l'intégrité des données : fonction de hachage

Fonction de hachage

- La cryptographie sans secret est basée sur de fonctions de hachage cryptographique.
- En informatique, une fonction de hachage est une fonction mathématique qui prend en entrée des données de différentes tailles, et qui fournit en sortie des donnée de taille fixe appelée empreinte ou hash ou encore digest.
- Deux propriétés essentielles :
  - elle est irréversible : impossible de retrouver le message lorsqu'on connaît le hash ;
  - elle est résistante aux collisions : deux messages différents ne produiront jamais (en théorie) le même hash ⟹ une modification même infime du message entraîne une modification du hash.
- Si un message est transmis avec son hash, le destinataire peut vérifier son intégrité en recalculant son hash et en le comparant avec le hash reçu (voir figure).
- Il est beaucoup plus rapide de comparer des hashs (**format fixe** et **petite taille)** que des données (grande taille et variable) ⟹ fonctions de hachage très utilisées en informatique.



Contrôle de l'intégrité des données : fonction de hachage

Applications des fonctions de hachage cryptographiques

- Ces fonctions sont largement utilisées :
- **Contrôler l'intégrité d'un message**
  - 1ère application de fonction de hachage : protéger l'intégrité d'un message ou d'un fichier.
  - L'empreinte d'un fichier peut également servir d'identifiant unique du fichier, dans la mesure où le risque d'une collision est pratiquement nul. Ce type de fonction cryptographique est conçu de façon qu'une modification même infime du message initial entraîne une modification du hash.
  - Si un message est transmis avec son hash, le destinataire peut vérifier son intégrité en recalculant son hash et en le comparant avec le hash reçu.
- **Stocker des mots de passe**
  - Les mots de passe sont des données particulièrement sensibles.
  - Le serveur ne stocke donc pas le mot de passe dans sa base de données, mais plutôt son hash
  - Lorsque l'utilisateur fournit son mdp pour s'authentifier, le serveur calcule le hash de ce mdp et le compare avec la valeur du hash enregistré. Si les hashs sont identiques, le mdp est valide.
  - Si un attaquant vole la base de données du serveur, il ne pourra pas retrouver les mdp;
  - L'inconvénient est que si l'utilisateur oublie son mot de passe, le serveur ne pourra pas le lui redonner, car il ne le connaît pas lui-même.

Contrôle de l'intégrité des données : fonction de hachage

Hash : exemples

- Les algorithmes de hachage les plus utilisés sont :
  - MD5 (*Message Digest 5*) sur 128 bits (16 octets). il est possible de trouver des collisions.
  - SHA-1 (*Secure Hash Algorithm*) : Input message 264 octets (au max), output sur 160 bits (20 octets), aujourd'hui considérée comme vulnérable, car il est possible de trouver des collisions.
  - SHA-2 (2002) : fonction standard de hachage cryptographique, la plus utilisée actuellement. Plusieurs versions, dont SHA-256 dont le haché est de 256 bits.
- D'autres fonctions de hachage :
  - SHA-3 (2015) : alternative à SHA-2, ne la remplace pas mais servira d'alternative à l'avenir.
  - MD2 : opère sur des bloques de 16 octets, manipule des mots de 8 bits output 128 bits.
  - MD4 : manipule des mots de 32 bits, plus performant sur des processeurs 32 bits.

Exemple de hash :

MD5 (Jesaispas) =
EFB7929E6A5B7DCC6EBB79AA3C45AF13

sur 128 bits

Nom	Empreinte	Propriétés de l'algorithme
MD5	128 bits	Message Digest v5 (issu du MIT)
SHA1 SHA2	160 bits	Secure Hash Algorithm (issu du NIST) : algorithme de hachage **beaucoup plus sûr**

Authentification des utilisateurs : signature électronique

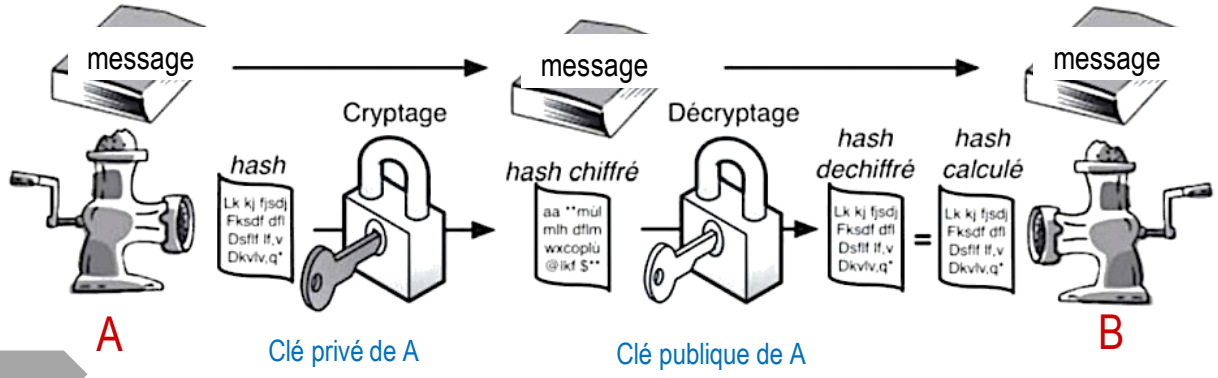
Signature

- Equivalent électronique de la signature physique des documents papiers.
- Signature utilise le chiffrement à clé publique (asymétrique). Les algorithmes RSA, DSA (*Digital Signature Algorithm*) et ElGamal sont souvent utilisés pour générer une signature.
- Pour signer électroniquement un message, il suffit de le chiffrer avec la clé privée.
- Le déchiffrement avec la clé publique correspondante prouve que seul le détenteur de la clé privée a pu créer la signature.
- Pas nécessaire de chiffrer tout un document pour le signer, il suffit de chiffrer son hash.
- La résistance aux collisions de la fonction de hachage permet de garantir que c'est bien ce document qui a été signé.
- Signature garantit :
  - l'authenticité de l'expéditeur : doit permettre au lecteur d'un document d'identifier la personne ou l'organisme qui a apposé sa signature.
  - l'intégrité du message reçu : doit garantir que le document n'a pas été altéré.
  - Une signature est infalsifiable, non réutilisable, inaltérable et irrévocable

Authentification des utilisateurs : signature électronique

Signature

- Exemple d'utilisation de la signature : messagerie électronique
- Coté émetteur du message :
  - Avant d'envoyer un message, l'outil de messagerie calcule d'abord l'empreinte du message.
  - Pour réaliser la signature, il chiffre ensuite cette empreinte avec la clé privée de l'émetteur.
  - Avant l'envoi, cette signature est ajoutée au message, qui devient un **message signé**.
- Coté destinataire :
  - L'outil de messagerie du destinataire déchiffre l'empreinte chiffrée avec la clé pub de l'émetteur.
  - Puis il recalcule le hash sur le message reçu et compare le résultat avec l'empreinte déchiffrée.
  - Si les 2 sont égaux, cela veut dire que le message n'a pas été modifié durant le transfert et que l'émetteur est authentifié.



Authentification des utilisateurs : signature électronique

Signature RSA

- **Paramètres** :
  - Choisir deux nombres premiers $\boldsymbol{p}$ et $\boldsymbol{q}$, les deux étant plus grands que 2^{10}.
  - Calculer $n = p \cdot q$ (n est le modulus) et $\varphi(n) = (p-1) \cdot (q-1)$ (fonction d'Euler)
  - Choisir $\boldsymbol{e}$ aléatoire tel que e et $\varphi(n)$ soient premiers entre eux (c-à-d $pgcd(e, \varphi(n)) = 1$).
  - Trouver $\boldsymbol{d}$ tel que : $e * d = 1 \ mod(\varphi(n))$ (c-à-d $e = d^{-1} \ mod(\varphi(n))$)
  - Clé publique : (n, e).
  - Clé privée : (n, d) ou (p, q, d) si on désire garder p et q.
- **Signature** :
  - L'expéditeur signe le message m avec sa clé privée d :

    $s = m^d \ mod(n)$ et envoie m avec s
- **Vérification signature** :
  - Le destinataire vérifie sa signature avec la clé publique de l'émetteur e :

    $sd = s^e \ mod(n)$ et vérifie si $m = sd$
- La signature RSA utilise les algorithmes de hachage MD5 et SHA-1.

Authentification des utilisateurs : signature électronique

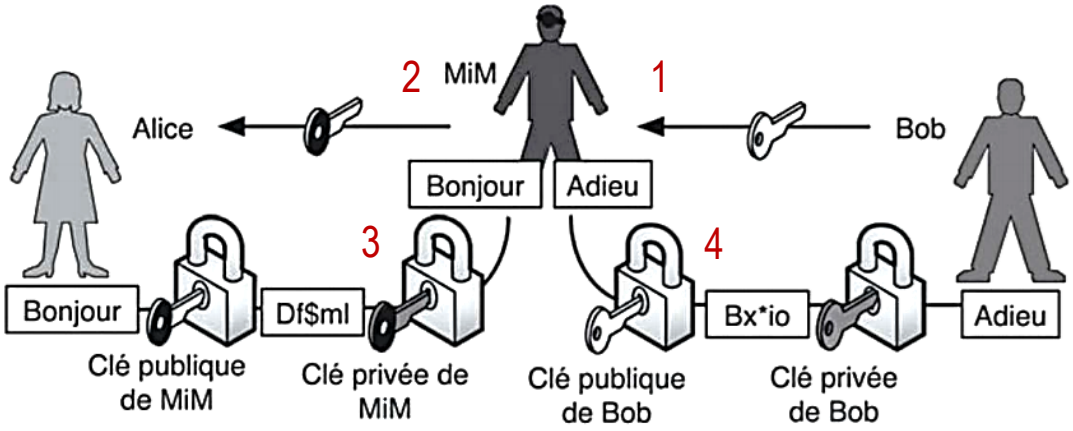
Signature DSA, DSS, EL Gamal

- ElGamal : crypto-système à clé publique utilisé pour la signature et basé sur les algorithmes discrets. Versions récentes implantent les courbes elliptiques.
- Amélioration de cette algorithme par DSS (*Digital Signature Standard*) avec une signature plus courte qu'El Gamal pour une sécurité identique.
- La signature DSA utilise l'algorithme de hachage SHA-1.
- DSS spécifie l'algorithme de signature numérique DSA développé par la NSA (*National Security Agency*) et a été proposé par le NIST en 1994 pour devenir la norme du gouvernement des USA pour authentifier des documents électroniques.
- DSS est spécifié dans la norme FIPS (*Federal Information Processing Standard*).
- 4 révisions ont été apportées par rapport aux spécifications initiales : FIPS 186-1 (en 1998), FIPS 186-2 (en 2000), FIPS 186-3 (en 2009) et FIPS 186-4 (en 2013).

Vérification identité émetteur de clé publique : certificats

Certificat numérique : intérêt

- Dans l'utilisation de la cryptographie asymétrique un utilisateur obtient la clé publique d'une personne en consultant un annuaire (ou un serveur web). Mais qu'est-ce qui garantit que cette clé publique ainsi récupérée est la bonne ? Elle peut être sujette à l'interception.
  - Un pirate (MiM, *Man in the Middle*) peut modifier l'annuaire ou le serveur web. Scénario :
    1. intercepte la clé publique,
    2. remplace la clé publique par la sienne et la transmet à l'expéditeur,
    3. déchiffre avec sa fausse clé privée correspondante les messages envoyés par l'expéditeur et
    4. les retransmet au destinataire après les avoir éventuellement modifiés.
    5. Une fois cette mascarade commise, MiM pourra lire les courriers confidentiels envoyés et signer des messages en se faisant passer pour l'émetteur.



Vérification identité émetteur de clé publique : certificats

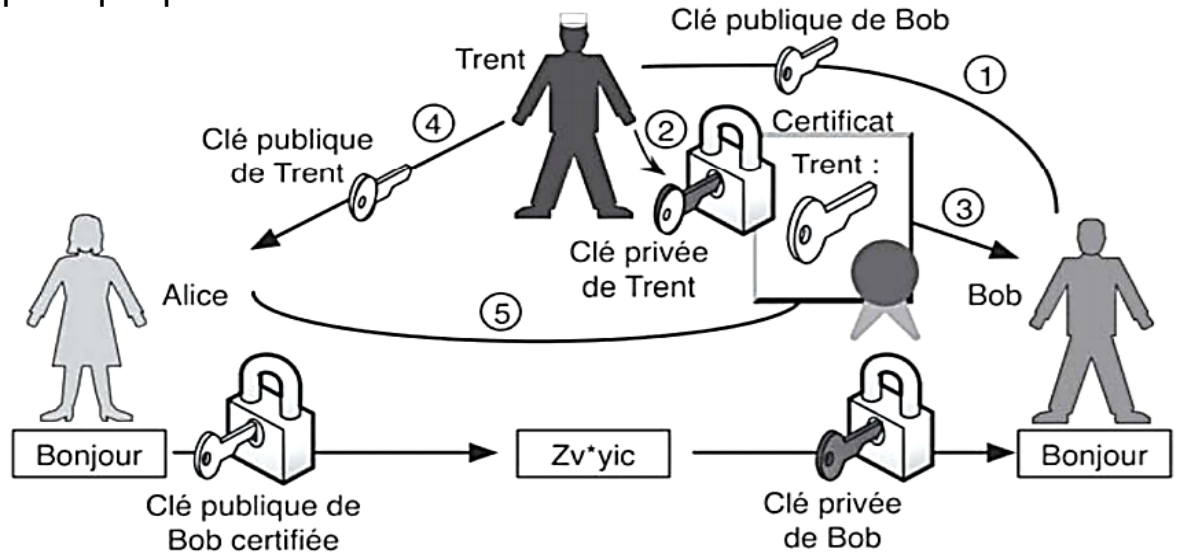
Certificat numérique :

- Pour garantir l'origine d'une clé publique, l'une des solutions est d'utiliser un certificat électronique (numérique) fourni en même temps que la clé :
  - C'est l'équivalent électronique d'une carte d'identité ou d'un passeport.
  - Réalise l'association d'une clé publique à une entité (personne, machine, ...) afin d'en assurer la validité ;
  - Permet aux sites web, personnes et appareils de prouver l'authenticité de leur identité.
  - Il est donc nécessaire que le certificat provienne d'un tiers de confiance.
  - C'est le mécanisme de signature qui est utilisé pour garantir l'identité de ce tiers.
- Les certificats peuvent être :
  - certificats auto-signés : un périphérique génère son propre certificat et le signe en tant qu'étant valide. Ce type de certificat devrait avoir une utilisation limité.
  - Certificat signé par une autorité de certification : un tiers valide et authentifie 2 nœuds ou plus qui tentent de communiquer. Chaque nœud a une clé publique et une clé privée. Puisqu'ils ont obtenu leurs certificats de la même source, ils peuvent être assurés de leurs identités respectives.

Vérification identité émetteur de clé publique : certificats

Certificat d'autorité de certification

- Bob ne transmet pas directement sa clé publique à Alice.
- Il la transmet d'abord à Trent (1), le tiers de confiance.
- Ce dernier dispose lui aussi d'une paire de clés asymétriques, il utilise sa clé privée pour signer le certificat contenant la clé publique de Bob (2) et lui transmet celui-ci (3).
- Bob peut alors envoyer le certificat signé à Alice (5).
- Celle-ci, qui a également reçu la clé publique de Trent (4), peut vérifier la signature et être sûre que la clé publique provient bien de Bob.



Vérification identité émetteur de clé publique : certificats

Certificats et PKI sur internet

- La génération et la distribution sur Internet des certificats sont organisées autour des infrastructures à clé publique PKI (*Public Key Infrastructure*).
- Une PKI est une infrastructure composée d'un ensemble de systèmes, de procédures et de politiques très strictes, dont les fonctions sont les suivantes :
  - enregistrer les entités désirant obtenir des certificats électroniques ;
  - fabriquer des paires de clés privée et publique ;
  - certifier des clés publiques afin de créer des certificats et de publier ces derniers sur des annuaires publics, généralement des serveurs LDAP ;
  - révoquer des certificats et gérer des listes de révocation.

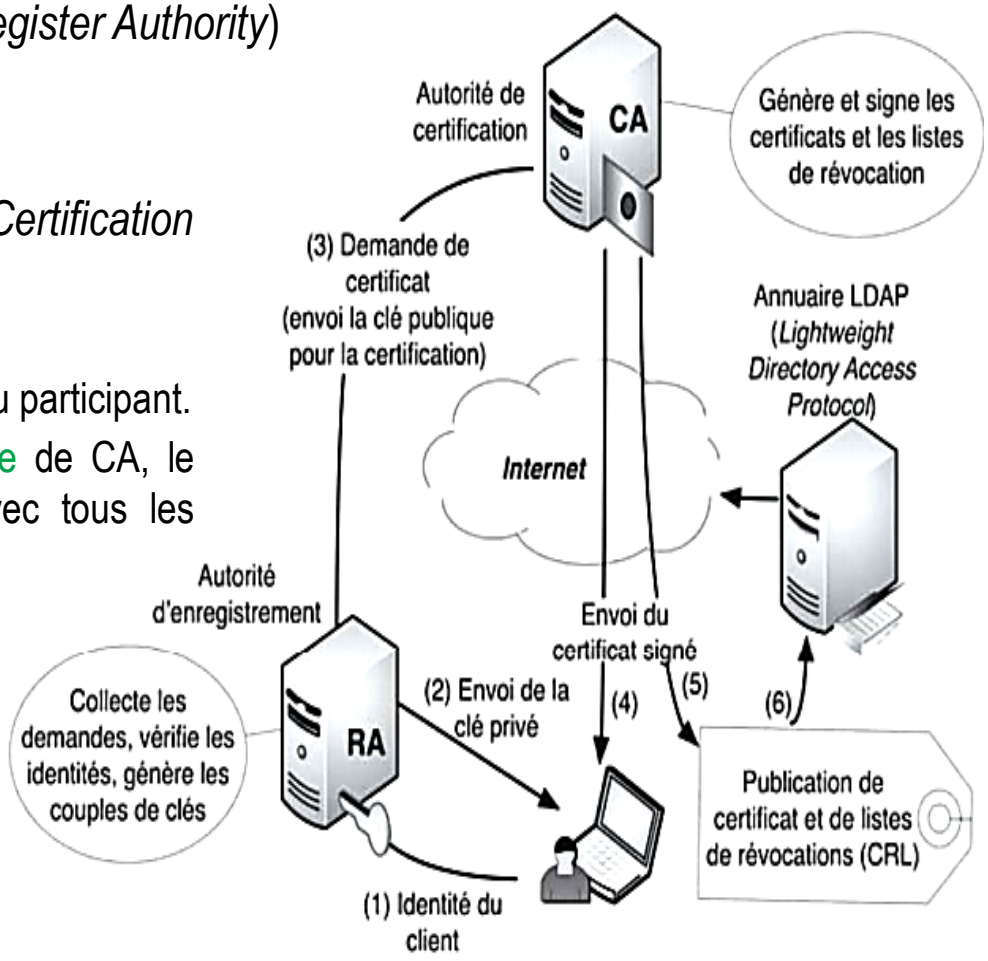
- **Certification avec PGP**
  - PGP (*Pretty Good Privacy*) a pour fonction d'offrir des services de confidentialité et d'authentification pour la messagerie électronique et le stockage de données.
  - gratuit et disponible sur la plupart des systèmes d'exploitation actuels.
  - crypto-système hybride (combinaison crypto à clé publique RSA et à clé secrète IDEA)

Vérification identité émetteur de clé publique : certificats

Certificats et PKI sur internet

- Une PKI est constituée des éléments suivants :

1. Une autorité d'enregistrement ou RA (*Register Authority*)
   - authentifie chaque nouveau participant.
   - génère une paire de clés publique/privée ;
2. Une autorité de certification ou CA (*Certification Authority*).
   - crée et signe les certificats
   - fournit une copie de sa propre clé publique au participant.
   - Muni de son certificat et de la clé publique de CA, le nouveau participant peut communiquer avec tous les autres participants certifiés par la même CA.
   - RA peut être intégrée à la CA.
2. annuaire de certification. .
   - généralement des serveurs LDAP
   - Contient les certificats publiés et certificats révoqués.



UVS
UNIVERSITE VIRTUELLE DU SENEGAL

Vérification identité émetteur de clé publique : certificats

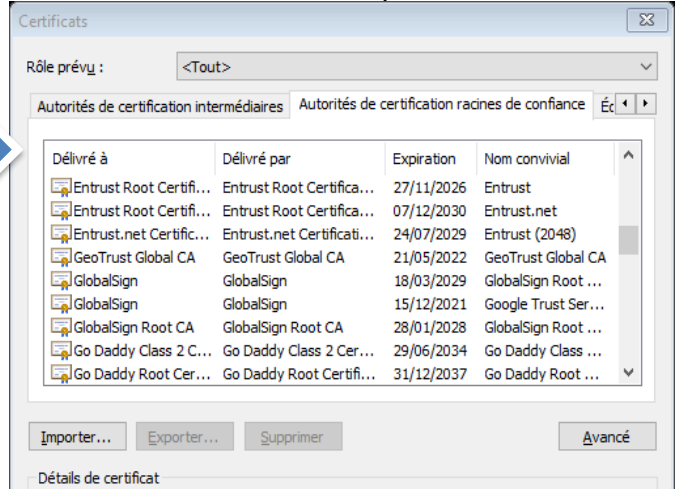
Certificats et PKI sur internet

- La plupart des certificats utilisés sur Internet sont au format X.509 (de l'UIT), utilisé par HTTPS, IPsec, PGP et SSH. Ce fichier transmis par la CA sur l'ordinateur du client contient :
  - version du certificat (ex: X.509 v3);
  - numéro de série fourni par la CA ;
  - algos de hachage et de signature utilisés ;
  - nom et clé publique du destinataire du certificat ;
  - l'identité de l'émetteur (CA) ;
  - la signature de la CA réalisée par chiffrement avec la clé privée du hash des informations précédentes ;
  - dates de validité du certificat, …
- Tout utilisateur possédant la clé publique de cette CA peut déchiffrer le hash et le comparer au calcul de son propre hash du certificat.

Vérification identité émetteur de clé publique : certificats

Autorités de Certification et les Hiérarchies de Confiance

- Un certificat SSL est un type de certificat numérique largement utilisé qui lie les informations de propriété d'un serveur web (d'un site web) à une clé cryptographique.
  - Pour qu'un navigateur puisse faire confiance à un certificat SSL il doit
    - contenir le nom de domaine du site web pour lequel il est utilisé,
    - avoir été émis par une CA de confiance et être valide.
  - Les navigateurs et les appareils décident de faire confiance à une CA en acceptant son certificat racine dans leur magasin de certificats racine (base de données contenant une liste de CA autorisées) qui a été préinstallée dans le navigateur ou sur l'appareil.
  - Les clés publiques des principales CA sont mémorisées (et éventuellement révoquées) dans les navigateurs (Internet Explorer, Opera, Firefox, Google Chrome, Safari, …). Pour voir les infos sur un certificat SSL sur googleChrome, aller à
    - ***paramètres***, puis ***confidentialité et sécurité*** , puis ***sécurité***, puis ***gérer les certificats***, puis ***autorités de certification intermédiaires*** ou ***racine de confiance***
  - Egalement, la majorité des fabricants d'appareils mobiles ont leur propre magasin de certificats racine.



UVS
UNIVERSITE VIRTUELLE DU SENEGAL

Vérification identité émetteur de clé publique : certificats

Utilisation des certificats numériques dans les applications Internet

- Applications Internet les plus courantes, qui utilisent les certificats numériques :
  - SSL : protocole utilisé par les serveurs web pour sécuriser les connexions entre ces derniers et les navigateurs web. SSL utilise des certificats numérique pour l'échange de clés.
  - Authentification client : le serveur demande et authentifie le certificat numérique du client lors de l'établissement de liaison et décide s'il fait confiance à l'AC ou pas.
  - Courrier électronique sécurisé : de nombreux systèmes de courrier électronique, utilisant des normes telles que PEM (*Privacy Enhanced Mail*) ou S/MIME (*Secure/Multipurpose Internet Mail Extensions*) pour la sécurisation du courrier, utilisent des certificats numériques pour signatures.
  - Réseaux privés virtuels VPN : configurés entre des pare-feu afin d'activer des connexions protégées (trafic crypté) entre des réseaux sécurisés ou entre un client éloigné (ex : un salarié travaillant à son domicile) et un réseau sécurisé via internet.
  - SET (Secure Electronic Transaction) : norme permettant d'effectuer des paiements par carte bancaire sécurisés dans des réseaux non sécurisés (tels que Internet). Des certificats numériques sont utilisés pour les détenteurs de carte de crédit, commerçants et banques. Les transactions crées sont sécurisées et incontestables, et ne peuvent pas être contrefaites.

Protocoles de sécurité

C'est quoi un protocole de sécurité

- Les protocoles qui utilisent les principes de cryptage, d'authentification ou de certification permettent de mettre en place des solutions de sécurité pour :
  - *différentes architectures* de réseau et
  - en intervenant à *différents niveaux* du modèle OSI.

Communication layers	Security protocols
Application layer	ssh, S/MIME, PGP
Transport layer	SSL, TLS, WTLS
Network layer	IPsec
Data Link layer	PPTP, L2TP
Physical layer	Scrambling, Hopping, Quantum Communications

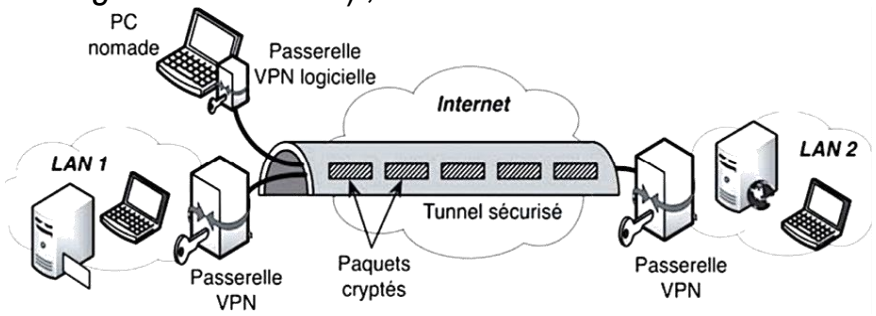
MPLS

- Certains protocoles concernent :
  - les VPN avec du cryptage aux niveaux 2 ou 3,
  - d'autres sont utilisés pour sécuriser les applications en cryptant toutes les données encapsulées au niveau 7. SSL/TLS (*Secure Socket Layer/Transport Layer Security*) et SSH (*Secure Shell*) permettent de chiffrer les messages encapsulés dans des segments TCP.
  - d'autres encore sont dédiés à l'authentification de l'utilisateur externe lors de l'accès à un réseau privé.

Protocoles pour les tunnels VPN

VPN (Virtual Private Network) et protocoles pour réaliser une connexion

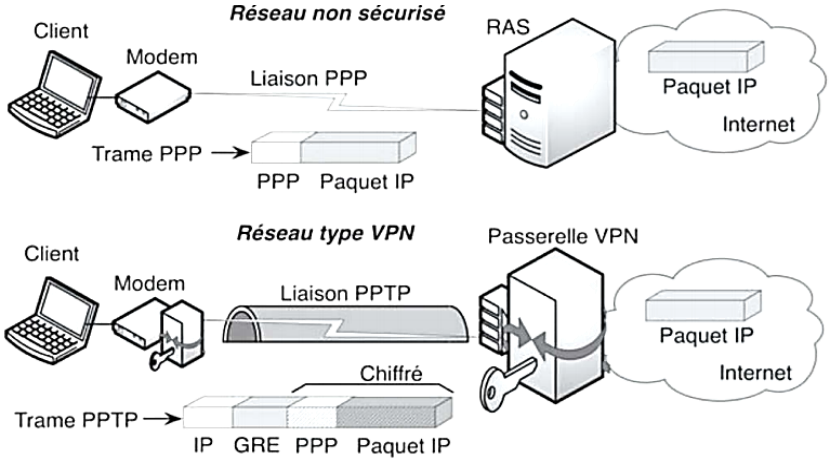
- VNP : repose sur le tunneling, est un élément essentiel pour une architecture moderne de sécurité. Les postes distants faisant partie du même VPN communiquent de manière sécurisée via Internet par tunnel sécurisé (données cryptées).
- Au niveau 2, les 2 protocoles les plus utilisés pour mettre en place un tunnel VPN sont :
  - PPTP (*Point to Point Tunneling Protocol*) proposé au départ par Microsoft et
  - L2TP (*Layer 2 TP*) : résultat de la fusion des protocoles PPTP et L2F (*L2 Forwarding*) de Cisco.
- Au niveau 3, IPSec (*IP Security*) défini par l'IETF permet de sécuriser les paquets IP.
  - Pour l'authentification :
    - AH (*Authentication Header*) : pour authentification ;
    - ESP (*Encapsulating Security Payload*) : pour authentification + cryptage.
  - La gestion des clefs pour IPSec est assurée par :
    - ISAKMP (*Internet Security Association and Key Management Protocol*) ;
    - IKE (*Internet Key Exchange*).



Protocoles pour les tunnels VPN

Protocoles PPTP et L2TP

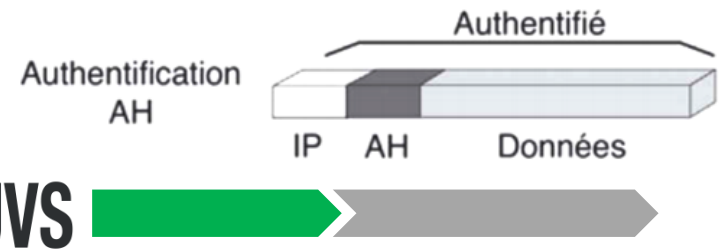
- La méthode standard pour accéder à distance à un réseau non sécurisé (ex : à l'Internet via son FAI) est de se connecter par un modem à un serveur d'accès distant RAS (*Remote Access Server*) à l'aide du protocole PPP (*Point to Point Protocol*).
- Dans le cas d'un VPN, le serveur RAS devient une passerelle VPN à laquelle on accède par le protocole PPTP. Le rôle de PPTP est de chiffrer et d'encapsuler, en les faisant passer par un tunnel crypté, les datagrammes IP dans le cadre d'une connexion point à point.
- Points faibles de PPTP :
  - Toute la partie négociation de la connexion n'est pas protégée.
  - Découvertes de vulnérabilités et incapacité à protéger efficacement les mots de passe des users
  - authentification moins fiable que celle d'IPSec.
- Caractéristiques de L2TP :
  - Permet de terminer une connexion non pas à l'autre modem mais à une adresse IP quelconque.
  - paquets encapsulés dans des paquets UDP
  - N'a pas de chiffrement.

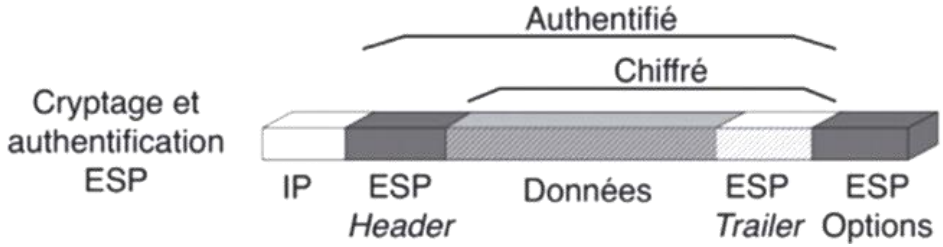


Protocoles pour les tunnels VPN

Protocole IPSec

- Lié à IPV4 et V6, assure l'authentification et l'encryptage des paquets IP au sur Internet.
  - IPSec peut être utilisé pour ne faire que de l'authentification : dans ce cas, l'ajout d'une **entête** AH permet de vérifier l'authenticité et l'intégrité des paquets.
  - AH ne spécifie pas d'algorithme de signature particulier.
  - Dans la plupart des applications IPSec, l'enveloppe ESP qui permet de chiffrer et d'authentifier les paquets est utilisée.
  - ESP ne spécifie pas d'algorithmes de signature ou de chiffrement particuliers, ceux-ci sont décrits séparément.
- Deux modes correspondant à deux architectures sont possibles avec IPSec :
  - mode transport : ne protège que les données des paquets transmis ;
  - mode tunnel : le paquet entier est protégé en l'encapsulant dans un nouveau paquet IP.





Protocoles pour les tunnels VPN

Protocole IPSec

- Sur chaque système susceptible d'utiliser IPSec, une base de données nommée SPD (*Security Policy Database*) doit être présente. Sa forme précise est laissée au choix de l'implémentation ; elle permet de préciser la politique de sécurité à appliquer au système.
- Une communication protégée entre deux systèmes à l'aide d'IPSec est appelée une SA (*Security Association*).
- Une SA est une entrée de la SPD, c'est-à-dire un enregistrement contenant des paramètres de communication IPSec :
  - algorithmes,
  - types de clés,
  - durée de validité,
  - identité des partenaires.
- Pour éviter d'avoir à gérer les clés de cryptage et d'authentification manuellement, IPSec intègre un *protocole d'échange automatique de clé* nommé IKE. IKE, utilisé dans la phase d'initialisation, négocie une SA et échange les clés choisies via un certificat (ex: X509)

Protocoles pour les tunnels VPN

Tableau comparatif :

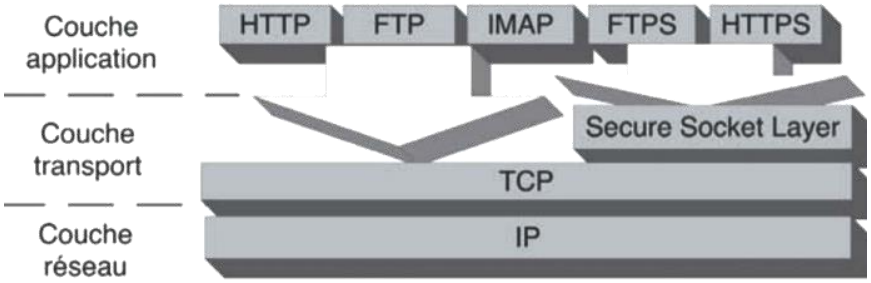
- Caractéristiques des protocoles d'accès distants

	L2TP	PPTP	IPsec
Mode	Client-serveur, tunnel opérateur (L2F)	Client-serveur	Client-client, tunnel passerelle
Utilisation	Accès distant *via* un tunnel	Accès distant *via* un tunnel	Intranet, extranet, accès distant
Protocole transporté	IP, IPX, NetBEUI, etc.	IP, IPX, NetBEUI, etc.	IP
Service de tunnel	Point-à-point	Point-à-point	Multipoint
Niveau OSI	2 (encapsulé dans IP)	2 (encapsulé dans IP)	3
Partage du tunnel	Oui	Oui	Oui
Authentification utilisateur	PAP, CHAP, EAP, SPAP	PAP, CHAP, EAP, SPAP	Non
Authentification du paquet	Le tunnel peut être authentifié.	Le tunnel peut être authentifié.	Oui, *via* l'en-tête AH
Chiffrement du paquet	Oui, *via* un tunnel IPsec	Oui, *via* la couche MPPE spécifique de Microsoft	Oui, *via* l'en-tête ESP
Affectation d'adresses dynamique	Oui (PPP NCP)	Oui (PPP NCP)	Selon les implémentations
Gestion des clés	Non	Non	IKE, SKIP
Résistance aux attaques	Non	Non	Oui

Protocoles pour sécuriser les applications

SSL/TLS

- Le protocole SSL (*Secure Socket Layer*) a été proposé au départ par *Netscape* (jusqu'à la version 2.0) pour permettre des connexions sécurisées sur des serveurs web.
- La version 3.0 est standardisée par l'IETF.
- TLS (*Transport Layer Security*), proposé par l'IETF, est la version 3.1 de SSL.
- Le protocole TLS n'impose pas de méthodes de chiffrement spécifiques.
- SSL intervient au-dessus de la couche transport et peut être utilisé pour sécuriser pratiquement n'importe quel protocole utilisant TCP/IP (SMTP, POP3, IMAP...) en créant un tunnel dans lequel toutes les données échangées seront automatiquement chiffrées.
- Permet la sécurisation des échanges entre la couche applicative et la couche transport :
- Génère un tunnel chiffré et créé une session entre le client et le serveur.



Protocoles pour sécuriser les applications

SSL/TLS

- **Caractéristiques et utilisations de SSL :**
- Certains protocoles applicatifs ont été spécialement adaptés pour supporter SSL :
  - HTTPS (HTTP+SSL) est inclus dans tous les navigateurs et permet par exemple de consulter des comptes bancaires par le web de façon sécurisée ;
  - FTPS est une extension de FTP utilisant SSL.
- SSL/TLS utilise un cryptage asymétrique par clé publique/clé privée pour :
  - authentifier le serveur (et éventuellement le client)
  - échanger la clé maîtresse (clé secrète)
  - assurer l'intégrité des messages.
- Permet par exemple de consulter des comptes bancaires par le web de façon sécurisée ;
- Connue des deux extrémités, la clé maitresse permet un cryptage symétrique efficace des données pendant toute la durée de la session.
- Implanté sur les grands navigateurs (clients)
- Côté serveur : apache + openSSL (50 % pdm), Microsos IIS 4 ou 5 (40 %).

Protocoles pour sécuriser les applications

SSL/TLS

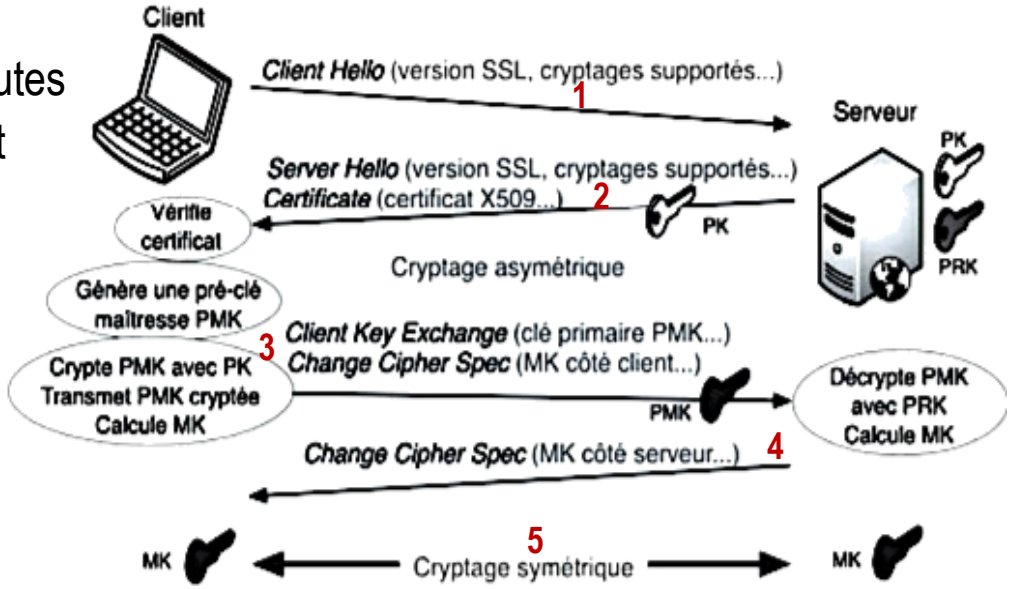
- **Caractéristiques et utilisations de SSL :**
- Sites dont le contenu est sensible :
  - Comptes bancaires consultables en ligne
  - Déclarations en ligne (consultation)
- Sites recevant des informations sensibles :
  - Le cas d'utilisation le plus courant est la connexion à un site bancaire ou marchand à partir de son navigateur. Ce site doit être certifié par une CA.
  - Commerce en ligne (paiements sécurisés)
  - Bourse (ordre de transactions)
  - Places de marché
  - Déclarations administratives, vote en ligne
- Authentification sur un réseau d'entreprise
- SSL permet d'établir une session client serveur sécurisée.

Protocoles pour sécuriser les applications

SSL/TLS

- Différentes phases pour l'authentification et la génération de clé maîtresse (clé secrète)
  1. Phase 1: après établissement de la connexion TCP, un 1er dialogue permet de choisir la version SSL et les types de cryptage qui seront utilisés ;
  2. Phase 2: au cours de ce dialogue, le serveur envoie au client un certificat X509 qui contient la clé publique du serveur *PK* signée par une autorité de certification (CA) ;
  3. Phase 3: le client vérifie le certificat, génère une pré-clé maîtresse PMK, crypte PMK avec PK, calcule la clé maîtresse MK à partir de PMK, et transmet PMK cryptée au serveur ;
  4. Phase 4: le serveur reçoit PMK crypté, la décrypte avec sa clé privée, calcule à son tour MK et indique que le *chiffrage est effectif* ;
  5. Phase 5: le tunnel sécurisé est créé, toutes les données applicatives seront cryptées et décryptées avec la clé symétrique MK.

- Pour sécuriser davantage, un certificat peut aussi être installé sur le client.



Protocoles pour sécuriser les applications

SSH

- La possibilité de travailler à distance a toujours été une fonctionnalité très appréciée des utilisateurs de machines UNIX. Cela est traditionnellement possible suivant plusieurs modes de connexion : telnet, rlogin, rsh, ftp, ...
- Le gros problème avec ces modes de connexions c'est que le mot de passe fourni pour vous connecter sur la machine distante circule en clair sur le réseau ⟹ n'importe quelle personne qui a accès au réseau où l'on se trouve peut à l'aide de programmes appelés sniffers récupérer le mot de passe.

- L'environnement SSH (*Secure Shell*) s'adresse aux utilisateurs qui souhaitent accéder de manière sécurisée à des systèmes Linux distants.
- Ses composants remplacent des programmes peu sécurisés comme
  - *telnet* pour **établir à partir d'un terminal une session sur un serveur** ou
  - *FTP* pour les **échanges de fichiers**.

Protocoles pour sécuriser les applications

SSH

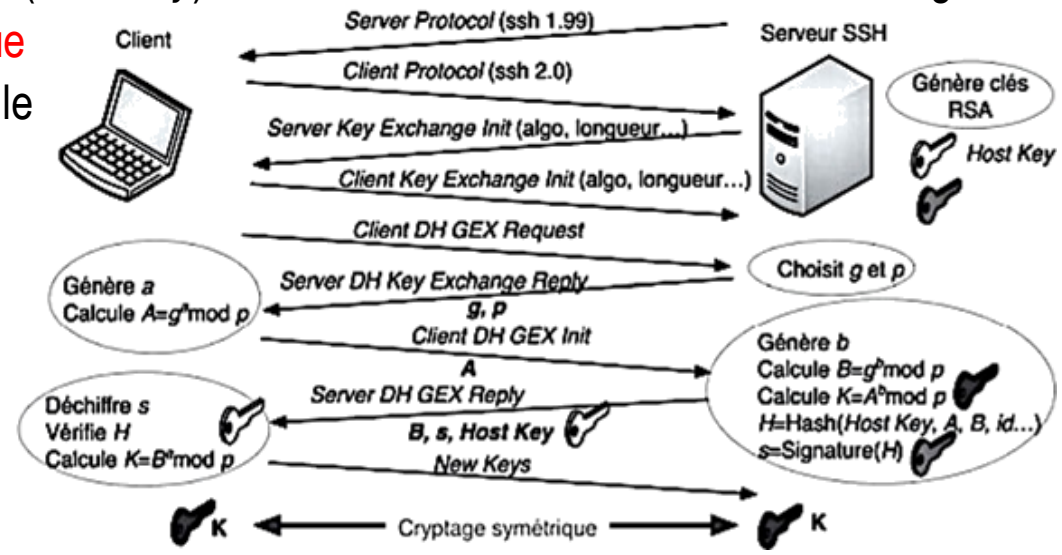
- La sécurité est garantie par une authentification à l'établissement de chaque connexion et par l'encryptage des données (y compris les mots de passe).
- SSH intervient donc au niveau 7, la connexion TCP est établie sur le ***port 22***.
- Le tunnel sécurisé dans lequel les communications sont chiffrées permet, comme pour SSL, d'encapsuler n'importe quel dialogue applicatif.

- Deux protocoles de transfert ont été prévus pour fonctionner avec une connexion SSH :
  - SCP (*Secure CoPy*), utilisé généralement en mode commande, permet de télécharger des fichiers de manière sécurisée dans un tunnel SSH ;
  - SFTP, version SSH de FTP, peut également être utilisée pour les transferts de fichiers. SFTP ne nécessite pas de clients ou de serveur FTP puisque le transfert se fait par le « *shell* ».

Protocoles pour sécuriser les applications

SSH

- Différentes phases pour authentification et création de connexion sécurisée SSH :
  1. Phase 1: client demande un échange de clé de type Diffie-Hellman ;
  2. Phase 2: serveur choisit 2 nombres g et p et les transmet au client ;
  3. Phase 3: client génère un nombre aléatoire a, calcule $A = g^a \ mod \ p$ et transmet A ;
  4. Phase 4: serveur génère b, calcule $B = g^b mod \ p$ et clé de session $\mathrm{K} = \mathrm{K_A} = \mathrm{A^b mod\ p}$, un hash H, signature s de H avec sa clé privée RSA. Il transmet B, s et clé publique RSA Host Key.
  5. Phase 5: client décrypte s avec la clé publique RSA reçue et compare le résultat avec son propre calcul de H. Si l'authenticité du serveur est vérifié, il calcule sa clé $\mathrm{K} = \mathrm{K_B} = \mathrm{A^b mod\ p}$;
  6. Phase 6: après confirmation du client (*New Key*), le *reste des communications est chiffré* grâce à un algorithme de chiffrement symétrique utilisant la clé K partagée par le client et le serveur.

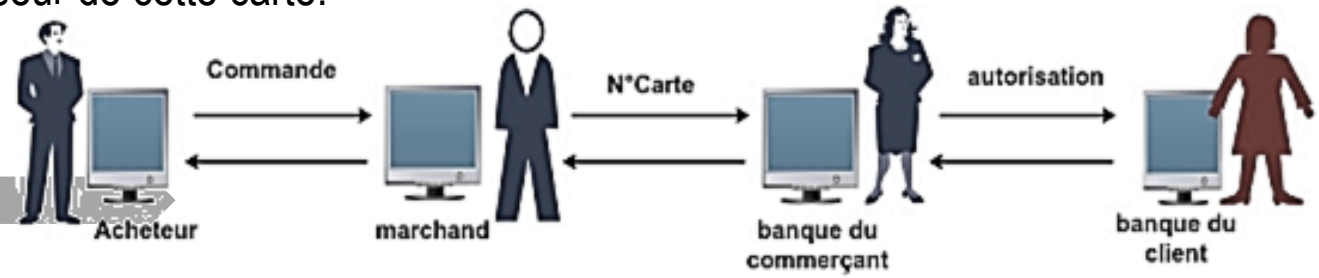


Protocoles pour sécuriser les applications

SET

- SET (*Secure Electronic Transaction*) est un protocole de sécurisation des transactions électronique mis au point par Visa et MasterCard et s'appuie sur SSL. Il est basé sur l'utilisation d'une signature au niveau de l'acheteur et une transaction mettant en jeu :
  - non seulement l'acheteur et le vendeur,
  - mais aussi leurs banques respectives.
- Principe d'une transaction sécurisée avec SET :
  1. Les données sont envoyées par le client au serveur du vendeur
  2. Le vendeur ne récupère que la commande
  3. Le numéro de carte bleue est envoyée directement à la banque du commerçant pour :
     - être en mesure de lire les coordonnées bancaires de l'acheteur,
     - et donc contacter sa banque afin de les vérifier en temps réel.

  ➢ Nécessité d'une signature électronique au niveau de l'utilisateur de la carte pour certifier qu'il s'agit bien du possesseur de cette carte.



Protocoles pour sécuriser les applications

S/MIME

- Standard MIME (*Multipurpose Internet Mail Extension*) :
  - Permet d'inclure dans les message électroniques des fichiers attachées autres que des fichiers texte.
- La version sécurisée S/MIME (*Secure MIME*) :
  - Procédé de sécurisation des échanges par courrier électronique, encapsulé au format MIME.
  - Utilise la cryptographie à clé publique pour signer, chiffrer et déchiffrer les message électroniques.
  - Assure l'intégrité, l'authentification, la non-répudiation et la confidentialité des messages électroniques.

  - Mis au point à l'origine par la société *RSA Data Security*.
  - Ratifié en juillet 1999 par l'IETF, S/MIME est devenu un standard, dont les spécifications sont contenues dans les RFC 2630 à 2633.

Protocoles pour l'authentification sur un réseau

Authentification

- Garantit que seuls les utilisateurs ayant le droit d'accéder au système y accèdent.
- Protocoles pour authentifier un utilisateur lorsqu'il cherche à se connecter sur un réseau :
- Pour une connexion distante point à point vers son FAI par ex, le protocole PPP définit une méthode d'authentification CHAP (*Challenge Handshake Authentication Protocol*) :
  - permet d'éviter que le mot de passe (mdp) ne soit transféré.
  - basé sur la résolution d'un « défi » lancé par le serveur au client.
  - Il ne permet cependant pas au client de s'assurer de l'identité du serveur.
  - autre point faible de CHAP : stockage en clair des mdp dans le serveur d'authentification.
- Améliorations de CHAP proposées par Microsoft MS-CHAP v1 puis v2 mais ce protocole reste vulnérable aux attaques de type dictionnaire $\Longrightarrow$ d'autres améliorations proposées.
- Pour compenser les faiblesses de MS-CHAP et pour proposer aux FAI d'autres méthodes d'authentification que par le mot de passe (carte à puce, certificats électroniques...), l'IEEE a proposé en 2001 le standard 802.1x et ses nombreuses déclinaisons sont considérés comme les plus sécurisés à l'heure actuelle pour une connexion sans fil.
- 802.1x repose sur le protocole EAP (*Extensible Authentification Protocol*) défini par IETF.

Protocoles pour l'authentification sur un réseau

802.1 x/EAP

- Les méthodes d'authentification supportées de base par EAP :

Type d'EAP	Description
EAP/MD5	basée sur le protocole CHAP associé à un algorithme de hachage MD5
EAP MS-CHAP-v2	basée sur la dernière version CHAP de Microsoft
EAP/OTP (*One Time Password*)	basée sur l'utilisation unique d'un mot de passe non nécessairement crypté
EAP/SIM	permet de s'identifier grâce la carte SIM de son téléphone portable
EAP/TLS	authentification par certificat proposé par SSL/TLS, tunnel TLS pas utilisé
EAP/PEAP (*Protected EAP*)	authentification, CHAP/MD5 par exemple, à l'intérieur d'un tunnel TLS
EAP/TTLS (*Tuneled TLS*)	authentification dans un tunnel TLS avec davantage de méthodes d'authentification possibles.

	EAP-MD5	EAP-LEAP	EAP-TLS	EAP-TTLS	EAP-PEAP
Authentification du serveur	Aucune	Mot de passe (hash)	Certificat	Certificat	Certificat
Authentification du client	Mot de passe (hash)	Mot de passe (hash)	Certificat	Certificat, compte/ mot de passe	Certificat, compte/ mot de passe
Distribution dynamique des clés	Non	Oui	Oui	Oui	Oui

Cryptographie et applications mobiles

- Les applications mobiles permettent par exemple un paiement à partir du téléphone ou encore les applications bancaires pose de sérieux enjeux de sécurité.
- D'autres équipements comme les objets connectés peuvent avoir à manipuler des données qui doivent rester secrètes.
- $\Longrightarrow$ sécurisation des applications mobiles est primordial aussi bien pour les utilisateurs mais aussi pour les fournisseurs de services qui y ont un intérêt financier.
- Ainsi, les outils cryptographiques sont utilisés pour protéger ces applications mobiles.
- La sécurité des implémentations logicielles des algorithmes cryptographiques dans un environnement mobile doit être vérifiée.
- Si la sécurité est défaillante un attaquant peut contrôler l'environnement d'exécution ou avoir accès aux implémentations logicielles.
- Si une clé secrète est révélée, la sécurité de l'algorithme de chiffrement n'est plus compromise. Dans ce cas la dernière ligne de défense est l'implémentation elle-même.

Cryptographie dans les applications web et mobiles

Fourniture d'un code et une application fiables

- Pour 2 personnes communiquant à l'aide d'applications sur leurs téléphones mobiles, la chaîne de confiance se présente comme suit :
  - un programmeur écrit un bon code de chiffrement,
  - le compile dans une application,
  - signe l'application avec une signature numérique et
  - la télécharge dans un store d'applications via TLS,
  - l'utilisateur télécharge une application avec TLS,
  - le système d'exploitation vérifie si la signature numérique est approuvée et
  - l'utilisateur exécute l'application pour avoir sa communication chiffrée.
- Certains problèmes pourraient compromettre la sécurité de l'application :
  - fournisseur du système d'exploitation pourrait saper le contrôle de la signature sur l'application.
  - Un attaquant pourrait inciter l'utilisateur à installer une version malveillante de l'application, …