Dernière mise à jour le 23/03/2023

URCA

Notes de cours RT0802

Cours dispensé par Mr. Fouchal

Jade Marquette
2022-2023

Table des matières

I. Préambule

1. Mise en place de PKI :

Mise en place :

- D'une infrastructure :
  - Cryptographie dans son ensemble
  - L'ensemble de standards de formats, de structure et de codage associés à la cryptographie pour permettre l'interopérabilité.
    La dimension offre au niveau logicielle et infrastructure de PKI
  - L'organisation en termes de gestion technique et de responsabilité associé aux différentes habilitations
  - La régulation au niveau institutionnelle quant à l'usage, la responsabilité, la qualification et le juridique (RGPD, CNIL)
- D'une politique de certification (**CP**) : où on met les certificats
- D'une politique de sécurité (**SP**) : comment mettre des protocoles en place.
- **Analyse de risque** : étude qui permet de montrer que ça vaut le cout de développer une solution car les risques sont importants.

2. Architecture et protocoles de communication

Architecture de communication en couches. Ce qui est un avantage et un inconvénient en même temps. Une faille peut arriver dans n'importe quelle couche, et entre n'importe quelle couche.

Un système ouvert est l'utilisation d'algorithmes connus. Les failles sont corrigées dès qu'elles sont découvertes. Ce sont des protocoles simples.

a. Ethernet

b. IP

- Le checksum est seulement fait sur l'en-tête de IP.
- TOS : toujours à 0 mais possibilité de rajouter des choses dedans.

c. TCP

- Tout est transmis en clair
- Possibilité de substituer une connexion
- Les champs peuvent subir des marquages
- Altération possible des tailles de fenêtres pour ralentir les échanges

d. Applicatif

- ASCII
- Echangées en claires

Aujourd'hui un firewall est programmé avec des algorithmes de Machine Learning.

3. Sécurité

Paradigme de la sécurité : la sécurité cause un ralentissement des performances.

a. Typologie des attaques

- Choix des implémentations : 5% patch (logiciel ou matériel)
- Design des protocoles : 10% filtrage (courriel, ftp, TCP con…)
- Usurpation d'adresse ou d'identité : 25% cryptographie, PKI
- Défaut d'implémentation : 60% patchs

b. Principe de sécurité

- Confidentialité
    - **Privacy** : ajout d'un anonymat. Un service donne un identifiant que seul l'utilisateur connaît. Utilisation d'un double hash.
- Intégrité (hash)
- Authentification (signature) } Souvent les mêmes mécanismes
- Non-répudiation : incapacité de remise en cause d'une action à laquelle on a participé, incapacité de remise en cause d'un message reçu ou émis.
- Répudiation : dans un échange où sont impliqués deux ou plusieurs entités, l'une de celles renie d'avoir participé à tout ou partie de l'échange.
- Horodatage : permet lors de l'envoi d'une requête le rejeu (avec le numéro de séquence ou les horloges)
- Disponibilité
- Traçabilité : dilemme : s'il y a un problème pouvoir remonter mais garder sa privacy.

Il existe un article de norme par principe.

II. Chiffrement symétrique et asymétrique

1. Chiffrement symétrique

a. Fonctionnement et généralités

- AES : utilisé sur TLS
- Codage de César
- DES : pas du tout fiable, historique
- 3DES
- RC4
- Blowfish

On dispose de :

- **L'algorithme**
- **La clef**

On dispose de la même clef pour chiffrer et déchiffrer. Plus la taille de la clef est grande plus il est difficile de tester toutes les combinaisons de clef. La clef ne doit jamais échanger en clair sur le réseau et ne doit pas être stockée en clair sur un équipement.

TPM : processeur cryptographique qui est sur toutes les machines depuis 2018. Permet de stocker les clefs secrètes.
Il existe des emplacements mémoire où personne ne peut lire sur les téléphones.

Permet :

- Confidentialité (chiffrement)
- Intégrité
- Authentification (gestion des mots de passe)

Cf. commandes `openssl`.

L'échange des clefs nécessite souvent de la personnalisation. La taille de la clef dépend de la législation.

10/01/2023

b. AES : Advanced Encryption Standard, un algorithme symétrique

En 1997 pour remplacer DES (qui était utilisé dans les échanges bancaires). En 2001, AES est devenu le standard.

- Substitution : flux binaire chiffré de manière originale. Chaque octet va être traduit. `SubBytes`.
- Permutation : mélange des octets après la substitution. Les algorithmes de permutation sont connus. `ShiftRows` : décalage circulaire. `MixColums` pour multiplier la matrice par un vecteur.
- Utilisation d'une clef pour faire un XOR. `AddRoundKey` : XOR de la matrice.
- Au deuxième tour la clef est modifiée, pour ajouter une complexité supplémentaire. Expansion de la clef : génération de la clef du tour i en fonction de celle du tour i-1.

Les blocs sont des multiples de 32.

La sécurité est basée sur l'algèbre et les groupes. Un octet est transformé en polynôme. On divise ce polynôme par un autre et le reste est le code chiffré.

XOR :

ET	*1*	*0*
1	1	0
0	0	0
OU	*1*	*0*
1	1	1
0	1	0
XOR	*1*	*0*
1	0	1
0	1	0

Exemple :
X XOR 0 = X
X XOR 1 = Inverse de X

Multiplication :
0110 → 1100 (décale à gauche quand < 8)
Si > 8 il faut faire un XOR.

00 → 00
01 → 02
…
80 →
95 →
Mettre en binaire le chiffre, décaler à gauche, faire le XOR de ça avec 1B en binaire.
Exemple :

95	10010101
Multiplication par 2 (décale à gauche), on le note *d*	00101010

1B	00011011
1B XOR *d*	00110001
Résultat en hexadécimal	31

15/03/2023

2. Fonctions de hachage MD5 et SHA, algorithmes de génération d'empreintes

Le hachage permet de générer une empreinte d'un message. Quelque soit la taille du départ, l'arrivée sera toujours la même. L'empreinte est une garantie d'avoir le bon fichier.
Un fichier transporte son empreinte. Le récepteur recalcule l'empreinte et la compare avec celle du fichier reçu.
Si les empreintes sont les mêmes les principes d'intégrité et d'authenticité sont conservés.

Le principe est que le hachage est à sens unique, facile à calculer et sans collision. Il existe deux types de fonctions de hachages :

- MDC : La fonction de hachage peut être calculée sans clef secrète.
- MAC : Hachage avec une clef.

a. MD5

Une étape de MD5 (parmi les 64) compte 4 rondes de 16 opérations. Les 4 tours permettent de rajouter de la confusion. L'ANSII ne recommande pas le MD5 pour des applications très sécurisées. En effet, il existe quelques cas de collisions.

- Complétion par remplissage (padding) du message hacher. Si le message fait moins de 128 bits on le complète avec des bits de padding.
- Définition de 4 variables A, B, C et D de 32 bits. Ces 4 variables sont connues. Après calcul / transformation elles deviennent l'empreinte. Ces transformations ne sont pas bijectives pour ne pas revenir en arrière.
  Les variables rentrent dans des transformations (F, G, H et I) qui génèrent des A', B', C' et D'.
  L'étape se reproduit un certain nombre de fois ce qui génère des A'''…, B'''…, C'''… et D'''….
- Calcul itératif : le calcul comprend 4 tours. Chaque tour comporte 16 opérations similaires basées sur les fonctionnements F, G, H et I. Une addition $\%2^{32}$ et un décalage à gauche de s positions.
  K[i] est une constante qui veut la partie entière de $2^{32}|\sin(i)|$, i correspondant à la $i^{ième}$ étape de calculs (parmi 4*16= 64 étapes).
  Slide 6 : M(0) → 1^{ier} bloc de 32 bits.
- Résultat

Un flux binaire est coupé au minimum en 4 blocs de 128 bits, le plus petit message étant de 512 bits.

b. SHA 1

Même esprit que le MD5 mais :

- ~~4 variables~~ → 5 variables
- ~~64 étapes~~ → 80 étapes
- ~~16 opérations~~ → 20 opérations

3. Echanges asymétriques

Schématisation du but :

Alice et Tom ont un secret. Alice envoie le message par la boîte de transport et ferme cette boîte de transport à clef. Bob ouvre la boite, récupère le message et referme la boîte de transport à clef.

Réalité schématisée :

Distribution des cadenas ouvert à tout le monde, mais seulement l'expéditeur détient la clef.

Fonction à sens unique qui donne un résultat facilement. L'opération inverse doit être très difficile → le temps pour retrouver la clef doit être le plus long possible.
Cependant cette fonction doit avoir une « brèche » secrète, une information secrète s telle qu'il est facile de retrouver le message.

a. Diffie-Hellman

Algorithme de la cryptologie à clef publique.

Il faut que les éléments soient authentifiés pour que cet algorithme fonctionne.

Schématisation :
Alice et Tom se mettent d'accord sur deux valeurs, α et β, connues de tout le monde. β doit être un nombre premier.

Alice calcule $K_A = \alpha^a \%\beta$

Tom calcule $K_B = \alpha^b \%\beta$

} Même si α et β sont connus, *exemple* $2047 = 317^a \%97$, à cause du modulo il est compliqué, long, de retrouver a, l'inconnue.

Tom envoie son K_B à Alice et Alice envoie son K_A à Tom. Puisque $(\alpha^a)^b = (\alpha^b)^a$ le message peut être récupéré.

La faille de Diffie Hellman est à l'attaque de l'homme du milieu.

b. RSA Rivest Shamir Adleman 1977

Le plus populaire. Création de clés publiques et privées.
Le chiffrement est fait à l'aide de la clé publique du destinataire.
Le déchiffrement est réalisé à l'aide de la clé privée du destinataire.

Création des clés

1- Choisir p et q des nombres premiers très grands. On calcule n = pq. n doit être très grand pour rendre plus difficile la factorisation. Il est important de ne pas utiliser le même n pour différentes clefs. On supprime p et q.
2- Choisir un entier e premier avec N, tel que N = (p-1)(q-1).
Il ne faut pas donner N puisqu'il permet de calculer (p-1)(q-1).

⇨ La clef publique sera (e, n).

3- On calcule d tel que ed%(p-1)(q-1)=1

⇨ La clef privée est (d,n)

Envoi de messages

Découper le message que l'on souhaite chiffrer, m, en blocs de m_i, tel que chaque bloc ait une représentation unique modulo n.
Chiffrer chaque bloc m_i par $c_i = m_i^e \%n$, avec e et n la clef publique du destinataire.

Déchiffrer pour chaque bloc c_i par $m_i = c_i^d \%n$, avec d et n la clef privée du destinataire.

Attaque

Il faut essayer de factoriser N pour retrouver p et q.
Pour les éviter ne pas chiffrer des messages trop courts.

4. Asymétrique ou symétrique ?

- Asymétrique : pour échanger le secret initial, certificat. Ils sont plus longs. On dispose de deux clefs qui sont basées sur des fondements mathématiques
  - Une clef chiffre, privée (signature)
  - Une clef déchiffre, publique
- Symétrique : tout le reste du temps. Ils sont plus rapides. Chiffrement des données.

III. Authentification

- Être certain de l'expéditeur d'un message : signature numérique. Assure l'authentification d'un envoi / l'identité de l'émetteur ne peut pas être usurpée et non répudiation.
- Être certain du destinataire d'un message : certificat numérique. Assurer que l'on s'adresse bien au bon destinataire (*exemple* payement par carte bleue). Le principe important dans le certificat est **la clef publique et la signature de l'autorité**.

La signature se fait avec la clef privée. La clef publique permet de vérifier la signature.

Les buts de l'authentification sont :

- L'authenticité
- La non-falsification
- La non-réutilisation
- Le fait que le document soit inaltérable
- La non-répudiation : l'expéditeur ne peut pas nier qu'il a envoyé le message.

Certifier revient à signer. Les certificats sont en notation ASN1. Les certificats sont publics. La première chose utilisée est la validité de la date. L'algorithme de hachage est dans le certificat (norme X509). Il existe des algorithmes qui permettent de créer des certificats respectant X509. Il existe des chaînes de confiances pour les certificats. Si une autorité de certificat n'est pas connue d'une machine elle remonte la chaîne de confiance de certifications jusqu'à retrouver une autorité de certification qu'elle connaisse. L'autorité a sa clef publique qui est chez tout le monde.

Une signature : permet de garantir que le message vient de quelqu'un de confiance.

On parle de :

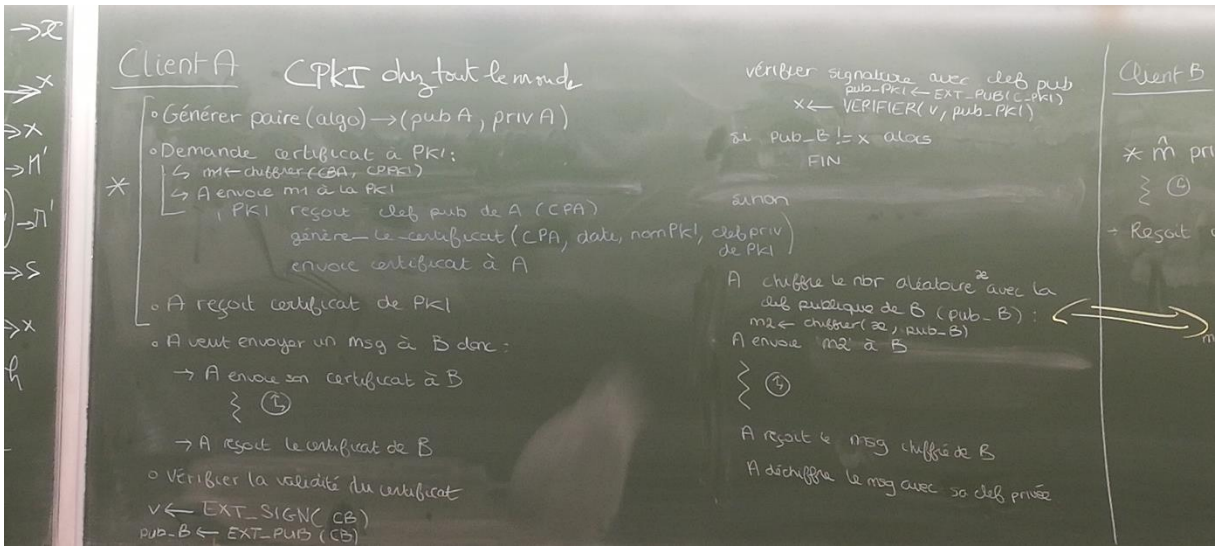
- Couche authentification
- Couche confidentialité

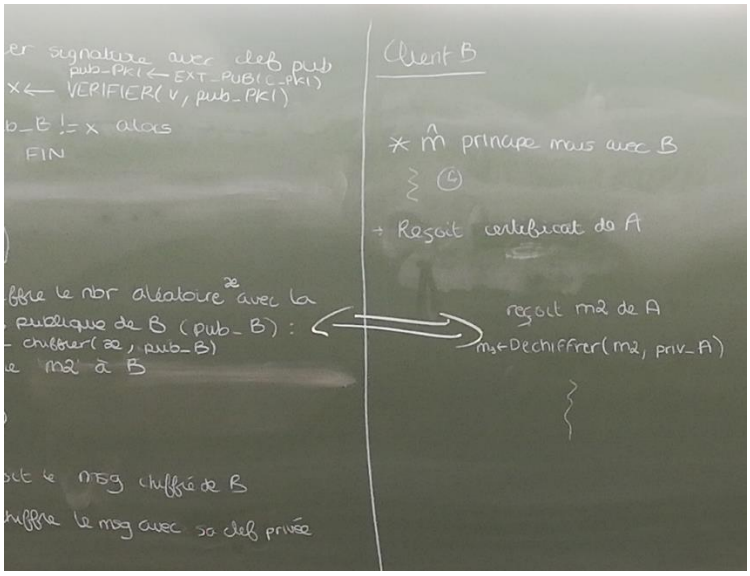
Revoir avec les slides du CM (signature avec fonction de hachage).

Résumé sur les certificats :

1- Réception d'un message signé
2- Authentification de l'émetteur son certificat
3- Authentification du certificat lui-même

Cf. correction ci-dessous :





1. PGP

Système fermé, où tout le monde se connait. Certificat auto-signé. Chacun dispose la clef publique des autres. Les échanges sont donc « entre-nous » et les certificats pas donnés par une autorité.

IV. PKI – autorité de certification

La clef publique des PKI est stockée en local (en dur). Les certificats sont stockés dans un endroit spécial. Il faut faire attention qu'une application ne rajoute pas un certificat « dangereux » dans la liste des certificats de sont téléphone par *exemple*.

Une partie de la PKI est accessible par internet et l'autre non. Afin de faire collaborer le monde PGP et le monde des certificats on utilise des serveurs intermédiaires.

Attention la politique de sécurité (quels sont les algorithmes) est différente de la politique de certification.

Secure boot : contrôle de l'intégrité du logiciel. S'il est différent de la normal la machine ne démarre pas.

Il y a eu différentes versions :

- Version 1 :
  - Certificat
  - Clef
  - Date
  - Algorithme de la clef publique
- Version 2 : rajout de 2 nouveaux champs
  - Identifiant unique
  - Sujet
- Version 3 : rajout d'un nouveau champ :
  - Les extensions : il en existe de nombreuses dont le niveau de criticité :
    - Niveau 1
    - …
    - Niveau 3 : seulement certaines autorités
    - …
    - Niveau 5 : échanges physiques uniquement

    Si le niveau n'est pas celui demandé → refusé. Ces règles sont négociées lors du premier échange.

1. Enregistrement et émission

Il existe deux méthodes de création des clefs privées :

- Je génère (moi client) la paire de clef sur ma machine directement.
- La PKI me génère ma clef privée et ma clef publique et me les envoie par un canal spécial. La PKI est constituée de modules qui permettent de créer des clefs où la mémoire n'est pas lisible, où la mémoire est cryptée :
  - HSM : Hardware Security Module → carte sur les serveurs.
  - TPM : Trusted Platform Module → microcontrôleur dédié à la sécurisation d'un système par intégration de clefs de chiffrement dans le matériel.

2. Stockage et distribution

3. Révocation des certificats

Le mécanisme de révocation se fait par le biais de protocoles normalisés.

Il existe un document au sein des entreprises nommé « politique de révocation ». Il permet de rassembler au sein d'un seul document la politique de certification (quand les donner, quel type…).

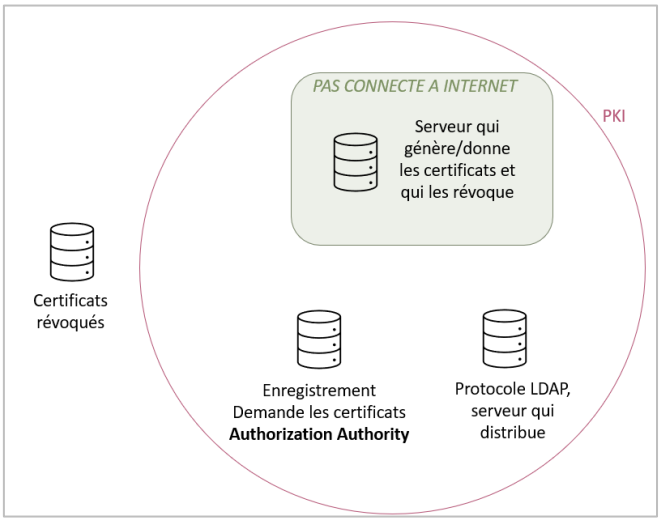
Il existe 8 raisons de révocation d'un certificat :

- La clef privée de l'autorité est compromise : c'est pour cela qu'il faut des algorithmes robustes qui génère des nombres premiers de manière aléatoire. Utilisation de l'heure qu'on

modifie avec la luminosité ou la température par exemple.
Très gros impacte après la compromission. Il est possible de générer des faux certificats. Donc il faut refaire tous les certificats.

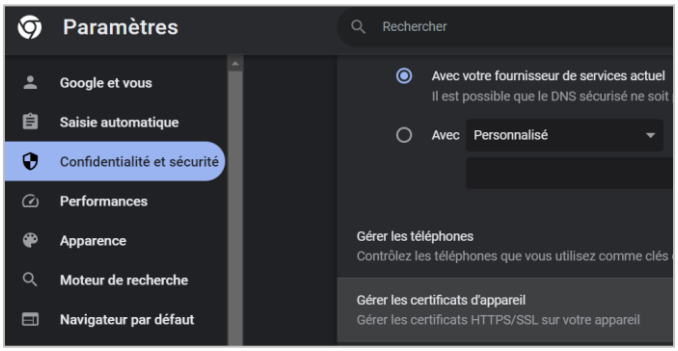
- La clef privée associée est compromise
- Changement de statut du détenteur du certificat. *Exemple* utilisation personnel à professionnel.
- Suspension du détenteur du certificat (à cause d'actes illégales)
- Un certificat a été obtenu frauduleusement
- Un changement intervenu dans l'état du sujet (activité) du certificat en tant qu'entité approuvée

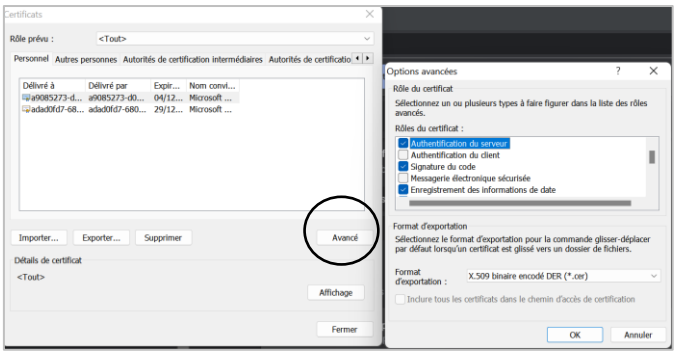
L'autorité de révocation et l'autorité de certification peuvent être la même entité.



a. Méthodes de révocation

- CRL distribution Points : distribuer les révocations sur différents serveurs. Le découpage peut être par ordre alphabétique, par géographie ou autre. Les avantages sont que le problème de taille est résolu et qu'il peut être caché. Mais problème d'implémentation.
- Delta CRLs : ne contient que ce qui a changé depuis la dernière fois. Il faut un référentiel qui fixé par le serveur.
- OCSP : vérification de la validation. Il est en temps réel. La réponse est rapide. Cela nécessite une haute **disponibilité**. Le serveur envoi l'état du certificat dans un message **signé**, c'est son défaut. Il est activé automatiquement sur les navigateurs :





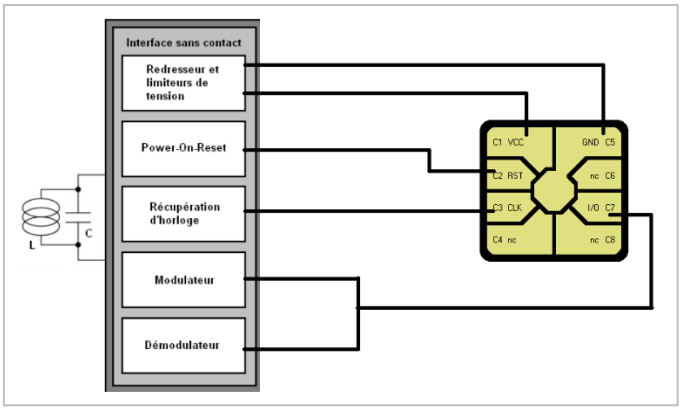
- CRL : modèle traditionnel. Liste noire des certificats révoqué. On peut trouver dans la liste des autorités, … Cette liste doit être signée. Mais le serveur doit publier périodiquement une nouvelle CRL.
- SCVP : modèle client/serveur. Garde en cache les certificats révoqués donc nécessite un serveur sur site. Cette solution est restée en essai. Les avantages est la décharge des postes clients et la centralisation des politiques de validation.

4. Sauvegarde et récupération

La PKI se charge de sauvegarder les clefs qu'elle a généré sur demande des clients. Comme cela, si un utilisateur perd sa clef privée la PKI peut lui renvoyer.

5. Exemple – La fonctionnement d'une carte bancaire

Une carte bancaire détient une puce avec un process qui fonctionne seulement quand la puce est insérée dans un TPE. Une puce est constituée de nombreux transistors.



A la création des cartes la clef publique de la banque est placée dans la puce.

RAM Partie calculs	**ROM** Partie pour mettre des informations.	**Mémoire invisible** Stockage de la clef privée. La clef ne sort jamais de cette zone. Elle signe seulement. Cette partie est invisible mais pas pour le calculateur.

Un TPE qui ne reconnait pas une carte bancaire → le TPE n'a sûrement pas le certificat de la banque.

1- Authentification **locale** via RSA.
   A la construction de la carte :
   - Application d'une fonction de hachage
   - Application d'une signature qui est signée avec la clef privée de la banque.
2- Entrée du code client : programme qui dit oui ou non.
3- Autorisation de la banque :
   - DES était initialement présent sur les puces des cartes mais étant trop faible il a été remplacé par TDES, puis AES.
   - TDES : trois fois DES, utilisation de deux clefs. Il est aussi robuste qu'AES 128. Il fonctionne avec des échanges symétriques.

V. TLS et IPSec

1. Protocole TLS

Canal chiffré - l'information qui est envoyée est intègre - permet d'authentifier le serveur (et le client optionnellement).

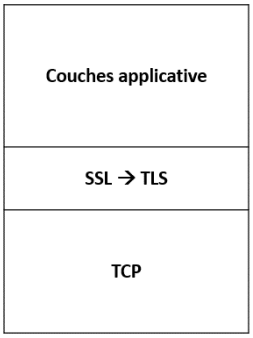
Initialement appelé SSL : Secure Socket Layer. Elle est sur la couche transport.

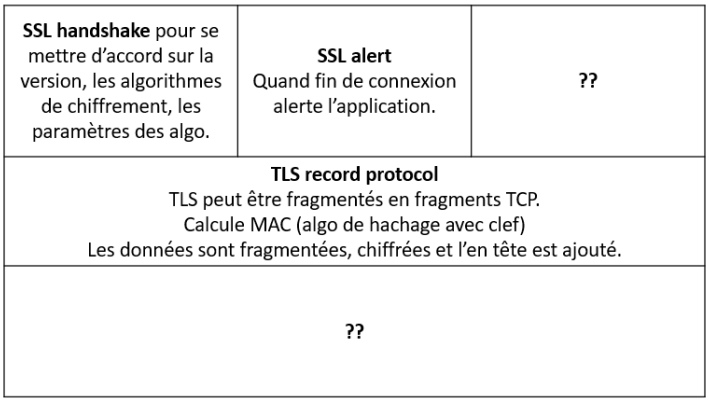
Sécurité qui va de la couche transport (process → port) du client 1 jusqu'à la même couche côté client 2.

TLS suit un modèle client/serveur : un doit écouter, donc est obligé d'être serveur.

TLS offre :
- La **confidentialité**
- La non-répudiation
- L'authentification
- L'intégrité

TLS utilise X509 v3 mais le protocole peut être autre chose dedans.
L'échange des clefs en asymétrique se fait à l'établissement de la session. Puis pour la confidentialité, il faut choisir un algorithme de chiffrement symétrique (tirer un nombre aléatoire et l'envoyer en la chiffrant avec sa clef).
Pour l'intégrité utilisation de la fonction de hachage. Le non-rejeu est fait avec l'horodatage ou un ID.





L'association entre le serveur et un client est créée par le protocole handshake, avec l'échange des certificats. Génération d'une clef maître qui sera dérivée.

Lors d'une connexion TLS offre la **confidentialité** et **intégrité** (calcule de la clef secrète pour le hachage type MAC, (HMAC)).

L'algorithme de chiffrement peut être **AES**, 3DES.

a. TLS a 3 phases :

Phases dans le handshake :

1- Etablir les paramètres de sécurité : processus initié par le client et le serveur répond.
Le client envoie des informations concernant la version de TLS, l'identificateur de la session, il envoi donc sa **Cipher Suits** (algo + clef proposée par le client).
Le serveur répond alors avec une Cipher Suit. Elle détient l'algorithme choisi qui est l'algorithme le plus sécurisé qu'il supporte dans la liste proposée/envoyée par le client – *Paramètres.*
Cette étape se fait en clair, donc possibilité de savoir quel algorithme est utilisé.

2- Authentification serveur et échange de clefs : le serveur envoie son certificat, **le client envoie la clef secrète** en utilisant un des algorithmes à clef publique. Le serveur peut demander un certificat client. *Exemple :* lors d'un achat sur internet pas besoin. Les schémas de signature sont RSA et DSA. *Authentifier le serveur.*

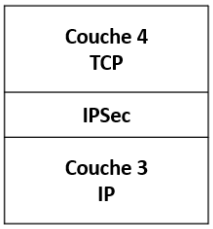
3- Optionnelle - authentification du client et échange de clés : le client envoie au serveur son certificat. *Authentifier le client*

4- Fin de la négociation : le client envoie le message final pour confirmer que l'échange de clefs et le processus d'authentification.

Chaîne de CA : liste de certificats jusqu'à arriver au certificat root. C'est une chaîne de confiance.

TLS a été décliné partout. *Exemple : SET : a*ncien protocole qui fait du pseudo-TLS (pour les cartes bancaires).

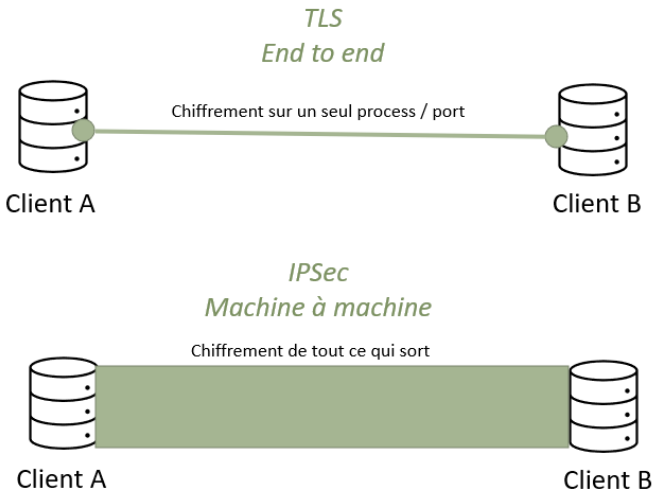
2. IPSec (IP Security)



IPSec est intégré dans l'IPv6. Il ajout une notion de privée. Travail sur l'authentification, pas forcémment la confidentialité car pas toujours de chiffrement. En effet, il existe différentes versions d'IPSec. Les clients n'ont pas besoin d'initier les échanges.

Il faut une négotiation au départ pour prévenir qu'on souhaite faire de l'IPSec.

Ici la sécurité se fait de machine à machine. C'est une des différences avec TLS :



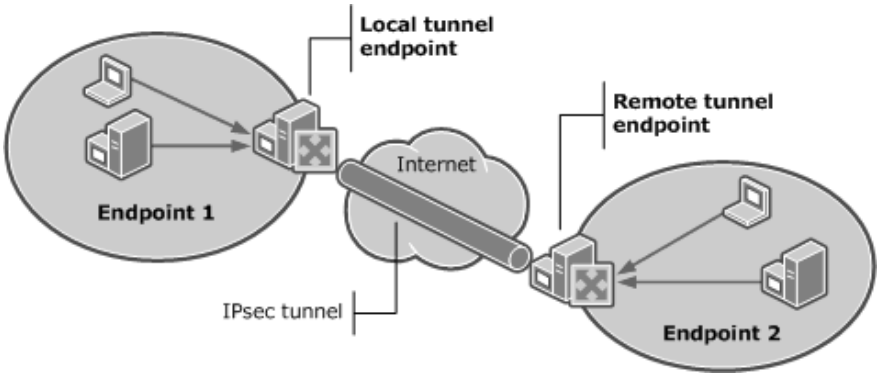
Le schéma reprend le mode transport sinon il est possible d'avoir un mode tunnel.

a. Architecture mode transport

La passerelle ne fait rien. Tous les headers ne sont pas encryptés.

ESP authentification : empreinte + signature.

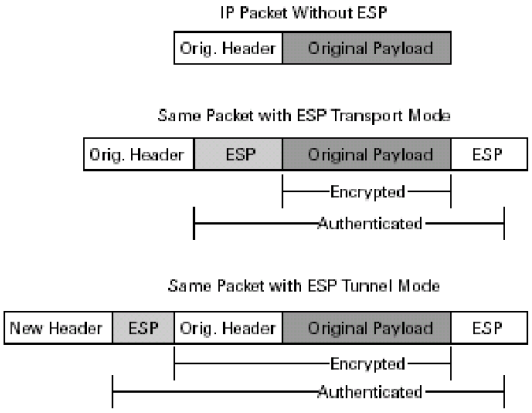
b. Architecture mode tunnel



Rajoute un en-tête au niveau IP, AH - Authentication Header, l'adresse source de la Gateway, avant d'envoyer dans le tunnel. Donc aucune information n'est disponible contrairement au mode transport où l'adresse source était lisible.

c. AH et ESP :

- AH pour l'authentification
- ESP : permet de combiner plusieurs services de sécurité : confidentialité, authentification et intégrité. Il chiffre les données (symétriquement) puis les encapsulent.



3. Conclusion

IPSec permet de sécuriser l'entreprise du monde extérieur et le TLS permet de sécuriser les échanges au sein même de l'entreprise.