Formation Générale en Cryptographie

Robert Rolland

rolland@iml.univ-mrs.fr

C.N.R.S., Institut de Mathématiques de Luminy

F13288 Marseille cedex 9, France

I-1. Les buts à atteindre

- **Secret, confidentialité.**
- **Intégrité des données.**
- **Authentification.**
- **Non-répudiation.**
- **Signature.**
- **Certification.**
- **Contrôle d'accès.**
- **Gestion des clés.**

I-1.1 Secret, confidentialité

La **confidentialité** est la propriété qu'une information n'est ni disponible ni divulguée aux personnes, entités ou processus non autorisés (norme ISO 7498-2).
Lors d'une communication, il s'agit d'empêcher un tiers de prendre connaissance de l'information contenue dans un message transmis sur un canal non sécurisé.

I-1.2 Intégrité des données

On doit éviter que les données transmises soient modifiées ou forgées par un adversaire. Plus précisément : l'intégrité est la prévention d'une modification non autorisée de l'information. L'intégrité du système et de l'information garantit que ceux-ci ne sont modifiés que par une action volontaire et légitime. Les attaques contre l'intégrité sont appelées **substitutions**.

I-1.3 Authentification

L'authentification consiste à vérifier l'identité des différents éléments impliqués dans un dialogue. Il peut s'agir d'authentifier une personne : on parle aussi dans ce cas **d'identification**. On parlera par exemple d'identification de l'expéditeur, du destinataire. Il peut s'agir aussi d'authentifier une machine, notamment dans le cadre d'une relation client-serveur à travers un réseau ouvert ou un réseau fermé.

I-1.3 Authentification (suite)

On peut vouloir également authentifier un document, son auteur, le serveur sur lequel on l'a récupéré, etc. La notion d'authentification recouvre différentes variantes plus ou moins similaires. Dans le cas d'un message, il s'agit de prouver son origine. Les attaques contre l'authentification sont appelées **mascarades**.

I-1.4 Non-répudiation

C’est un mécanisme qui empêche de nier un contrat. La non-répudiation consiste à prouver par exemple qu’un message a bien été émis par son expéditeur ou a bien été reçu par son destinataire. L’auteur d’un message ne peut nier l’avoir écrit ou transmis. Cette fonctionnalité doit donc permettre à un tiers de juger un conflit éventuel entre l’expéditeur et le destinataire.

I-1.5 Signature

C’est un mécanisme qui garantit l’authentification de l’expéditeur, l’intégrité des données et la non répudiation. On peut en outre vouloir l’empêchement d’un **rejeu** par l’expéditeur lui-même ou par un tiers.

I-1.6 Certification

Une entité connue, digne de confiance, valide une certaine information. Cette entité est alors appelée **authorité de certification**.
Typiquement, il peut s'agir d'un tiers de confiance, qui dispose d'un mécanisme pour certifier que la clé publique d'un utilisateur est bien celle qui est présente sur un serveur de clés.

I-1.7 Contrôle d’accès

L’accès à certaines ressources est limité aux personnes autorisées. Par exemple retrait d’argent à un terminal bancaire (ATM : Automated Teller Machine) ou encore connexion à un ordinateur.

I-1.8 Gestion des clés

En général les systèmes cryptographiques utilisent des clés (clés secrètes, clés privées, clés publiques). La gestion de ces clés (génération, distribution, stockage, intégrité, recouvrement, utilisation) est un problème difficile.

I-2. Techniques cryptographiques

- **Chiffrement.**
- **Signature.**
- **Authentification.**
- **Échange de clés.**

I-2.1 Chiffrement

Le **chiffrement** consiste à transformer un **texte clair** en **texte chiffré**. Le **déchiffrement** est l'opération qui consiste à retrouver le texte clair à partir du texte chiffré lorsqu'on dispose de la clé. Le **décryptage** consiste à retrouver le texte clair à partir du chiffré lorsqu'on ne connaît pas la clé. Ainsi un **expéditeur** chiffre un texte clair et envoie le chiffré à un **destinataire**. Celui-ci le déchiffre et récupère ainsi le texte clair. Un **attaquant (ou ennemi) passif** écoute la communication et tente à partir du chiffré une **cryptanalyse** afin de le décrypter.

I-2.1 Chiffrement (suite)

- Chiffrement à clé secrète
  - Chiffrement à flot
  - Chiffrement par bloc
- Chiffrement à clé publique

I-2.1.1 Chiffrement à clé secrète

Dans un système **à clé secrète ou symétrique** un **expéditeur** et un **destinataire** partagent une même **clé secrète**. Cette clé est utilisée à la fois pour le **chiffrement** et pour le **déchiffrement** et doit rester secrète de tout **observateur ennemi**. Le système consiste en une fonction de chiffrement publique $\mathcal{C}$ et en une fonction de déchiffrement publique $\mathcal{D}$.

I-2.1.1 Chiffrement à clé secrète (suite)

La fonction $\mathcal{C}$ prend en entrée la clé secrète K et un texte clair x et renvoie en sortie le texte chiffré $y = \mathcal{C}(K, x)$. La fonction $\mathcal{D}$ prend en entrée la clé secrète K et le chiffré y et renvoie le texte clair $x = \mathcal{D}(K, y)$.

I-2.1.1 Chiffrement à clé secrète (suite)

A chaque clé K sont associées une fonction de chiffrement $\mathcal{C}_K = \mathcal{C}(K, .)$ et une fonction de déchiffrement $\mathcal{D}_K = \mathcal{D}(K, .)$. L'expéditeur chiffre le **texte clair** x pour obtenir le **texte chiffré (ou cryptogramme)** $y = \mathcal{C}_K(x)$ et envoie y au destinataire. Le destinataire rétablit le texte clair en calculant $x = \mathcal{D}_K(y)$. Autrement dit $\mathcal{D}_K \circ \mathcal{C}_K = Id$.
Les fonctions $\mathcal{C}_K$ et $\mathcal{D}_K$ sont secrètes.

I-2.1.1 Chiffrement à clé secrète (suite)

Les problèmes difficiles principaux liés à un tel système sont : **échanger la clé secrète**, **la stocker**, **éviter de l'exposer** lors de son utilisation pour chiffrer ou déchiffrer. Le problème d'échange de la clé est spécifique de la cryptographie à clé secrète. La clé doit être communiquée par un canal sûr. L'utilisation d'une clé secrète s'accompagne de la notion de **sphère de confiance** pour le partage de la clé. Le partage de clés qui est acceptable en réseau fermé n'est plus envisageable en réseau ouvert.

I-2.1.1.1 Chiffrement à flot

Une méthode de chiffrement à flot **opère individuellement sur chaque bit** de texte clair en utilisant une transformation qui varie en fonction de la place du bit d'entrée.

Le **cryptosystème de Vernam** appelé aussi **one-time-pad** ou encore **masque jetable** est le prototype de ces systèmes. Il utilise une clé secrète très longue qui devrait de manière idéale représenter une suite aléatoire de bits.

I-2.1.1.1 Chiffrement à flot (suite)

Si on a un message m de n bits à chiffrer, on considère les n premiers bits de la clé qui constituent un mot K et on calcule le "ou exclusif bit à bit" entre le message et cette partie de la clé, c'est-à-dire que le texte chiffré s'écrit sous la forme $c = m \oplus K$. Le destinataire qui partage la même clé extrait de la même façon la partie K et récupère alors le texte clair m en calculant $m = c \oplus K$. Les deux interlocuteurs jettent la partie K utilisée et peuvent effectuer une nouvelle transaction.

I-2.1.1.1 Chiffrement à flot (suite)

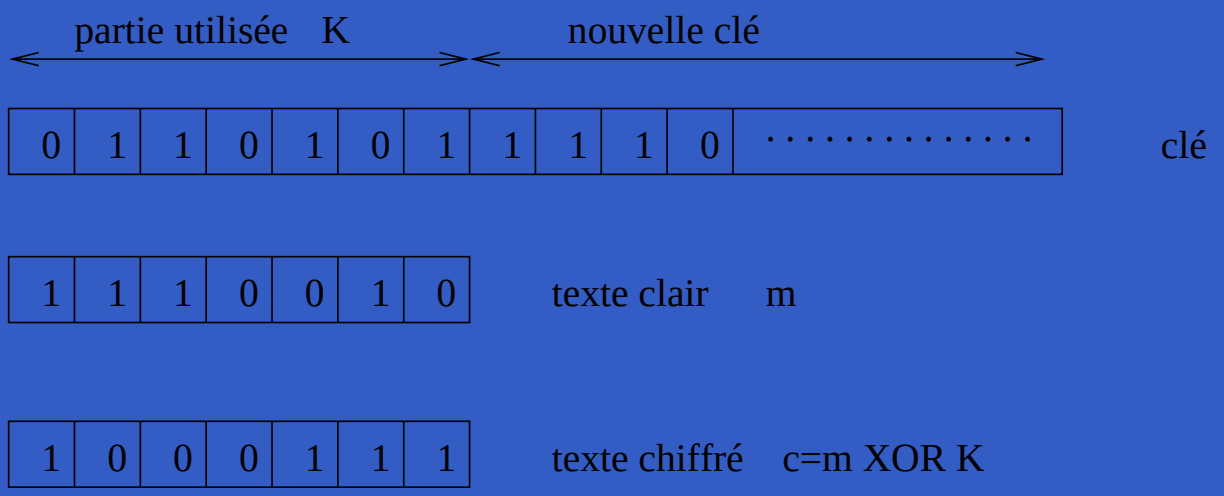


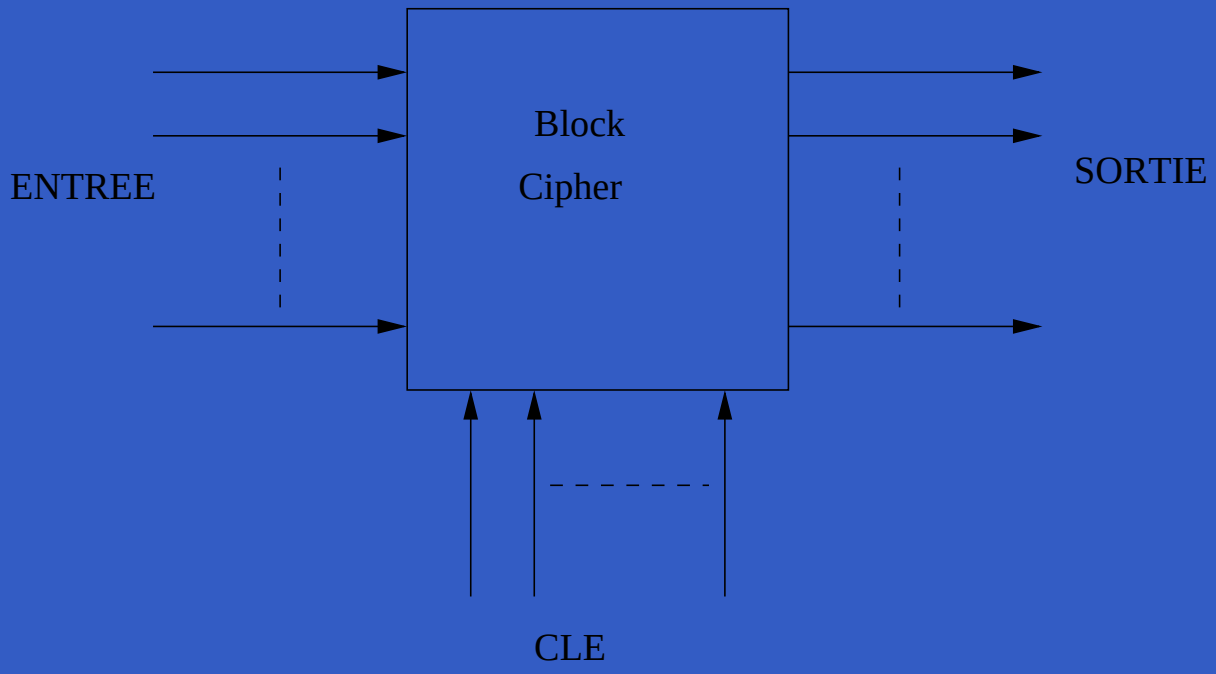
Figure 1: **Le one-time pad**

I-2.1.1.1 Chiffrement à flot (suite)

Construire une clé aussi longue et qui de plus soit une suite aléatoire de bits n'est pas chose facile. Il est possible de réaliser approximativement un tel système, à partir d'un **générateur pseudo-aléatoire** et d'un **germe**.

I-2.1.1.2 Chiffrement par bloc

Un système de chiffrement par bloc opère avec une transformation fixe qui s'applique sur des blocs de texte clair, de taille fixe.



I-2.1.1.2 Chiffrement par bloc (suite)

- **DES** (Data Encryption Standard). C'est le standard ANSI X3.92, proposé en 1974, publié dans le Federal Register en 1975, adopté comme standard en 1977 (FIPS-46). Il utilise une clé de 56 bits, des données de 64 bits. Ce système est maintenant considéré commme trop faible (taille trop petite des clés). Il est encore utilisé principalement dans le 3-DES.

I-2.1.1.2 Chiffrement par bloc (suite)

- **3-DES**. Appelé encore Triple DES, c'est le standard ANSI X9.71 et ISO 9732 (1985). On compose 3 circuits précédents de la façon suivante : on utilise deux clés DES k_1 et k_2 . On chiffre le texte clair avec k_1 on déchiffre la sortie avec k_2 et on chiffre cette deuxième sortie avec k_1. On a donc 112 bits de clé, une entrée de 64 bits et une sortie de 64 bits.
- **MISTY1**. Il utilise une clé de 128 bits et chiffre des blocs de données de 64 bits. La sortie est aussi un bloc de 64 bits.

I-2.1.1.2 Chiffrement par bloc (suite)

- **IDEA** (International Data Encryption Algorithm). Proposé en 1992 par Lai et Massey. Il utilise une clé de 128 bits, des doneées de 64 bits.
- **AES** (Advanced Encryption Standard). Standard américain qui remplace le DES. Proposé en 1998 par J. Daemen et V. Rijmen sous le nom de Rijndael, retenu en partie par le NIST en 2000. Standard FIPS-197 en 2001. Les données sont de 128 bits, les clés : 128, 196 ou 256 bits.

I-2.1.1.2 Chiffrement par bloc (suite)

- **Camellia.** Ce système offre aussi un choix pour la taille des clés : 128, 196 ou 256 bits. La taille des textes clairs et chiffrés est fixée à 128 bits.
- **SHACAL-2.** Il utilise une clé secrète de 512 bits et travaille sur des blocs de 256 bits.

MISTY1, AES (version 128 bits de données), Camellia, SHACAL-2 ont été retenus par le projet européen **NESSIE** (**N**ew **E**uropean **S**chemes for **S**ignatures, **I**ntegrity, and **E**ncryption).

I-2.1.2 Chiffrement à clé publique

En 1976, Diffie et Hellman introduisent l'idée de système cryptographique à clé publique. En 1977, Rivest, Shamir et Adleman proposent le système cryptographique RSA. Aujourd'hui, il existe divers systèmes à clé publique. Dans un cryptosystème à clé publique, chaque utilisateur A dispose d'une paire de clés : une **clé privée** d_A et une **clé publique** e_A. La clé privée de A n'est connue que de A. La clé publique est publiée et connue de tous.

I-2.1.2 Chiffrement à clé publique (suit

Il doit, bien entendu, être impossible en pratique de calculer d_A à partir de e_A. On dispose en outre d'une fonction publique de chiffrement $\mathcal{E}$ qui à une clé e_A et un texte clair x fait correspondre $y = \mathcal{E}(e_A, x)$, le chiffré de x à destination de A. On dispose également d'une fonction publique de déchiffrement $\mathcal{D}$ qui à la clé privée d_A de A et à un chiffré y à destination de A fait correspondre $x = \mathcal{D}(d_A, y)$, le texte clair associé à y. Remarquons que seule la clé privée est secrète; les fonctions $\mathcal{E}$ et $\mathcal{D}$ sont publiques.

I-2.1.2 Chiffrement à clé publique (suit

Notons $E_A = \mathcal{E}(e_A, .)$ la fonction de chiffrement à destination de A et $D_A = \mathcal{D}(d_A, .)$ la fonction de déchiffrement de A. Donc

$$D_A \circ E_A = \text{Identité}.$$

Si l'expéditeur B veut communiquer le texte clair m à A, il calcule le texte chiffré $c = E_A(m)$ en utilisant la clé publique de A, et il envoie c à A. Le destinataire A retrouve le texte clair en calculant $m = D_A(c)$ grâce à sa clé privée.

I-2.1.2 Chiffrement à clé publique (suit

L'existence d'un tel cryptosystème est basée sur la possibilité de construire des paires de fonctions réciproques l'une de l'autre, E_A et D_A, qui sont faciles à calculer et où E_A est très dure à inverser. Les systèmes à clé publique reposent sur la difficulté d'effectuer en pratique certains calculs.

Dans un tel système il n'y a pas de clé secrète à échanger : la clé privée ne sort pas de chez A et la clé publique est connue de tous.

I-2.1.2 Chiffrement à clé publique (suit

Il reste les problèmes de gestion de clé :

- comment éviter que la clé publique soit corrompue par une attaque malveillante du serveur de clés ou même forgée?
- comment éviter que la clé privée soit exposée, soit lorsqu'elle est stockée (faiblesse du moyen de stockage), soit lors de son utilisation (inspection de la mémoire de la machine utilisant la clé, virus malveillants, etc)?

I-2.1.2 Chiffrement à clé publique (suit

- **RSA** (Rivest-Shamir-Adleman). Ce système est basé sur la difficulté de factoriser un produit de deux nombres premiers. Associé à l'encodage **KEM** (**K**ey **E**ncapsulation **M**echanism) il a été retenu par le projet NESSIE. Associé à **OAEP** (**O**ptimal **A**symmetric **E**ncryption **P**adding) c'est le standard de RSALAB.
- **ElGamal**. Ce système est basé sur la difficulté du problème du logarithme discret dans certains groupes.

I-2.1.2 Chiffrement à clé publique (suit

- **PSEC** (**P**rovable **S**ecure **E**lliptic **C**urve Encryption). Ce chiffrement utilise le calcul sur le groupe des points d'une courbe elliptique. Associé à KEM, il a été également retenu par le projet NESSIE.
- **Rabin**. Ce système est basé sur la difficulté d'extraire une racine carrée modulo un produit n de deux grands nombres premiers.
- **McEliece**. Ce système est basé sur la difficulté du décodage d'un code correcteur linéaire binaire quelconque.

I-2.2 Signature

Un algorithme de signature numérique a plusieurs volets.

1) Un **condensé** du message est calculé avec une fonction de hachage publique h. Le message M est transformé en un message $m = h(M)$ de longueur fixée.

2) Chaque utilisateur A dispose d'une clé publique e_A et d'une clé privée d_A. Une fonction de signature $\mathcal{S}$ fait correspondre à une clé privée d_A et à un message condensé m un **appendice** $s = \mathcal{S}(d_A, m)$.

I-2.2 Signature (suite)

L'utilisateur A, qui veut signer un message M, commence par en faire un condensé $m = h(M)$, puis il calcule, grâce à sa clé privée, l'appendice $s = \mathcal{S}(d_A, m)$. Il peut alors transmettre son message signé (M, s). Le destinataire utilise la fonction de vérification $\mathcal{V}$ qui à une clé publique e_A, et à un message signé (M, s) fait correspondre $\mathcal{V}(e_A, M, s)$, valant "vrai" si (M, s) correspond bien à un message signé par A et "faux" sinon.

I-2.2 Signature (suite)

Remarquons que le processus de signature implique la **non-répudiation**. En effet, puisque tout le monde connaît la clé publique de A, tout le monde peut s'assurer que la signature est bien conforme. Comme A est le seul à avoir accès à la fonction $S_A = \mathcal{S}(d_A, .)$, tout le monde peut s'assurer que c'est bien A qui a signé le message M. Il existe aussi des signatures **à recouvrement de message** dans lesquels le message est récupéré à partir de la signature.

I-2.2 Signature (suite)

- **RSA** (Rivest-Shamir-Adleman). C'est l'algorithme RSA utilisé "à l'envers" : celui qui signe, chiffre avec sa clé privée. Tout le monde peut vérifier avec la clé publique de celui qui a signé. La signature RSA associée à l'encodage **PSS** (**P**robabilistic **S**ignature **S**cheme) a été retenu par le projet NESSIE.

I-2.2 Signature (suite)

- **DSS** (Digital Signature Standard) Utilise **DSA** (Digital Signature Algorithm) basé sur la difficulté du problème du logarithme discret sur le groupe multiplicatif $\mathbb{Z}/p\mathbb{Z}$ (p étant un grand nombre premier). D'autres alternatives sont possibles dans le standard DSS : on peut utiliser aussi **ECDSA** (Elliptic Curve Digital Signature Algorithm) basé sur la difficulté du problème du logarithme discret sur le groupe des points d'une courbe elliptique, ou encore **RSA**. Le standard ECDSA a été retenu par le projet NESSIE.

I-2.3 Authentification

L'authentification est un mécanisme qui couvre plusieurs fonctionnalités voisines. Il garantit que l'expéditeur d'un message est bien celui qu'il prétend être et que le message n'est pas corrompu ou forgé (identification et intégrité des données). On peut penser à utiliser un processus de signature pour obtenir l'authentification. La signature demande des fonctionnalités différentes de celles de l'authentification. L'authentification ne requiert pas la non-répudiation.

I-2.3 Authentification (suite)

- Il est possible de construire des **MAC** (Message Autentication Code) en utilisant des systèmes de chiffrement par bloc à clé secrète (DES, AES).
- Le système d'authentification de **Feige-Fiat-Shamir** qui fournit un processus d'identification basé sur la difficulté de l'extraction d'une racine carrée modulo n, lorsque n est un nombre composé et que sa factorisation n'est pas connue.

I-2.3 Authentification (suite)

- **UMAC** (retenu par le projet NESSIE). La conception de ce système utilise une famille très intéressante de fonctions de hachage.
- Certains systèmes d'identification mutuelle sont basés sur des protocoles d'échanges de signatures.
- Plus généralement, les preuves de savoir, éventuellement à divulgation nulle.

I-2.4 Échange de clés

- Utilisation de chiffrement (par exemple RSA)
- **Diffie-Hellman**. Ce système qui permet d'échanger des clés a été le premier protocole utilisant le principe de la cryptographie à clé publique.

I-3. Techniques mathématiques

- **Fonctions.**
  - ▷ **Fonctions à sens unique**
  - ▷ **Fonctions à sens unique avec trappe**
  - ▷ **Fonctions de hachage**
- **Arithmétique et algèbre.**
  - ▷ **Arithmetique dans** $\mathbb{Z}$ et dans $\mathbb{Z}/n\mathbb{Z}$
  - ▷ **Les corps finis et les courbes elliptiques sur les corps finis**
- **Générateurs pseudo-aléatoires.**
- **Complexité des algorithmes.**

I-3.1 Fonction à sens unique

Une **fonction à sens unique** est une fonction qui est **facile à calculer** et **difficile à inverser** (c'est-à-dire que $f^{-1}(y)$ est difficile à calculer **en pratique** pour "presque tout" y). Les significations exactes de "facile" et "difficile" dans la définition précédente demandent quelques notions de **théorie de la complexité des algorithmes**.

Exemples: fonction RSA, fonction puissance modulo p.

I-3.1 Fonction à sens unique avec trapp

Une **fonction à sens unique avec trappe** est une fonction à sens unique, mais qui devient facile à inverser si on connaît un secret (la trappe).

Exemple: fonction RSA.

I-3.2 Fonction de hachage

Une fonction de hachage transforme un message de taille quelconque en un **résumé court de taille fixe.** L'image d'un message par une fonction de hachage s'appelle le **condensé** du message, l'**empreinte** du message, le **résumé** du message ou encore le message **haché.** Une fonction de hachage n'est surement pas injective, cependant elle doit en pratique vérifier les conditions suivantes :

I-3.2 Fonction de hachage (suite)

- **résistance à la détermination d'une pré-image**, ce qui signifie qu' il doit être impossible **en pratique**, à partir d'un résumé m, de retrouver un message M ayant ce résumé, i.e. tel que $m = h(M)$.
- **résistance aux collisions**, ce qui signifie qu'il est impossible en pratique de construire deux messages M_1 et M_2 ayant le même résumé : $h(M_1) = h(M_2)$.

I-3.2 Fonction de hachage (suite)

- **SHA1** (Secure Hash Algorithm), qui fournit en sortie un bloc de 160 bits.
- **MD5** (Message Digest), qui donne une empreinte sur 128 bits.
- **SHA-256, SHA-384, SHA-512**. Ce sont des améliorations de SHA1 de manière à fournir une résistance aux attaques brutales comparable à la résistance des diverses versions de AES (SHA-256 à mettre en rapport avec AES-128, etc). Le projet NESSIE les a retenues.

I-3.2 Fonction de hachage (suite)

- **Whirlpool**. Cette fonction a une sortie de 512 bits. Whirlpool a été retenu par le projet NESSIE.
- Utilisation de systèmes de chiffrement par bloc (par exemple AES).

I-4. Les mises en œuvre, les protocoles

- **PGP.**
- **SSH .**
- **Station to station protocol.**
- **Kerberos.**
- **SSL.**

I-5. La normalisation

- l'**ISO** (**I**nternational **S**tandards **O**rganization),
- l'**ITU** (**I**nternational **T**elecommunication **U**nion),
- l'**ANSI** (**A**merican **N**ational **S**tandards **I**nstitute),
- l'**ECMA** (**E**uropean **C**omputer **M**anufacturers **A**ssociation),
- l'**IEEE** (**I**nstitute of **E**lectrical and **E**lectronics **I**ngineers),

I-5. La normalisation (suite)

- le **NIST** (**N**ational **I**nstitute of **S**tandards and **T**echnology),
- l'**AFNOR** (**A**ssociation **F**rançaise pour la **NOR**malisation).

Dans le cas de l'Internet, les standards sont appelés des **RFC** (**R**equest **F**or **C**omments). Il peut s'agir de définitions de protocoles, de projets, de compte-rendus de réunions, de spécifications de standards, etc.

I-5. La normalisation (suite)

Les RFC sont des documents publics, accessibles par exemple sur http://www.rfc-editor.org. Il en existe deux sous-catégories notables, les STD et les FYI : les STD sont les standards officiels et les FYI sont les documents d'apprentissage (For Your Information). Les "drafts" sont les documents des groupes de travail, généralement mis à la disposition de tous et sur lesquels chacun peut émettre des remarques et suggestions.