

Cryptographie

Une introduction

A. Bonnecaze
Polytech/IML



La cryptographie est un ensemble de **mécanismes** permettant d'assurer les **services** suivants

La confidentialité

L'intégrité des données

L'authentification

Le contrôle d'accès

La non répudiation

Alice et Bob décident d'utiliser la cryptographie pour communiquer

La cryptographie moderne

- Les travaux
 - d'Auguste **Kerckhoffs** de 1883
 - de Claude **Shannon** de 1949 sur la théorie de l'information

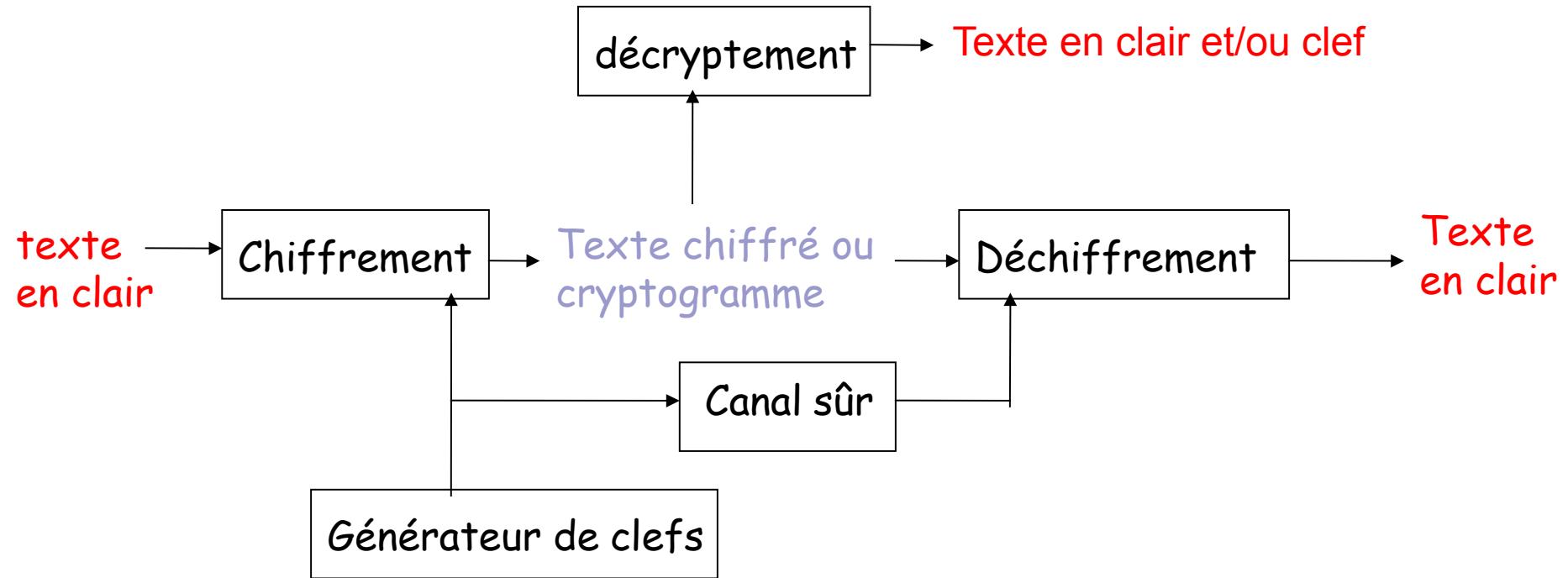
Ont profondément influencé la conception de la cryptographie moderne

Principes d'Auguste Kerckhoffs

1883

- Le système doit être **matériellement**, sinon mathématiquement, indéchiffrable ;
- Il faut qu'il n'exige pas le secret, et qu'il puisse **sans inconvénient tomber entre les mains de l'ennemi** ;
- La clé doit pouvoir être communiquée et retenue sans le secours de notes écrites, et être changée ou modifiée au gré des correspondants ;
- Il faut qu'il soit applicable à la correspondance télégraphique ;
- Il faut qu'il soit **portatif**, et que son maniement ou son fonctionnement n'exige pas le concours de plusieurs personnes ;
- Enfin il est nécessaire, vu les circonstances qui en commandent l'application, que le système soit d'un **usage facile**, ne demandant ni tension d'esprit, ni la connaissance d'une longue série de règles à observer.

Terminologie



Cryptographie (*cryptos logos* : « mot caché » en grec)

Chiffrement symétrique

- Alice et Bob partagent la même **clé** (secrète)
- Personne d'autre ne doit connaître la clé
- Le schéma de chiffrement consiste en
- Une **fonction de chiffrement** publique E
- Une **fonction de déchiffrement** publique D
- Une clé secrète
- ...et des messages à chiffrer

Schéma de chiffrement

- Si m est le message à chiffrer
- Si k est la clé secrète

Alice chiffre m en utilisant la fonction de chiffrement E et la clé k

$$c = E(k, m)$$

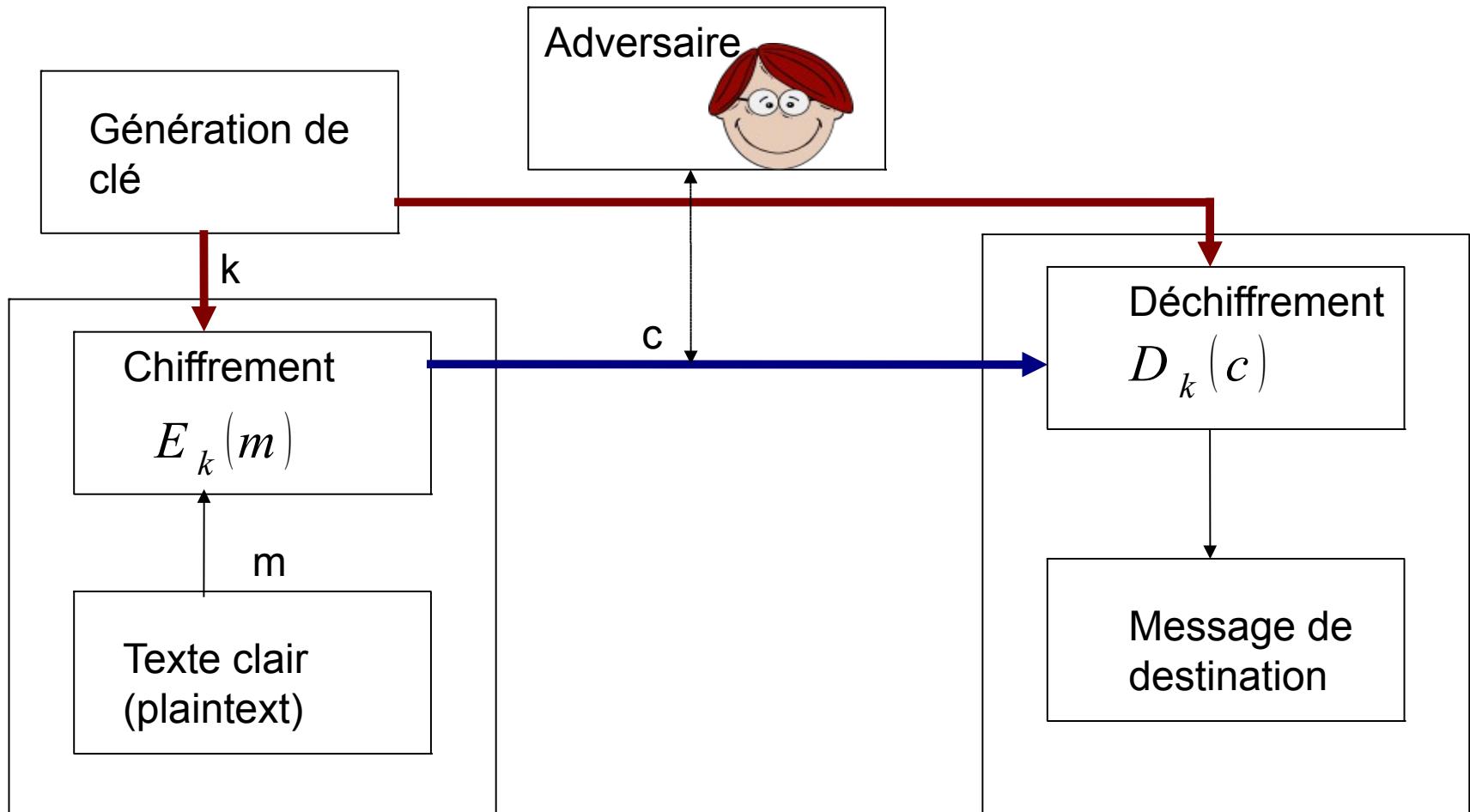
- Elle envoie le chiffré c à Bob
- Bob déchiffre en utilisant la fonction D et la clé k

$$m = D(k, c)$$

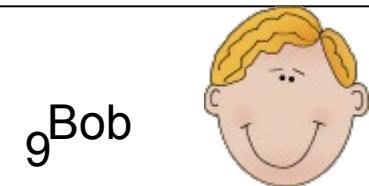
Chiffrement symétrique

- Si on note
 - E_k pour $E(k \cdot)$
 - D_k pour $D(k \cdot)$
- Alors on a
$$E_k \cdot D_k = Id$$
- Les fonctions E_k et D_k sont secrètes

Communication par chiffrement symétrique



Gestion des clés?



Types de chiffrement

- Chiffrement par **fLOTS** (one time pad, RC4,Salsa20...)
 - Bit à bit ou octets par octets
 - Très rapide
 - Utilise un PRG
- Chiffrement par **bLOCS** (DES, AES,...)
 - Le clair est divisé en blocs : 64 bits, 128 bits, ...
 - Doit utiliser un mode : CBS, CTR, GCM, ...
 - Clé de petite taille (128b,156b, ...) et réutilisable
 - rapide

Chiffrement par flot

Il faut un PRG qui prend en input un germe et rend une clé k aussi longue que le clair m

Le chiffré est : $c = m \text{ XOR } k$

La sécurité du chiffrement repose sur la qualité du PRG : si le PRG est indistinguorable d'un RG, alors le chiffrement est sûr

La clé k ne doit pas être réutilisée

Chiffrement par blocs (multiple time key)

- La transformation s'applique sur des blocs (de textes clairs) de taille fixe
- DES : adopté comme standard en 1977 (FIPS-46)
 - Clé de 56 bits, blocs de 64 bits
- 3-DES (triple DES) standard 1985. Se compose de 3 circuits de DES et 2 clés DES.
 - On chiffre le clair avec k_1 , on déchiffre la sortie avec k_2 , on chiffre la nouvelle sortie avec k_1
 - On a donc 112 bits de clé, une entrée et une sortie de 64 bits



Chercher sur internet : FIPS 46-3, RFC 1851

Chiffrement par blocs

- **AES** : Advanced Encryption Standard
- Par J. Daemen et V. Rijmen
- Standard américain qui remplace DES
- Standard **FIPS-197** en 2001
- Blocs de 128 bits
- clés de 128, 196 ou 256 bits (suivant sécurité voulue)

Authentification



- je veux être sûre que le message est bien celui de Bob
- Puisque la clé n'est connue que de Bob et moi, si j'arrive à déchiffrer, je sais que le message provient de Bob
- Je peux donc authentifier le message
- Mais un tiers ne peut pas authentifier le message

Clé publique clé privée



- Lorsque je suis en déplacement, je veux pouvoir chiffrer sans avoir de clé secrète sur moi
- Je veux aussi que tout le monde puisse authentifier mes messages

Pour cela il faut utiliser le concept de clé publique qui date de la fin des années 70

Clé publique/privée

- Chacun de vous dispose d'une paire de clés (publique/privée)
- La clé privée n'est connue que de vous
- La clé publique e est publiée et donc connue de tous
- Il doit être impossible de calculer d à partir de e (problème de math...)

Clé publique/privée

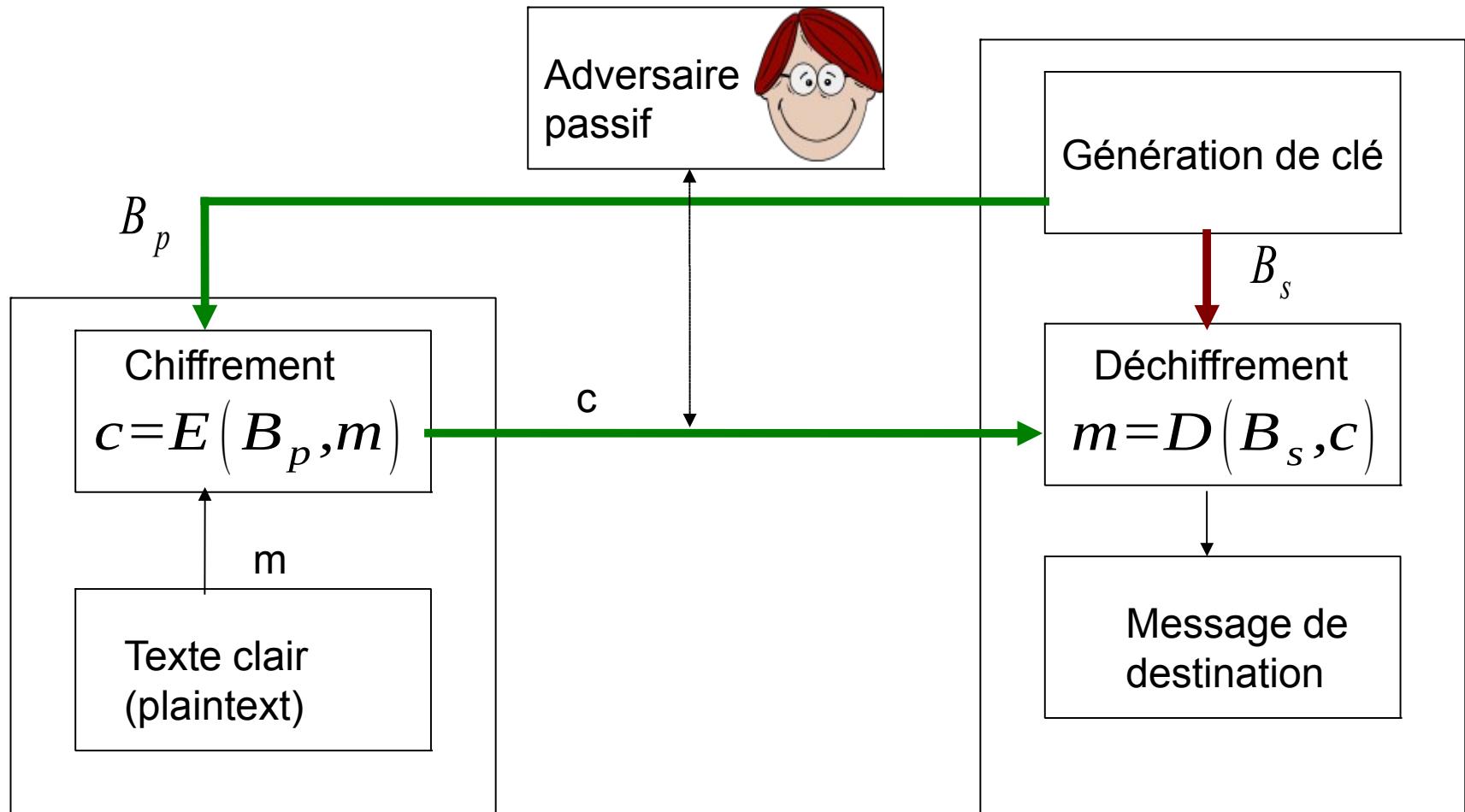
- 1976 : Diffie et Hellman : concept de cryptosystème à clé publique
- 1977 : RSA (Rivest Shamir Adleman)
- Repose sur des fonctions mathématiques
- Beaucoup plus lent que le chiffrement symétrique : **On ne peut pas chiffrer de longs messages, seulement des clés (< 10 000 bits)**



En pratique

- J'ai une paire de clés (A_s, A_p)
- Bob a une paire de clés (B_s, B_p)
- A_s, B_p et les fonctions de chiffrement sont connues de tous
- Je veux envoyer m à Bob
- J'envoie $c = E(B_p, m)$
- Bob utilise sa clé privée pour déchiffrer
$$m = D(B_s, c)$$
- **Pour chiffrer, je n'ai besoin que de la clé publique de Bob!**

Communication par chiffrement asymétrique



Gestion des clés?

Alice



Bob
19



La théorie

- Comment construire des paires de fonctions réciproques l'une de l'autre?
- Existence de fonctions difficiles à inverser
- Exemple simple : RSA
- Avantage : pas de clés à échanger
- Désavantage : calculs lourds pour messages courts



Pour chiffrer un fichier

- Je choisis une clé secrète s
- Je chiffre (algo symétrique) mon message avec cette clé
- Je chiffre (algo asymétrique) s avec la clé publique de Bob
- J'envoie à Bob les 2 chiffrés
- Bob retrouve s grâce à sa clé privée et déchiffre le message
- **Pas besoin de partager une clé secrète !**

Exemples d'algo de chiffrement asymétrique

- **RSA** (pb de factorisation)
- **ElGamal** (pb du log discret)
- **PSEC** (Provable Secure Elliptic Curve Encryption)
- **Rabin** (extraire une racine carrée)
- **McEliece** (pb de décodage)
- ...



Authentification

- Si je chiffre un message en utilisant ma clé privée, tout le monde peut savoir que le message vient de moi (**RSA à l'envers**)
- Il s'agit d'une signature électronique
 $E(A_s, m)$
- Mais que faire si le message est long?
- Le chiffrement asymétrique prend trop de temps...

Signature

- Primitive fondamentale pour l'authentification, l'autorisation et la non répudiation
- Ensemble des messages \mathcal{M} , des signatures \mathcal{S}
- $S_A : \mathcal{M} \rightarrow \mathcal{S}$: transformation de signature
- S_A est gardée **secrète** par A
- $V_A : \mathcal{M} \times \mathcal{S} \rightarrow \{\text{Vrai}, \text{Faux}\}$
- V_A est **publique** et permet à tout le monde de vérifier la signature de A
- S_A et V_A produisent un schéma de signature



Fonction de hachage

J'ai écrit un programme de 150 000 lignes en c++

Je veux l'envoyer à Bob

Je veux être sûre que le fichier ne sera pas modifié lors du transfert

- Il 'agit d'assurer **l'intégrité** du fichier
- Pour cela, utiliser des **fonctions de hachage**
- Une fonction de hachage produit une **empreinte** du fichier

Fonction de hachage H

- Elle prend en entrée n'importe quel fichier
- Donne en sortie un condensé de taille fixe
- SHA-1 donne des sorties de 160 bits
- SHA-256 donne 256 bits...
- $H(m) = h$
- H est une **fonction à sens unique**
- Connaissant h, il est calculatoirement impossible de trouver m' tel que $H(m') = h$



Intégrité d'une donnée

- Je publie sur ma page web le haché d'une donnée M
- Alice utilise cette donnée et veut s'assurer qu'elle n'a pas été modifiée par un tiers
- Alice hache la donnée et compare l'empreinte avec la valeur que j'ai publiée sur ma page
- Alice est alors sûre que la donnée ne contient pas de virus
- Je peux aussi signer le haché et envoyer la signature à Alice



Je peux utiliser une fonction de hachage pour signer !!

- Je hache mon message $H(m) = h$
- Je signe h avec ma clé privée
- J'envoie le message concaténé avec la signature
- Tout le monde peut vérifier, en utilisant ma clé publique que c'est bien moi qui ai signé.
- Comment ?

Exemples d'algo de Signature

- RSA
- DSS (FIPS PUB 186-3)
- ECDSA
- BLS
- ...

Lire la page



- http://csrc.nist.gov/groups/ST/toolkit/digital_signatures.html

Non répudiation

- Attention : Si Bob reçoit un message signé d'Alice, celle-ci ne pourra pas dire qu'elle ne l'a pas signé
- Si Iwan vole la clé privée d'Alice, il peut se faire passer pour Alice (Mascarade)
- Les clés privées sont des données sensibles



et



veulent partager un secret
sans utiliser leurs clés
publiques

- Bob publie les valeurs g et n
- Bob choisit au hasard une valeur b et envoie $g^b \text{ mod}$ à Alice
- Alice choisit au hasard une valeur a et envoie $g^a \text{ mod}$ à Bob
- Alice et Bob partage le secret $g^b \text{ mod}$



peut-il obtenir le secret à partir des communications entre Alice et Bob?

Quelques applications de la crypto

Communications anonymes

Digital cash anonyme

- peut-on acheter anonymement ?
- comment empêcher de dépenser plusieurs fois le même argent ?

Elections

Calculs multi-partie

Calculs privés sur un serveur externe

Zero knowledge ($N=p \cdot q$)

Protocoles connus

Communications sûres :

Web trafic : HTTPS

Wireless trafic : 802.11i WPA2, GSM, Bluetooth

Chiffrement de fichiers :

EFS, TrueCrypt

Protection de contenu :

DVD : CSS, Blue-ray : AACS

...

La crypto en 3 étapes

- Définir le modèle des menaces (threat model)
- Proposer une construction
- Prouver que casser cette construction sous ce modèle de menaces résout un problème difficile

Techniques de cryptanalyse

- **Hypothèse** : Oscar connaît le système cryptographique utilisé (principe de Kerkhoff)
- **Texte chiffré connu** : Oscar ne connaît que le message chiffré
- **Texte clair connu** : Oscar dispose d'un texte clair et de son chiffré
- **Texte clair choisi** : Oscar choisit un texte clair et a la possibilité d'obtenir son chiffrement
- **Texte chiffré choisi** : Oscar a temporairement accès à une machine déchiffrante. Il peut donc obtenir le texte clair du chiffré choisi.

Qui est l'adversaire ?

L'attaquant est

Le plus intelligent possible ; il peut faire toutes les opérations qu'il veut.

Il dispose d'un temps « raisonnable » qui dépend de l'application

Il ne doit pas pouvoir énumérer toutes les clefs (attaque par force brute)

Sécurité : indistinguabilité

L'adversaire a accès à des **oracles** :

- Chiffrement des messages de son choix
- Déchiffrement des chiffrés de son choix

Hypothèses :

- Chosen-Plaintext Attacks (**Ind-cpa**)
- Non adaptative Chosen-Ciphertext Attacks (**Ind-cca1**)
 - Accès à l'oracle seulement avant le challenge
- Adaptative Chosen-Ciphertext (**Ind-cca2**)
 - Accès illimité à l'oracle (sauf pour le challenge)

Sécurité par réduction

- Réduction d'un problème à un autre :

P1 se réduit à P2 ($P1 \leq P2$), s'il est possible de résoudre, en temps polynômial,
P1 à l'aide d'un oracle résolvant P2

- Réduction vers problèmes algorithmiques conjecturés comme difficile
- Preuve par l'absurde

On montre que s'il existe un adversaire contre le schéma cryptographique (algo A), alors il existe un attaquant (algo B) contre le problème algorithmique

Comme le problème algorithmique est supposé difficile, il n'existe pas d'algo B et donc pas d'algo A

Résumé

- Chiffrement (assure la confidentialité)
 - De données : symétrique (bloc ou flot)
 - De clés : asymétrique
- Fonction de hachage (à sens unique)
 - Intégrité des données
 - Très utile dans les protocoles crypto
- Signature (assure l'authentification)
 - Utilise une fonction de hachage
 - Crypto asymétrique
 - Assure la non répudiation