Rappel des concepts et fondamentaux sur le chiffrement

P. Gaudry

Caramel – LORIA
CNRS, Université de Lorraine, Inria

Demi-journée sur le chiffrement – 2RCE

Plan

Ordres de grandeur, entropie, mots de passe

Les grands concepts de la crypto

Le chiffrement symétrique

Plan

Ordres de grandeur, entropie, mots de passe

Les grands concepts de la crypto

Le chiffrement symétrique

La sécurité absolue n'existe (presque) pas

- Les systèmes mathématiquement parfaits sont peu nombreux et peu pratiques.
  - Exemple du chiffrement de Vernam (masque jetable).
  - Les preuves s'appuient sur la théorie de l'information.
- Sécurité calculatoire
  - La meilleure attaque connue prend au moins X siècles.
  - Si on a une notion de clef ou de mot de passe, on peut toujours tout essayer.
- Sécurité probabiliste
  - Avec une probabilité très faible, on peut casser en quelques secondes.
  - Si j'ai de la chance, je tombe sur la bonne clef dès le premier essai !

Ce que sait faire un ordinateur

- Imaginons un «gros» attaquant avec
  - Un million de cœurs ;
  - Un an de calcul ;
  - Chaque cœur tourne à 5 GHz ;
  - À chaque cycle, 1 cœur effectue une opération élémentaire.
- Au total : $10^6 \times 3600 \times 24 \times 365 \times 5 \times 10^9 \approx 1.5 \times 10^{23}$ opérations élémentaires disponibles pour monter une attaque. En crypto, on préfère les puissances de 2. On arrondit à 2^{80}.
- Ordres de grandeur courramment retenus en crypto :
  - 2^{65} – Calcul faisable par une petite organisation (le LORIA).
  - 2^{80} – Calcul à la limite du faisable par une très grosse organisation (Google, organisation gouvernementale).
  - 2^{128} – Calcul infaisable avec la technologie actuelle, et dans l'avenir prévisible.

Mots de passe

Monde idéal :

- 8 caractères choisis aléatoirement parmi 2×26 lettres, 10 chiffres, 20 caractères spéciaux : $(52 + 10 + 20)^8 \approx 2^{51}$ choix possibles.
- Si on est dans une configuration où l'on peut «tout essayer», facile à casser avec un cluster.

Monde réel :

- L'utilisateur veut s'en souvenir ! En général, bien moins de choix.
- Les sysadmin sont obligés de forcer des règles qui limitent aussi le nombre de choix.
- Exercice : de combien a-t'on diminué la sécurité théorique avec les règles au Loria ?

Qu'est ce que l'entropie ?

L'entropie est une notion de théorie de l'information qui permet de donner un sens mathématique précis à ce qu'est un bit d'information.

Cf page wikipedia pour les formules et leur sens.

- Permet d'estimer la redondance dans une langue.
- Permet de calculer plus précisément le temps moyen pour trouver un mot de passe, en prenant en compte le fait que certains sont plus probables que d'autres.

Exemple d'utilisation : `http://xkcd.com/936/`.

Conclusion sur les mots de passe (1)

C'est le point faible de la crypto, car on est obligé de s'appuyer sur un système hardware défectueux : le cerveau humain.

- Pour avoir un niveau de sécurité élevé : s'assurer d'avoir une entropie d'au moins 80 bits.
- Cela signifie au moins 12 caractères complètement aléatoires !
- Si possible, préférer une phrase plus longue, formée de mots simples.

Un mécanisme hardware supplémentaire peut-être nécessaire (carte à puce, dongle usb, . . .)

L'idée est que ce hardware n'autorise pas la recherche exhaustive (cf la carte bleue qui se bloque après 3 essais invalides).

Conclusion sur les mots de passe (2)

En pratique, pour la majorité d'entre nous, on veut se prémunir contre une attaque non ciblée :

- Un mot de passe standard (entropie de 40 bits) suffit.
- Faire la promotion d'outils de génération de mots de passe (`pwgen`).
- Les mesures trop coercitives (changer de mot de passe tous les 3 mois), ont des effets néfastes.

D'autres considérations sortent du domaine de la crypto :

- Risque de perte de données, si oubli du mot de passe.
- Loi US : on peut vous demander le mot de passe à la douane.

Plan

Ordres de grandeur, entropie, mots de passe

Les grands concepts de la crypto

Le chiffrement symétrique

Liste de concepts de la crypto

- Générateur pseudo-aléatoire ;
- Fonction de hachage ;
- Chiffrement symétrique ;
- Chiffrement asymétrique ;
- Signature numérique.

Générateur pseudo-aléatoire

L'aléa est difficile à obtenir pour une machine. C'est pourtant nécessaire pour de nombreux protocoles (et évidemment pour fabriquer des clefs).

Source de graves problèmes de sécurité :

- La mésaventure Debian.
- «Cassage» d'un certain nombre de certificats.

Sous Linux, deux versions :

- `/dev/random`. Chaque bit fourni est censé correspondre exactement à un bit d'entropie. Peut être bloquant.
- `/dev/urandom`. Version non-bloquante, avec des fonctions cryptographiques en boucle et des bits d'entropie pur insérés de temps en temps.

Rem. Pour des situations sensibles, il existe des périphériques hardware spécifiques.

Fonction de hachage

But. Obtenir une «empreinte digitale» d'un fichier.

On veut :

- Le haché est unique.
- Le haché est de taille fixe (256 bits).

Ces deux propritétés sont mathématiquement contradictoires.

Utilisé pour vérifier l'intégrité des paquets logiciels (et plus généralement, présent dans presque tous les algorithmes cryptographiques).

Nouveau standard en cours d'établissement : **SHA3**. Le vainqueur de la compétition est Keccak. Les paramètres standard ne sont pas complètement arrêtés.

En attendant, SHA1, SHA256.

Chiffrement symétrique

Historiquement, c'est la première fonctionalité de la crypto.

But. On a une clef (un secret) qui sert à chiffrer et à déchiffrer.

On veut :

- Si un espion voit des paires (clair, chiffré), il ne peut pas en déduire la clef avec un algorithme meilleur que tout essayer.
- On peut même donner à l'espion l'opportunité de choisir des clairs ou des chiffrés.
- Si on change un bit du clair, le chiffré n'a rien à voir.

L'algorithme le plus standard aujourd'hui est **AES**, mais la mise en œuvre (mode opératoire) dépend de l'usage.

Chiffrement asymétrique

Inventé dans les années 70.
Idée. La clef qui sert à chiffrer (publique) n'est pas la même que celle qui sert à déchiffrer (secrète).

Exemple : OpenGPG, qui s'appuie sur l'algorithme RSA.

Principe :

- Je construis mes clefs. Je garde ma clef de déchiffrement secrète. Je publie l'autre (page web, annuaire, dots)
- Quelqu'un qui veut m'envoyer un message récupère la clef de chiffrement, et l'utilise pour chiffrer le message.
- Même lui (!) ne peut plus déchiffrer s'il perd l'original.
- Je peux ainsi recevoir des messages chiffrés de gens que je ne connais pas.

Rem : il faut que le chiffrement soit randomisé.

Signature numérique

Même principe que dans le chiffrement asymétrique : paire de clefs (publique, privée).

Principe :

- Pour «signer» un document, j'utilise ma clef secrète.
- Ma clef publique permet à tout le monde de vérifier que c'est bien moi qui a signé le document.

Approximation : pour signer un document, je fais comme si c'était un chiffré. Je le déchiffre, et publie ce déchiffrement.

Attention : quel est le sens d'une signature classique ?

- Je prouve qui je suis ? – émargement
- Je m'engage ? – signature d'un contrat

En crypto, l'usage (de loin) le plus fréquent est le premier, alors que dans la vie réelle, c'est plutôt le second (?).

Plan

Ordres de grandeur, entropie, mots de passe

Les grands concepts de la crypto

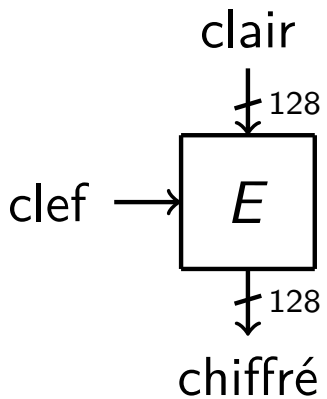
Le chiffrement symétrique

Principe général de conception

Deux approches pour chiffrer des messages de longueur arbitraire :

- Chiffrement à flot. On génère une suite pseudo-aléatoire basée sur la clef, qui sert de masque pour le message à chiffrer.
- On décompose en deux notions : chiffrement par bloc et mode opératoire.

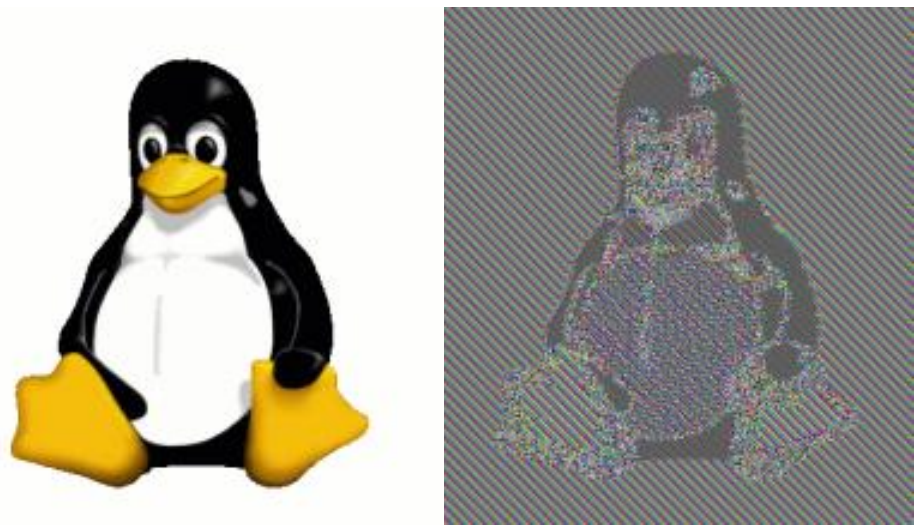
Chiffrement par bloc :



Principe général de conception (suite)

Mode opératoire : comment fait-on pour chiffrer un fichier qui est plus grand que 128 bits ?

L'approche naïve (mode ECB) n'est pas sûre, car un même bloc est toujours chiffré de manière identique.



© Larry Ewing

Taille de bloc / taille de clef

Deux paramètres fondamentaux du chiffrement par bloc :

Taille de bloc :

- Si trop gros, cela pèse sur l'efficacité (notamment en terme de mémoire utilisée par le chiffreur).
- Si trop petit, on met trop de pression sur le mode opératoire.
- On souhaite que l'événement «deux chiffrés identiques» n'arrive pas par accident.
- Aujourd'hui, 128 bits semblent optimal.

Taille de clef :

- Une clef donne une permutation de l'ensemble des 2^{128} blocs. On a $(2^{128})!$ permutations possibles.
- Ainsi, prendre une clef de taille supérieure à la taille de bloc est pertinent.
- Aujourd'hui, entre 128 et 256 bits pour la taille de clef.

Quelques mots sur AES

Historique :

- Dans les années 90, le standard était le DES (en place depuis les années 70). Mais DES est devenu trop faible, avec une taille de clef de 56 bits.
- Concours publique lancé par le NIST.
- Vainqueur = Rijndael, inventé par deux chercheurs belges.
- Devenu standard officiel aux USA en 2001.

Quelques mots sur AES (suite)

Attaques connues sur AES :

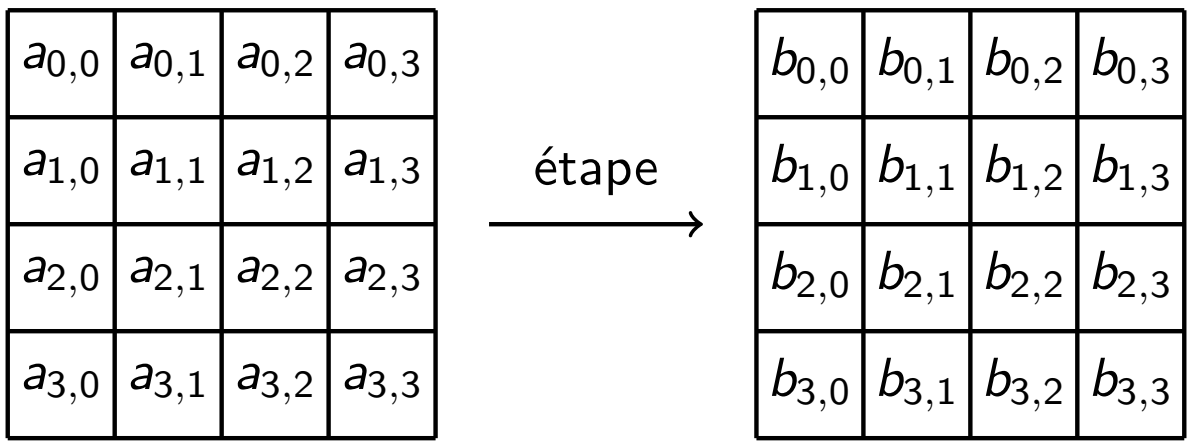
- Recherche exhaustive.
- Quelques faiblesses potentielles structurelles, identifiées dès le début, et désormais reconnues comme n'aboutissant pas.
- Polémique autour de 2002 : attaque annoncée, s'appuyant sur la conversion du problème en système d'équations polynomiales.
  L'attaque repose sur des hypothèses mathématiques qui ne semblent pas vérifiées.
- Fin des années 2000 : attaques sur des versions réduites d'AES.
- 2011 : première attaque sur le vrai AES, en $2^{126.1}$ opérations (au lieu de 2^{128} pour la recherche exhaustive).

Quelques mots sur AES (suite)

État interne de 128 bits :

- Initialisé avec le message clair.
- Organisé en une matrice 4×4 d'octets.
- Chaque octet est vu comme un élément du corps fini $GF(2^8)$.

AES procède par étape, chacune modifiant l'état interne.



Les étapes d'AES

Chiffrement organisé en 10 tours (rounds) identiques, comprenant 4 étapes :

- `SubBytes`. On applique une fonction mathématique (l'inverse modulaire) à chaque octet.
- `ShiftRows`. Permutation circulaire des octets sur chaque ligne.
- `MixColumns`. Chaque colonne, vue comme un polynôme sur $GF(2^8)$ est multipliée par un polynôme constant $C(x)$ modulo $X^4 + 1$.
- `AddRoundKey`. On fait un XOR bit-à-bit avec la sous-clef du tour.

Les sous-clefs de tours sont dérivées de la clef principale par un procédé indépendant.

Efficacité d'AES

Outre la sécurité, la conception d'AES a été guidée par son efficacité, en hard et en soft.

De nos jours :

- En soft pur, une vingtaine de cycles par octet.
- Avec les instructions AES-NI : quelques cycles par octet.

On est proche du Go/s par cœur.

Conclusion : Pas de surcoût visible pour l'utilisateur, même si tout le système est chiffré, même s'il n'y a pas de disque chiffrant.

Un mode-op pour les disques : XTS

Idée :

- On veut garder le «random-access» du mode naïf ECB.
- Il faut sécuriser pour éviter qu'un même bloc soit toujours chiffré de manière identique.
- On randomise avec une version chiffrée du numéro de secteur.

Formule :

$$C = \mathrm{AES}_K(M \oplus \mathrm{mask}) \oplus \mathrm{mask}, \text{ où } \mathrm{mask} = \mathrm{AES}_K(\mathrm{secteur}) \otimes \alpha^{\mathrm{block}}.$$

Faible surcoût par rapport à un mode naïf. Utilisé par `dm-crypt`.

Conclusion

- Les solutions cryptographiques pour le chiffrement de disque sont mûres.
- On a de plus une bonne modularité des produits (si une faiblesse est révélée dans une des briques, on peut facilement changer).
- Le surcoût en temps de calcul est faible. Probablement invisible pour l'utilisateur.
- Les outils sont de mieux en mieux intégrés au système d'exploitation.