# Cryptographie

Yannick Chevalier
Université de Toulouse
CSA M1, Security

# PLAN

## SURVOL DES PRIMITIVES CRYPTOGRAPHIQUES

## CHIFFREMENT ASYMÉTRIQUE

## SIGNATURE DIGITALE

## FONCTION DE HACHAGE

## CHIFFREMENT SYMÉTRIQUE

# CRYPTOGRAPHIE *VS.* CONTRÔLE D'ACCÈS

## CONTRÔLE D'ACCÈS

- une politique
- un mécanisme d'application doit mettre en œuvre cette politique
- mécanisme applicable uniquement sur des composants qu'on maîtrise

## CRYPTOGRAPHIE

- transformations de données
- les propriétés obtenues sont indépendantes de la localisation de ces données
- permet la communication de données en dehors des composants maîtrisés

# Primitives Cryptographiques

## Primitives ?

- type d'opérations sur les messages/textes
- classement en fonction de leur effet

## Grands types de primitives

- fonctions de hachage (pour la sûreté et la sécurité)
- chiffrement symétrique (2 personnes, 1 clef partagée pour le chiffrement et le déchiffrement)
- chiffrement asymétrique (1 personne, 2 clefs, une pour le chiffrement, une pour le déchiffrement)
- signature digitale (1 personne, 2 clefs, une pour la signature, une pour la validation des signatures)

# Base de la Cryptographie

## Ingrédients essentiels

- toutes les primitives robustes nécessitent un générateur aléatoire
- toutes les primitives robustes partent d'un problème à “porte dérobée” :
  - il est facile d'effectuer un calcul dans un sens
  - il est très difficile d'inverser ce calcul
  - sauf si on peut passer par une porte dérobée

## Évaluation

Pour chaque primitive :

- certaines opérations faciles, généralement implémentées dans des librairies
- la robustesse est définie par un devinette qui conclut que sans porte dérobée, il n'est pas possible de faire mieux que deviner au hasard la réponse

# PLAN

# CHIFFREMENT ASYMÉTRIQUE

## CONTEXTE

- découverte au début des années 70
- exemple connu historiquement : RSA
- chaque personne dispose de deux clefs :
  - une clef distribuée à tout le monde, qui sert à chiffrer des messages
  - une clef gardée secrète, qui permet de déchiffrer des messages

## DANS LA VIE DE TOUS LES JOURS

laisser un message sur la messagerie d'une personne :

- avec un annuaire, tout le monde peut connaître le numéro de téléphone de cette personne et laisser un message
- seule la personne ayant le téléphone pourra connaître le contenu des messages sur sa messagerie

# Cycle d'utilisation : Initialisation

## Procédure

- un paramètre de difficulté (un nombre de bits)
- une personne $A$ crée un couple $(KA, KA^{-1})$ de clefs à partir d'un générateur aléatoire
- pour simplifier, $KA$ est distribuée à tout le monde, et $KA^{-1}$ reste connue de $A$ seul

## Risques

- si le générateur aléatoire marche mal, une personne extérieure peut tenter de deviner les clefs qui seront produites
- réutilisation de paramètres d'une application à l'autre
- réseaux : pour le Web, supporter de vieux navigateurs signifie en général accepter des communications avec des paramètres trop faibles

# CYCLE D'UTILISATION : UTILISATION

## OPÉRATIONS FACILES

- en connaissant un message $m$ et une clef publique $k$, il est **facile** d'obtenir $c = encp(m, k)$, le chiffré de $m$ par $k$
- en connaissant un message chiffré $c = encp(m, k)$ et la clef privée correspondante $k^{-1}$, il est **facile** de déchiffrer $c$ pour calculer $m$

## MODÈLE LOGIQUE

$$\begin{cases} m, k \rightarrow encp(m, k) \\ encp(m, k), k^{-1} \rightarrow m \end{cases}$$

# Cycle d'utilisation : robustesse

## Jeu de base

1. le joueur choisit deux messages de même longueur $m$ et $m'$
2. il reçoit en retour soit $encp(m, k)$, soit $encp(m', k)$
3. le joueur doit deviner lequel des deux messages il a reçu

## Améliorations

- à tout moment, le joueur peut demander à déchiffrer n'importe quel message qu'il construit tant que ce n'est pas le message qu'il a reçu à l'étape 2
- la chiffrement est robuste si la probabilité qu'il a deviné correctement est $\frac{1}{2}$, quelque soit le message renvoyé

# EXEMPLE

## RSA

1. publique : un module $N$, un entier $k$
2. privé : un entier $k^{-1}$ tel que :

$$\forall 0 \leq m < N, \left(m^k\right)^{k^{-1}} = m \mod N$$

3. chiffrement : $encp(m, k) = m^k \mod N$
4. déchiffrement $c^{k^{-1}} \mod N$

## NOTE

- propriété importante : le déchiffrement “réussit” toujours, mais si on utilise la mauvaise clef on obtient un message sans signification
- pirmitive pas robuste (pourquoi ?)

## MORALITÉ

- lorsqu’on chiffre deux fois un même message, on doit obtenir deux résultats indépendants
- on a aussi besoin d’aléatoire pour le chiffrement

# PLAN

# SIGNATURE DIGITALE

## CONTEXTE

- historiquement basée sur le chiffrement asymétrique
- exemple connu historiquement : DSA (signature RSA)
- chaque personne dispose de deux clefs :
  - une clef distribuée à tout le monde, qui sert à valider des signatures
  - une clef gardée secrète, qui permet de signer des messages

## DANS LA VIE DE TOUS LES JOURS

affichage du numéro d'un correspondant :

- avec un annuaire/liste de contact, tout le monde peut connaître la personne ayant le numéro appelant
- seule la personne ayant le téléphone pourra appeler avec ce numéro
- ne pas montrer son numéro revient à ne pas signer son coup de fil

# Cycle d'utilisation : Initialisation

## Procédure (idem chiffrement

- un paramètre de difficulté (un nombre de bits)
- une personne $A$ crée un couple $(KA, KA^{-1})$ de clefs à partir d'un générateur aléatoire
- pour simplifier, $KA$ est distribuée à tout le monde, et $KA^{-1}$ reste connue de $A$ seul

## Risques

- idem chiffrement

# Cycle d'utilisation : Utilisation

## Opérations faciles

- en connaissant un message $m$ et la clef de signature publique $k^{-1}$, il est **facile** d'obtenir $c = sigp(m, k^{-1})$, la signature digitale de $m$ par $k^{-1}$
- en connaissant un message signé $s = sigp(m, k^{-1})$ et la clef de validation correspondante $k$, il est **facile** de vérifier que $s$ est la signature de $m$

## Modèle logique

$$\begin{cases} m, k^{-1} \rightarrow sigp(m, k^{-1}) \\ m, sigp(m, k^{-1}), k \rightarrow \top \end{cases}$$

# Cycle d'utilisation : robustesse

## Jeu de base

1. le joueur demande la signature de messages de son choix
2. le joueur gagne s'il peut produire un couple $(m, \sigma)$ où $\sigma = sigp(m, k^{-1})$

## Améliorations

- la signature est robuste si la probabilité de produire un couple correct est négligeable (zéro)
- (multiplication) utiliser la même mécanisme que pur RSA n'est pas très robuste
- on utilise en général une fonction de hachage pour rendre la signature robuste

# Plan

# FONCTION DE HACHAGE

## CONTEXTE

- vient des codes correcteurs d'erreur
- exemples connus : SHA1, MD5
  - une fonction qui calcule un message de taille fixée en fonction d'un message de taille quelconque

## DANS LA VIE DE TOUS LES JOURS

enregistrer ses contacts avec un surnom :

- peu de chances de connaître 2 personnes différentes ayant les mêmes nom etprénoms
- sans la liste de contacts, impossible de contacter une personne à partir du surnom
- si on choisit les surnoms au hasard, peu de chances que quelqu'un d'autre puisse trouver un des surnoms utilisés

# Cycle d'utilisation : Utilisation

## Opérations faciles

- en connaissant un message $m$ et la fonction $h(\cdot)$, il est **facile** d'obtenir $c = h(m)$, le haché de $m$
- en connaissant un message haché $b = h(m)$ et le message original $m$, il est **facile** de vérifier si $b$ est le haché de $m$

## Modèle logique

$$\left\{ \; m \rightarrow h(m) \right.$$

# Cycle d'utilisation : robustesse

## Recherche d'antécédents

du plus faible au plus dangereux

- il est possible de calculer $x, y$ tels que $h(x) = h(y)$
- connaissant $x, h(x)$, il est possible de calculer $y$ tel que $h(x) = h(y)$

## Notes

- Pour les fonctions de hachage communément utilisées, pas de critère basé sur des jeux
- (taille des espaces entrée/sortie) : il y a toujours une infinité de collisions, mais selon la fonction il sera plus ou moins dur d'en calculer

# Variante : HMAC

HMAC = hash + nombre aléatoire partagé

## HMAC et signature digitale

- connaissant $m$, la valeur $h(m,r)$ peut être produite par toute personne connaissant $r$
- connaissant $m, c$, toute personne connaissant $r$ peut vérifier que $c = h(m,r)$
- signature light, symétrique, propre à un groupe de sujets
- avantage : très rapide à calculer

## Signature digitale et H

- RSA avec application de la clef privée sur le haché de $m$ est très sûr
- en plus les temps de calcul sont plus courts

# Plan

# CHIFFREMENT SYMÉTRIQUE

## CONTEXTE

- historiquement le seul chiffrement jusque dans les années 70
- chiffrement de César, de Vigenère, Enigma,
- notion de chiffrement parfait, mais pas de jeu difficile pour les chiffrements en pratique

## CHIFFREMENT PARFAIT

- Vernam : un générateur aléatoire parfait $k$
- chiffrement : $m \oplus k$ (ou exclusif bit à bit)
- déchiffrement $(m \oplus k) \oplus k = m$

# Cycle d'utilisation : initialisation

## Entente

- 2 personnes souhaitant communiquer s'entendent sur un nombre aléatoire (contexte de sécurité ou clef secrète)
- le contexte de sécurité est utilisé pour initialiser un générateur de nombre aléatoires (puis ou exclusif avec les nombres générés)
- la clef est utilisée pour chiffrer bloc par bloc un message avec chaînage entre les différents blocs

# Cycle d'utilisation : Utilisation

## Opérations faciles

- en connaissant un message $m$ et la clef $k$, il est **facile** d'obtenir $c = encs(m, k)$, le chiffré de $m$
- en connaissant un message chiffré $c = encs(m, k)$ et la clef $k$, il est **facile** de calculer $m$

## Modèle logique

$$\begin{cases} m, k \to encs(m, k) \\ encs(m, k), k \to m \end{cases}$$

# Cycle d'utilisation : robustesse

## Jeux cryptographiques

- des jeux peuvent être définis comme pour le chiffrement asymétrique
- mais les chiffrements utilisés en pratique ne sont pas montrés sûrs

## Notes

- le chiffrement par blocs est déterministe (ex : AES), donc moins sécurisé
- pour le rendre moins déterministe, on utilise en plus le bloc chiffré précédent pour calculer le chiffré du bloc courant
- vecteur d'initialisation : valeur utilisée pour chiffré le premier bloc