

Signature et architectures à clé publique

DUT S4

Pierre Ramet: ramet@labri.fr

2013-2014

Plan

Plan

La cryptologie à clé publique

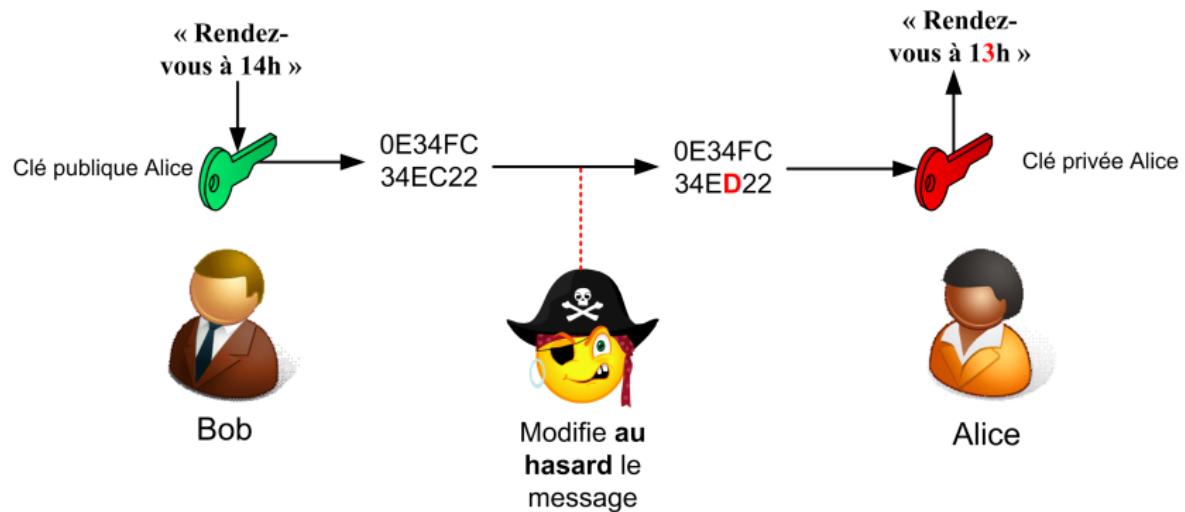
La cryptologie à clé publique

- Permet de garantir la **confidentialité** des messages échangés
- Ne garantit pas **l'intégrité** du message
- Surtout si le canal de communication n'est **pas sûr** (ex : Internet)

Definition

Garantir l'intégrité d'un message consiste à pouvoir **détecter toute modification malicieuse** du message entre le moment de son émission et l'instant où il est reçu

Exemple d'attaque



Dangers

- L'attaquant ne peut déchiffrer le message, il ne peut donc modifier que le **chiffré**
- La plupart du temps, la modification aura peu d'incidence (pour un message en français par exemple)
- Parfois, les conséquences peuvent être désastreuses
 - Exemple précédent
 - Pas acceptable pour les organisations
- A votre avis, comment garantir l'intégrité d'un message ?

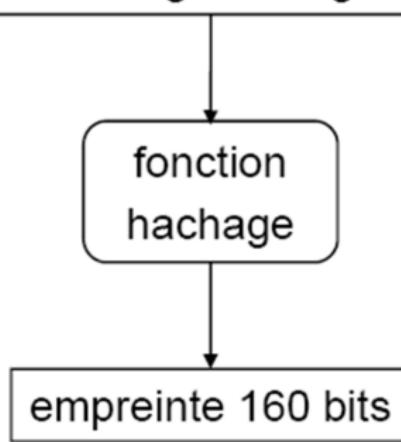
Plan

Prise d'empreinte d'un message

- Une empreinte numérique est un "*résumé*" d'un message, aussi appelé condensat
- Elle est de taille fixe (souvent entre 128 à 160bits)
- Deux documents légèrement différents ont une empreinte très différente
 - Propriété de **non colusion**
- Il est très difficile de contrefaire un document qui ait la même empreinte qu'un autre document
 - Propriété de "**sens unique**"
- Algorithmes utilisés : MD5, SHA1
- Ces algorithmes sont appelés **fonction de hachage**

Fonction de hachage

Ceci peut être un très long message de plusieurs megs



Utilisation

- L'empreinte est utilisée pour vérifier **l'intégrité** d'un message
 - Si le message et son empreinte ne correspondent plus, il y a sûrement eu une manipulation du message
 - L'empreinte ne doit pas pouvoir être recalculée facilement par un attaquant ⇒ **chiffrement du message+empreinte**
 - Si le message et l'empreinte sont chiffrés et que l'attaquant ne peut les déchiffrer, il ne pourra les modifier qu'**aléatoirement** et il y aura de fortes chances que l'empreinte ne corresponde plus au message modifié

Qualités d'une fonction de hachage

- Chaque bit de l'empreinte dépend de l'ensemble des bits du message
- le changement d'un bit du message d'entrée provoque en moyenne le changement de la moitié des bits de l'empreinte
- Connaissant $H(m)$, il est difficile de trouver m' tel que $H(m) = H(m')$
 - Que penser de la fonction de hachage *parité* ?
- Rapide à calculer
 - Mais pas trop, pour freiner une attaque exhaustive
 - Exemple : la fonction *crypt Unix*

Fonctions de hachages connues

- MD4 et MD5
 - Fonctions produisant des empreintes de 128 bits
 - Ne sont plus résistantes aux collisions. Il est possible de trouver des messages ayant la même empreinte
- RIPEMD 128-256
 - Fonctions produisant des empreintes de 128 ou de 256 bits
- SHA-1 (Secure Hashing Algorithm)
 - Standard américain produisant des empreintes de 160 bits
 - N'est plus résistante aux collisions. Il est possible de trouver des messages ayant la même empreinte en 2^{63} opérations (moins que les 2^{80} opérations requises pour une attaque exhaustive)
- SHA-256 et SHA-512
 - Nouveaux standards (256 et 512bits)

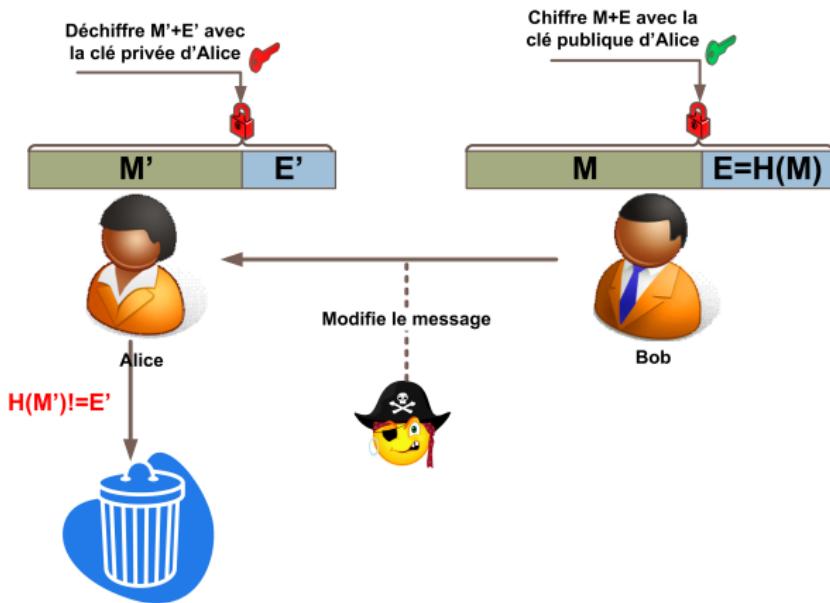
Plan

L'idée

Bob veut envoyer un message M à Alice :

- Il concatène à la fin du message M l'empreinte $E = H(M)$ du-dit message
- Le message $M + E$ ainsi obtenu est chiffré avec la clé publique d'Alice et envoyé à Alice
- Alice reçoit et déchiffre le message (éventuellement modifié par un attaquant) $M' + E'$
 - Si $E' = H(M')$, alors le message M' n'a pas été altéré (c'est le message M original)
 - Dans le cas contraire, le message est mis à la poubelle

Le protocole



- Pourquoi ça marche ?

Attaques

Ce protocole permet de résister aux attaques suivantes (dans tous les cas, l'empreinte et le message sont chiffrés) :

- L'attaquant intercepte un message. Il modifie le message sans modifier l'empreinte et l'envoie
- L'attaquant intercepte un message. Il modifie le message ainsi que (aléatoirement) l'empreinte

Paradoxe des anniversaires

Pourquoi augmenter la taille de l'empreinte ?

- Il suffit de 23 personnes pour qu'il y ait plus d'une chance sur deux pour que parmi elles deux personnes soient nées le même jour !
- Une fonction de hachage générant une empreinte de n bits donne 2^n sorties différentes
- D'après le paradoxe des anniversaires on a une chance sur deux de retrouver un message ayant une empreinte donnée en $2^{n/2}$ essai
- Augmenter n (Actuellement, plus de 128 bits)

Plan

Problème

Le problème :

- Le protocole présenté précédemment garantit la **confidentialité** ainsi que l'**intégrité** du message
- Mais pas son **authenticité** !
- Qu'est ce qui empêche Oscar de :
 - Chiffrer un message avec la clé publique d'Alice
 - Lui envoyer en se faisant passer pour Bob ?

Definition

Un message est dit authentique si son expéditeur est bien celui auquel on s'attend, il n'y a pas eu usurpation d'identité.

Exemple



- Qu'est-ce qui garantit l'intégrité ?
- Qu'est-ce qui garantit l'authenticité ?

Plan

Objectifs

Transposition au monde numérique

- La signature permet de :
 - Garantir l'**intégrité** du message
 - Garantir l'**authenticité** du message !
- Presque équivalent à une signature "classique" manuscrite
 - **Intégrité** : ajout d'une empreinte au message
 - **Authenticité** : Utilisation de cryptographie à clé publique
 - A votre avis, comment garantir l'authenticité ?

Plan

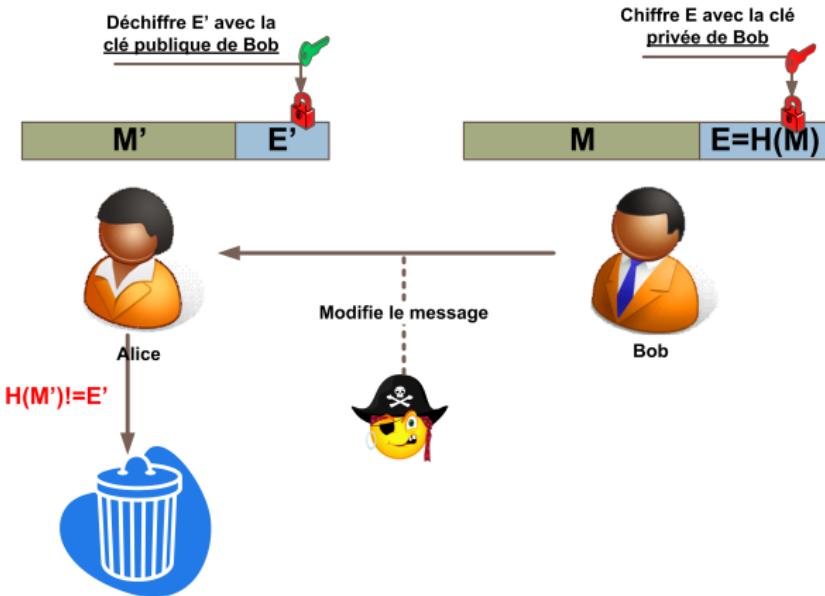
Protocole de signature numérique

- Bob veut envoyer un message M de manière authentifiée et sûre à Alice :
 - 1 Il va calculer une empreinte $E = H(M)$ du message
 - 2 Il va chiffrer l'empreinte avec sa **clé privée**
 - Seul Bob peut le faire
 - 3 Il va concaténer l'empreinte chiffrée au message M et envoyer $M + E$ à Alice
 - 4 Le message final est éventuellement chiffré avec la clé publique d'Alice si l'on veut en plus assurer la **confidentialité**

Vérification de la signature

- Alice reçoit le message signé de Bob. Elle va :
 - 1 Déchiffrer le message avec sa clé privée si il a été chiffré par Bob
 - 2 Séparer M' et E'
 - 3 Déchiffrer E' avec la **clé publique** de Bob
 - 4 **Vérifier** si l'empreinte E' est bien valide pour M'
- Si c'est le cas, elle sait que :
 - C'est bien Bob qui a envoyé le message. Pourquoi ?
 - Le message n'a pas été modifié par un tiers. Pourquoi ?

Illustration du protocole (sans confidentialité)



Algorithmes utilisés

- La signature **RSA**
 - On chiffre l'empreinte via l'algorithme RSA (et la clé privée évidemment)
- La signature **DSA**
 - Chiffrement de l'empreinte basé sur le problème du Logarithme Discret
- Et bien d'autres
 - Courbes elliptiques, etc
 - Toute fonction à sens unique et à brêche secrète peut être utilisée

Récapitulatif

- Pour **authentifier** un message (ainsi qu'assurer son **intégrité**) on peut le **signer** :
 - Calcul de l'empreinte du message \rightarrow **intégrité**
 - Chiffrement de l'empreinte avec la clé privée via un algorithme asymétrique \rightarrow **authenticité**
- Tout algorithme de chiffrement asymétrique peut être utilisé
 - Signature RSA
 - Signature DSA
 - Courbe elliptiques, etc.

Plan

Le point noir de la cryptographie à clé publique

- La cryptographie asymétrique repose essentiellement sur l'authenticité des clés publiques
 - Bob publie sa clé publique sur son site web perso
 - Oscar s'introduit sur le site et remplace la clé publique de Bob par la sienne
 - Une personne veut envoyer un message à Bob, elle récupère la clé publique sur le site de Bob
 - Oscar peut alors déchiffrer le message adressé à Bob !
- Ces clés n'ont pas forcément besoin d'être transmises de façon confidentielle, mais elles se doivent d'être transmises de façon à s'assurer de leurs authenticités
- Solution : utiliser un organisme de **confiance** pour distribuer les clés publiques

Plan

Infrastructure à clé publique

Deux modèles :

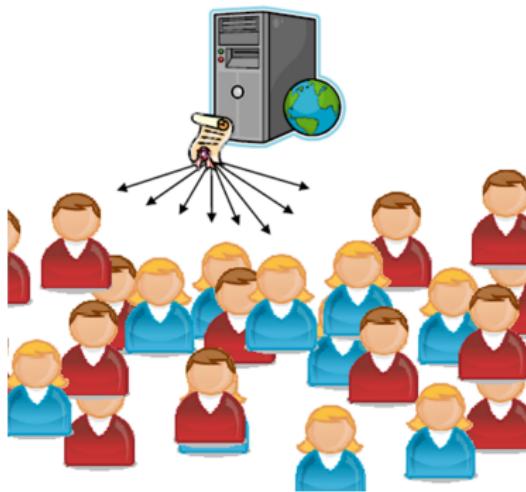
- Modèle hiérarchique
 - Infrastructure à clé publique (ICP). En anglais : Public key infrastructure (PKI)
 - Utilisé par les organismes commerciaux et gouvernementaux
 - Autorité centrale
- Réseau de confiance (Web of trust)
 - Modèle distribué sans autorité centrale
 - Utilisé par le logiciel PGP

Modèle hiérarchique (2)

- Le modèle repose sur une autorité certifiant (signant avec sa clé privée) les clés des divers intervenants
- Le rôle de l'autorité de certification (Certification Authority ou CA) est de s'assurer de la validité de la correspondance entre le nom d'une personne et une clé publique
 - Le CA émet des certificats X.509 aux personnes qu'elle a pu authentifier
- Il n'y a plus qu'un seul acteur à qui faire confiance : le CA

Comment ça marche ?

- Distribution du certificat "root" du CA à tous les intervenants
 - Certificat autosigné càd que le CA signe son propre certificat
 - Le certificat est distribué de façon sécurisée : par exemple avec le système d'exploitation



Certification du client

- Chaque intervenant s'inscrit (une seule fois) au CA afin qu'il puisse être identifié par les autres intervenants
- Il donne des preuves de son identité (carte d'identité, passeport)
 - L'intervenant reçoit un certificat qui l'identifie, signé par le CA (avec la clé privée du CA)



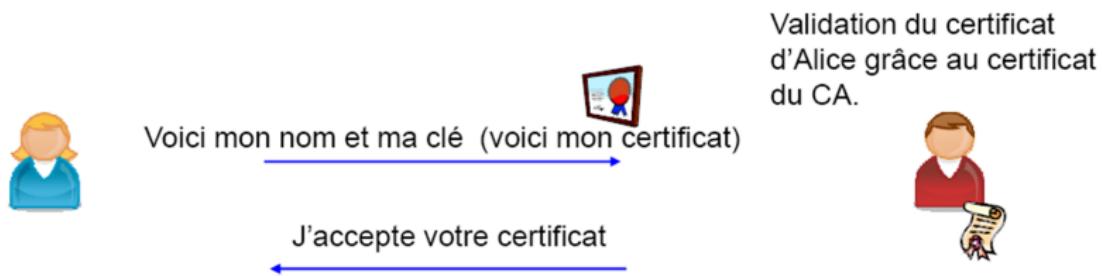
Voici mon nom, ma clé et mes preuves



Voici le certificat X.509 d'Alice signé par CA

Identification d'un intervenant

- Chaque intervenant ayant un certificat peut "prouver" son identité à tout autre intervenant ayant confiance au CA



- Bob valide le certificat avec la clé publique du CA.
- Si le certificat d'Alice est valide, Bob accepte la clé publique d'Alice

Problèmes du modèle hierarchique

Problèmes du modèle hierarchique :

- Comment distribuer de manière sûre le certificat root du CA ?
- Comment intégrer les certificats dans une application usager ?
- Si le CA et tous ses sous-serveurs tombent en panne ?
- La certification est **payante**

Plan

Modèle du réseau de confiance

- Ce modèle du **réseau de confiance** est simple et sans autorité centrale
- Utilisé par PGP
- Chaque intervenant construit son réseau de confiance
 - Il peut décider de faire confiance à un autre intervenant qui lui donne directement son certificat (+ des preuves type carte d'identité)
 - Il peut décider de faire pleine confiance à tout intervenant "connu" par **un autre intervenant** en qui il a pleine confiance
 - Plusieurs variantes de cette règle :
 - Doit être connu par plus d'un intervenant en qui il a pleine confiance
 - Peut avoir une confiance relative en ce nouvel intervenant

Problèmes

Problèmes du réseau de confiance

- Est-ce que les amis de mes amis sont mes amis ?
- Idéal pour un milieu informel, mais non pour une application où le cadre juridique est important
 - Judiciaire
 - Gouvernemental

Récapitulatif

- Pour **authentifier** un message (ainsi qu'assurer son **intégrité**) on peut le **signer** :
 - Calcul de l'empreinte du message
 - Chiffrement de l'empreinte avec la clé privée via un algorithme asymétrique
- Mais pour vérifier la signature, il faut avoir **confiance dans la clé publique**
- La certification essaye de résoudre ce problème
 - Le modèle hierarchique
 - Une autorité de certification signe des clés publique en échange de preuves d'identité
 - Tout le monde fait confiance au CA
 - Le réseau de confiance
 - Les amis de mes amis sont mes amis