

## Cloud computing Travaux pratiques

### Objectif

Se donner une vision théorique et pratique de la sécurité de la correspondance par mail.  
En particulier sont abordés : le chiffrement et l'authenticité des contenus.

### Un peu de théorie

#### *Chiffrement symétrique*

Alice chiffre un message pour l'envoyer à Bob :

$\text{Message}_{\text{chiffré}} = \text{Fonction\_Chiffrer}(\text{Clé}_{\text{symétrique}}, \text{Message}_{\text{initial}})$

Bob déchiffre le message :

$\text{Fonction\_Déchiffrer}(\text{Clé}_{\text{symétrique}}, \text{Message}_{\text{chiffré}}) = \text{Message}_{\text{initial}}$

Le chiffrement et l'authenticité sont bien traités, mais si Alice possède 150 correspondants, elle devra partager 150 clés, ce qui n'est pas pratique.

#### *Chiffrement asymétrique*

Le chiffrement asymétrique permet d'avoir un couple de clés (clé privée, clé publique) unique pour l'ensemble de ses correspondants :

Pour envoyer un message chiffré à Bob, Alice utilise la clé publique de Bob

$\text{Message}_{\text{chiffré}} = \text{Fonction\_Chiffrer}(\text{Clé}_{\text{publique de Bob}}, \text{Message}_{\text{initial}})$

Bob reçoit le message chiffré avec sa clé publique et il est le seul à pouvoir déchiffrer le message :

$\text{Message}_{\text{initial}} = \text{Fonction\_Déchiffrer}(\text{Clé}_{\text{privée de Bob}}, \text{Message}_{\text{chiffré}})$

Ce message a pu être envoyé par n'importe qui et rien ne prouve que Alice est bien

l'expéditeur.

## **Signature numérique**

Alice doit donc prouver qu'elle est bien l'expéditeur, elle utilise donc un mécanisme de signature numérique en suivant les étapes suivante :

- 1) création d'un condensat (empreinte) du message à l'aide d'une fonction de hachage :

Condensat = Fonction\_Hash (Message<sub>initial</sub>)

- 2) chiffrer le condensat avec sa clé privée :

Signature = Fonction\_Chiffrer (Clé<sub>privée de Alice</sub> , Condensat)

Finalement :

Signature = Fonction\_Chiffrer (Clé<sub>privée de Alice</sub> , Fonction\_Hash (Message<sub>initial</sub>))

Bob reçoit Message<sub>initial</sub> et Signature et pour vérifier la signature, il doit donc vérifier l'égalité :

Fonction\_Hash (Message<sub>initial</sub>) = Fonction\_Déchiffrer (Clé<sub>publique de Alice</sub> , Signature)

## **Chiffrement et signature**

Alice souhaite envoyer un message à Bob

- 1) fabrication de la signature :

Signature = Fonction\_Chiffrer (Clé<sub>privée de Alice</sub> , Fonction\_Hash (Message<sub>initial</sub>))

- 2) chiffrer le message :

Message<sub>chiffré</sub> = Fonction\_Chiffrer (Clé<sub>publique de Bob</sub> , Message<sub>initial</sub>)

- 3) envoi des deux éléments Message<sub>chiffré</sub> et Signature à Bob

- 4) Bob déchiffre le message :

Message<sub>initial</sub> = Fonction\_Déchiffrer (Clé<sub>privée de Bob</sub> , Message<sub>chiffré</sub>)

- 5) Bob vérifie l'authenticité du message : il fabrique le condensat

Fonction\_Hash (Message<sub>initial</sub>),

il déchiffre la signature

Fonction\_Déchiffrer (Clé<sub>publique de Alice</sub> , Signature)

et le compare au condensat.

## **Définitions**

Enigmail : extension pour client de messagerie qui utilise GnuPG

GnuPG ou Gnu Privacy Guard ou GPG : implémentation libre de PGP

PGP ou Pretty Good Privacy : PGP : logiciel créé par Phil Zimmermann en 1991, code source ouvert mais non libre ?

OpenPGP : standard (RFC 4880) qui décrit le protocole

## Un peu de pratique

Installer un client de messagerie :

apt-get install icedove-l10n-fr

passer l'étape de la configuration automatique et configurer les éléments suivants où XX est votre numéro d'étudiant sur la feuille de présence :

nom et prénom : imacXX

Adresse électronique : imacXX@test.wargon.org

mot de passe : h5H7kVXX

la configuration trouvée par icedove doit être bonne, cliquer sur Terminé

accepter les certificats qui sont des certificats autosignés pour le serveur de lecture des mails d'une part et pour le serveur d'envoi de mail d'autre part.

Envoyer et recevoir des mails avec un autre compte imacXX permettra de vérifier que les configurations sont bonnes.

installer l'extension Enigmail :

Menu / Modules complémentaires / chercher enigmail / installer / redémarrer icedove

Utiliser l'assistant - Suivant

Ne pas chiffrer mes messages par défaut - Suivant

Ne pas signer mes messages par défaut - Suivant

Modifier les paramètres pour que Enigmail fonctionne mieux - Suivant

Créer une nouvelle paire de clés - Suivant

clé = « ma cle mon secret XX »

générer le certificat de révocation de la clé

ne pas déverrouiller automatiquement la clé, cela permet d'avoir une visibilité sur les mails dans leur format chiffré.

(B) envoie d'un mail chiffré à (A) :

(A) envoie sa clé publique à (B) : Menu / Enigmail / Gestion des clés / Fichier / Envoyer des clés publiques par courrier électronique

(B) importe la clé publique de (A) clic droit sur le fichier joint puis Importer une clef OpenPGP

(B) peut envoyer un mail chiffré à (A) la clé en bas à droite de la fenêtre doit être jaune.

(A) envoie un mail signé à (B) :

le crayon en bas à droite de la fenêtre doit être jaune.

(B) reçoit le mail, il peut vérifier la signature avec la clé publique de (A) et définir avec le menu « Détail » le niveau de confiance de l'expéditeur, essayer la confiance absolue.

(B) envoie un mail chiffré et signé à (A)

le crayon et la clé en bas à droite de la fenêtre doivent être jaunes

(A) peut déchiffrer le mail, mais ne peut pas vérifier la signature, pourquoi ?

Trouver une solution.

Enfin envoyer un mail chiffré et signé de (A) vers B)

Finalement, télécharger la clé publique de imac20

[http://blog.wargon.org/wp-content/uploads/cours/2014-2015/0x10C49F8E\\_pub.asc](http://blog.wargon.org/wp-content/uploads/cours/2014-2015/0x10C49F8E_pub.asc)

importer la clé :

Menu / Enigmail / Gestion de clefs / Fichier / Importer des clefs depuis un fichier

Envoyer votre clé publique à imac20 afin qu'il puisse vérifier votre signature, puis envoyer un mail chiffré et signé à imac20.

Références :

wikipedia

une introduction intéressante se trouve là :

<http://www.controle-tes-donnees.net/outils/GnuPG.html>