

# HTTPS : HTTP Sécurisé

Décembre 2000

sa@cru.fr

## Objectifs

- Crypter la communication entre le client et le serveur
- Authentifier le serveur
- Authentifier la personne

## Principes élémentaires de crypto

- Cryptage symétrique : pour chaque couple communicant, une clef secrète partagée par les deux personnes.
- $\text{crypt}(\mathbf{key}, \text{crypt}(\mathbf{key}, \text{msg})) = \text{msg}$
- comment partager les clefs ?
- Nombre de clefs à gérer :  $\mathbf{n(n-1)/2}$

## Principes élémentaires de crypto

- Cryptage asymétrique : 2 clefs associées
- $\text{crypt}(\mathbf{key2}, \text{crypt}(\mathbf{key1}, \text{msg})) = \text{msg}$
- 1 clef publique, 1 clef privée par personne
- algorithmes coûteux : on les utilise uniquement pour crypter une clef (aléatoire et à usage unique) beaucoup plus courte que le message. Cette clef est utilisée pour crypter le message avec un algorithme symétrique.

## Principes élémentaires de crypto

- Cryptage : crypte le message avec la clef publique du destinataire
- Signature : on crypte une *empreinte* du message avec la clef privée de l'émetteur.
- Comment être certain de la provenance de la clef publique ?

## Autorité de certification

- Solution : une autorité est chargée de signer les clefs publiques : elle crypte une empreinte de la clef publique de la personne avec sa clef privée.
- On s'assure de la provenance de la clef publique en vérifiant la signature qui y a été apposée avec la clef publique de l'autorité de certification.

## Certificat X509

- Le modèle est récursif (les clefs des autorités de certification sont elles-mêmes certifiées, ou auto-signées)
- Faire confiance dans un certificat, c'est:
  - vérifier la chaîne de signature
  - reconnaître une des autorités de certification de cette chaîne parmi les autorités de confiance
- La clef publique et sa ou ses signatures, le «distinguish name» et quelques extensions constituent un certificat X509

## Certificat X509

- Un certificat contient :
  - le sujet (sous forme d'un DN)
  - la clef publique,
  - la signature de cette clef par une CA,
  - un numéro de série
  - parfois le certificat des CA signataires
  - période de validité
  - usage (email, serveur, signature d'application, ca)

MD5 Fingerprint=34:21:86:66:0B:20:64:56:E5:9F:EE:D6:6C:7D:DA:42

PEM Data:

Certificate:

Data:

Version: 3 (0x2)

Serial Number: 0 (0x0)

Signature Algorithm: md5WithRSAEncryption

Issuer: CN=test lustu, O=cru, C=fr

Validity

Not Before: Mar 21 10:26:22 2000 GMT

Not After : Mar 21 10:26:22 2002 GMT

Subject: CN=test lustu, O=cru, C=fr

Subject Public Key Info:

Public Key Algorithm: rsaEncryption

RSA Public Key: (2048 bit)

Modulus (2048 bit):

00:af:52:14:0f:32:3b:78:73:cb:d5:38:a1:9f:42:

72:e9:71:9f:33:2c:23:1e:28:bb:9f:f1:ca:eb:73:

.....

e8:af:b7:1e:a7:49:8d:86:4d:17:13:04:5f:3d:e2:

77:2f

Exponent: 65537 (0x10001)

X509v3 extensions:

X509v3 Basic Constraints:

CA:TRUE

X509v3 Subject Key Identifier:

## IGC ( PKI )

- La PKI délivre, publie, renouvelle et révoque des certificats. Parfois elle assure aussi un service de recouvrement des clefs privées.
- Divisée en autorité d'enregistrement (RA) et une autorité de certification.
- Fonctionne avec des cahiers des charges : politique d'enregistrement, politique de certification qui dépendent des usages et des contraintes légales.

## Domaines d'application

- IMAPS
- S/MIME
- SMTP/TLS
- IPSec
- ...
- HTTPS
  - url : https://... port par défaut 443

## HTTPS

- mode chiffrement : un certificat X509 émis par une autorité doit être installé sur le serveur.
- Si la chaîne de vérification du certificat serveur ne croise pas une des autorités de confiance du client, la session est chiffrée, mais le serveur n'est pas authentifié. (Le client reçoit un tas de messages inquiétants).
- Importance de la liste des CAs par défaut des clients

## HTTPS

- Mode chiffré avec authentification du client.
- 2 certificats X509 : client + serveur
- La communication est cryptée, le serveur récupère les infos extraites du certificat client en particulier son DN

## Que faire avec HTTPS ?

- 1 seul certificat (coté serveur) pour crypter la saisie d'informations sensibles.
- E-commerce genre *camif.fr* se limite à cela !
- Crypter la saisie de mots de passe.
- Crypter les données d'un formulaire.
- Crypter des cookies.

## Que faire avec HTTPS ?

- 1 certificat serveur + 1 certificat par client
- permet de sécuriser fortement l'authentification des clients,
- Remplace des méthodes d'authentification peu pratique : plus de mot de passe, plus de formulaire préalable, plus de cookie ...

## Que faire avec HTTPS ?

- restreindre les droits d'accès à une partie du client. Logiques possibles :
  - distribuer des certificats aux gens autorisés. Un utilisateur peut avoir beaucoup de certificats. La gestion des droits est faite par la PKI.
  - lister dans le serveur les DN autorisés. Pas de notion de privilège dans le certificat. (réplication des privilèges sur chaque serveur)
  - on peut panacher ces méthodes : classe de certificats

## Apache

- open\_ssl <http://www.openssl.org>
- mod\_ssl <http://www.modssl.org>
- mod\_ssl peut être utilisé en mode DSO
- Sur red hat 7.0 le rpm d'apache installe open\_ssl et mod\_ssl. Le démarrage standard d'apache ouvre le port 80 et le port 443.

## HTTPD.conf

- Listen 443
- SSLEngine on|off
- SSLCertificateFile SSLCertificatekeyFile
- SSLCACertificate(File|Path)
- SSLCARevocation(File|Path)
- SSLEngine on|off
- SSLVerifyClient none|optional|require
- SSLVerifyDepth n

## HTTPD.conf

- `SSLRequireSSL` : pour éviter une « back door » accidentelle
- `SSLOptions` +/- `<option>`
  - `StdEnvVars` : extraction des infos du certificat vers des variables à usage des CGI
  - `OptRenegotiate` : Optimisation des re-négociations liées aux directives `<directory>`.

## HTTPD.conf

- `FakebasicAuth` : le DN est utilisé comme nom d'utilisateur dans le fichier d'authentification basique. (le mot de passe est alors `xxj31ZMTZzkVA`)
- `StrictRequire` utile avec « satisfy any » si on veut `SSLRequire` ou `SSLRequireSLL` et d'autres conditions. Grâce à cette directive le « any » ne concerne que les autres conditions

## le TP

- installer un certificat X509 côté serveur
- installer un certificat côté client
- configurer apache pour restreindre l'accès à certains certificats
- exécuter un cgi qui affiche les infos du certificat

## PKI

- L'apport de HTTPS (et des autres applications des certificats X509) est considérable
- Les applications sont disponibles (apache, roxen, netscape, iis, outlook, messenger, ...)
- Sans PKI, il faut renoncer ou accepter de se soumettre au bon vouloir des PKI commerciales
- Les projets démarrent : cnrs, pki ministérielle, projets européens, ...