Programmation Réseau

TLS (aka SSL)

Jean-Baptiste.Yunes@liafa.jussieu.fr

UFR Informatique

2011-2012

TLS (aka SSL)

- Transport Layer Security (aka Secure Socket Layer) est une couche intermédiaire entre les couches Transport et Application de la pile Internet et joue un rôle équivalent à la couche Session du modèle OSI

- elle fournit des services de sécurité suivant :
  - authentification
  - confidentialité
  - intégrité

- D’autre part elle possède deux caractéristiques intéressantes :
  - la transparence (les protocoles applicatifs ne sont pas modifiés)
  - la spontanéité (établissement à partir de zéro d’un certain niveau de sécurité, i.e. pas de configuration a-priori)
- la première normalisation correspondante est la RFC 2246 (01/1999)
  - il en existe une version « paquet », DTLS normalisée dans RFC 4347

- L’idée est d’utiliser un système de cryptographie asymétrique (type RSA) pour établir une liaison sûre utilisant de la cryptographie symétrique (type DES)
  - la cryptographie asymétrique est utilisée pour échanger la clef de cryptage de la communication
- En général une authentification est aussi réalisée
  - pour s’assurer que le serveur est bien qui il affirme être…

- pour le **serveur** il faut une **paire de clefs** cryptographiques (une privée, une publique)

  - ces clefs seront stockées dans une armoire à clefs (keystore), cette armoire étant elle-même protégée par un mot de passe

```
keytool -genkey
        -alias nomclef
        -keyalg RSA
        -keystore armoire.jks
        [-storepass mpd]
```

- pour le **client** il faut une armoire de clefs contenant le certificat correspondant à la clef publique du serveur

  - extraire le certificat de l'armoire du serveur

```
keytool -export
        -alias nomclef
        -keystore armoire.jks
        [-storepass mdp]
        -file certificat.crt
```

- importer le certificat dans l'armoire du client, tout en le reconnaissant comme certificat acceptable

```
keytool -importcert
        -trustcacerts
        -alias nomclef
        -file certificat.cer
        -keystore armoireclient.jks
        [-storepass mdp]
```

- Côté serveur la création de la socket s'effectue par

```
ServerSocketFactory ssocketFactory =
      SSLServerSocketFactory.getDefault();

ServerSocket ssocket =
      ssocketFactory.createServerSocket(port);
```

- Côté client la création de la socket s'effectue par

```
SocketFactory socketFactory =
      SSLSocketFactory.getDefault();

Socket socket =
      socketFactory.createSocket(hostname,port);
```

- Attention les instructions précédentes ne permettent pas d'obtenir
  - la spontanéité
  - l'authentification du client (double authentification)
- Elles ne doivent pas être utilisées pour établir une liaison « commerciale »
  - le certificat est auto-signé... (pas de sens du point de vue de la sécurité et sensibilité à des attaques)