

# Confidentialité des informations

V1.1

C. Drocourt

[drocourt@e-conseil.fr](mailto:drocourt@e-conseil.fr)

## SOMMAIRE

1 - Infrastructure à clé publique, certificats, signature électronique et empreinte numérique.....	3
2 - Sécurité du transport d'informations avec HTTPS.....	3
3 - Gérer ses certificats serveurs, le standard X509.....	3
4 - Quelle autorité de certification choisir ?.....	3
5 - Dans quel cas être sa propre autorité de certification ?.....	3
6 - Accélérateurs SSL : comment choisir entre appliance et carte sur serveur.....	3
7 - L'unité de mesure TPS : Transactions Par Seconde.....	3
8 - Chiffrement.....	4
9 - Certificats.....	12
10 - Travaux pratiques sous Windows.....	29

# 1 - Chiffrement

## 1.1 - Chiffrement symétrique

- DES : 56 bits, algorithme trop long, clés trop petites, chiffrement par blocs (bloc ciphers),
- Triple DES : 168 bits théoriques mais 112 bits effectifs (attaque par rencontre au milieu), même problème de lenteur, chiffrement par blocs (bloc ciphers),
- AES : 128, 192 ou 256 bits, standard américain (128 bits : Niveau Secret, 192-256 bits : Niveau Top Secret), chiffrement par blocs (bloc ciphers),
- RC4 : « Ron's Code » ou « Rivest's Cipher », clé variable de 1 à 256 octets (2048 bits), souvent 5 (40 bits) ou 16 (128 bits), rapide (basé sur XOR), utilisé dans WEP et WPA, chiffrement par flux (stream ciphers),
- RC5 : « Ron's Code » ou « Rivest's Cipher », clé variable de 1 à 256 octets (2048 bits), rapide (basé sur XOR), chiffrement par blocs (bloc ciphers) de taille variable, propriété de RSA Security,

- ECB (Electronic Codebook) : On applique l'algorithme au texte clair en transformant normalement chaque bloc de texte clair.
- CBC (Cipher Block Chaining) : On applique une opération XOR entre le bloc courant en clair et le bloc précédent chiffré, et on chiffre le bloc obtenu. Le premier bloc utilise un vecteur d'initialisation normalement aléatoire.
- CFB (Cipher Feedback) : On applique une opération XOR entre le bloc courant en clair et le bloc précédent chiffré. Le premier bloc utilise un vecteur d'initialisation normalement aléatoire.
- OFB (Output Feedback) : Utilisation également d'opération XOR et de blocs temporaires.

Attaque exhaustive : nombre de solutions =  $2^{\text{taille de la clef}}$ .

Application :

- `openssl des -e -a -in fichier.txt > fichier.des.txt`
- Utilisation de la fonction `crypt()` en C sous unix

## 1.2 - Fonctions de hachage

- MD4 : 128 bits, « Message Digest », utilisé dans les mots de passe NTLM de Microsoft, gros problème de sécurité car collision possible avec  $2^8$  opérations.
- MD5 : 128 bits, « Message Digest », basé sur md4.
- SHA-1 : 160 bits, « Secure Hash Algorithm », problèmes récents de collisions réduit à  $2^{52}$  au lieu de  $2^{80}$ ,
- SHA-2 : 224, 256, 384 ou 512 bits, « Secure Hash Algorithm », algorithme similaire à SHA-1,
- RIPEMD-128 : 128 bits, « RACE Integrity Primitives Evaluation Message Digest », collision trouvée en 2004,
- RIPEMD-160 : 160 bits, « RACE Integrity Primitives Evaluation Message Digest »,
- SHA-3 : Appel par le NIST en 2007, doit se terminer en 2012.

Le salage : (Salt) Important pour les mots de passe !!

Application :

- Créez un fichier texte,
- Utilisez md5sum dessus,
- Modifiez 1 seul caractère du fichier,
- Appliquez à nouveau md5sum dessus, alors ?

## 1.3 - Chiffrement asymétrique

Utilisation : Chiffrement, Authentification, Signature, ...

- RSA : « Rivest, Shamir, et Adelman » - 1977, basé sur la factorisation et les nombres premiers, breveté mais expiré en 2000, 1024 bits minimum mais déconseillé, 2048 recommandé actuellement pour les données sensibles, 4096 pour les très sensibles.
- DSA : « Digital Signature Algorithm » - 1991, basé sur les logarithmes, uniquement pour la signature.
- ElGamal : Non couvert par un brevet, signature et chiffrement, basé sur les logarithmes,

### Application :

Pour générer une clé privée RSA de longueur 1024 dans un fichier `rsa.priv` :

```
$ openssl genrsa -out rsa.priv
```

On peut ajouter du chiffrement avec l'option `-algo` qui peut être `-des`, `-des3`, etc. Puis extraire la clé publique RSA dans un fichier `rsa.pub` à partir de la clé privée :

```
$ openssl rsa -in rsa.priv -pubout -out rsa.pub
```

Voyons comment chiffrer un fichier `fic.txt` en un fichier `fic.enc` :

```
$ openssl rsautl -encrypt -pubin -inkey rsa.pub -in fic.txt  
-out fic.enc
```

Puis comment le déchiffrer dans un fichier `fic.dec` :

```
$ openssl rsautl -decrypt -inkey rsa.priv -in fic.enc -out  
fic.dec
```

Chiffrer avec la clef privé :

```
$ openssl rsautl -sign -inkey rsa.pem -in fichier.txt -out  
fic.dat
```

Déchiffrer :

```
$ openssl rsautl -verify -pubin -inkey rsa.pub -in fic.dat  
-out decod.txt
```

## 1.4 - Autres outils

- Utilisation de ssh, (connexion standard, ssh-keygen, ...),
- Utilisation de GPG PGP : chiffrement, signature, fonctionnement,

## 2 - Certificats

### 2.1 - Certificats électroniques x509

Quatre types de certificats :

- Classe 1 : Vérification sommaire (existence de l'adresse mail ou du domaine),
- Classe 2 : Vérification standard (photocopie de papiers d'identité, de documents officiels, ...),
- Classe 3 : Vérification forte (rencontre physique des personnes),
- Classe 3+ : Idem Classe 3 avec stockage sur support physique.

## 2.2 - Structure

- Version de X509,
- Numéro de série,
- Algorithme de signature utilisé,
- Nom de l'autorité de certification émettrice,
- Date de début et de fin de validité,
- Clé publique,
- Algorithme de la clé publique,
- Signature (empreinte du certificat signé avec la clé privé du CA),

## 2.3 - Fonctionnement

- le client demande le certificat au serveur,
- le serveur retourne le certificat au client,
- le client calcul l'empreinte du certificat,
- le client déchiffre l'empreinte du certificat avec la clé publique du CA,
- le client compare les deux empreintes,

### Utilisation :

- web (https),
- mail (smime),
- authentification (PKI),
- ...

## 2.4 - Révocation

Une liste de révocation des certificats, signée également par le CA est nécessaire :  
CRL (Certificate Revocation List).

Problème : Demande un travail important du client,

Solution : Ajout du protocole OCSP (Online Certificate Status Protocol),

## 2.5 - Formats de fichiers

- DER : Distinguished Encoding Rules (en ASN.1),
- PEM : DER en base64,
- CSR : Certificate Signing Request,
- CER et CRT : Certificats binaires x509,

Exercice :

Consulter les certificats présents dans le navigateur,

## 3 - Transport Layer Security

### 3.1 - Introduction

Anciennement nommé SSL (Secure Socket Layer) :

- SSLv1 : 1994 par Netscape,
- SSLv2 : 1995 par Netscape,
- SSLv3 : 1996 par Netscape,
- TLSv1 (SSLv3.1) : 1999 par l'IETF, rachat du brevet de Netscape en 2001,
- TLSv1.1 : 2006
- TLSv1.2 : 2008

Se situe au niveau 5 du modèle OSI (session), et utilisable avec tous les protocoles de niveau supérieur sans modification : HTTPS, SMTPS, IMAPS, ... Il permet l'authentification, la confidentialité et l'intégrité des données.

Remarque : Dialogue SSL avant le dialogue HTTP -> problème du virtualhost.

### Déroulement d'une connexion :

- Le client envoie une requête au serveur contenant en outre la plus haute version du protocole SSL qu'il peut utiliser, ainsi qu'une liste d'algorithmes de chiffrement classés par ordre de priorité décroissante (protocole Handshake),
- Le serveur répond par la version SSL choisie, l'algorithme choisi, il envoie également son certificat (x509),
- Le serveur peut également demander un certificat au client,
- Le client génère une pré-clé qu'il chiffre avec la clé publique du serveur et l'envoie à ce dernier,
- Le client et le serveur génèrent des clés de sessions qu'ils utiliseront pour dialoguer en chiffrement symétrique,

### Exercice :

Client : `openssl s_client -connect www.u13.org:443`

Serveur : `openssl s_server -cert certificat.pem -www`

### 3.2 - Sécurité de SSL/TLS :

Les seuls protocoles à utiliser sont au minimum TLSv1 et SSLv3, tous les précédents possédant des failles (Exemple du ver "slapper" pour SSLv2 en 2002). Utiliser un CIPHER trop faible pour la clé pourrait amener à un décodage de celle-ci. En effet, en utilisant openssl, on arrive à forcer les modes Protocol/Cipher que l'on veut.

Exemple :

```
# openssl s_client -connect ssl.u13.org:443
New, TLSv1/SSLv3, Cipher is DHE-RSA-AES256-SHA
...
SSL-Session:
    Protocol    : TLSv1
    Cipher      : DHE-RSA-AES256-SHA
...

```

Qui montre que l'on est passé en TLSv1 / DHE-RSA-AES256-SHA

Mais si l'on tente de désactiver TLSv1 :

```
# openssl s_client -connect ssl.u13.org:443 -no_tls1
```

Et désactiver également SSLv3 :

```
# openssl s_client -connect ssl.u13.org:443 -no_tls1  
-no_ssl3
```

Le serveur doit refuser, mais certains acceptent et permettent d'utiliser des protocoles très peu sécurisés.

Il est possible également de forcer les algorithmes comme DES :

```
openssl s_client -host ssl.u13.org -port 443 -cipher DES
```

Et de forcer le non chiffrement :

```
openssl s_client -host ssl.u13.org -port 443 -cipher NULL
```

Pour pallier à la faille, il faut modifier les instructions de configuration suivantes dans le fichier de configuration de apache : SSLProtocol et SSLCipherSuite.

### 3.3 - Création et utilisation des certificats avec openssl :

Générer une clef RSA de 2048 bits :

```
openssl genrsa -out cle-rsa.pem 2048
```

Exporter la clé publique correspondante :

```
openssl rsa -in cle-rsa.pem -pubout
```

Générer une requête de certification :

```
openssl req -new -nodes -newkey rsa:2048 -keyout mykey.pem  
-out myreq.pem
```

Idem si le clé existe déjà :

```
openssl req -new -key mykey.pem -out myreq.pem
```

Générer un certificat auto signé :

```
openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout  
mycert.pem -out mycert.pem
```

Vérifier un certificat :

```
openssl verify mycert.pem
```

### 3.4 - Devenir sa propre autorité de certification :

Générer une paire de clés pour la CA :

```
cd /etc/pki/CA/private  
openssl genrsa -out ca-prive.key 2048
```

Créer une requête de certification pour le CA (repondre aux questions) :

```
openssl req -new -key ca-prive.key -out ca.csr
```

Auto-signer le certificat du CA :

```
openssl x509 -req -days 3650 -in ca.csr -signkey ca-  
prive.key -out ca.crt
```

Créer une paire de clés pour le serveur :

```
cd /etc/pki/tls/private  
openssl genrsa -out serveur.key 2048
```

Créer une requête de certification pour le serveur :

```
cd /etc/pki/tls/certs  
openssl req -new -key ../private/serveur.key -out  
serveur.csr
```

Signer le certificat du serveur avec le certificat du CA (l'option -Ccreateserial n'est nécessaire que la première fois) :

```
openssl x509 -req -days 365 -in serveur.csr -out serveur.crt  
-CA ../..CA/private/ca.crt -CAkey ../..CA/private/ca-  
prive.key -Ccreateserial
```

Afficher les informations sur le certificat :

```
openssl x509 -text -in serveur.crt
```

Le vérifier :

```
openssl verify -CAfile ../..CA/private/ca.crt serveur.crt
```

Afin de ne pas avoir de message d'avertissement, il faut installer le certificat du CA sur les postes de travail, soit en utilisant directement le fichier ca.crt, soit en exportant la clé publique en pkcs12 :

```
openssl pkcs12 -export -in ca.crt -nokeys -out ca.p12
```

Tester le fonctionnement du certificat :

```
openssl s_server -cert serveur.crt -key  
../private/serveur.key -www
```

Et dans une autre fenêtre :

```
openssl s_client -connect localhost:4433
```

Modifier la configuration du serveur « Apache » :

```
/etc/httpd/conf.d/ssl.conf
```

Modifier les lignes suivantes :

```
SSLCertificateFile /etc/pki/tls/certs/serveur.crt  
SSLCertificateKeyFile /etc/pki/tls/private/serveur.key
```

Démarrez « Apache » :

```
service httpd start
```

Tentez une connexion avec votre navigateur sur l'adresse suivante :

```
https://localhost/
```

Tester les performances :

```
openssl s_time -connect localhost:443 -www /
```

## 4 - Mise en place sous Windows 2003 Server

Mise en oeuvre de S.S.L. sous Internet Information Serveur avec autorité de certification Microsoft.

### 4.1 - Active Directory

- Panneau de Configuration / Outils d'administration / Gérer votre serveur
- Ajouter ou supprimer un rôle
- Contrôleur de Domaine Active Directory AD,
- Nom DNS complet : societe13.com
- Ajout et configuration du DNS,
- Attente ... et redémarrage,

## 4.2 - Autorité de certification :

- Ajout/suppression de programmes,
- Ajout Serveur d'applications/Services IIS (Normalement déjà installé),
- Ajout Services de certificats,
- Autorité racine d'entreprise,
- Cocher la case : Utiliser les paramètres personnalisés ...
- Cocher la case : Autoriser ce fournisseur de services cryptographiques à interagir avec le bureau,
- Entrez le nom du CA, par exemple : autorite,
  - > Génération des clés,
  - > Activation des ASP,

## 4.3 - DNS

- Panneau de Configuration / Outils d'administration / Services
- Serveur DNS, Type de démarrage : automatique,
- Démarrer maintenant,

## 4.4 - Certificat de site

- Gestionnaire de service IIS, Sélection Sites Web (colonne gauche),
- Sélection Site par défaut (colonne droite), bouton droit, propriété,
- Onglet sécurité de répertoire,
- Section « Communications sécurisées »,
- Bouton « Certificat de serveur », Sélectionner « Créer un certificat »,
- Sélectionner « Envoyer immédiatement la demande à une Autorité de certification en ligne »,
- Sélectionner un nom et une taille de clé,
- Entrez le nom de l'organisation et le nom de l'unité d'organisation,
- Entrez le nom DNS, conservez la proposition de nom (exemple mw172) mais ajouter le domaine (exemple : mw172.societe13.com),
- Entrez Pays/Région/Ville,
- Port SSL : Laissez 443 (port https par défaut),
- Sélectionner une autorité de certification : Normalement une seule, la votre !!
- Consultez le certificat,

## 4.5 - Test

- Création d'une page par défaut « index.html » dans « c:\Inetpub\wwwroot »,
- Connexion avec le navigateur <http://localhost>,
- Connexion sécurisée avec le navigateur <https://localhost/> -> Normalement Erreur !!  
Pourquoi ?
- Connexion sécurisée avec le navigateur <https://mw172.societe13.com/> -> Normalement OK.

## 4.6 - Vérification du certificat délivré

- Panneau de Configuration / Outils d'administration / Autorité de Certification,
- Section « Certificats délivrés »,