NFA083 – Réseau et Administration Web
Eléments de sécurité[1]

Tristan Crolard

Laboratoire CEDRIC
Equipe « Systèmes Sûrs »

tristan.crolard@cnam.fr

cedric.cnam.fr/sys/crolard

1. Cours basé sur les supports de Sami Taktak

Introduction à la sécurité

Principaux objectifs :

- **Confidentialité** des données échangées (ou stockées)
- **Authentification** des parties (utilisateurs, sites, serveurs)
- **Intégrité** des données échangées (ou stockées)
- **Disponibilité** du système (à préserver)

Introduction à la sécurité

Principales sources de vulnérabilités :

- Vulnérabilités théoriques :
  - faiblesses des algorithmes ou des protocoles
  - hypothèses ou conjectures obsolètes

- Vulnérabilités pratiques :
  - faiblesses des implantations (algorithme ou programme)
  - faiblesses du système ou du compilateur
  - faiblesses au niveau matériel

Techniques de chiffrement

- Chiffrement symétrique :
  - clé de chiffrement **secrète** connue des 2 parties
  - clé utilisée pour chiffrer et déchiffrer
- Chiffrement asymétrique :
  - chaque partie possède une clé **privée** et une clé **publique**
  - la clé **privée** doit rester **secrète**
  - la clé **publique** est **connue** de tous

Usages

- Chiffrement symétrique (peu coûteux) :
  - **confidentialité**
    pour chiffrer des données ou des sessions (AES pour TLS)

- Chiffrement asymétrique (plus coûteux) :
  - **intégrité** et **authentification**
    signatures et certificats (RSA pour HTTPS)
  - **confidentialité**
    échange de clé secrète (RSA et DH pour TLS)

Usages : chiffrement asymétrique

- **intégrité** et **authentification**
  - message chiffré avec la clé privée de l'émetteur
  - message déchiffré par le destinataire avec la clé publique de l'emetteur
  - tout le monde peut déchiffrer le message
    chiffrer seulement un code de contrôle suffit donc
    (notion de signature electronique)

- **confidentialité**
  - message chiffré avec la clé publique du destinataire
  - message est déchiffré par le destinataire avec sa clé privée
  - seul le destinataire peut déchiffrer le message

Chiffrement symétrique : exemple (XOR)

– Voici le mot « **MESSAGE** » converti en binaire :

lettres	M	E	S	S	A	G	E
ASCII	77	69	83	83	65	71	69
binaire	01001101	01000101	01010011	01010011	01000001	01000111	01000101

– Le mot « **CLE** » en binaire est lui représenté par 01000011 - 01001100 - 01000101.

message	01001101	01000101	01010011	01010011	01000001	01000111	01000101
clé (répétée)	01000011	01001100	01000101	01000011	01001100	01000101	01000011
message chiffré (XOR)	00001110	00001001	00010110	00010000	00001101	00000010	00000110
...puis déchiffré (XOR)	01001101	01000101	01010011	01010011	01000001	01000111	01000101

Remarque : répéter simplement la clé permet des attaques basée sur un « dictionnaire de code », il est préférable d'utiliser un générateur de nombres pseudo-aléatoires pour « étirer » la clé (et se rapprocher ainsi du chiffre parfait de Vernam).

Chiffrement symétrique

- **Chiffrement par flot**

Utilise un générateur de nombres pseudo-aléatoires cryptographique.
Utilisés aujourd'hui : RC4

- **Chiffrement par bloc**

Utilisés aujourd'hui : DES (obsolète[2]) et AES (128, 192 et 256)
Définissent des fonctions plus complexes que le XOR.

Deux principaux paramètres de sécurité :

- La taille du bloc (e.g. $n = 64$ ou 128 bits pour l'AES).
- La taille de clé (e.g. $k = 128$, 192 ou 256 bits).

2. https://www.securiteinfo.com/cryptographie/cracked.shtml

Chiffrement asymétrique : RSA

Algorithme dû à Ronald Rivest, Adi Shamir et Leonard Adleman (1977)

Un exemple simple (en pratique il faut de très grands nombres premiers) :

1. on choisit deux nombres premiers $p = 3$, $q = 11$
2. leur produit $n = 3 \times 11 = 33$ est le module de chiffrement
3. $\varphi(n) = (3 - 1) \times (11 - 1) = 2 \times 10 = 20$
4. on choisit $e = 3$ (premier avec 20) comme exposant de chiffrement
5. l'exposant de déchiffrement est $d = 7$, l'inverse de 3 modulo 20 (en effet $e.d = 3 \times 7 \equiv 1 \bmod 20$).

La clé publique d'Alice est $(n, e) = (33, 3)$, et sa clé privée est $(n, d) = (33, 7)$. Bob transmet un message à Alice.

- Chiffrement de $M = 4$ par Bob avec la *clé publique* d'Alice : $4^3 \equiv 31 \bmod 33$, le chiffré est $C = 31$ que Bob transmet à Alice ;
- Déchiffrement de $C = 31$ par Alice avec sa *clé privée* : $31^7 \equiv 4 \bmod 33$, Alice retrouve le message initial $M = 4$.

Chiffrement asymétrique : DH

Algorithme clé dû à Whitfield Diffie and Martin Hellman (1976)

- Protocole utilisé pour établir une clé secrète partagée.

Un exemple simple (en pratique il faut de très grands nombres premiers) :

1. Alice et Bob ont choisi un nombre premier p et une base g (échangés en clair). Par exemple, $p = 23$ et $g = 5$.
2. Alice choisit un nombre secret a, par exemple $a = 6$.
   Bob choisit aussi un nombre secret b, par exemple $b = 15$.
3. Alice envoie à Bob la valeur $A = g^a \bmod p = 5^6 \bmod 23 = 8$
4. Bob envoie à Alice la valeur $B = g^b \bmod p = 5^{15} \bmod 23 = 19$
5. Alice peut maintenant calculer la clé secrète : $B^a \bmod p = 19^6 \bmod 23 = 2$
6. Bob fait de même et obtient la même clé qu'Alice : $A^b \bmod p = 8^{15} \bmod 23 = 2$

Recommendations de l'ANSSI
concernant la taille des clés[3]

Date	Symétrique	Factorisation Module	Logarithme discret Clef	Logarithme discret Groupe	Courbe elliptique GF(p)	Courbe elliptique GF(2^n)	Hash
2014 - 2020	100	2048	200	2048	200	200	200
2021 - 2030	128	2048	200	2048	256	256	256
> 2030	128	3072	200	3072	256	256	256

Les tailles de clef sont exprimées en bit. Ces résultats garantissent une sécurité minimale.
Cliquer sur une valeur pour la comparer avec les autres méthodes.

Remarques et algorithmes recommandés pour les systèmes symétriques :

La taille recommandée pour les systèmes symétriques est de 128 bits.
La taille minimale des blocs de chiffrement par bloc est de 64 bits (128 bits recommandés et obligatoires après 2020).
Il est recommandé d'employer des algorithmes par bloc et non des algorithmes de chiffrement par flot.
Algorithme de chiffrement : AES-CBC (FIPS 197)
Algorithme d'authentification et d'intégrité : CBC-MAC "retail" avec AES et 2 clefs distinctes.

Remarques et algorithmes recommandés pour les systèmes asymétriques :

Pour le chiffrement RSA, les exposants publics doivent être strictement supérieurs à 2^{16}=65536.
Les exposants secrets doivent être de la même taille que le module (3072 bits recommandés).
Les nombres premiers constituant le module RSA sont choisis aléatoirement uniformément et d'une taille égale à la moitié de celle du module.
Algorithme de chiffrement : RSAES-OAEP (PKCS#1)
Algorithme de signature : RSA-SSA-PSS (PKCS#1)
Algorithme de signature : ECDSA/ECKCDSA avec FRP256v1 ou P-256, P-384, P-521, B-283, B-409, B-571 dans FIPS 186-4
Courbes elliptiques GF(p) : FRP256v1 et P-256, P-384, P-521 dans FIPS 186-4
Courbes elliptiques GF(2^n) : B-283, B-409 et B-571 (FIPS 186-4)

Algorithme recommandé pour les fonctions de hachage : SHA-256 (FIPS 180-4)

3. https://www.keylength.com/fr/5/

Suites cryptographiques

Exemples d'algorithmes utilisés :

Echange de clés. RSA, Diffie-Hellman, ECDH, SRP, PSK.

Authentification. RSA, DSA, ECDSA.

Chiffrement symétrique. RC4, Triple DES, AES, IDEA, DES, ChaCha20 ou Camellia.

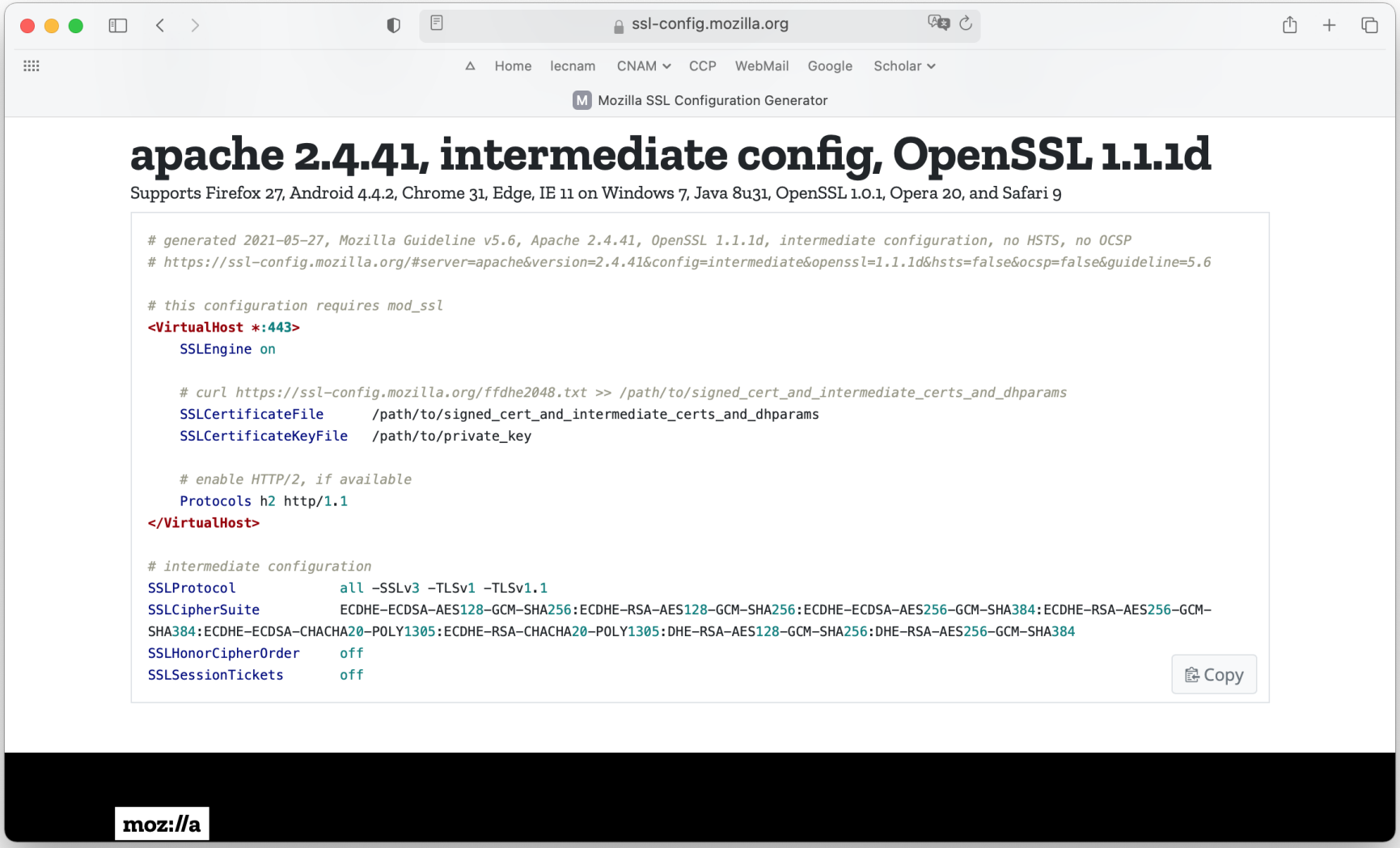
Authentification de message (code de contrôle).

– Pour TLS (< 1.3), un HMAC basé sur l'algorithme MD5 ou l'un des algorithmes de hachage SHA peut être utilisé.

– Pour SSL, SHA, MD5, MD4 et MD2 pouvaient être utilisés.

Pour TLS < 1.3, plus de 30 combinaisons acceptées !

Configuration de Apache (TLS 1.2)



Suites cryptographiques (TLS 1.3)

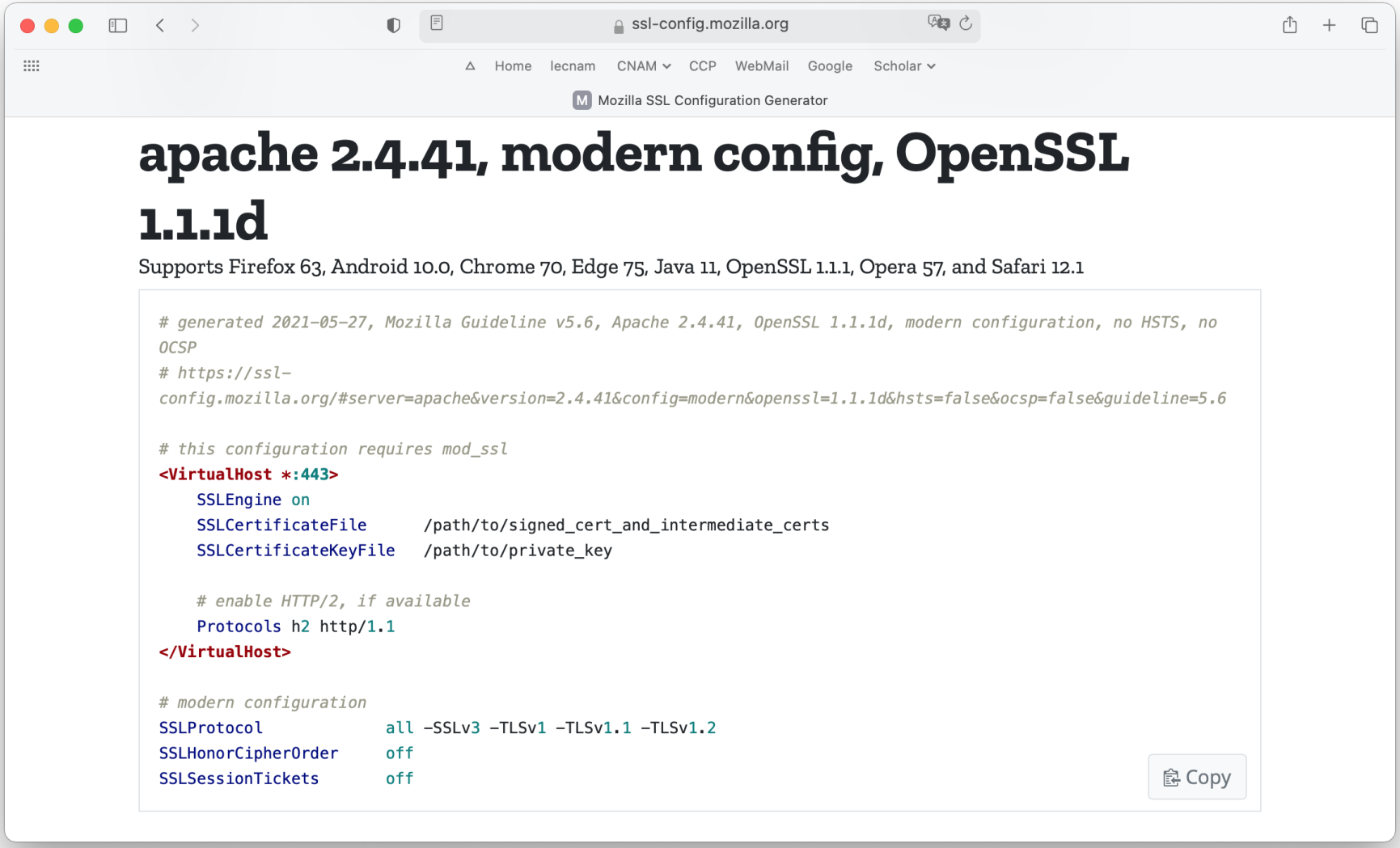
Les 4 suites recommandées[4] par l'ANSSI en 2020 :

Code TLS	Suite cryptographique
0x1302	TLS_AES_256_GCM_SHA384
0x1301	TLS_AES_128_GCM_SHA256
0x1304	TLS_AES_128_CCM_SHA256
0x1303	TLS_CHACHA20_POLY1305_SHA256

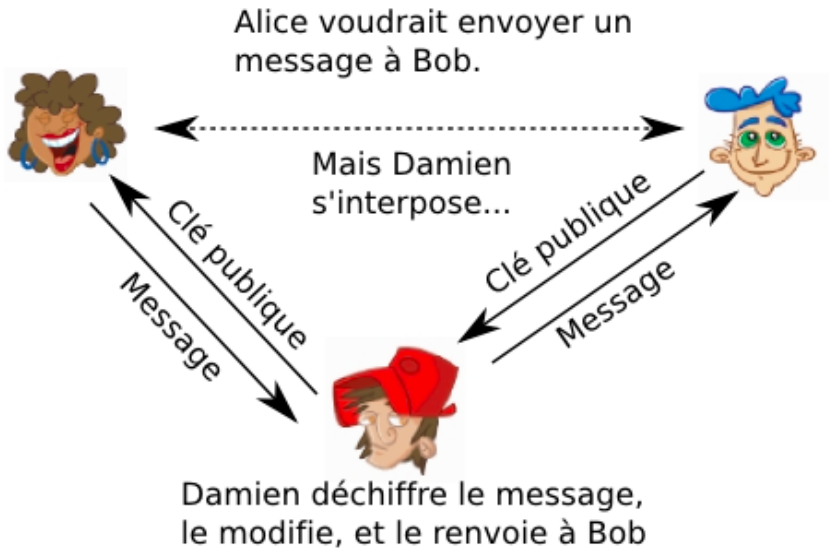
Table 2.1 – Suites TLS 1.3 recommandées

4. https://www.ssi.gouv.fr/guide/recommandations-de-securite-relatives-a-tls/

Configuration de Apache (TLS 1.3)



Attaque de l'homme du milieu[5]

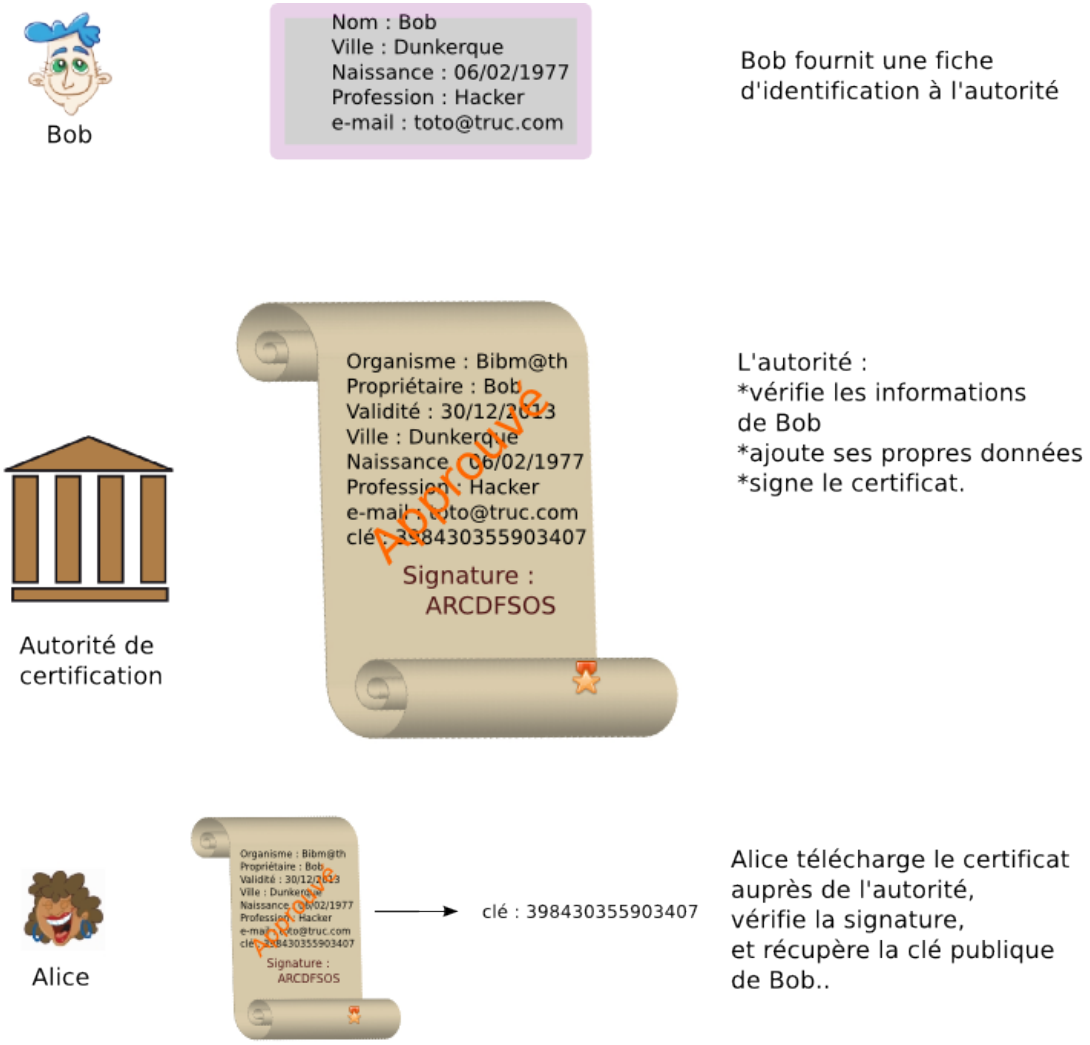


Comment être sur que l'émetteur est bien celui qu'il prétend être ?

- Notion de certificat
- Chaîne de confiance (Web of Trust)
- Autorité de certification (CA: Certificate Authority)

5. **http://www.bibmath.net/crypto/index.php?action=affiche&quoi=moderne/certificat**

Autorité de certification[6]



6. http://www.bibmath.net/crypto/index.php?action=affiche&quoi=moderne/certificat

Certificats

- Clé publique
- Informations spécifiques au certificat
- Signature par une CA

```
Certificate:
    Data:
        Version: 1 (0x0)
        Serial Number: 12103226332656409189 (0xa7f74c3952239a65)
    Signature Algorithm: sha1WithRSAEncryption
        Issuer: C=FR, ST=IdF, L=Paris, O=NFA083, OU=CA, CN=localhost
        Validity
            Not Before: Jun 12 14:19:55 2014 GMT
            Not After : Jul 12 14:19:55 2014 GMT
        Subject: C=FR, ST=IdF, L=Paris, O=NFA083, OU=TP, CN=localhost
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (1024 bit)
                Modulus:
                    00:ae:8d:ac:57:76:14:5b:c3:23:17:dd:c4:be:57:
                    c1:da:07:45:c5:aa:1f:64:c9:38:82:5f:23:6f:6f: [...]
                Exponent: 65537 (0x10001)
    Signature Algorithm: sha1WithRSAEncryption
         52:91:2e:a4:70:19:e8:8f:8d:db:5d:e4:5b:fd:79:82:12:31:
         af:f1:80:a1:86:ed:10:25:0d:e1:4b:df:ba:97:b5:4a:b8:6d: [...]
```

Autorité de Certification (CA)

- Doit être de confiance
- Signe les demandes de certificats
- Possède aussi une clé privée et un certificat
- Possibilité de certificat autosigné
  (mais avertissement du navigateur)

Plusieurs types de certificats

- Domain Validation (DV)
- Organization Validation (OV)
- Extended Validation (EV)

Seul la délivrance de certificats DV peut être automatisée.

Automatic Certificate Management Environment (ACME)[7]

> "Different types of certificates reflect different kinds of CA verification of information about the certificate subject. ‘‘Domain Validation’’ (DV) certificates are by far the most common type. For DV validation, the CA merely verifies that the requester has effective control of the web server and/or DNS server for the domain, but does not explicitly attempt to verify their real-world identity. (This is as opposed to ‘‘Organization Validation’’ (OV) and ‘‘Extended Validation’’ (EV) certificates, where the process is intended to also verify the real-world identity of the requester.)"

7. https://tools.ietf.org/id/draft-barnes-acme-03.txt

OpenSSL

- Implémentation des protocoles SSL et TLS
- Fournit les protocoles de chiffrement et authentification
- Permet de créer des clés publiques/privées, des certificats
- Permet de créer sa propre autorité de certification
- Possède un module Apache

Activation du module `ssl` d'Apache:

```
a2enmod ssl
```

Création d'un Certificat

- Génération d'une clé privé:

```
openssl genrsa 1024 > serveur.key

-----BEGIN RSA PRIVATE KEY-----
MIICXQIBAAKBgQC9FNacrVW8vwmvJWCyHEhpBzcoszCW
vMe4j8P27EW3ZHaTJLMCQQDo/cH6EQUS2z5GZlVFdtZS
bUwmbp4ByihybVwHhoipF6GnVJvfSgRSo16nHBezpNud
-----END RSA PRIVATE KEY-----
```

Création d'un Certificat

- Création de la demande de signature du certificat associé:

```
openssl req -new -key serveur.key > serveur.csr

Country Name (2 letter code) [AU]:FR
State or Province Name (full name) [Some-State]:IdF
Locality Name (eg, city) []:Paris
Organization Name (eg, company) []:NFA083
Common Name (e.g. server FQDN or YOUR name) []:localhost
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

Création d'un Certificat

- Envoie de la demande à une CA
- Signature du certificat par la CA

```
openssl x509 -req -in serveur.csr -out serveur.crt
```

```
Signature ok
subject=/C=FR/ST=IdF/L=Paris/O=NFA083/OU=TP/CN=localhost
Getting CA Private Key
Enter pass phrase for ca.key:
```

Configuration d'un Serveur Virtuel Sécurisé

```
<IfModule mod_ssl.c>
<VirtualHost _default_:443>
  ServerAdmin webmaster@localhost

  DocumentRoot /var/www
  <Directory />
    Options FollowSymLinks
    AllowOverride None
  </Directory>
  <Directory /var/www/>
    Options Indexes FollowSymLinks MultiViews
    AllowOverride None
    Order allow,deny
    allow from all
  </Directory>

  #   SSL Engine Switch:
  #   Enable/Disable SSL for this virtual host.
  SSLEngine on

  SSLCertificateFile    /etc/ssl/certs/ssl-cert.crt
  SSLCertificateKeyFile /etc/ssl/private/ssl-cert.key
</VirtualHost>
</IfModule>
```

Certificat non vérifiable

This Connection is Untrusted

You have asked Iceweasel to connect securely to **127.0.0.1:2443**, but we can't confirm that your connection is secure.

Normally, when you try to connect securely, sites will present trusted identification to prove that you are going to the right place. However, this site's identity can't be verified.

What Should I Do?

If you usually connect to this site without problems, this error could mean that someone is trying to impersonate the site, and you shouldn't continue.

Get me out of here!

Technical Details

127.0.0.1:2443 uses an invalid security certificate.

The certificate is not trusted because no issuer chain was provided.
The certificate is only valid for localhost

(Error code: sec_error_unknown_issuer)

I Understand the Risks

Certificat non valide

This Connection is Untrusted

You have asked Iceweasel to connect securely to **127.0.0.1:2443**, but we can't confirm that your connection is secure.

Normally, when you try to connect securely, sites will present trusted identification to prove that you are going to the right place. However, this site's identity can't be verified.

What Should I Do?

If you usually connect to this site without problems, this error could mean that someone is trying to impersonate the site, and you shouldn't continue.

Get me out of here!

Technical Details

127.0.0.1:2443 uses an invalid security certificate.

The certificate is only valid for localhost

(Error code: ssl_error_bad_cert_domain)

I Understand the Risks