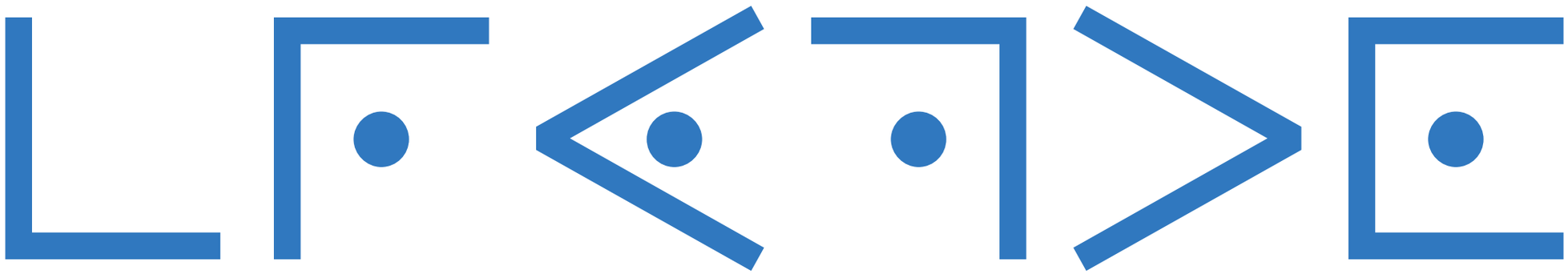


signatures et certificats

Nicolas Ollinger

M1 informatique — 2025/2026

Public-Key Cryptography Standards: PKCS

Yongge Wang, Ph.D., University of North Carolina at Charlotte

Outline: 1 Introduction, 2 PKCS #1: RSA Cryptography Standard, 3 PKCS #3: Diffie-Hellman Key Agreement Standard (Outdated), 4 PKCS #5: Password-Based Cryptography Standard, 5 PKCS #6: Extended-Certificate Syntax Standard (Historic), 6 PKCS #7 and RFC 3369: Cryptographic Message Syntax (CMS), 7 PKCS #8: Private-Key Information Syntax Standard, 8 PKCS #9: Selected Object Classes and Attribute Types, 9 PKCS #10: Certification Request Syntax Standard, 10 PKCS #11: Cryptographic Token Interface Standard, 11 PKCS #12: Personal Information Exchange Syntax Standard, 12 PKCS #15: Cryptographic Token Information Syntax Standard, 13 An Example.

Key works. ASN.1, public key cryptography, digital signature, encryption, key establishment scheme, public key certificate, cryptographic message syntax, cryptographic token interface (cryptoki).

Abstract

Cryptographic standards serve two important goals: making different implementations interoperable and avoiding various known pitfalls in commonly used schemes. This chapter discusses Public-Key Cryptography Standards (PKCS) which have significant impact on the use of public key cryptography in practice. PKCS standards are a set of standards, called PKCS #1 through #15. These standards cover RSA encryption, RSA signature, password-based encryption, cryptographic message syntax, private-key information syntax, selected object classes and attribute types, certification request syntax, cryptographic token interface, personal information exchange syntax, and cryptographic token information syntax. The PKCS standards are published by RSA Laboratories. Though RSA Laboratories solicits public opinions and advice for PKCS standards, RSA Laboratories retain sole decision-making authority on all aspects of PKCS standards. PKCS has been the basis for many other standards such as S/MIME.

Signature

Schémas de signature numérique

Repose sur la cryptographie à clé publique.

Permet **contrôle d'intégrité** et **authentification**.

La signature dépend à la fois du contenu du message et de l'identité du signataire.

Elle doit être non falsifiable et non répudiable.

Ça me rappelle les codes MAC

Pas besoin de secret partagé.

Pas besoin d’une clé par interlocuteur.

La preuve de validité est opposable à un tiers.

Mais les codes MAC sont généralement plus courts et se calculent beaucoup plus vite.

Finalement c’est juste l’inverse…

Signer c’est l’inverse de chiffrer, non ?

Pour signer m, je le chiffre avec ma clé privée, comme ça n’importe qui peut le déchiffrer avec ma clé publique pour le vérifier…

…bof, même s’il y a *un peu* de ça, en général cette manière de procéder ne produit pas des mécanismes de signature **sûrs**.

Formellement

Un schéma de signature est défini par 3 algorithmes PPT :

$\mathrm{Gen}(1^k) = (pk, sk)$ algo probabiliste de génération de clés

$\mathrm{Sign}_{sk}(m) = \sigma$ algo probabiliste de signature

$\mathrm{Vrfy}_{pk}(m, \sigma) = b$ algo déterministe de vérification

$$\forall (pk, sk) = Gen(1^n) \quad \forall m \quad \mathrm{Vrfy}_{pk}(m, \mathrm{Sign}_{sk}(m)) = 1$$

Sécurité avec Sig-forge$_{A,\Pi}(n)$

1. $(pk,sk) = \text{Gen}(1^n)$

2. L'adversaire dispose de pk et d'un oracle $\text{Sign}_{sk}()$. Il doit produire (m,σ). On note Q l'ensemble des requêtes faites à l'oracle.

3. L'adversaire gagne si $\text{Vrfy}_{pk}(m,\sigma)=1$ et $m \notin Q$.

Un schéma de signature est sûr si un adversaire PPT ne peut gagner qu'avec une probabilité négligeable devant n.

Hache et signe

$\mathrm{Gen}'(1^n) = ((pk, s), (sk, s))$ avec $(pk, sk) = \mathrm{Gen}(1^n)$ et $s = \mathrm{Gen}_H(1^n)$

$$\mathrm{Sign}'_{(sk,s)}(m) = \mathrm{Sign}_{sk}(H^s(m))$$

$$\mathrm{Vrfy}'_{(pk,s)}(m, \sigma) = \mathrm{Vrfy}_{pk}(H^s(m), \sigma)$$

Si Π est un schéma de signature sûr et que Π_H est résistant aux collisions alors Π’ est sûr.

Schéma RSA dit «Textbook»

Gen : sur l'entrée 1^n, générer deux nombres premiers de n bits p et q et deux paramètres d et e pour obtenir les clés (pq,e) et (pq,d).

Sign : étant donnés (N,d) et m calcule

$$\sigma = m^d \pmod{N}$$

Vrfy : étant donnés (N,e) et m tester

$$m \stackrel{?}{=} \sigma^e \pmod{N}$$

Sécurité de RSA Textbook

Cette façon d'utiliser RSA n'est pas sûre !

no message l'adversaire tire σ et calcule *m* avec la clé publique…

forger une signature pour *m* en faisant signer m_1 et m_2 puis en faisant le produit $\sigma=\sigma_1.\sigma_2$ des signatures !

RSA-FDH : sécurité grâce au hachage

On fixe une *bonne* fonction de hachage H à image uniforme dans $\mathbb{Z}_N^*$.

Gen : sur l'entrée 1^n, générer deux nombres premiers de *n* bits *p* et *q* et deux paramètres *d* et *e* pour obtenir les clés (*pq*,*e*) et (*pq*,*d*).

Sign : étant donnés (N,d) et m calcule

$$\sigma = H(m)^d \pmod{N}$$

Vrfy : étant donnés (N,e) et m tester

$$H(m) =^? \sigma^e \pmod{N}$$

FIPS PUB 186-4

FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION

Digital Signature Standard (DSS)

CATEGORY: COMPUTER SECURITY **SUBCATEGORY: CRYPTOGRAPHY**

Information Technology Laboratory
National Institute of Standards and Technology
Gaithersburg, MD 20899-8900

Issued July 2013

DSA (depuis 1991)

Schéma de signature autour de **DLP**.

DSA raccourci la taille des signatures par rapport à une application directe de ElGamal.

ECDSA est la version elliptique !

DSA

Soit G un groupe cyclique d'ordre premier q et g un générateur de G. H une fonction de hachage vers $\mathbb{Z}_q$, F code G dans $\mathbb{Z}_q$.

Alice choisit une clé privée $0 \leq x < q$ et transmet sa clé publique (G,q,g,y) où $y=g^x$.

Pour signer m, Alice choisit $0 \leq k < q$ et calcule $r=F(g^k)$ puis $s=k^{-1}(H(m)+xr) \bmod q$. *($r$ et $s \neq 0$)* La signature est la paire (r,s).

Pour vérifier $(m,(r,s))$, Bob teste

$$r =^{?} F\left(g^{H(m)s^{-1}} y^{rs^{-1}}\right)$$

Certificats

Certificat ?

(pk_A, sk_A) clés d'Alice

(pk_B, sk_B) clés de Bob

$\text{cert}_{A \to B} = \text{Sign}_{sk_A}(\text{"La clé de Bob est } pk_B\text{"})$

Si Charlie connaît la clé publique d'Alice mais pas celle de Bob, Bob peut s'authentifier auprès de Charlie en lui transmettant la paire (pk_B, $\text{cert}_{A \to B}$).

PKI permettant la distribution de clés publiques.

Autorité de certification (CA)

Entité en laquelle tous les acteurs ont confiance. Elle publie sa clé publique.

Elle certifie des clés en vérifiant l'identité qu'elle associe à la clé publique.

Le modèle se décline avec des CA multiples…

…et avec une notion de délégation entre CA : on obtient alors une chaîne de certification.

Gestion de la durée de validité et de la révocation des certificats.

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

X.509

(10/2016)

SERIES X: DATA NETWORKS, OPEN SYSTEM COMMUNICATIONS AND SECURITY

Directory

Information technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks

Certificats X509

```
$ echo | \
openssl s_client -connect cnrs.fr:443 | \
awk 'BEGIN{inside=0} /BEGIN CERT/
{inside=1} /END CERT/{inside=0; print $0}
{if (inside) print $0 }' | \
openssl x509 -text -noout
```

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            0a:9b:7f:0d:bd:1c:ca:dd:68:2e:14:aa:c3:ed:45:e8
    Signature Algorithm: sha256WithRSAEncryption
        Issuer: C=NL, ST=Noord-Holland, L=Amsterdam, O=TERENA, CN=TERENA SSL CA 3
        Validity
            Not Before: Feb  9 00:00:00 2018 GMT
            Not After : Feb 17 12:00:00 2021 GMT
        Subject: C=FR, L=Paris, O=Centre national de la recherche scientifique,
CN=www.cnrs.fr
    Subject Public Key Info:
        Public Key Algorithm: rsaEncryption
            Public-Key: (2048 bit)
            Modulus:
                00:d2:9d:f6:19:8a:38:20:9a:dd:7a:10:f9:24:8e:
                e6:5d:50:a1:c2:8c:31:38:86:87:6d:32:bd:7f:fe:
                (snip)
                e9:ab:ce:63:78:69:2a:cd:4d:52:a7:89:00:d9:c0:
                f2:44:b3:2c:84:e1:82:36:fd:8a:c6:9f:47:f1:43:
                c2:ed
            Exponent: 65537 (0x10001)
```

```
X509v3 extensions:
    X509v3 Authority Key Identifier:
        keyid:67:FD:88:20:14:27:98:C7:09:D2:25:19:BB:E9:51:11:63:75:50:62

    X509v3 Subject Key Identifier:
        AB:0B:50:0C:FD:E8:86:86:96:60:73:FB:42:3E:D3:23:C9:73:90:5E
    X509v3 Subject Alternative Name:
        DNS:www.cnrs.fr, DNS:cnrs.fr
    X509v3 Key Usage: critical
        Digital Signature, Key Encipherment
    X509v3 Extended Key Usage:
        TLS Web Server Authentication, TLS Web Client Authentication
    X509v3 CRL Distribution Points:

        Full Name:
          URI:http://crl3.digicert.com/TERENASSLCA3.crl

        Full Name:
          URI:http://crl4.digicert.com/TERENASSLCA3.crl

X509v3 Certificate Policies:
    Policy: 2.16.840.1.114412.1.1
      CPS: https://www.digicert.com/CPS
    Policy: 2.23.140.1.2.2
(snip)
```

```
Signature Algorithm: sha256WithRSAEncryption
     88:2f:24:46:7a:9f:4e:dd:12:8c:c7:fc:aa:d0:ba:17:76:a5:
     3a:9b:ae:61:31:94:e5:a2:7a:63:ec:03:58:ba:b4:aa:d6:52:
     3c:e4:aa:02:d1:5f:02:4f:fd:b0:5c:b6:4f:1c:b8:d3:83:2b:
     83:6b:12:a4:97:ac:39:d8:ac:10:ae:1d:fe:a8:03:43:e4:61:
     a1:b7:33:44:14:66:fd:a0:41:f1:0a:9a:4b:29:73:79:b8:7f:
     18:ed:da:e4:d0:00:23:8d:67:af:ad:28:b9:e8:82:1b:be:51:
     ce:c1:53:38:58:a4:fb:fc:10:5e:9b:96:85:b0:a4:21:7e:08:
     15:ea:3c:03:74:72:00:a8:ca:21:c7:83:59:9b:cc:a6:f6:a1:
     18:0e:96:53:1d:73:16:55:91:6f:03:78:59:38:9b:34:0d:e2:
     43:5d:74:39:81:cf:08:39:82:e6:e6:eb:7c:c7:92:e7:ad:2b:
     40:81:45:99:c8:9e:55:20:3d:6b:1e:21:98:b9:13:1d:ca:de:
     27:44:9c:6e:6e:0d:aa:54:5f:bc:6f:dd:9b:0c:e5:a9:3c:a2:
     35:98:11:bb:ee:34:ef:ec:40:55:be:1f:13:c6:90:d6:fb:39:
     1a:17:1c:50:53:81:5e:aa:e1:52:3c:1f:10:3a:d5:a3:6d:e5:
     e6:b5:55:02
```

(5) SSL/TLS

RFC 5246

De SSL à TLS

1994 **SSL 1.0**, Netscape, jamais diffusé

1995 **SSL 2.0**, Netscape, **gros problèmes de sécurité**

1996 **SSL 3.0**, Netscape, mieux… mais **pas en 2014 :-(**

1996 **TLS 1.0**, IETF (= SSL 3.1), **RFC 2246** **Do Not Use TLSv1.0**

2006 **TLS 1.1**, IETF, **RFC 4346** **Do Not Use TLSv1.1**

2008 **TLS 1.2**, IETF, **RFC 5246**

2018 **TLS 1.3**, IETF, **RFC 8446**

Internet Engineering Task Force (IETF)
Request for Comments: 7525
BCP: 195
Category: Best Current Practice
ISSN: 2070-1721

Y. Sheffer
Intuit
R. Holz
NICTA
P. Saint-Andre
&yet

May 2015

Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)

Abstract

Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS) are widely used to protect data exchanged over application protocols such as HTTP, SMTP, IMAP, POP, SIP, and XMPP. Over the last few years, several serious attacks on TLS have emerged, including attacks on its most commonly used cipher suites and their modes of operation. This document provides recommendations for improving the security of deployed services that use TLS and DTLS. The recommendations are applicable to the majority of use cases.

Transport Layer Security

The primary goal of the TLS protocol is to provide privacy and data integrity between two communicating applications. The protocol is composed of two layers: the TLS Record Protocol and the TLS Handshake Protocol. At the lowest level, layered on top of some reliable transport protocol (e.g., TCP), is the TLS Record Protocol.
The TLS Record Protocol provides connection security that has two basic properties:

- **The connection is private**. Symmetric cryptography is used for data encryption (e.g., AES, RC4, etc.). The keys for this symmetric encryption are generated uniquely for each connection and are based on a secret negotiated by another protocol (such as the TLS Handshake Protocol). The Record Protocol can also be used without encryption.

- **The connection is reliable**. Message transport includes a message integrity check using a keyed MAC. Secure hash functions (e.g., SHA-1, etc.) are used for MAC computations. The Record Protocol can operate without a MAC, but is generally only used in this mode while another protocol is using the Record Protocol as a transport for negotiating security parameters.

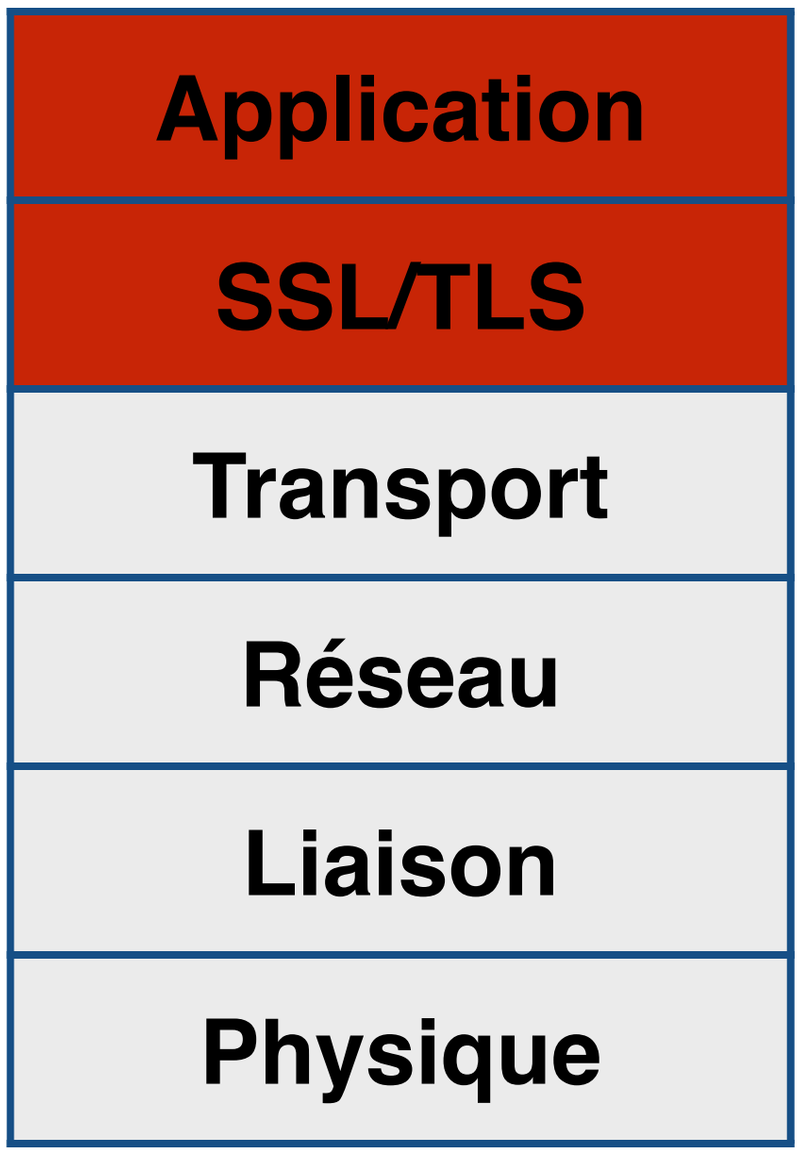
The TLS Record Protocol is used for encapsulation of various higher-level protocols. One such encapsulated protocol, the TLS Handshake Protocol, allows the server and client to authenticate each other and to negotiate an encryption algorithm and cryptographic keys before the application protocol transmits or receives its first byte of data. The TLS Handshake Protocol provides connection security that has three basic properties:

- **The peer's identity can be authenticated** using asymmetric, or public key, cryptography (e.g., RSA, DSA, etc.). This authentication can be made optional, but is generally required for at least one of the peers.

- **The negotiation of a shared secret is secure**: the negotiated secret is unavailable to eavesdroppers, and for any authenticated connection the secret cannot be obtained, even by an attacker who can place himself in the middle of the connection.

- **The negotiation is reliable**: no attacker can modify the negotiation communication without being detected by the parties to the communication.

SSL/TLS



HTTP, SMTP, IMAP, ...

surcouche à TCP

session **TCP**

Mise en œuvre

- Version **old school** : encapsuler un service existant dans un tunnel SSL/TLS sur un autre port :
  - ➡ HTTP (80), IMAP (143), POP3 (110), ...
  - ➡ HTTPS (443), IMAPS (993), POP3S (995), ...

- Version **moderne** : étendre le protocole avec un message STARTLS pour basculer sous TLS : SMTP, FTP, IMAP, POP3, XMPP, LDAP, NNTP, *etc*.

```
220 smtp.terrier.net ESMTP Postfix (Debian/GNU)
EHLO jeannot.lapin.org
250-smtp.terrier.net
250-PIPELINING
250-SIZE 51200000
250-VRFY
250-ETRN
250-STARTTLS
250-ENHANCEDSTATUSCODES
250-8BITMIME
250 DSN
STARTTLS
220 2.0.0 Ready to start TLS
```

En pratique

- Bibliothèques : **`Openssl`**, **`GnuTLS`**, **`JSSE`** (`Java`), **`Schannel`** (`Windows`), `...`

- Encapsulation applicative avec **`stunnel`**.

Cipher suites

TLS_NULL_WITH_NULL_NULL

TLS_RSA_WITH_NULL_MD5
TLS_RSA_WITH_NULL_SHA
TLS_RSA_WITH_NULL_SHA256

TLS_RSA_WITH_3DES_EDE_CBC_SHA
TLS_RSA_WITH_AES_128_CBC_SHA
TLS_RSA_WITH_AES_256_CBC_SHA
TLS_RSA_WITH_AES_128_CBC_SHA256
TLS_RSA_WITH_AES_256_CBC_SHA256

TLS_DH_DSS_WITH_3DES_EDE_CBC_SHA
TLS_DH_RSA_WITH_3DES_EDE_CBC_SHA
TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA
TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA
TLS_DH_DSS_WITH_AES_128_CBC_SHA
TLS_DH_RSA_WITH_AES_128_CBC_SHA
TLS_DHE_DSS_WITH_AES_128_CBC_SHA
TLS_DHE_RSA_WITH_AES_128_CBC_SHA
TLS_DH_DSS_WITH_AES_256_CBC_SHA
TLS_DH_RSA_WITH_AES_256_CBC_SHA
TLS_DHE_DSS_WITH_AES_256_CBC_SHA
TLS_DHE_RSA_WITH_AES_256_CBC_SHA
TLS_DH_DSS_WITH_AES_128_CBC_SHA256
TLS_DH_RSA_WITH_AES_128_CBC_SHA256
TLS_DHE_DSS_WITH_AES_128_CBC_SHA256
TLS_DHE_RSA_WITH_AES_128_CBC_SHA256
TLS_DH_DSS_WITH_AES_256_CBC_SHA256
TLS_DH_RSA_WITH_AES_256_CBC_SHA256
TLS_DHE_DSS_WITH_AES_256_CBC_SHA256
TLS_DHE_RSA_WITH_AES_256_CBC_SHA256

TLS_DH_anon_WITH_3DES_EDE_CBC_SHA
TLS_DH_anon_WITH_AES_128_CBC_SHA
TLS_DH_anon_WITH_AES_256_CBC_SHA
TLS_DH_anon_WITH_AES_128_CBC_SHA256
TLS_DH_anon_WITH_AES_256_CBC_SHA256

TLS Record protocol

- Protocole de transport pour tous les messages TLS :
  - ➡ Change Cipher (20) ;
  - ➡ Alert (21) ;
  - ➡ Handshake protocol (22) ;
  - ➡ Application Data (23).

- Succession de **`Record`**, séquences d'au plus 2^14 octets éventuellement compressés avec MAC et chiffrés, avec entête :
  - ➡ Content type (20 / 21 / 22 / 23);
  - ➡ Version du protocole (TLS 1.2 = (3,3), TLS 1.0 = (3,1), ...);
  - ➡ Longueur de la charge utile.

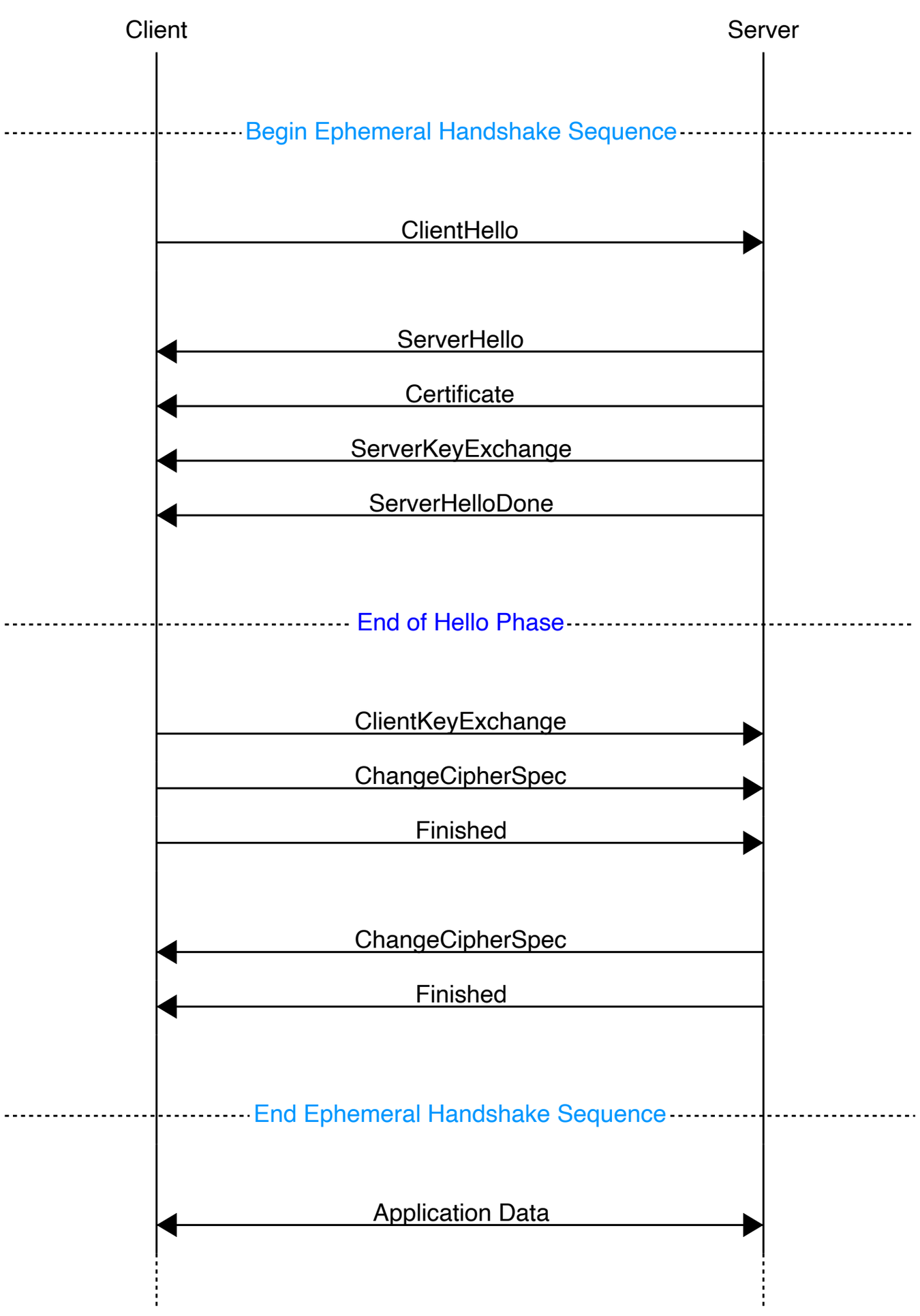
TLS Handshake protocol

- **Négociation** des paramètres de sécurité de la session et **identification** du client et/ou du serveur.

- Paramètres de session :
  - ➡ session identifier ;
  - ➡ certificat X509 du pair (peut être `null`) ;
  - ➡ méthode de compression ;
  - ➡ algorithmes crypto (PRF + cipher + MAC) ;
  - ➡ clé secrète de session (48 octets).

- Différentes versions du protocole existent selon les choix cryptographique (anonyme ou non, PFS ou non, *etc*)

Handshake Ephemeral Diffie-Hellman avec certificat serveur



ClientHello / ServerHello

- ➡ version du protocole
- ➡ GMT unix time
- ➡ random (28 octets de qualité cryptographique)
- ➡ session id
- ➡ liste algo crypto (`TLS_RSA_WITH_RC4_128_MD5`, `TLS_RSA_WITH_AES_256_CBC_SHA`, `...`)
- ➡ liste méthodes de compression (`null`, `zlib`, `...`)

Certificate

- Contient une suite de certifications X509v3, certificat du serveur en tête, certificats CA derrière.

- Codé au format ASN.1

Server/ClientKeyExchange

- Mise en œuvre du protocole d'échange de clé secrète de session à travers un canal non sûr.

- version RSA ou version Diffie-Hellman.

ServerHelloDone
ChangeCipher
Finished

- `ServerHelloDone` : le serveur a terminé de parler, le client doit **vérifier le certificat** du serveur.

- `ChangeCipher` : indique que les prochains messages utiliseront les nouveau paramètres de sécurité.

- `Finished` : message de **vérification** chiffré.

Application Data / Alert

- `Application Data` : Données brutes (encapsulées dans un `Record`).

- `Alert` : Message d'erreur ou d'avertissement de terminaison de connexion.

DTLS

RFC 6347

Datagram Transport Layer Security

The basic design philosophy of DTLS is to construct "TLS over datagram transport". The reason that TLS cannot be used directly in datagram environments is simply that packets may be lost or reordered. TLS has no internal facilities to handle this kind of unreliability; therefore, TLS implementations break when rehosted on datagram transport. The purpose of DTLS is to make only the minimal changes to TLS required to fix this problem. To the greatest extent possible, DTLS is identical to TLS. Whenever we need to invent new mechanisms, we attempt to do so in such a way that preserves the style of TLS.

(6) SSH

RFC 4251

Secure Shell

1995 **SSH 1.x**, Tatu Ylönen, HUT, Finlande,
logiciel libre puis propriétaire

1999 **OpenSSH**, OpenBSD, libre et portable

2006 **SSH-2**, IETF, protocole normalisé

Mises en œuvre : `OpenSSH, lsh, Dropbear, ...`

Sous Debian GNU/Linux

- Serveur SSH avec **`openssh-server`**, à configurer dans `/etc/ssh/sshd_config`.

- Client SSH avec **`openssh-client`**, à configurer dans `/etc/ssh/ssh_config` et `~/.ssh/config`.

```
bob:~# /etc/init.d/ssh start
```

```
alice:~# ssh guest@bob
The authenticity of host 'bob (10.0.0.2)' can't be established.
RSA key fingerprint is 53:d4:13:1a:f0:a3:e9:43:6b:04:6c:d0:bc:63:10:29.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added ‘bob’ (RSA) to the list of known hosts.
guest@bob's password:
Last login: Sat Mar 28 13:37:00 2015 from alice
guest@bob:~$
```

Protocole

Secure Shell (SSH) is a protocol for secure remote login and other secure network services over an insecure network. It consists of three major components:

- **The Transport Layer Protocol** RFC 4253 provides **server authentication**, confidentiality, and integrity. It may optionally also provide compression. The transport layer will typically be run over a TCP/IP connection, but might also be used on top of any other reliable data stream.

- **The User Authentication Protocol** RFC 4252 **authenticates the client-side user** to the server. It runs over the transport layer protocol.

- **The Connection Protocol** RFC 4254 multiplexes the encrypted tunnel into several logical channels. It runs over the user authentication protocol.

Authentification serveur

- **Côté serveur :** Paires de clés publiques/privées (RSA ou DSA) dans `/etc/ssh/ssh_host_*_key*`

- **Côté client :** Liste des clés publiques connues dans `~/.ssh/known_hosts`

- Empreinte (fingerprint) accessible grâce à

```
$ ssh-keygen -l -f /etc/ssh/ssh_host_rsa_key.pub
2048 53:d4:13:1a(…)
```

Authentification utilisateur

- Par la saisie d'un **mot de passe**.
- Ou mieux, grâce à la cryptographie asymétrique : utilisation d'une **clé publique** côté serveur.
- Clés générées avec `ssh-keygen` côté client.
- Clé publique dans `~/.ssh/authorized_keys` sur le compte utilisateur côté serveur.

```
alice:~# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
08:02:38:c9:2d:0c:67:70:3e:06:89:84:f4:dc:ca:85 root@alice
The key's randomart image is:
+--[ RSA 2048]----+
|#*=              |
|B@o.o            |
| .*E.o           |
| ..oo. .         |
|   o  . S        |
|                 |
|                 |
|                 |
|                 |
+-----------------+
```

```
alice:~# ssh-agent bash
alice:~# ssh-add
Enter passphrase for /root/.ssh/id_rsa:
Identity added: /root/.ssh/id_rsa (/root/.ssh/id_rsa)

(…)

alice:~# ssh guest@bob
Last login: Sat Mar 28 13:37:00 2015 from alice
guest@bob:~$

(…)
```

Transfert de fichiers

- **`scp`** permet de transférer des fichiers à travers une connexion SSH sur le principe de cp.

```
$ scp guest@bob:/etc/motd /tmp/
```

- **`sftp`** permet de transférer des fichiers à travers une connexion SSH sur le principe de ftp.

```
$ sftp guest@bob
sftp> help
```

- **`rsync`** est un logiciel de synchronisation de répertoires à travers une connexion SSH.

Redirection de ports TCP

- Joindre un service à travers une connexion SSH :
  `$ ssh -L 5555:serveur:143 passerelle`

- Partager un service à travers une connexion SSH :
  `$ ssh -R 5555:serveur:143 passerelle`

- Créer un proxy SOCKS à travers SSH :
  `$ ssh -D localhost:6666 passerelle`

- Créer un tunnel/VPN à travers SSH :
  `$ ssh -w 0:1 passerelle`

Pour quelques € de plus

- Possibilité de créer des tunnels à la volée pour une connexion existante grâce à **`~C`** (pour l'aide **`~?`**).

- Configuration avancée dans `~/.ssh/config` avec `Host`, `User` et `ProxyJump`.

(7) S/MIME

RFC 5751

RFC 822

From: Nicolas Ollinger <nicolas.ollinger@ens-lyon.org>
Subject: Message basique
Date: Mon, 30 Mar 2015 13:37:42 +0200
To: Nicolas Ollinger <nicolas.ollinger@univ-orleans.fr>

Bonjour,

Essai, 1, 2, 3 !

Bonne nuit,
N.
--

MIME

RFC 2045

```
From: Nicolas Ollinger <nicolas.ollinger@ens-lyon.org>
Content-Type: multipart/mixed; boundary="blop"
Subject: =?iso-8859-1?b?QXbp?= l'accent !
Date: Mon, 30 Mar 2015 11:39:58 +0200
To: Nicolas Ollinger <nicolas.ollinger@univ-orleans.fr>
MIME-Version: 1.0

This message is in MIME format.

--blop
Content-Type: text/plain; charset=ISO-8859-1; format="flowed"
Content-Disposition: inline
Content-Transfer-Encoding: quoted-printable

Bonjour,

Un message avec une pi=E8ce-jointe.

Amiti=E9s,
N.
--=20
```

--blop
Content-Type: image/png; name="couak.png"
Content-Disposition: attachment; filename="couak.png"
Content-Transfer-Encoding: base64

iVBORw0KGgoAAAANSUhEUgAAABMAAAAUCAAAAABKPlmoAAABFklEQVQY02WQvS9DYRSHn/tWGwYR
TK9B0mtoRMSgk/Em0n9ARKM1YWJQg8Vg9S+IySgxGDQxsPhsTSIhkZBqxLcguP2gPYa3t5c40zlP
8ss557GEf6W8Jr0UPfR6ERGRUlprHTkwg2WyJzGAjlXbz25PAHAzV6mxgqwl8v3DgcBQdG/y8xoQ
Oe7W2inJyrKUHd3Z9y0KHl4hHiKeJDjG11MeBY+N0FUZtVTCjUD1EgUbRfh42313dwousI6CGSDb
ls2dZ9q3oHURRG5trcOZ+0Hn7iistZ4WS3jpAagdT+8mCpqnADwXsTIKGmZH6kqCqVTIRHIDdXjR
ZP69GvfdJZ8Nmz/12f6CWWcXf0luOcOq/vVuIfwAEpFmNswwQrAAAAAASUVORK5CYII=

--blop--

S/MIME

RFC 5751

- **Secure**/MIME (Multipurpose Internet Mail Extension)
- Ajoute la possibilité de **signer** et/ou **chiffrer** les mails

```
▸ Content-Type: multipart/signed
▸ Content-Type: application/pkcs7-signature
▸ Content-Type: application/pkcs7-mime
    ➡ smime-type=enveloped-data
    ➡ smime-type=signed-data
    ➡ smime-type=certs-only
    ➡ …
```

```
From: Nicolas Ollinger <nicolas.ollinger@univ-orleans.fr>
Content-Type: multipart/signed; boundary="blop";
    protocol="application/pkcs7-signature"; micalg=sha1
Subject: =?iso-8859-1?Q?sign=E9?=
Date: Mon, 30 Mar 2015 12:09:36 +0200
To: Nicolas Ollinger <nicolas.ollinger@ens-lyon.org>
MIME-Version: 1.0

--blop
Content-Transfer-Encoding: quoted-printable
Content-Type: text/plain; charset=iso-8859-1

Cher coll=E8gue,

C'est sign=E9,
N.
--=20

--blop
Content-Disposition: attachment; filename=smime.p7s
Content-Type: application/pkcs7-signature; name=smime.p7s
Content-Transfer-Encoding: base64

MIAGCSqGSIb3DQEHAqCAMIACAQExCzAJBgUrDgMCGgUAMIAGCSqGSIb3DQEHAQAAoIIG9DCCBvAw
ggTYoAMCAQICAQQwDQYJKoZIhvcNAQEFBQAwdjELMAkGA1UEBhMCRlIxDzANBgNVBAgTBkxvaXJl
(…)
zdbfuQf6EG5zXKx8EIjpbcdMZ0XjhmBjkRAznmKq0R1XsOptyhTA2GL+ccnvz5iwEjOtG491slRH
dGFVSkMRLYpeFHFGz3/+sXb+p6WUtotFrl5+EMAmFE2bfgiCchqsuHIAAAAAAAA=
--blop--
```

From: Nicolas Ollinger <nicolas.ollinger@univ-orleans.fr>
Content-Type: **application/pkcs7-mime**; name=smime.p7m; **smime-type=enveloped-data**
Content-Transfer-Encoding: base64
Subject: =?iso-8859-1?Q?chiffr=E9?=
Date: Mon, 30 Mar 2015 13:37:42 +0200
To: Nicolas Ollinger <nicolas.ollinger@univ-orleans.fr>
MIME-Version: 1.0
Content-Disposition: attachment; filename=smime.p7m

MIAGCSqGSIb3DQEHA6CAMIACAQAxggKXMIICkwIBADB7MHYxCzAJBgNVBAYTAkZSMQ8wDQYDVQQI
EwZMb2lyZXQxEDAOBgNVBAcTB09ybGVhbnMxETAPBgNVBAoTCE9sbGluZ2VyMRMwEQYDVQQDEwpS
T09ULW5vcGlkMRwwGgYJKoZIhvcNAQkBFg1ub3BpZEBmcmVlLmZyAgEEMA0GCSqGSIb3DQEBAQUA
BIICAF+MU0wSAMVXvz7grZ4dEMDpUGRpIzyuWjnb5e8FPJGVNrQuiMkQ1dEFl1KfJF0KqnKvztCk
VRpPRJTpLP6l8mGDkEo1YjGKwQO4wyeeX3dh/y80AI+4vctx2JUlcYUuLMpwIEF37RA89O8GkEPp
PZfPiGLUJnx+R/fZGkAvIYLQw3iD6/aP7CYNwYvCS6dMyjsmttMtF9ZF3ncjrNcbKqnVyW/1iAWF
5lfsNN759ORknngOTGIyMQnK3ULoRkwT1gC3SyfWeKiCvwpyAy9SQqx2qCROjn4Cn3t9Dhv8xSZ1
OGm22+4Rl92Sw0HlFXF5KZHFa/eo5/Etx3TdwKj3U99O0b0H+TZaoj0y8y5124qbR/ATXoZEl2on
M3IUZYeYnk4cS6IjhPOM5mdE4ZFr6ITSg0zBDAoqhy8LRuuD4q4Rw7y4xFoPj5sWgYUfJ7pwzBbh
uqZGTu8C2Kck6JQmxBcP+srY2wl8FP9YsZo6zcv/Ce1i2I0zptcImzcAnR1i5Ry1e1dmyo1UXX6+
Yr5vGHHWSQUYxiTqwNF7AIpFoPlTSUv+yxJe4vAkKnWhsH+9ZxG//1QGSnT4GEDxotYqMf0O0ODk
YjYcbeRiKj2PHkmjTkpNRHwDIESxFcxITmyXX1JIZ/qi+hbQChDVEXcKAK+h2+vvTgMbSMbYlkdD
ON2cMIAGCSqGSIb3DQEHATAUBggqhkiG9w0DBwQIMAPhQpSxUgqggAQoNO5mORBNb8e/ZnL4ewwB
ryo9p/51iR8f6d9TbxXdBNTNfX4kMYmhsQQY/Ax/XYLLFmNkmrFa6C+xmQjNu//LJuOxBAgZeSqJ
YKCK7gQQwRHahYcRfIoqC537jLsjCgQQGvwcDOC4NPjV17o0Ruj+hQQIYh0nEzzU5dcEEO3KBy5S
am1KeNBhDIDOukYECLCJNOsEpPqoBAirjRoZG9D5fgQIzzyTQWMzBrUAAAAAAAAAAAAA

CMS

RFC 5652

This document describes the Cryptographic Message Syntax (CMS). This syntax is used to digitally sign, digest, authenticate, or encrypt arbitrary message content.

The CMS describes an encapsulation syntax for data protection. It supports digital signatures and encryption. The syntax allows multiple encapsulations; one encapsulation envelope can be nested inside another. Likewise, one party can digitally sign some previously encapsulated data. It also allows arbitrary attributes, such as signing time, to be signed along with the message content, and it provides for other attributes such as countersignatures to be associated with a signature.

Algorithmes

- **Empreinte :** SHA-256, SHA-1
- **Signature :** RSA+SHA, DSA+SHA
- **Clé de session :** RSA, DH
- **Chiffrement de session :** 3DES, AES, …
- **MAC :** HMAC+SHA-1

En pratique

- Certificats **X509v3** avec CN = **adresse mail**.

- Clients de mail compatibles.

- Parfums :
  - ➡ message signé ;
  - ➡ message chiffré ;
  - ➡ message signé + chiffré ;
  - ➡ message en clair + signature.

- Attention à ne pas perdre sa clé privée ! (archivage)

(8) OpenPGP

RFC 4880

Pretty **G**ood **P**rivacy

- Crée par Ph. R. Zimmerman en 1991
- Un outil populaire pour signer et chiffrer des fichiers, des mails, *etc*
- Normalisé par l'IETF à la fin des années 90 sous le nom **OpenPGP**.
- Plusieurs implémentations dont GnuPG

Algorithmes

- **Clés publiques :** RSA, Elgamal, DSA, DH, …
- **Empreintes :** MD5, SHA-1, SHA-256, …
- **Signature :** RSA + SHA-1
- **Chiffrement de session :** IDEA, CAST5, AES, …

```
From: Nicolas Ollinger <nopid@free.fr>
Content-Type: multipart/signed; boundary="blop";
    protocol="application/pgp-signature"; micalg=pgp-sha512
Subject: =?iso-8859-1?Q?sign=E9?=
Date: Mon, 30 Mar 2015 12:53:15 +0200
To: Nicolas Ollinger <nicolas.ollinger@univ-orleans.fr>
MIME-Version: 1.0

--blop
Content-Transfer-Encoding: quoted-printable
Content-Type: text/plain; charset=iso-8859-1

Cher coll=E8gue,

C'est sign=E9 avec OpenPGP.

Amiti=E9s,
N.
--=20

--blop
Content-Transfer-Encoding: 7bit
Content-Disposition: attachment; filename=signature.asc
Content-Type: application/pgp-signature; name=signature.asc

-----BEGIN PGP SIGNATURE-----
Comment: GPGTools - https://gpgtools.org

iQEcBAEBCgAGBQJVGbe7AAoJEM5y2uR1++lOEvQIAI/yz6NKoNqBRf3FUTZgJZYw
(…)
675kJ6g+3gqqojQYdWjzsd5PnxssiuKwSYQUJKUJSyqODO4mmsNVYi0W8hdufCw=
=BSI+
-----END PGP SIGNATURE-----
```

```
From: Nicolas Ollinger <nicolas.ollinger@ens-lyon.org>
Date: Mon, 30 Mar 2015 12:54:51 +0200
Content-Type: multipart/encrypted; boundary="blop";
    protocol="application/pgp-encrypted";
Subject: =?iso-8859-1?Q?chiffr=E9?=
MIME-Version: 1.0
Content-Transfer-Encoding: 7bit
To: Nicolas Ollinger <nicolas.ollinger@univ-orleans.fr>

This is an OpenPGP/MIME encrypted message (RFC 2440 and 3156)
--blop
Content-Transfer-Encoding: 7bit
Content-Type: application/pgp-encrypted

Version: 1

--blop
Content-Transfer-Encoding: 7bit
Content-Disposition: inline; filename=encrypted.asc
Content-Type: application/octet-stream; name=encrypted.asc

-----BEGIN PGP MESSAGE-----
hQEMA70atxXC53EgAQf/b2AWKV5t+RcRwVC0QDKAkyjiWlQZtcOXJJyoCPqBgRv9
(…)
MQ7NXRND/MlCVtKo5Uu43d3mYGkHgXZsjtTmypwsanigSV/7TaylxE3bIea4NZUZ
iWLx
=SUDL
-----END PGP MESSAGE-----

--blop--
```

Web of trust

- Au lieu de certificats et de PKI à la X509, utilisation de **certificats auto-signés** signés par zéro, un ou plusieurs autres utilisateurs.

- Notion de **niveau de confiance** envers un autre utilisateur du système.

- Mécanisme de certification décentralisé.

- Mécanisme de révocation de certificat.

En pratique

- Utilitaire gpg
- Fichiers stockés dans `~/.gnupg`
- Serveurs de clés pour distribuer plus vite les clés

```
$ gpg --gen-key

(…)

 Empreinte de la clef = E9CF 8B27 C861 B66B 1461  BDF6 2B4A 4737 8B61 268F
uid                  Nicolas Ollinger <nicolas.ollinger@univ-orleans.fr>
sub   4096R/9B50E996 2015-03-30

$ gpg --send-key 9B50E996
gpg: envoi de la clef 8B61268F au serveur hkp keys.gnupg.net

$ gpg --fingerprint nicolas.ollinger@univ-orleans.fr
pub   4096R/8B61268F 2015-03-30
 Empreinte de la clef = E9CF 8B27 C861 B66B 1461  BDF6 2B4A 4737 8B61 268F
uid                  Nicolas Ollinger <nicolas.ollinger@univ-orleans.fr>
sub   4096R/9B50E996 2015-03-30

$ gpg –edit-key jeannot.lapin@terrier.net
(…)
```

```
$ gpg -u nicolas.ollinger@univ-orleans.fr --clearsign lapin.txt

Une phrase de passe est nécessaire pour déverrouiller la clef secrète de
l'utilisateur : « Nicolas Ollinger <nicolas.ollinger@univ-orleans.fr> »
clef RSA de 4096 bits, identifiant 8B61268F, créée le 2015-03-30

$ cat lapin.txt.asc
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA512

Un petit lapin qui mange du foin.
-----BEGIN PGP SIGNATURE-----
Version: GnuPG/MacGPG2 v2.0.22 (Darwin)
Comment: GPGTools - https://gpgtools.org
```


```
iQIcBAEBCgAGBQJVGcO9AAoJECtKRzeLYSaPRD0QAK5szWx7wa4Lu2a8ze4oXTY2
qOraDTj3XhilXNfwJEG3DXiN/KkC5+Mm0/n8RliXnjo/3DERvaLoRYbMD0DQUoLd
BpQbfaC0173bLhELSAu2Gm7d0JKDzaGSMMoSj1bTPMI90jGdL9itBX3vlBZXWbCS
FSPvYswnGXJ7RetP2oQQ9pffPBIIMHb6dfSNtbWfMO0OFSFtuoMKT04ByC5PcIup
0KkqAyrwteciBtB4yve3cLxJG30CWcKFNwAiGvI1W4121aUT2aFMl8+ePH/7eNY5
g6BFpTJll0sYIs/6rhOQTOm8mA4U8FITqCW58BWKHdweFRZ1UuonJ/6IOa5T/87w
Nf3W3cyOURB3tEp5KewM2bgoXQTQzcoIYiUOSwPWlE3Whl5Ej/C2lSCbFH4J3dxE
W0mAbJB9VhDqn0F1PzADUgv29n/pmaRLBJ+u++Y1FbLX8UGI9dPvBlr46yQzBvIK
VzqTXuTmhcOO5GErRH3veaX8ff+q9nMohdvotjnQ6Al0Veh5xHYAHQ9FBAZiTdTb
HFPoCyTLPQGqEKuXPEPO4dWkro6/EKIHNksJG5iBMltg77FFDs8HyUJhKbzb8ktU
kKjlrrnh/Nh1EpTD6P7ocdUk3xX6BV1iitbeyMllRHjcHagjYyYO4kC615JVQWxn
XfgA4iqjYZP92mwokZ16
=nWtx
```


```
-----END PGP SIGNATURE-----
```

```
$ gpg --verify lapin.txt.asc
gpg: Signature faite le Lun 30 mar 23:44:29 2015 CEST avec la clef RSA d'identifiant
8B61268F
gpg: Bonne signature de « Nicolas Ollinger <nicolas.ollinger@univ-orleans.fr> »

$ gpg -r jeannot.lapin@terrier.net \
-u nicolas.ollinger@univ-orleans.fr -e -a lapin.txt
$ cat lapin.txt.asc
-----BEGIN PGP MESSAGE-----
Version: GnuPG/MacGPG2 v2.0.22 (Darwin)
Comment: GPGTools - https://gpgtools.org

hQEMA70atxXC53EgAQgAjEl9HAEdQhOaJk8ImPBvfpYRZx+KTqgljOKn6dYF9kCi
0zudbfOCGqfBBJT+2pNKZmsi6dYDJBaMoNtH/+I77upjtudnBt0W9h0xr9VPy0ox
HE+iJxrR0vNO9u0n1W/j7Uu0ZICFH5im3bBb5iap9Cyv9zIp3vwZKCCOmAXxEq7u
9wgPWbKaOwaafF7mVHE/rTMjWAc1q30bLUN3JSRC2olYW9/uyZ6DSRf/bXUOdGDa
fzRHHEU/Cyif35kDYO/91QG1mgQM5eXy6bRGx39PBDnX9v+4ZbdJwXXmmQBhWD69
4slhliGBlg3SVkQJiAbf459svZfWBYVP2SJF/UOqvNJiAVVMM7hpCzobjyBv24Sx
QO1DmcjShAQjEqlOM0FhlGe/MElkV67B2W1hkDrq7XTh4VmTj/THo/8y7TsivQ1v
T68LPCFqbOL/0gvW7xcvsO/IYnvWzLAaY3UjX5LSo59PxVQ=
=FyUN
-----END PGP MESSAGE-----
```

(9) DNSSEC

RFC 4033

Couche application enregistrements DNS

- Les enregistrements **DNS** sont fournis par des résolveurs DNS qui effectuent des requêtes auprès des serveurs DNS de noms de domaines.
- Les données circulent en clair sans contrôle d'**intégrité** ou d'**origine**.

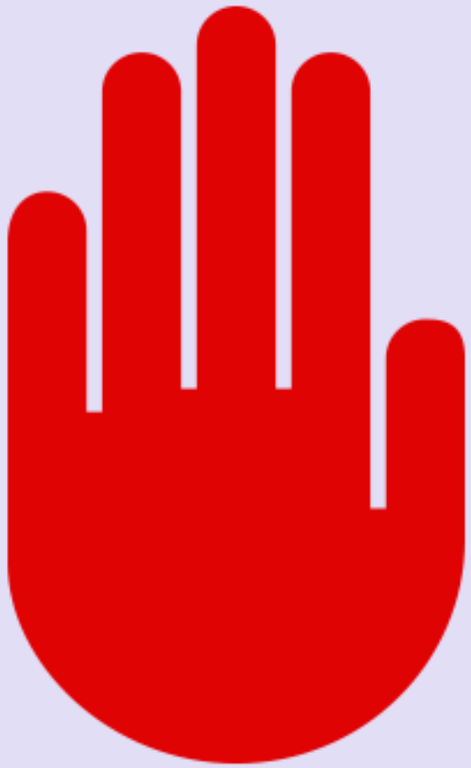
- Les résolveurs DNS peuvent mentir ! (pour votre bien ?)

Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

MINISTÈRE
DE
L'INTÉRIEUR

interieur.gouv.fr

MINISTÈRE DE L'INTÉRIEUR



VOUS AVEZ ÉTÉ REDIRIGÉ VERS CE SITE OFFICIEL CAR VOTRE ORDINATEUR ALLAIT SE CONNECTER À UN CONTENU ILLICITE

Cette redirection est conforme à l'article 6-1 de la loi du 21 juin 2004 pour la confiance dans l'économie numérique, modifiée par la loi du 13 novembre 2014 renforçant les dispositions relatives à la lutte contre le terrorisme

Si vous estimez que la page bloquée n'est pas illicite, vous pouvez contester la présente décision. Pour connaitre les voies et délais de recours, cliquez ici.

Votre redirection vers cette page d'information ne signifie pas que vous allez faire l'objet de poursuites judiciaires.

Que dit l'article 6-1 de la loi du 21 juin 2004 pour la confiance dans l'économie numérique ?

L'autorité administrative peut demander à l'éditeur ou à l'hébergeur d'un contenu pédopornographique ou constitutif d'apologie du terrorisme ou de provocation au terrorisme de le retirer.
Si l'éditeur et l'hébergeur du contenu illicite refusent de le retirer ou s'ils ne sont pas joignables, l'autorité administrative peut en demander le blocage aux fournisseurs d'accès à Internet.
La liste des adresses bloquées est élaborée par des enquêteurs de la direction centrale de la police judiciaire.
Les demandes de retrait et de blocage formulées par l'autorité administrative sont contrôlées par une personnalité qualifiée et indépendante.

Mise en œuvre

« Les adresses électroniques figurant sur la liste [noire] comportent soit un nom de domaine (DNS), soit un nom d'hôte caractérisé par un nom de domaine précédé d'un nom de serveur », obligation de rediriger vers « CE SITE OFFICIEL »

- En pratique les FAI utilisent des **résolveurs DNS menteurs** !

- Pour en savoir plus : **http://www.bortzmeyer.org/censure-francaise.html**

Pourquoi DNSSEC

- Le DNS joue un rôle critique dans l'identification des hôtes et services.

- Lieu logique pour stocker des clés publiques !

- Nécessite de rendre le DNS **vérifiable** :
  - ➡ authentification des enregistrements ;
  - ➡ intégrité des enregistrements ;
  - ➡ authentification du **déni d'existence**.

zone DNS

```
$TTL    60000
@       IN  SOA dnsjmail.jmail.com.root.dnsjmail.jmail.com. (
            1 ; serial
            28 ; refresh
            14 ; retry
            3600000 ; expire
            60000 ; negative cache ttl
            )
@       IN  NS  dnsjmail.jmail.com.
@       IN  MX 5    smtp.jmail.com.

dnsjmail    IN  A   173.194.66.108
imap        IN  A   173.194.66.108
smtp        IN  A   173.194.66.108
```

dnssec-tools

Générer la clé de zone (ZSK) :

```
$ dnssec-keygen -3 -b 512 -m ZONE -r /dev/urandom jmail.com
```

Générer la clé de clé (KSK) :

```
$ dnssec-keygen -3 -b 512 -m ZONE -r /dev/urandom -f KSK jmail.com
```

Ajouter les clés publiques à la zone :

```
$ cat Kjmail.com.*.key >> db.com.jmail
```

Signer la zone :

```
$ dnssec-signzone -t -3 CAFEBABE -o jmail.com db.com.jmail \
      kjmail.com.*.private
```

Transférer les enregistrements DS au domaine père (…)

```
; File written on Tue Mar 31 07:55:20 2015
; dnssec_signzone version 9.8.1-P1
jmail.com.          60000  IN SOA dnsjmail.jmail.com.
root.dnsjmail.jmail.com. (
                              1          ; serial
                              28         ; refresh (28 seconds)
                              14         ; retry (14 seconds)
                              3600000    ; expire (5 weeks 6 days
16 hours)
                              60000      ; minimum (16 hours 40
minutes)
                              )
                    60000  RRSIG  SOA 7 2 60000 20150430045520 (
                              20150331045520 5843 jmail.com.
                              mRKiXv0DWJcZH+NT8q/J71QGodToaMgt/BJL
                              p06L1XIsNnkEyq4ExwpMcqQk3hGu6Zzy6XuH
                              DwtxBuITrGO3ZA== )
                    60000  NS     dnsjmail.jmail.com.
                    60000  RRSIG  NS 7 2 60000 20150430045520 (
                              20150331045520 5843 jmail.com.
                              gfHReOpL4gxSzTxKEO7ey/yqEZXz/2U9H6qh
                              N2CwS2h5K7HU2CZrj6ouB5EZTbkV7yAwgx00
                              /BijvdrXsLRs9g== )
                    60000  MX     5 smtp.jmail.com.
                    60000  RRSIG  MX 7 2 60000 20150430045520 (
                              20150331045520 5843 jmail.com.
                              kQUZL8LWGgLnC6iJFjMGJ2HsF1ASzv6dvo6m
                              gyw2+NFaO0g8a7U7fKw8DfVXUiYzmK/aDMuG
                              SqGTPOBrzZCUYQ== )
          60000  DNSKEY 256 3 7 (
                              AwEAAbKv/OQCpxGwL2qqGKAexb7BakEWFLDl
                              Og3QQpsZ8sZdvwGoquuPxNSpU+neRhcwuJ7o
                              Gpn+LhJzRRrzDrD1CgE=
                              ) ; key id = 5843
          60000  DNSKEY 257 3 7 (
                              AwEAAdrW4WxN0/+zhkDiYYKiDSX0VNAY0XI/
                              Vuekyz8IPBv+sogld45Yb77GUNQODRNe1Pda
                              fkXwB2MCMhUjdI74N7U=
                              ) ; key id = 30420
          60000  RRSIG  DNSKEY 7 2 60000 20150430045520 (
                              20150331045520 5843 jmail.com.
                              Tw+GwQp2DdodTJAjqDerBEC2Z93wyAh0XvAN
                              CPxFHRvdQhxBeMOaHo13MgioAjurGF8OLiUq
                              lU754YJXdGhMBw== )
          60000  RRSIG  DNSKEY 7 2 60000 20150430045520 (
                              20150331045520 30420 jmail.com.
                              LIPlDHhuIPdSlSbCiF+wJ+vG/b9nLKJ5AAFQ
                              xXs+Z/cB6NPrPAS3yow8Rfygeh5CS+eUTtY9
                              t3OX+RFgq+tB8Q== )
          0      NSEC3PARAM 1 0 10 CAFEBABE
          0      RRSIG  NSEC3PARAM 7 2 0 20150430045520 (
                              20150331045520 5843 jmail.com.
                              PT4ygqNtAsJUBNatirN3KkVxlMOW5xJIrX6V
                              fhLEN7th5+srAQVWkNLxshfdbAGZHm+tc11i
                              /2anonYojbTtwQ== )
dnsjmail.jmail.com. 60000  IN A    173.194.66.108
                    60000  RRSIG  A 7 3 60000 20150430045520 (
                              20150331045520 5843 jmail.com.
                              qFlOyuqrwUGTUtC5j3K32/TrJXp2VhTOMOGZ
                              QCrtWmyKZaGvMHG+zgh47i0B/nLDug/lLW4S
                              qkPVoNsiz5FoNg== )
imap.jmail.com.     60000  IN A    173.194.66.108
                    60000  RRSIG  A 7 3 60000 20150430045520 (
                              20150331045520 5843 jmail.com.
                              CqW2eITtWcvN0d48VAG8vWiDoRnuWmMj/WWr
                              NGLur57pUubtSnsAmE/LBpbqKWVt+hlxC5wZ
                              khP0wbwWnr6j1A== )
smtp.jmail.com.     60000  IN A    173.194.66.108
                    60000  RRSIG  A 7 3 60000 20150430045520 (
                              20150331045520 5843 jmail.com.
                              IxRb+lGx4XvJYVDBhmVlC6tDbm+gxNTuBMwe
                              KG1U/zKMAOymAj7G4fVKdiib7SnMIoADwceP
                              jyb7oJr3zhqtXw== )
0J5QF8ETEGBED1UKTFGTMMK8JVP45HPH.jmail.com. 60000 IN NSEC3 1 0 10
CAFEBABE 4A7H1CMINJK5800GTUF1KKRKCB18I4AF NS SOA MX RRSIG DNSKEY
NSEC3PARAM
                    60000  RRSIG  NSEC3 7 3 60000 20150430045520 (
                              20150331045520 5843 jmail.com.
                              B3ur9KXn//qe22d8GZWKB6P2XusqqkBRfEP9
                              kha5ewK7ufE6rJ1rarIlK9yg0HqbToQl14EC
                              5Z9uWBeEsGSSVw== )
4A7H1CMINJK5800GTUF1KKRKCB18I4AF.jmail.com. 60000 IN NSEC3 1 0 10
CAFEBABE MJUI38Q1II10KUHV6JRB1I4C24QMM46L A RRSIG
                    60000  RRSIG  NSEC3 7 3 60000 20150430045520 (
                              20150331045520 5843 jmail.com.
                              TU6upk9hJClndfnVGmugJk7RC2Ih2fo5e9nC
                              x15l5ZvEM98E9qWMkUmefVWnf6E01P30SN1C
                              q+PpF0zHENOb1A== )
MJUI38Q1II10KUHV6JRB1I4C24QMM46L.jmail.com. 60000 IN NSEC3 1 0 10
CAFEBABE QN6BN3EHT42O8PGN8ILE8SIERMFL6P4C A RRSIG
                    60000  RRSIG  NSEC3 7 3 60000 20150430045520 (
                              20150331045520 5843 jmail.com.
                              a8PnGdMpDA0VmHnsJPKFWZ/EmDwUmGxURrTn
```

DNSKEY

```
60000 DNSKEY    256 3 7 (
                AwEAAbKv/OQCpxGwL2qqGKAexb7BakEWFLD1
                Og3QQpsZ8sZdvwGoquuPxNSpU+neRhcwuJ7o
                Gpn+LhJzRRrzDrD1CgE=
                ) ; key id = 5843
```

RRSIG

```
imap.jmail.com.      60000 IN A  173.194.66.108
          60000 RRSIG A 7 3 60000 20150430045520 (
                20150331045520 5843 jmail.com.
                CqW2eITtWcvN0d48VAG8vWiDoRnuWmMj/WWr
                NGLur57pUubtSnsAmE/LBpbqKWVt+hlxC5wZ
                khP0wbwWnr6j1A== )
```

NSEC3

```
                                0 NSEC3PARAM 1 0 10 CAFEBABE

0J5QF8ETEGBED1UKTFGTMMK8JVP45HPH.jmail.com. 60000 IN NSEC3 1 0 10
CAFEBABE 4A7H1CMINJK5800GTUF1KKRKCB18I4AF NS SOA MX RRSIG DNSKEY
NSEC3PARAM
60000  RRSIG  NSEC3 7 3 60000 20150430045520 (
       20150331045520 5843 jmail.com.
       B3ur9KXn//qe22d8GZWKB6P2XusqqkBRfEP9
       kha5ewK7ufE6rJ1rarIlK9yg0HqbToQll4EC
       5Z9uWBeEsGSSVw== )
```

DS

```
jmail.com.        IN DS 30420 7 1
D14580294B9C81415BCFB1494750CAF129C00812
jmail.com.        IN DS 30420 7 2
E5967DC1A68EABF609FF1A93B83EBF1F63743E456F7A9978A44E4097 A660A94D
```

- DS empreinte de la clé DNSKEY du sous-domaine
- Key Signing Key (**KSK**) : référencée dans DS, signe ZSK.
- Zone Signing Key (**ZSK**) : utilisée pour signer le reste.
- Clé racine : **https://data.iana.org/root-anchors/**
- Rotation régulière des clés !