

Protocoles et Tunnels de Sécurité

Dr. Nihel Ben Youssef

Références

R, E. Corvalan et Y. Le Corvic. « Les VPN, Principes, conception et déploiement des réseaux privés virtuels ».

S. Thomas. « SSL and TLS Essentials ».

G. Berton. Security Protocols: the case of Secure Sockets Layer (SSL) and Transport Layer Security (TLS)

Tunnels sur la couche Transport

SSL/TLS

**(Secure Socket Layer/Transport Layer
Security)**

SSL/TLS : Introduction

- Le Protocole SSL est développé en 1994 par Netscape.
- Permet la mise en œuvre de tunnels au niveau 4 du modèle OSI.
- N'est utilisable que pour la sécurisation du flux TCP.
- Largement utilisé pour HTTP qui devient HTTPS (port 443)
- Peut être implémenté pour d'autres protocoles applicatifs comme POP,FTP,SMTP,LDAP...
- La dernière version de SSL est 3. Ses fonctionnalités sont très similaires à celles du protocole TLS (Transport Layer Security) version 1. On dit souvent que : SSLv3=TLSv1
- En 2013, dernière version de TLS est TLSv1.2

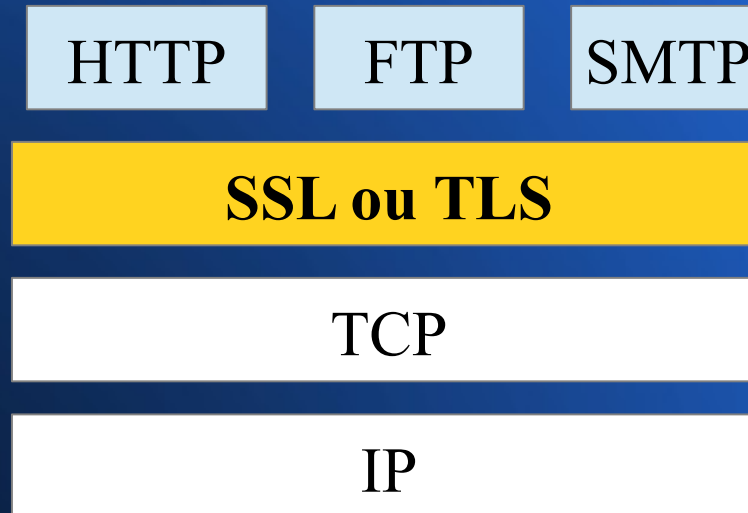
Ports au dessus de SSL

Protocole sécurisé	Port	Protocole non sécurisé	Application
HTTPS	443	HTTP	Transactions requête-réponse sécurisées
SSMTP	465	SMTP	Messagerie électronique
SNntp	563	NNTP	News sur le réseau Internet
SSL-LDAP	636	LDAP	Annuaire X.500 allégé
SPOP3	995	POP3	Accès distant à la boîte aux lettres avec rapatriement des messages

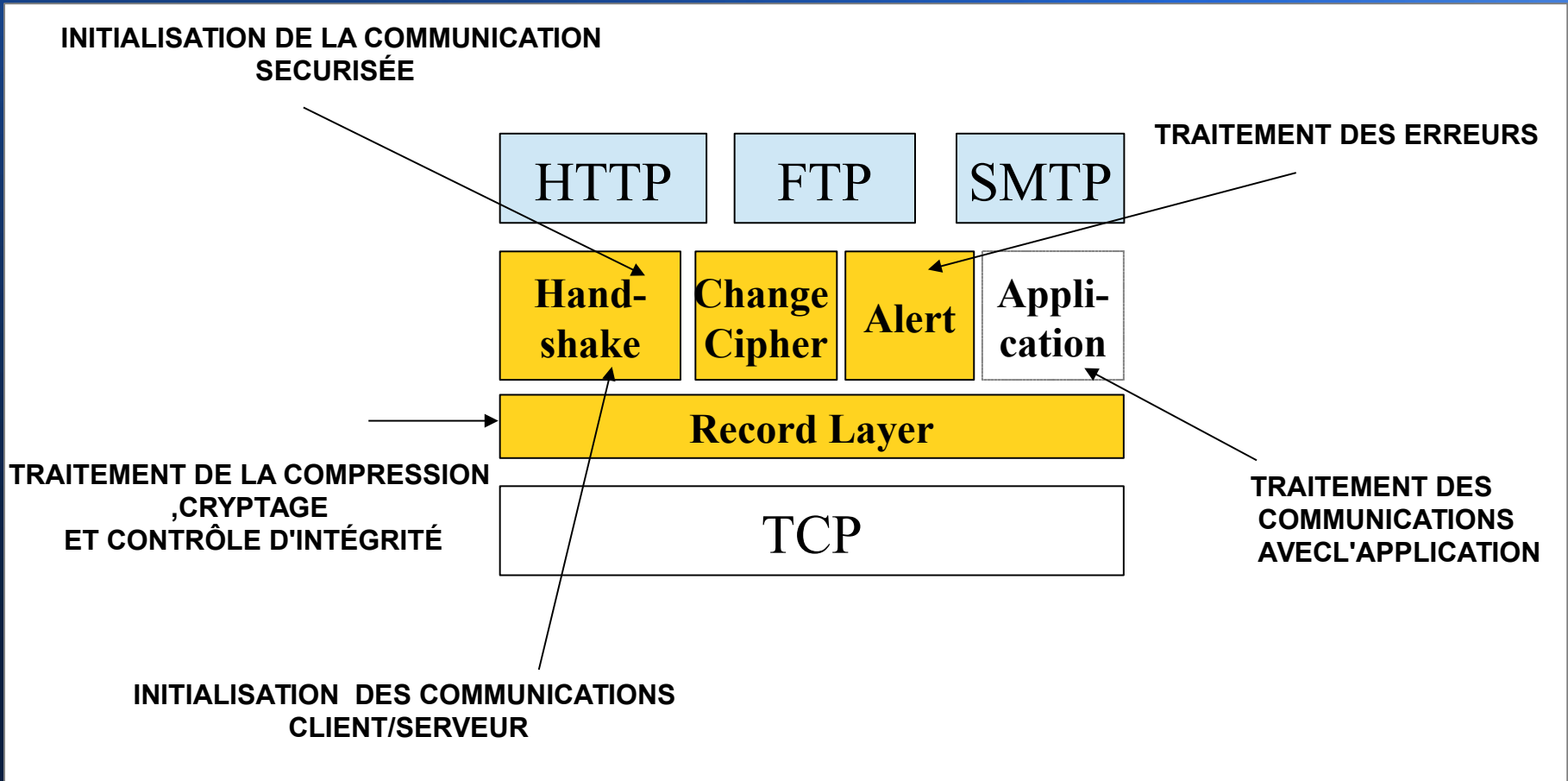
Ports au dessus de SSL

Protocole sécurisé	Port	Protocole non sécurisé	Application
FTP-DATA	889	FTP	Transfert de fichiers
FTPS	990	FTP	Contrôle du transfert de fichiers
IMAPS	991	IMAP4	Accès distant à la boîte aux lettres avec ou sans rapatriement des messages
TELNETS	992	Telnet	Protocole d'accès distant à un système informatique
IRCS	993	IRC	Protocole de conférence par l'écrit

SSL/TLS: Architecture



SSL/TLS: Architecture

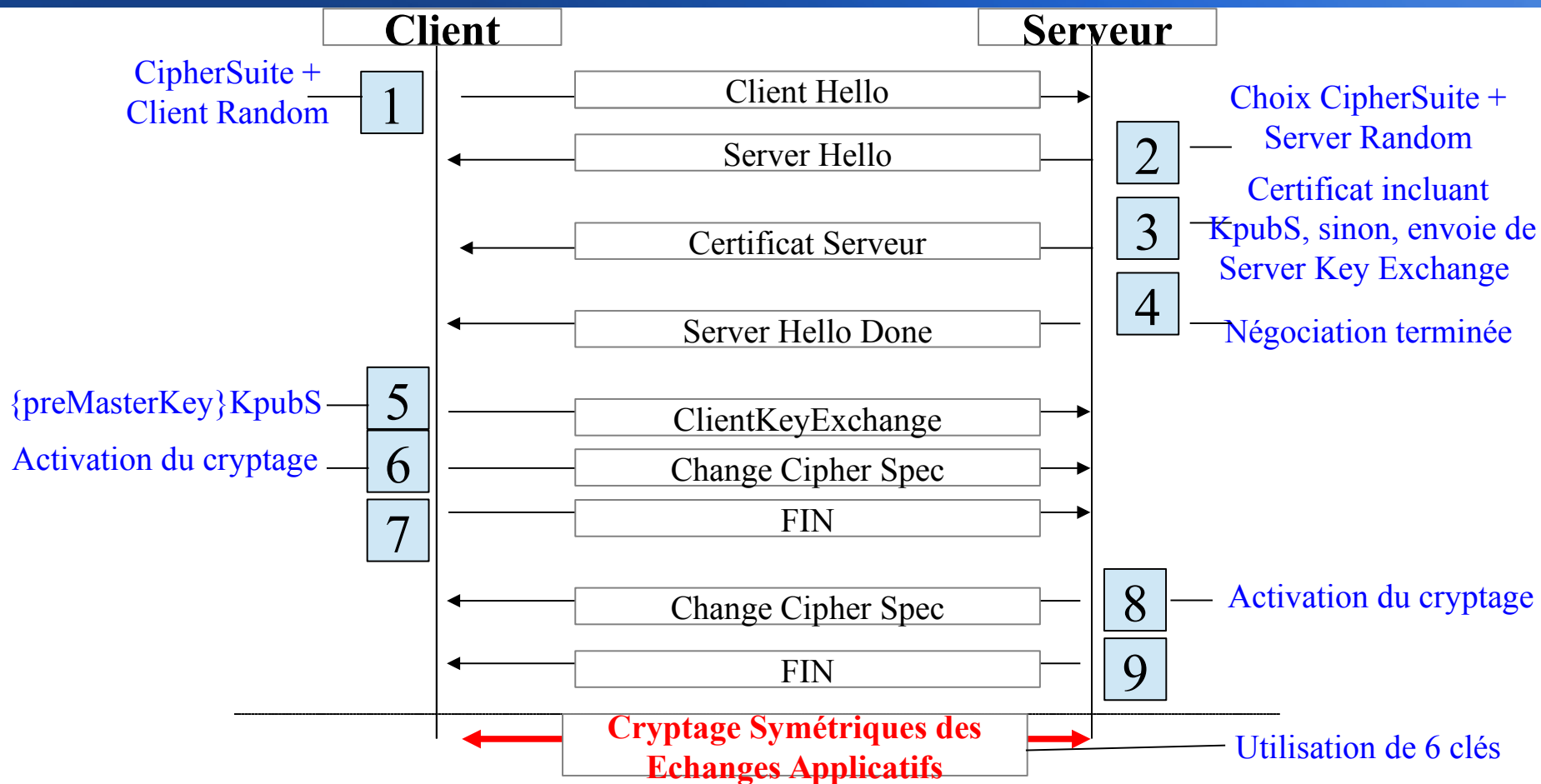


SSL/TLS

Propriétés de sécurité fournies

- **Échange sécurisé de clés de chiffrement**
- **Authentification du serveur** (optionnelle mais souvent utilisée)
- **Authentification du client** (optionnelle et très peu utilisée)
- **Confidentialité et Intégrité des messages**

Mécanisme d'établissement d'un tunnel TLS



TLS

Record Layer/Handshake Protocol Client Hello

No.	Time	Source	Destination	Protocol	Length	Info
112	11.381687	192.168.1.70	5.178.40.138	SSL	288	Client Hello

Transmission Control Protocol, Src Port: 48986 (48986), Dst Port: https (443), Seq: 1, Ack: 1, Len: 222

Secure Sockets Layer

- ▼ TLSv1.2 Record Layer: Handshake Protocol: Client Hello
 - Content Type: Handshake (22)
 - Version: TLS 1.0 (0x0301)
 - Length: 217
 - ▼ Handshake Protocol: Client Hello
 - Handshake Type: Client Hello (1)
 - Length: 213
 - Version: TLS 1.2 (0x0303)
 - ▶ Random
 - Session ID Length: 32
 - Session ID: df5eb658c8fb5389f5dbb8df2380ac2c8f35dd5d6eb17863...
 - Cipher Suites Length: 46
 - ▼ Cipher Suites (23 suites)
 - Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 (0xc02b)
 - Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)
 - Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA (0xc00a)
 - Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA (0xc009)
 - Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013)

Analyse de trafic TLS

Record Layer/Handshake Protocol

Server Hello

121 11.480608 5.178.40.138 192.168.1.70 TLSv1.2 1454 Server Hello

- ▶ Transmission Control Protocol, Src Port: https (443), Dst Port: 48981 (48981), Seq: 1, Ack: 223, Len: 1388
- ▼ Secure Sockets Layer
 - ▼ TLSv1.2 Record Layer: Handshake Protocol: Server Hello
 - Content Type: Handshake (22)
 - Version: TLS 1.2 (0x0303)
 - Length: 57
 - ▼ Handshake Protocol: Server Hello
 - Handshake Type: Server Hello (2)
 - Length: 53
 - Version: TLS 1.2 (0x0303)
 - ▶ Random
 - Session ID Length: 0
 - Cipher Suite: TLS_RSA_WITH_AES_128_CBC_SHA (0x002f)
 - Compression Method: null (0)

**RSA : Algorithme
Asymétrique
(Authentification)**

**AES: Algorithme symétrique
avec le mode opérateur CBC
(Cipher Block Chaining) et
une taille de clé de 128bits
(Confidentialité)**

**SHA: Algorithme de hachage
(Intégrité)**

Analyse de trafic TLS

Record Layer/Handshake Protocol

Certificate

125	11.486464	5.178.40.138	192.168.1.70	TLSv1.2	600	Certificate, Server Hello Done
-----	-----------	--------------	--------------	---------	-----	--------------------------------

Version: TLS 1.2 (0x0303)

Length: 3234

▼ Handshake Protocol: Certificate

Handshake Type: Certificate (11)

Length: 3230

Certificates Length: 3227

▼ Certificates (3227 bytes)

Certificate Length: 1114

▼ Certificate (id-at-commonName=a248.e.akamai.net,id-at-organizationName=Akamai Technologies, Inc.,id-at-localityName=Cambridge, MA)

▼ signedCertificate

version: v3 (2)

serialNumber : 0x01000000000013feace5c1353dcee

▶ signature (shaWithRSAEncryption)

▶ issuer: rdnSequence (0)

▶ validity

▶ subject: rdnSequence (0)

▼ subjectPublicKeyInfo

▶ algorithm (rsaEncryption)

Padding: 0

subjectPublicKey: 3082010a0282010100afb6532b2dc35b513095f9fb56a94d...

KpubS : Clé publique du serveur

Analyse de trafic TLS

Record Layer/Handshake Protocol

Client Key Exchange

128 | 11.487657 | 192.168.1.70 | 5.178.40.138 | TLSv1.2 | 408 | Client Key Exchange, Change Cipher Spec, Encrypted Handshake

Frame 128: 408 bytes on wire (3264 bits), 408 bytes captured (3264 bits)

Ethernet II, Src: HonHaiPr_6d:c4:5f (38:59:f9:6d:c4:5f), Dst: ThomsonT_89:e2:8e (00:1f:9f:89:e2:8e)

Internet Protocol Version 4, Src: 192.168.1.70 (192.168.1.70), Dst: 5.178.40.138 (5.178.40.138)

Transmission Control Protocol, Src Port: 48981 (48981), Dst Port: https (443), Seq: 223, Ack: 3311, Len: 342

Secure Sockets Layer

▼ TLSv1.2 Record Layer: Handshake Protocol: Client Key Exchange

Content Type: Handshake (22)

Version: TLS 1.2 (0x0303)

Length: 262

▼ Handshake Protocol: Client Key Exchange

Handshake Type: Client Key Exchange (16)

Length: 258

0030	05 99 b2 48 00 00 01 01 08 0a 01 80 b8 04 aa 60	...H....
0040	f6 f1 16 03 03 01 06 10 00 01 02 01 00 30 9e 6c0.l
0050	3f eb 33 50 d9 d6 c3 d3 cc b5 6a a6 54 e9 f9 1e	? .3P.... .j.T...
0060	fa 2e 5e 39 5d e6 25 a0 21 16 a7 ed 7e 23 6f 28	..^9].%. !...~#o(
0070	13 c9 46 84 ce 1c 62 2a dd 71 24 23 7d 07 60 6c	..F...b*.q\$#}.\l
0080	b6 f3 82 dc 8f 7f e1 98 e3 a9 f3 7e ea 80 22 bd~..".
0090	fb 8c c3 00 d9 60 cf 4e 95 75 ea 3b 93 5c 28 41` .N .u.;.\(A
00a0	b4 40 39 a3 ca 34 48 c7 d4 28 2e 50 fb 1e cb 7d	..@9..4H. .(.P...}
00b0	20 9f 67 77 b5 94 6d 1c dd cf 92 d8 1b ae c1 33	..gw..m.3
00c0	63 f7 36 83 ed ef 1c 1b 63 81 f7 05 a0 a6 bc 73	c.6.....C.....s
00d0	d0 fa c4 18 c5 14 6d 2c 79 db e7 60 3b 7f 08 3dm, y...;..=
00e0	8d da 58 64 8e 35 cc 82 10 1f 39 25 f5 65 c6 95	..Xd.5...9%.e..
00f0	c4 24 aa d0 8b 24 f0 d9 dd 5b e6 6a 2a f2 63 4e	..\$.\$. .[.j*.cN
0100	06 90 75 2e be 03 ac d0 2b 5f 27 7b d0 c7 ab 59	..u..... + '{...Y

PreMasterKey choisi par le client crypté par KpubS

TLS

Génération des 6 clés de session KDF(Key Derivation Function)

- A partir de **preMasterKey**, le client et le serveur génère simultanément une clé appelée **MasterKey**.
- A partir de **MasterKey**, ils génèrent les 6 clés suivantes :
 - **Client Cipher** : utilisée pour chiffrer les données du client vers le serveur.
 - **Server Cipher**: utilisée pour chiffrer les données du serveur vers le client.
 - **Client MAC** : utilisée dans la fonction cryptographique de hachage HMAC coté client pour le contrôle d'intégrité.
 - **Server MAC** : utilisée dans la fonction cryptographique de hachage HMAC par le serveur pour le contrôle d'intégrité.
 - **Client IV** : Vecteur d'initialisation utilisé par le client au niveau du mode CBC lors du chiffrement symétrique des données.
 - **Server IV** : Vecteur d'initialisation utilisé par le serveur au niveau du mode CBC lors du chiffrement symétrique des données.

Rappels- HMAC(Hash-Based Message Authentication Code)

(Wikipédia)

« La fonction **HMAC** est défini ainsi :

$$\text{HMAC}_K(m) = h\left((K \oplus \text{opad}) \parallel h((K \oplus \text{ipad}) \parallel m)\right)$$

avec :

h : une fonction de hachage itérative(MD5,SHA),

K : la clé secrète complétée avec des zéros pour qu'elle atteigne la taille de bloc de la fonction h . Dans notre cas, il s'agit soit de la clé client MAC ou server MAC

m : le message à authentifier,

"|" désigne une concaténation,

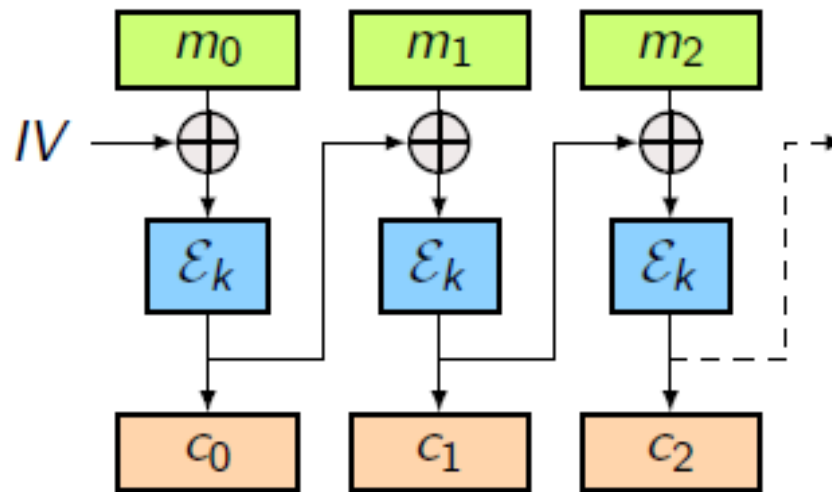
ipad et opad , chacune de la taille d'un bloc, sont définies par : $\text{ipad} = 0x363636\dots3636$ et $\text{opad} = 0x5c5c5c\dots5c5c$. Donc, si la taille de bloc de la fonction de hachage est 512, ipad et opad sont 64 répétitions des octets, respectivement, $0x36$ et $0x5c$. »

Rappels- mode de chiffrement CBC (Cipher Block Chaining)

(Wikipédia)

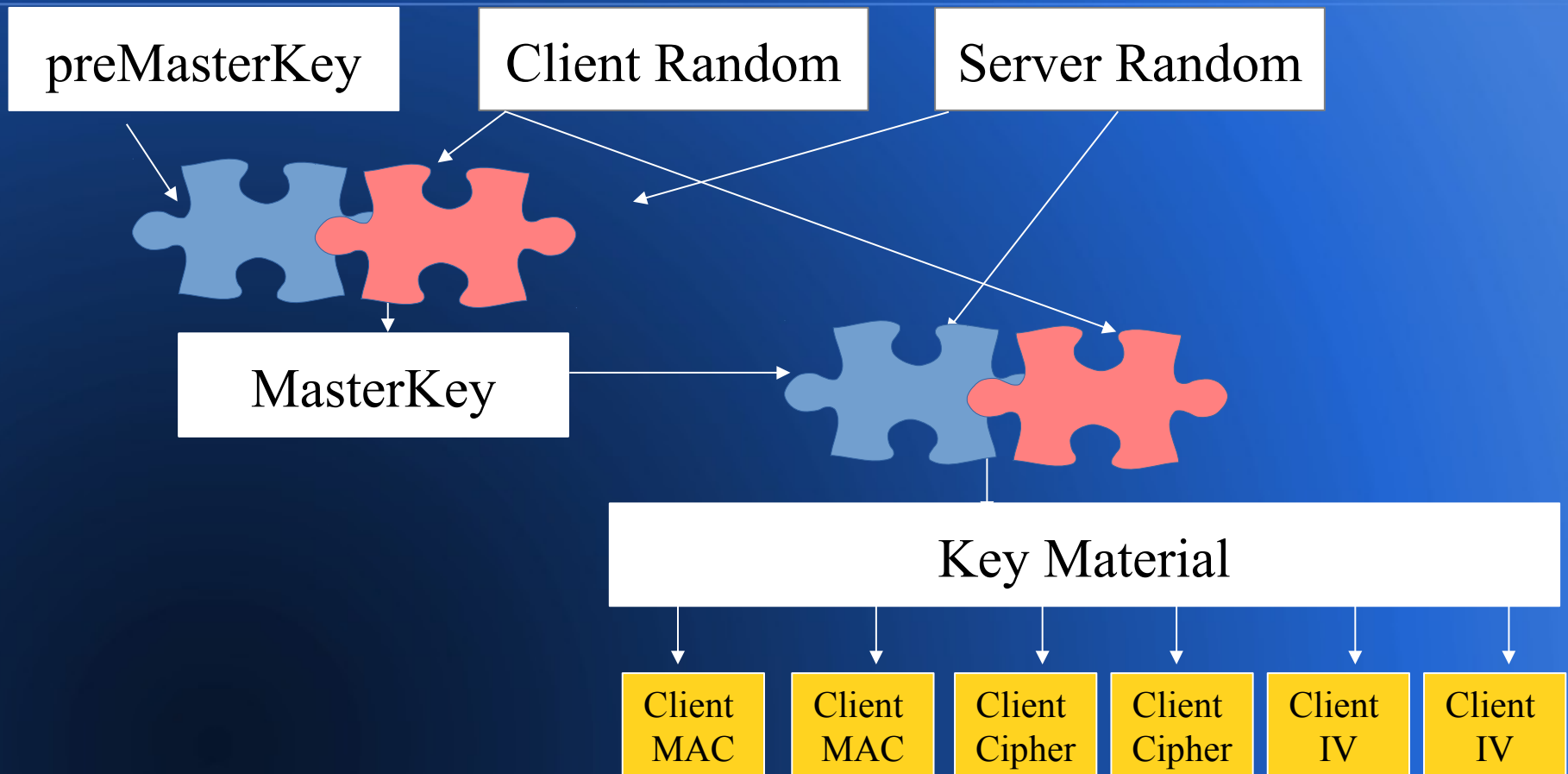
-Le message m à chiffrer est découpé en bloc (m_0, m_1, \dots). Un bloc dépend de tous les précédents. Mode randomisé par la présence d'une valeur aléatoire initiale IV . Dans notre cas, il s'agit soit du client IV ou Server IV .

- E_k est un algorithme de chiffrement symétrique au choix (DES, 3DES, AES...)



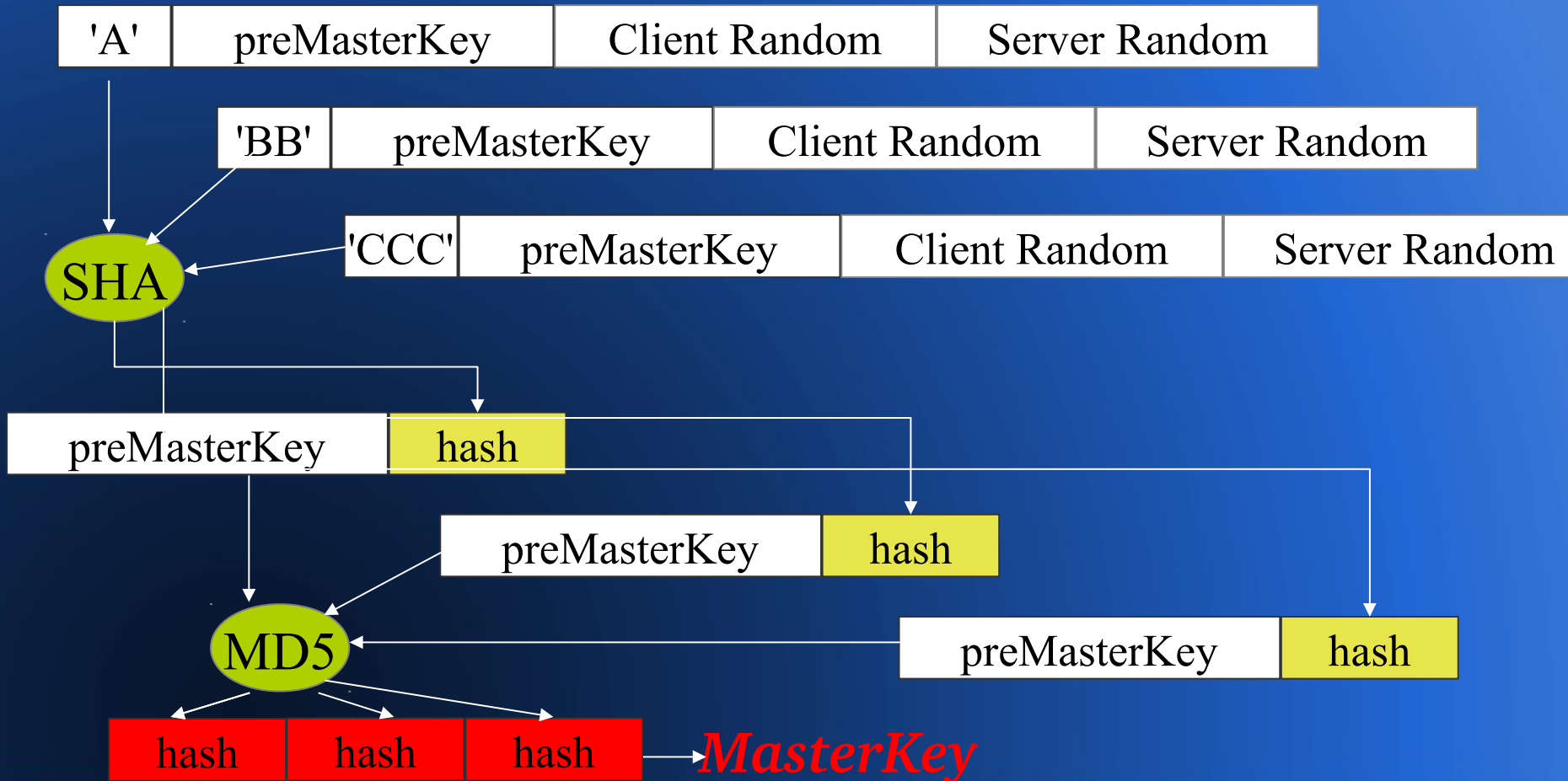
SSL/TLS

Génération des 6 clés de session KDF(Key Derivation Function)



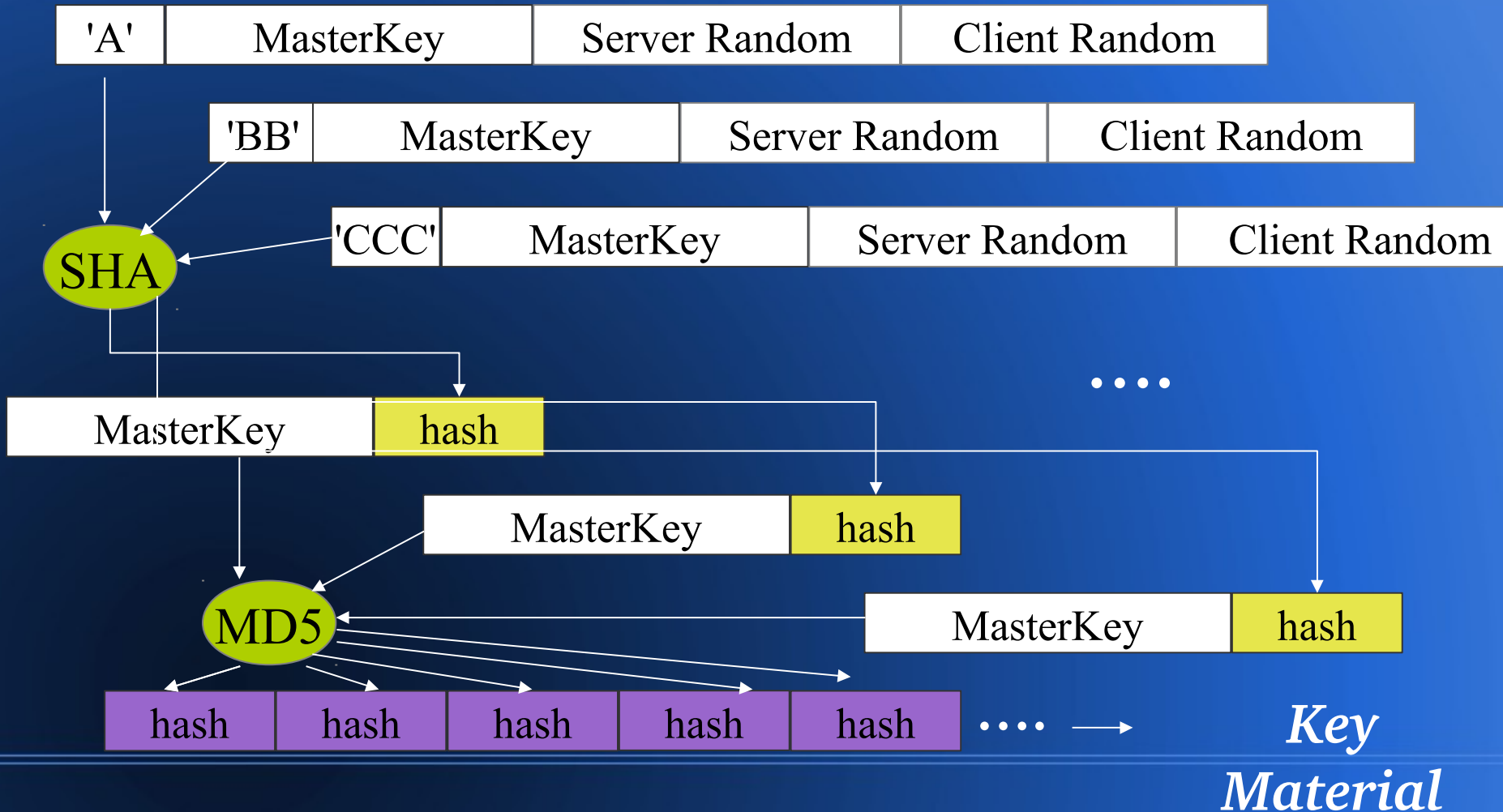
SSL

Dérivation de *MasterKey*



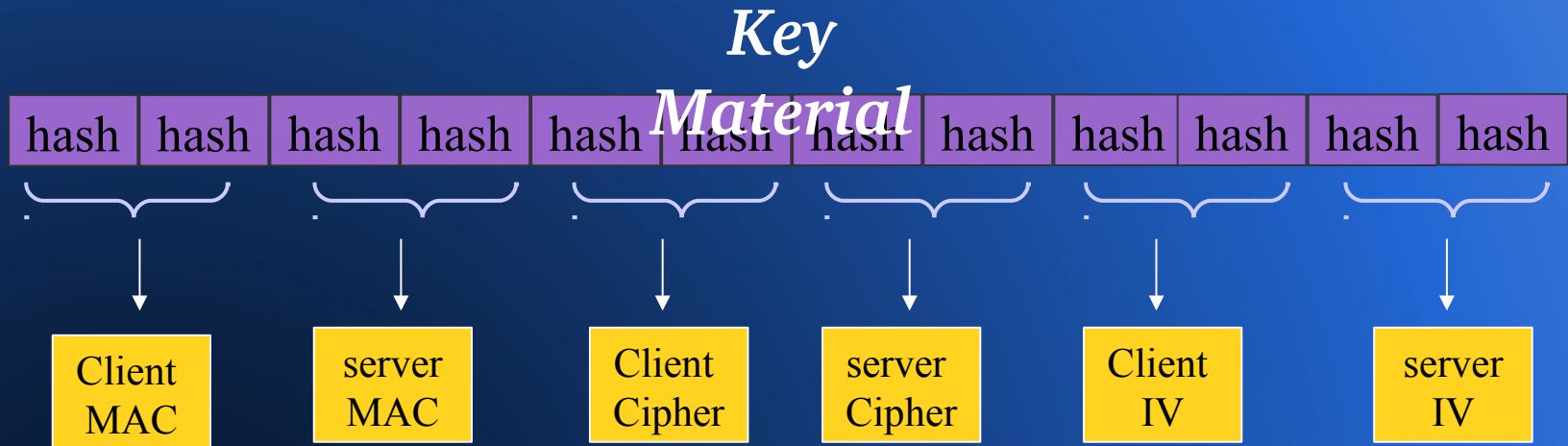
SSL

Dérivation de *Key Material*



SSL

Récupération des 6 clés



TLS-Record Protocol

Echange sécurisé des données

