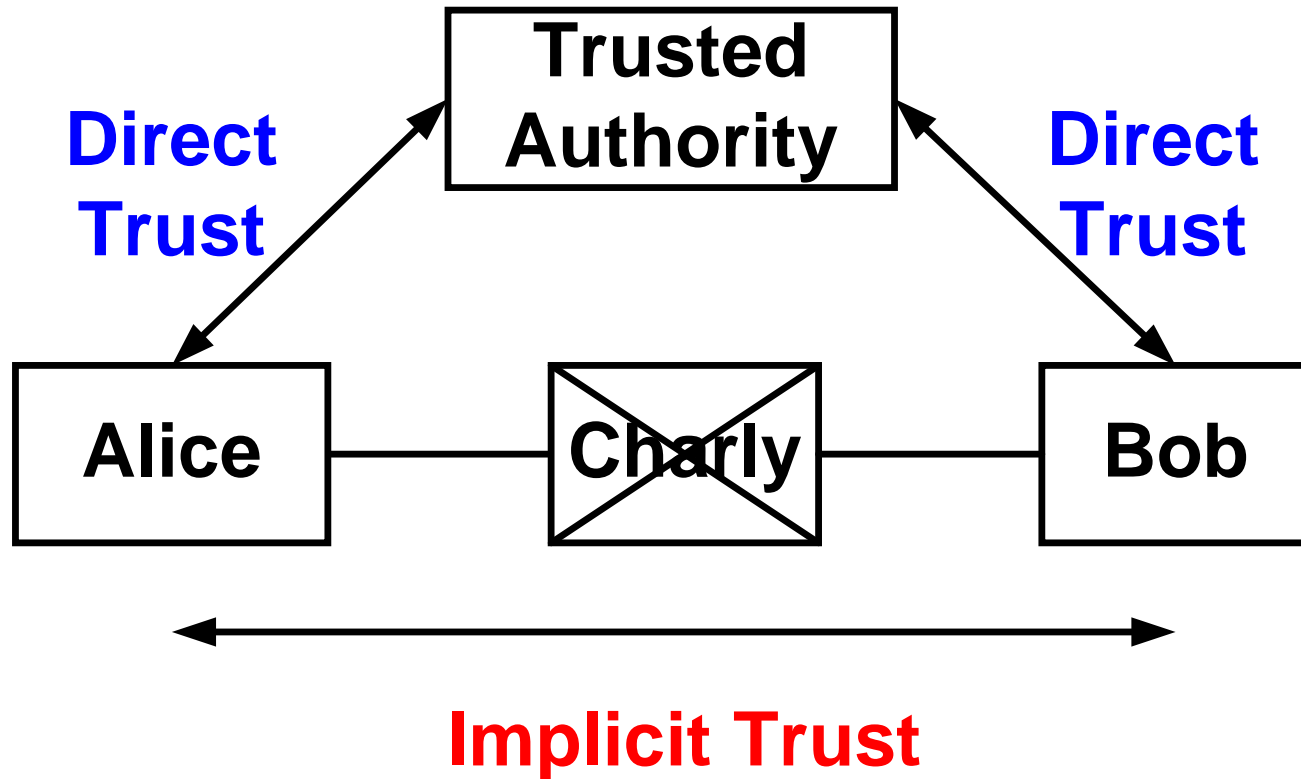


Public Key Infrastructure

- Définition, Certification Authority (CA)
- Certificat numérique, chaîne de certification, contrôle d'intégrité, durée de vie, révocation, authentification du serveur
- E-commerce sécurisé avec SSL, authentification du client
- Utilité, composants, mécanismes
- Etude détaillée du protocole SSL (*Secure Socket Layer*)
- **Les annexes du labo ne font pas partie du champ évalué**

Trusted Authority

- Tiers garant, tiers de confiance

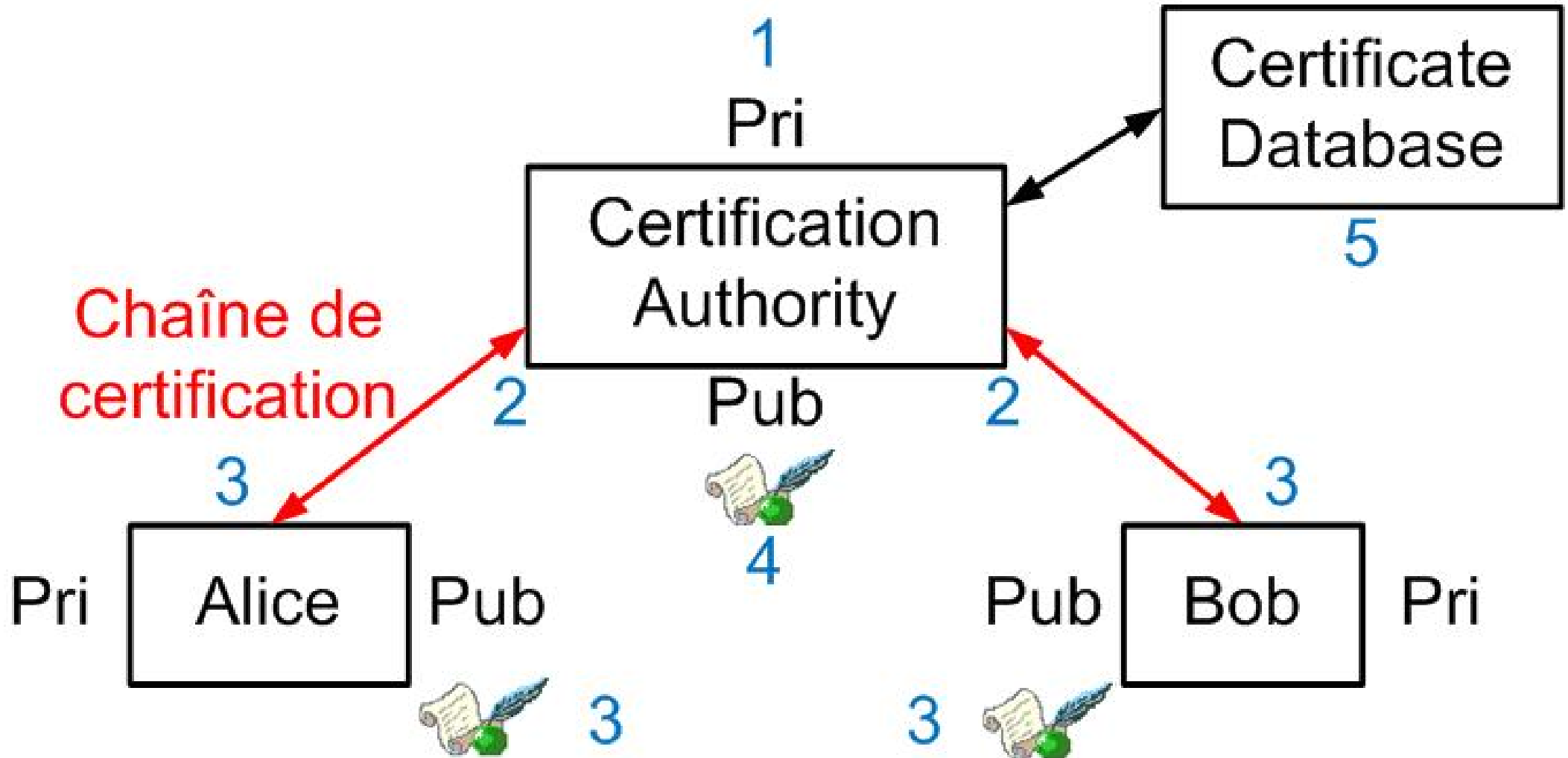


- Alice et Bob sans Charly (*Man in the Middle*) !

Public Key Infrastructure (PKI)

- Une infrastructure à clé publique (*Public Key Infrastructure*) crée un **espace de confiance** (*trust*) qui permet de gérer tous les aspects de sécurité : authentification des utilisateurs et des entités techniques, confidentialité - intégrité des données et non-répudiation des transactions
- **Comment obtenir la clé publique de mon correspondant ?**
→ moyen de communication jugé sûr comme le courrier postal, le téléphone, la valise diplomatique, ...,
- Une PKI est constituée par des services de **génération et de diffusion des certificats numériques** (sorte de carte d'identité virtuelle) et des **clés** nécessaires au chiffrement et au déchiffrement

Certification Authority (CA)



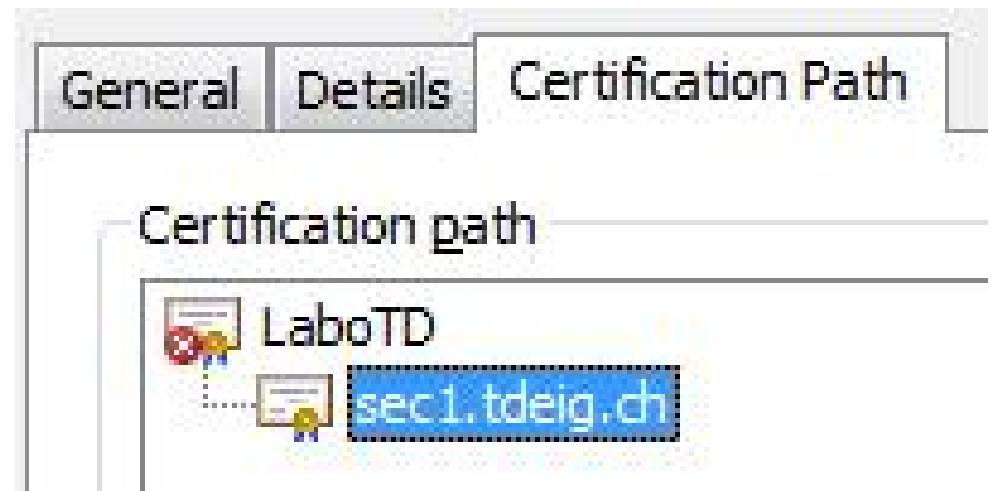
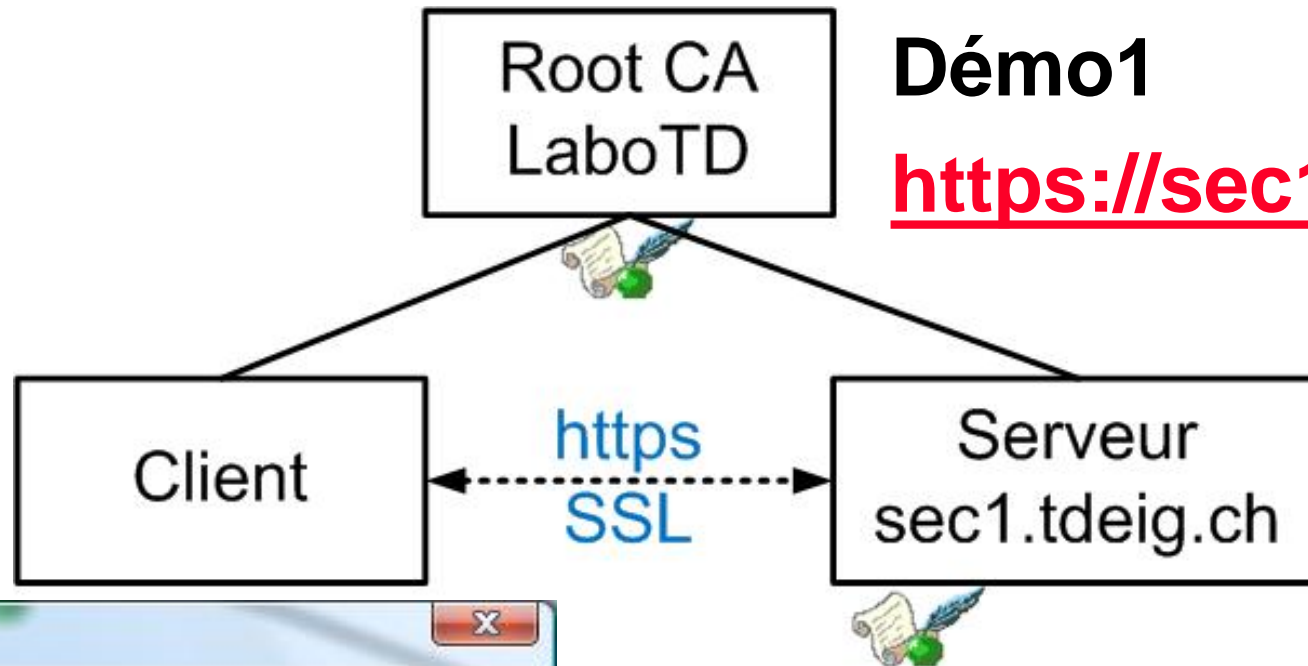
Autorité de certification

- 1 Une CA s'appuie sur une paire de clés asymétriques
- 2 Elle reçoit une demande de certificat
- 3 Elle génère un certificat (format X.509) signé
- 4 Elle met à disposition son certificat
Pourquoi ?
- 5 Elle met à disposition une liste des certificats révoqués (annulés)

E-commerce sécurisé par SSL

Démo1

<https://sec1.tdeig.ch/>



Certificat (1)

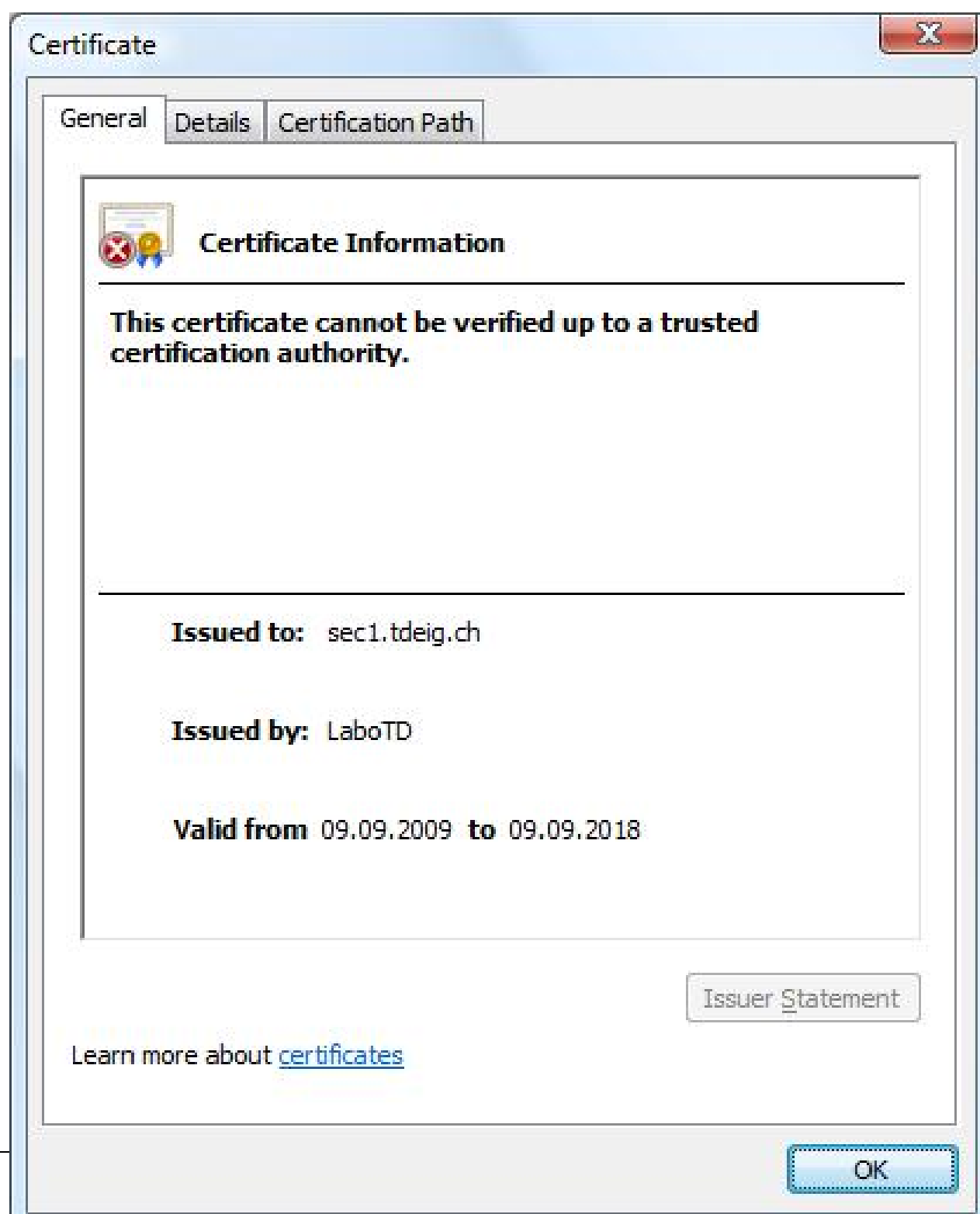
Onglet *General*

Problème

Issued to sec1.tdeig.ch
(chaîne de certification)

Issued by LaboTD

Validité



Certificat (2)

Onglet *Detail*

Sign. algorithm = sha1RSA

Issuer = LaboTD

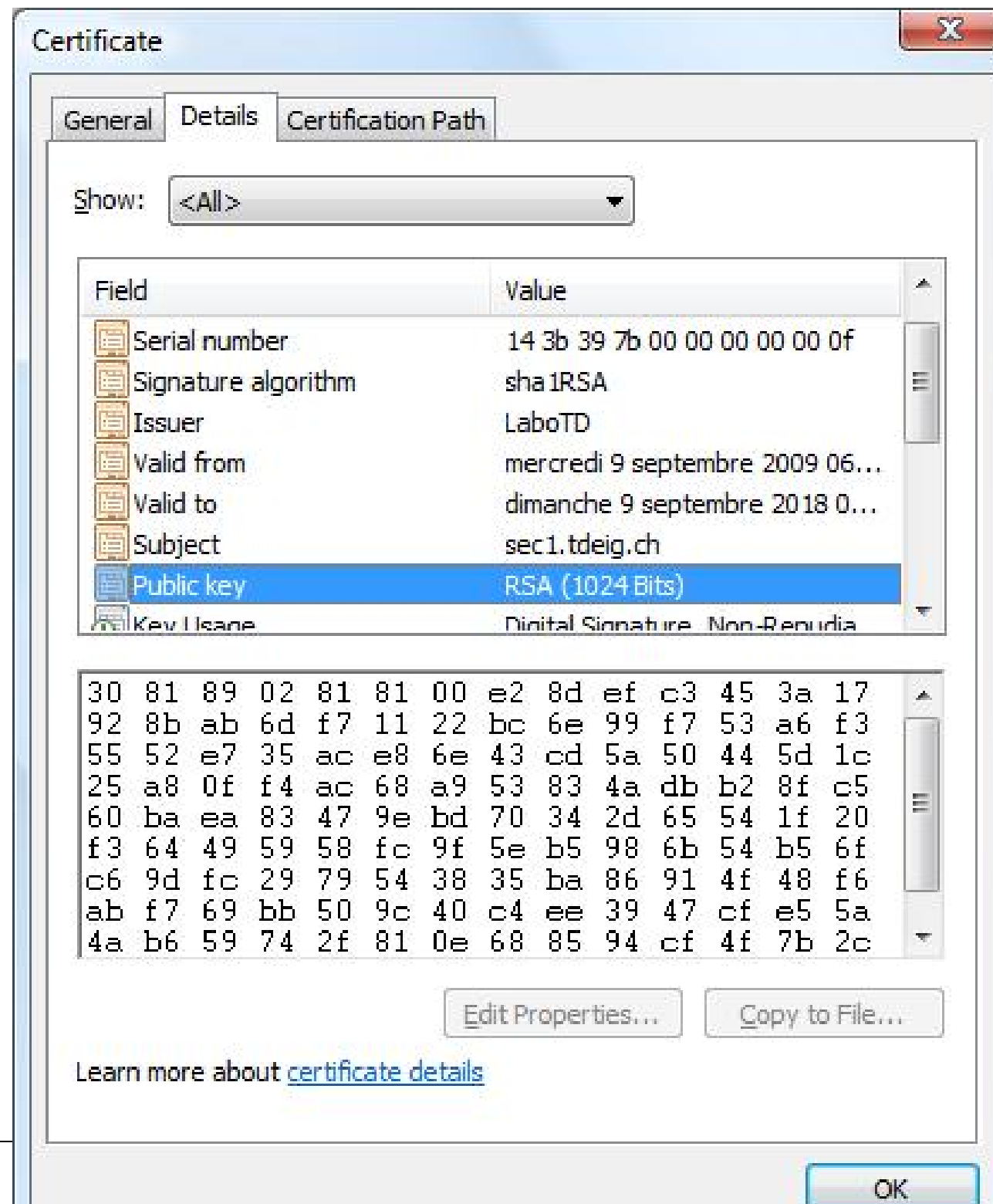
Valid from

Valid to

Subject = sec1.tdeig.ch

Public key = 3081 ...

RSA (1024 bits)



Certificat (3)

Onglet *Detail* (suite)

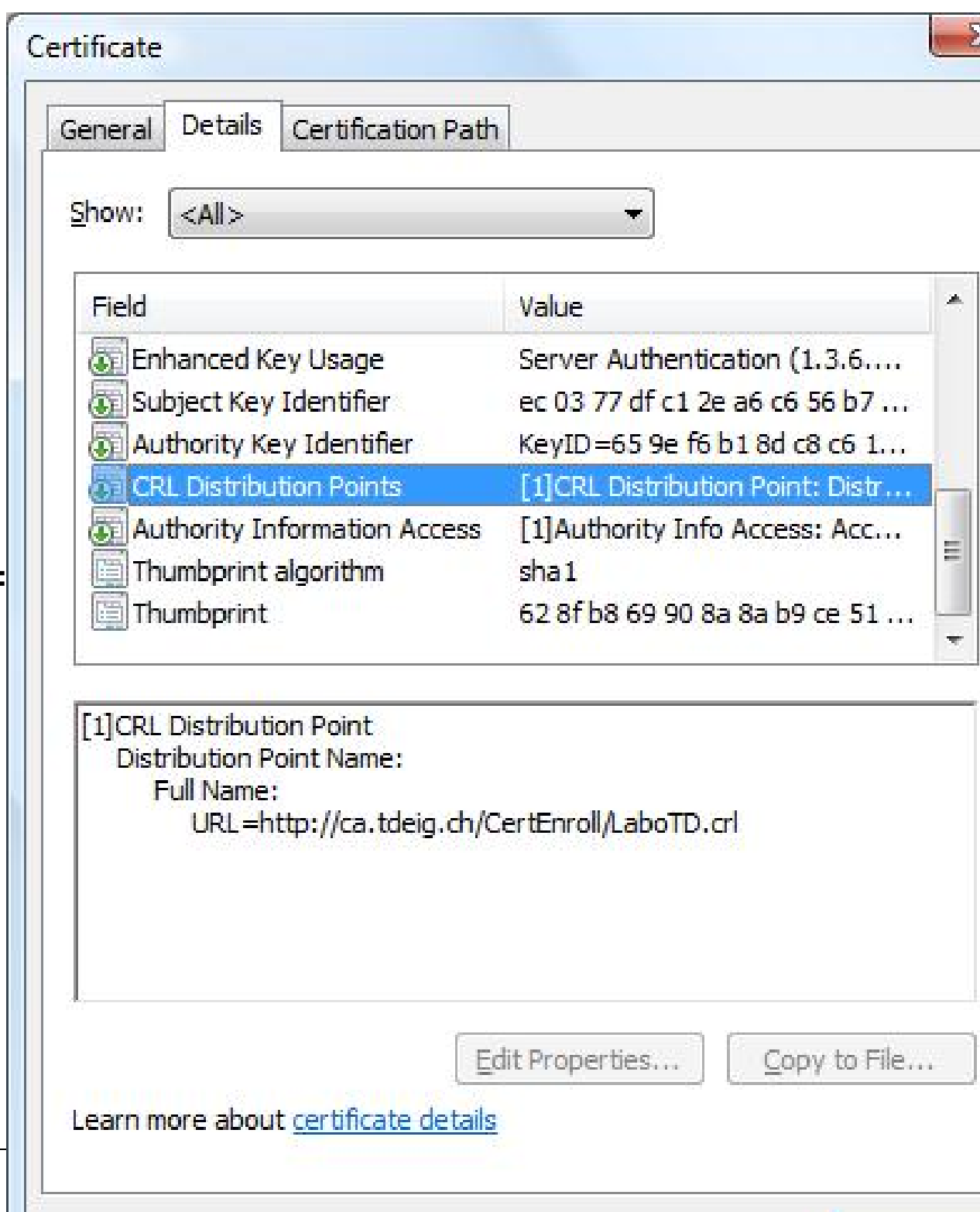
Utilité = Server Authentication
Enhanced Key Usage

Certificate Revocation List ... =
<http://.../CertEnroll/LaboTD.crl>

Thumbprint algorithm = sha1

Thumbprint = 62 8f ...

Thumbprint = signature



















Certificat numérique (résumé)

- Un certificat est un fichier d'un millier d'octets qui prouve un lien entre une **entité** et sa **clé publique**
- **Entité** = individu, *hardware (router, server), software process (Java applet)*, fichier, ...

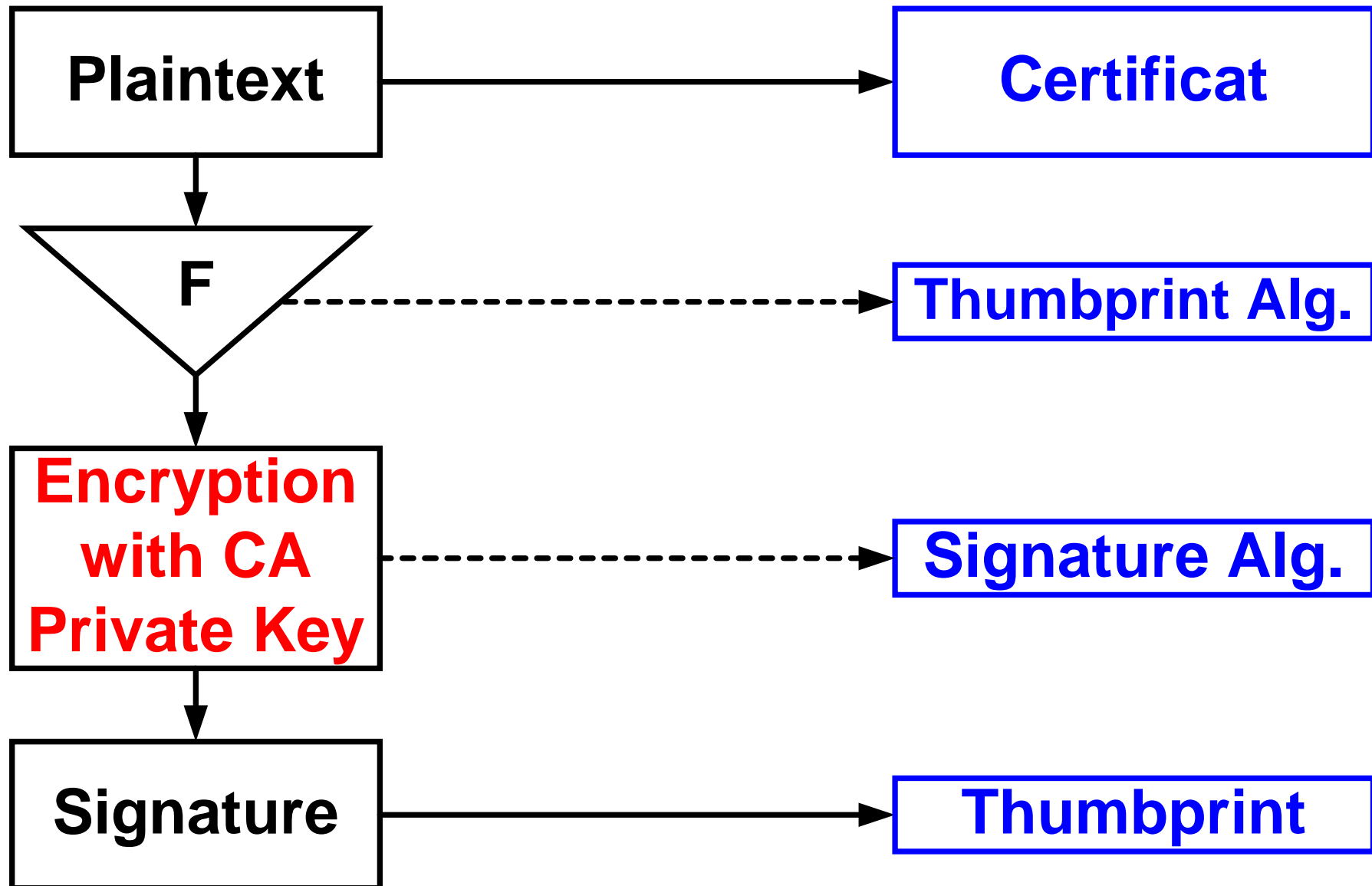
- Principaux champs obligatoires facultatifs selon X.509 (UIT)

- **Signé**

	Version
	Serial number
	Signature algorithm
	Issuer
	Valid from
	Valid to
	Subject
	Public key

	Subject Key Identifier
	Enhanced Key Usage
	Authority Key Identifier
	CRL Distribution Points
	Authority Information Ac
	Key Usage
	Thumbprint algorithm
	Thumbprint

Terminologie utilisée par Microsoft (labo)



Exercices → Solutions (slides 39 - 43)

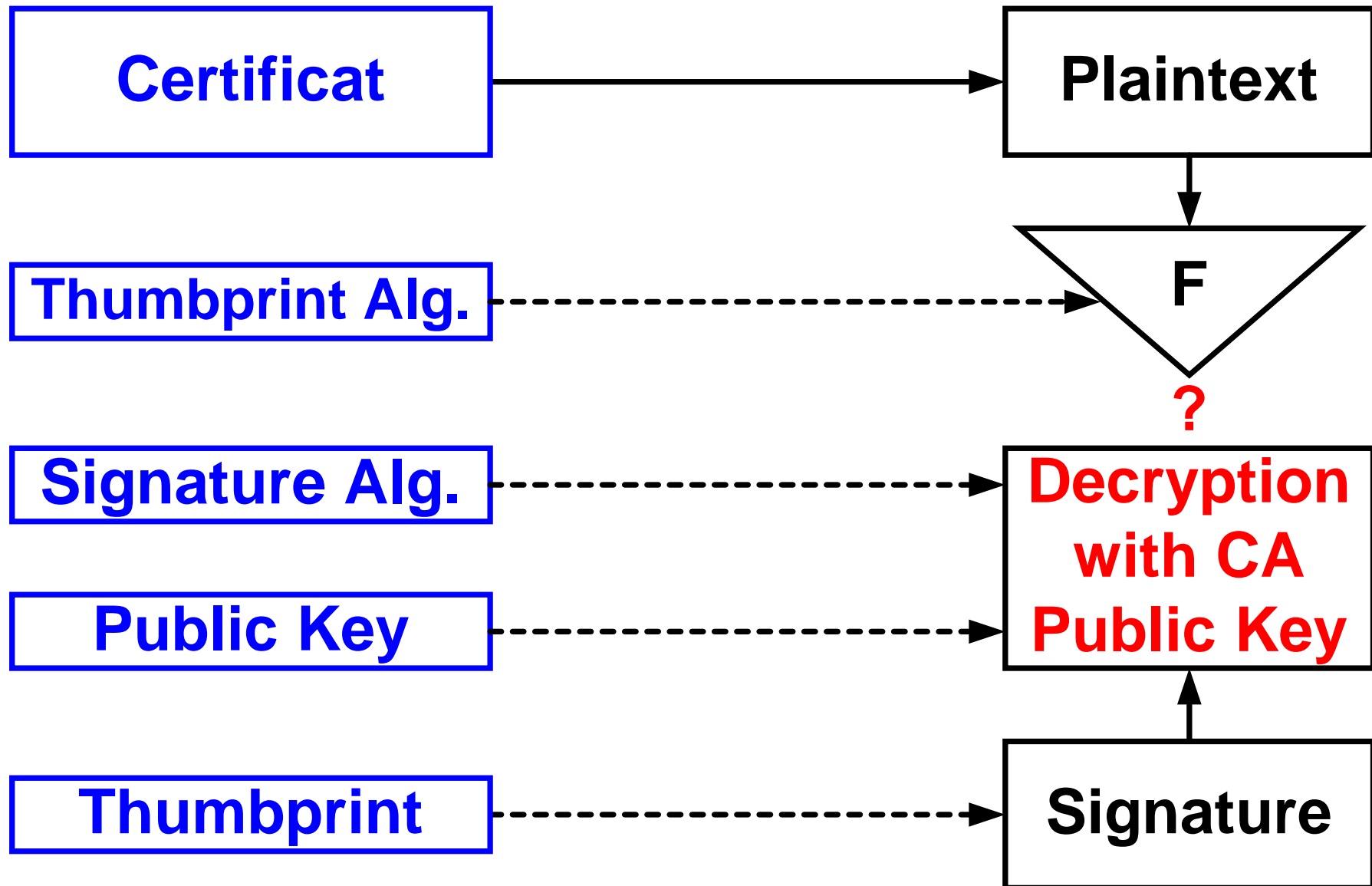
- Ex1 : Que faut-il faire pour supprimer le message d'erreur du slide 6 ?
- Ex2 : Quelles sont les commandes openssl pour produire un faux certificat LaboTD ?
- Ex3 : Déterminer les principaux tests effectués par mon navigateur lorsque je me connecte au site <https://sec1.tdeig.ch>
- Ex4 : Quel est l'intérêt d'avoir une arborescence de plusieurs étages dans la chaîne de certification <https://www.post.ch/> ?
- Ex5 : Peut-on, à partir du certificat sec1.tdeig.ch, signer d'autres certificats ?



Chaîne de certification

- Tous les navigateurs possèdent les certificats des principales autorités de certification dans le but **d'éviter de devoir les télécharger**
 - [IE : Tools - Internet Options - Content - Certificates – Trusted Root CA \(démonstration\)](#)
- **Tous les certificats Root sont autosignés !**
- Télécharger un certificat autosigné constitue toujours un danger (**imposture**)
- Il est conseillé de **restreindre** cette liste de *Trusted Root CA* aux autorités auxquelles vous voulez faire confiance
- **Il est facile de créer un faux certificat root** → [Annexe1](#) du labo

Contrôle d'intégrité d'un certificat



Durée de vie des certificats

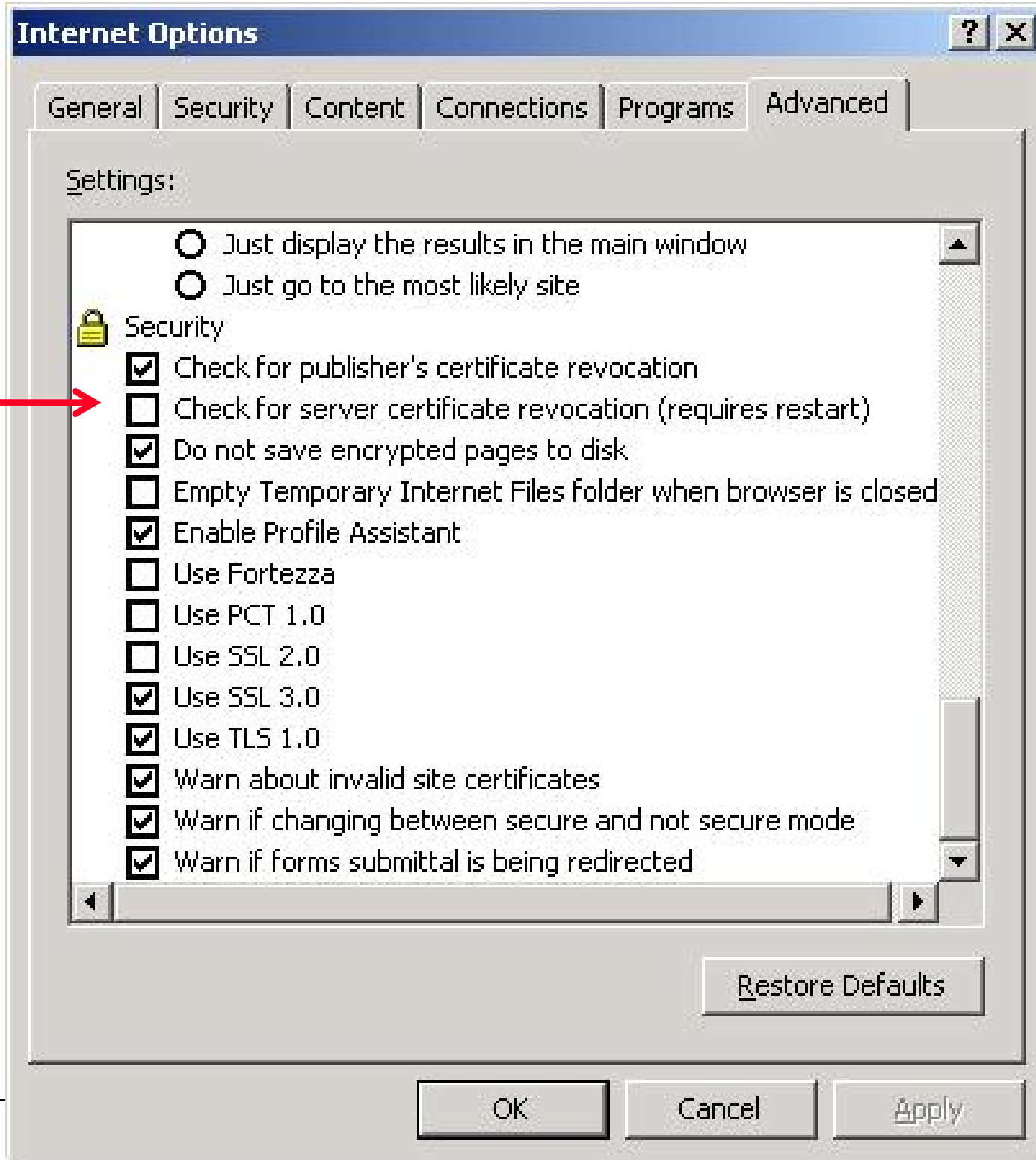
- Chaque certificat possède une **date d'expiration**
- Durant sa période de validité, le certificat peut être annulé (révoqué)
- La CA met à disposition une liste des certificats (*Certificate Database*) → slide 4

Application va chercher la liste des certificats révoqués à partir de l'URL indiqué dans le certificat → slide 9

URL=<http://ca.tdeig.ch/CertEnroll/LaboTD.crl>

Utilité du champ *Serial number* pour identifier un certificat révoqué

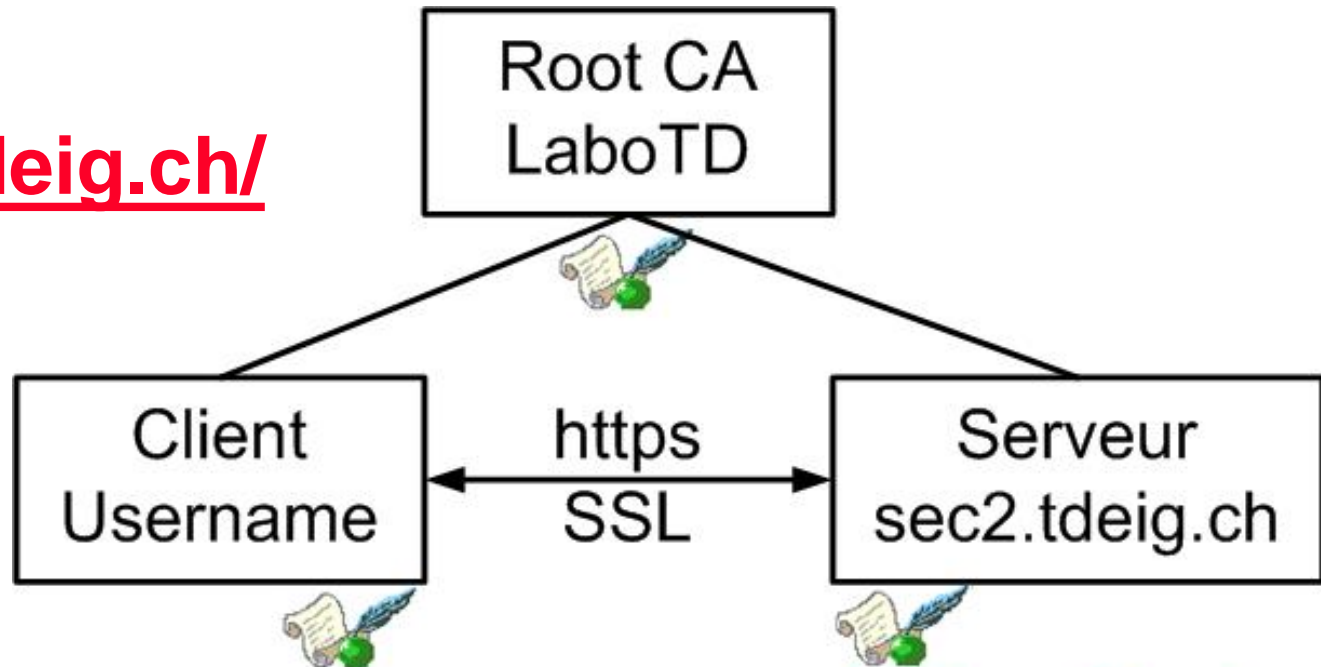
Contrôle de révocation avec IE



Authentification du client

Démo2

<http://ca.tdeig.ch/>



Authentification
du client (option)

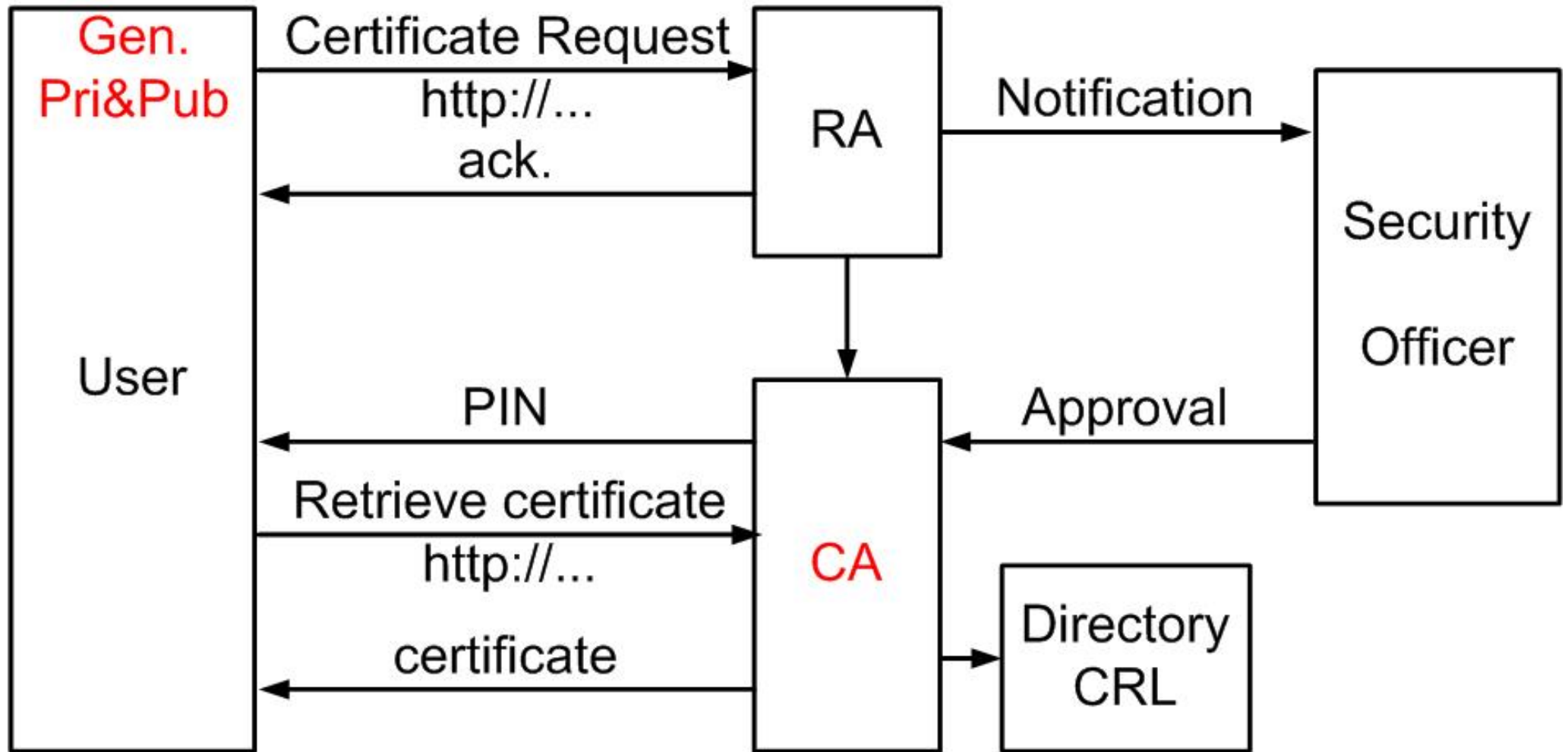
Démo3

<https://sec2.tdeig.ch:444/>

Utilité des certificats, d'une PKI

- Authentification du serveur (SSL)
Authentification du client (option SSL)
Portail web, e-commerce
- Authentification *smart card logon, 802.1x (switch)*
- Messagerie sécurisée (e-mails signés, chiffrés)
- Transactions électroniques (non-répudiation, ...)
- Exécutables signés (*update, driver, service pack, ActiveX*)
- *Encryption File System (Windows)*
- IPSec, Vista, ...

Composants du PKI (1)



- Aspects techniques (ce cours) & **organisationnels (procédures)**

Composants du PKI (2)

- *Registration Authority (RA)*

Contrôle physique de la personne (banques, postes),
Enregistrement effectué par chambres de commerce, ...

- *Certificate Authority (CA)*

Private CA : PKI d'entreprise → externaliser

Public CA : Verisign ... voir *browser list* → quel niveau confiance ?

Délivrer, renouveler un certificat

Bloquer → *Certificate Revocation List (CRL)*

- *Security Officer*

Habilité à approuver la demande

Composants du PKI (3)

- *Certification Practice Statement (CPS)*

Ce document légal détermine le niveau réel de sécurité

Verisign level 1 : adresse e-mail, ...

Exemple <https://ca.cern.ch/ca/crl/policy/cp-cps.pdf>

Voir [Annexe2](#) du labo

- **Validité du certificat** ... (talon d'Achille du PKI)

Application va chercher la liste de révocation (CRL)

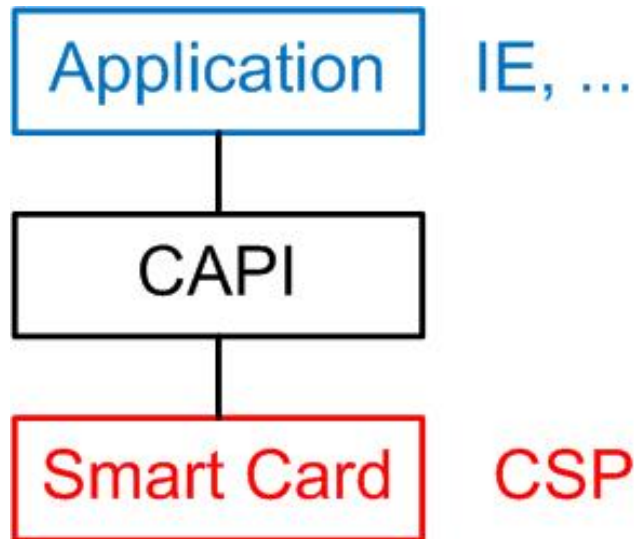
Taille du CRL, où est-il ?, transfert périodique (24h), ...

- Besoin d'effectuer ce contrôle avec un accès *online* à la CA avec le protocole OCSP (*Online Certificate Status Protocol*)

Mécanismes du PKI

- *Registration* Enregistrer la demande
Respecter la procédure de validation
- *Generation* Générer les clés publique et privée
Sont-elles aléatoires ?
- *Certification* Délivrer, renouveler, bloquer un certificat
- *Directory* Publier les certificats (annuaire), la CRL
- *Usage* Où sont stockés les certificats ?
Où est stockée la clé privée ?

MS Crypto Service Provider



Fonctions supportées ?
Stocker
Génération
Signature
Chiffrement

- CSP (software only) fourni par MS (RSA) fourni par entreprise tierce (Schlumberger, ...)

Démo4

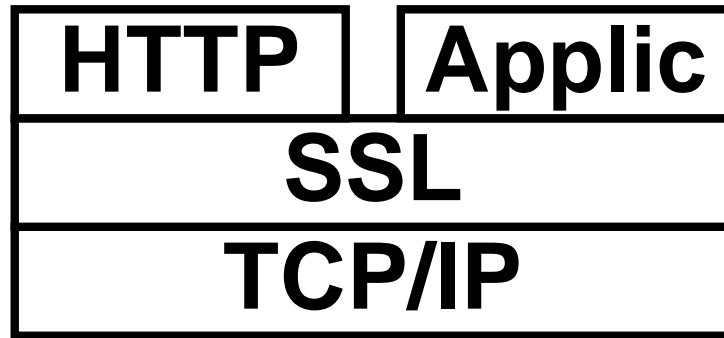
- CSP (software + **hardware**) fonctions minimales (mémoriser la clé privée, ...) fonctions évoluées avec cryptoprocasseur (générer les clés, signer, ...)



- La clé privée est-elle bien protégée ?

Voir Annexe3 du labo

Secure Socket Layer (SSL)



- Protège toute application : HTTP (80) → HTTPS (443), ...

- Chiffrement → confidentialité

- Signature → intégrité

- Authentification du serveur

- Authentification du client (option)

- SSL (Netscape) v1 (1994), v2 (1994), v3 (1995)

TLS (Transport Layer Security) = RFC 2246 (1999)

**Secure if endpoints =
Alice & Bob**

(no Man in the Middle)

→ Authentication

- Echanges en clair
 - *Client Hello* *client.random (32 bytes)*
 - ← *Server Hello* *server.random (32 bytes)*
 - ← *Certificate*
- Echange **chiffré avec la clé publique** du serveur
 - *Client Key Exchange* *premaster secret (48 bytes)*
 - *Change Cipher Spec* en clair
 - *Finished* **chiffré avec la clé sym.**
 - ← *Change Cipher Spec* en clair
 - ← *Finished* **chiffré avec la clé sym.**
- Données applicatives (http) **chiffrées**

Systeme hybride

- SSL utilise un système **hybride** pour des questions de performance
- Secret (*premaster secret*) chiffré avec la clé publique du serveur (RSA)
- Clés symétriques (chiffrement + signature) sont dérivées (calculées) de chaque côté
- Données utiles chiffrées avec des clés AES, ...



Détail (1)

→ **Client Hello** (*version*, 3.0
random, **client.random** (32 bytes)
session id, identificateur (32 bytes)
cipher list) algorithmes supportés

- *version* SSLv2 (non sûr), SSLv3, TLS
- *random* Grandeur aléatoire (32 bytes) produite par le client
- *session id* Ce champ n'est normalement pas présent
Il est envoyé si le client demande de réutiliser les clés symétriques d'une session précédente
- NB : le paquet est transmis en clair

Détail (2)

- *cipher list* Algorithmes (*cipher suites*) supportés par le client

Chiffrement	– Hachage (signature)
AES_256	– SHA1
AES_128	– SHA1
RC4_128	– SHA1
RC4_128	– MD5
DES_56	– MD5
EXP_40 – DES	– MD5

- Principe de sécurité : utiliser la longueur de clé la plus grande pour le chiffrement ainsi que la fonction de hachage qui produit le condensé le plus long (MD5 → 128 bit, SHA1 → 160 bit, ...)

Détail (3)

← **Server Hello** (*version*, 3.0
random, **server.random** (32 bytes)
session id, identificateur (32 bytes)
cipher,

- *version* SSLv2 (non sûr), SSLv3, TLS imposée par le serveur
- *random* Grandeur aléatoire (32 bytes) produite par le serveur
- *session id* Ce champ est présent si le serveur accepte de réutiliser les clés symétriques d'une session précédente
- *cipher* Choix imposé par le serveur (AES_128 – SHA1)

Détail (4)

← Certificate

- Chaque serveur SSL doit transmettre son certificat
- Situer ces 3 cas dans la solution Ex3



Détail (5)

→ **Client Key Exchange** (*premaster secret 48 bytes*)

- **Paquet chiffré avec la clé publique du serveur**

- *premaster secret* Grandeur aléatoire (48 bytes) produite par le client

- Client et serveur peuvent calculer (dériver) les différentes clés

- *Key Block* = PRF (*premaster secret, server.random, client.random*)

- PRF = *Pseudo Random Function* = P_MD5 () XOR P_SHA_1
(expansion)

Détail (6)

- Illustration pour AES_128 - SHA1 → AES (128 bit) & SHA1 (160 bit)

- Clés de signature

Client Write MAC secret = Key Block [0..19] 160 bit → 20 byte
Server Write MAC secret = Key Block [20..39] 160 bit → 20 byte

- Clés de chiffrement

Client Write key = Key Block [40..55] 128 bit → 16 byte
Server Write key = Key Block [56..71] 128 bit → 16 byte

- Key Block =

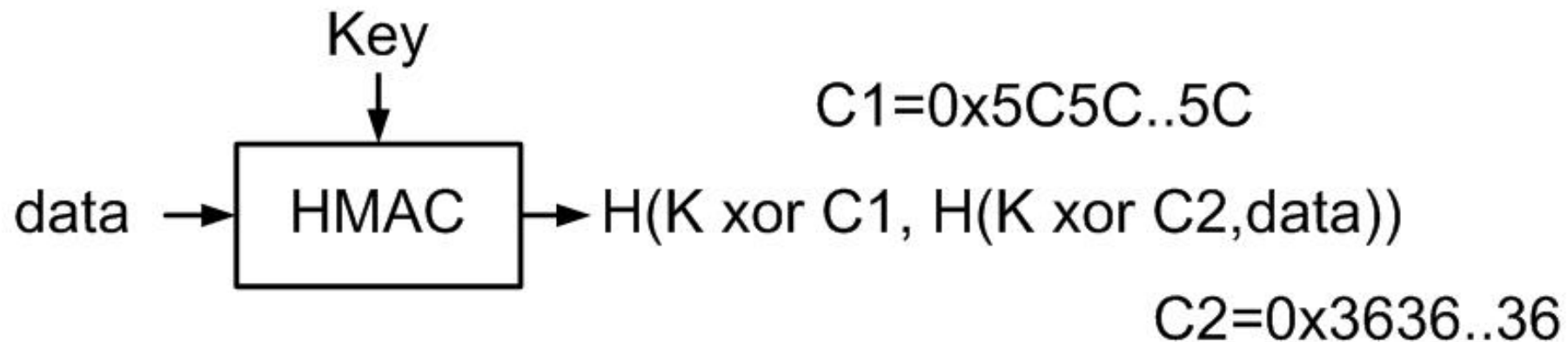
PRF (*premaster secret, server.random, client.random*) [0..71]

Message Authentication Code (MAC)

- SSL utilise une signature **basée sur un secret partagé** pour le contrôle d'intégrité pour chaque sens

C → S Client Write MAC secret

C ← S Server Write MAC secret



- **Signature → intégrité & authenticité du message !**
- **Signature basée sur un secret partagé → pas de non-répudiation**

TLS protocol (RFC 2246) page 30

Client

Server

ClientHello

----->

ServerHello

Certificate

ServerHelloDone

<-----

ClientKeyExchange

ChangeCipherSpec

en clair

Finished

----->

ChangeCipherSpec

Finished

en clair

<-----

Application Data

<----->

Application Data

- Chiffrement asymétrique / Chiffrement symétrique

Authentication du client est optionnelle

Client

Server

ClientHello

----->

ServerHello

Certificate

CertificateRequest

ServerHelloDone

<-----

Certificate

ClientKeyExchange

CertificateVerify

string signed with private key

ChangeCipherSpec

----->

- Le client doit prouver être en possession de la bonne clé privée
- On parle d'authentification mutuelle

Exercices → *Solutions* (slide 43)

- Ex6 : Que se passe-t-il si Charly mémorise tous les échanges SSL entre Alice et Bob et finit par trouver la clé privée ?

- Ex7 : Comment peut-on diminuer le risque de se faire voler la clé privée ?

Sécurité de la clé privée

- **Paire générée** de préférence **localement** (client - serveur)
Paire de clés aléatoire ?
- La clé privée du serveur est généralement en **RAM**
Risque de se la faire voler !
- *Hardware Security Module (HSM)* [Voir Annexe4](#) du labo
Module matériel pour serveur (analogue à une clé USB)
capable d'effectuer les calculs cryptographiques
- *Dual Key* pour chiffrer et signer
Sauvegarde si chiffrement
Pas de sauvegarde pour la signature (non-répudiation)

Annexes, liens et livre

- [Annexe5](#) = Config de CA LaboTD basée sur WindowsServer
- [Annexe6](#) = Config des serveurs SSL secx.tdeig.ch (Ubuntu)
- [Annexe7](#) = Ancien Labo OpenSSL

- [Annexe8](#) = Excellente présentation de Franck Davy (HSC)
http://www.hsc.fr/ressources/cours/pki/01_bases_crypto.pdf

- http://fr.wikipedia.org/wiki/Transport_Layer_Security

- RFC 2246 *The TLS Protocol Version 1.0*

- *SSL & TLS Building and Designing Secure Systems*
ISBN 0201615983 *Rescorla*

Solutions

- Ex1 : Que faut-il faire pour supprimer le message d'erreur du slide 6 ?

Faire confiance à la CA = LaboTD

→ télécharger certificat Root depuis <http://ca.tdeig.ch/certsrv/certcarc.asp>

- Ex2 : Quelles sont les commandes openssl pour produire un faux certificat LaboTD ?

Voir [Annexe1](#) du labo

Solutions

- Ex3 : Déterminer les principaux tests effectués par mon navigateur lorsque je me connecte au site <https://sec1.tdeig.ch>
 - 1 Ce certificat est-il délivré par une CA dont je fais confiance ?
 - 2 Intégrité du certificat (qui est signé)
 - 3 Format X.509 du certificat (champs obligatoires & facultatifs)
 - 4 Période de validité
 - 5 Certificat révoqué (annulé) → *Certificate Revocation List (CRL)*
 - 6 URL (sec1.tdeig.ch) = FQDN du certificat ?
 - authentification du serveur (par le client)

Solutions

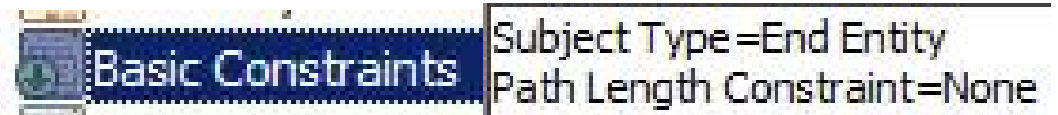
- Ex4 : Quel est l'intérêt d'avoir une arborescence de plusieurs étages dans la chaîne de certification <https://www.laposte.ch/> ?



- Limiter les risques que Charly obtienne la clé privée de la CA
- Ne pas offrir de connexion *online* à cette CA
- Créer un étage intermédiaire qui supporte les accès *online*

Solutions

- Ex5 : Peut-on, à partir du certificat sec1.tdeig.ch, signer d'autres certificats ?
- **Oui techniquement mais à éviter**



Solutions

- Ex6 : Que se passe-t-il si Charly mémorise tous les échanges SSL entre Alice et Bob et finit par trouver la clé privée ?

En disposant de la clé privée du serveur SSL, Charly pourra déchiffrer le *premaster secret* (slide 31) puis calculer toutes les clés du slide 32

→ **Charly peut déchiffrer tous les échanges !**

→ **plus besoin de décrypter !!!**

- Ex7 : Comment peut-on diminuer le risque de se faire voler la clé privée ?

En utilisant un dispositif matériel (USB, Hardware Security Module) qui garantit que la clé privée n'est pas accessible à Charly depuis le réseau