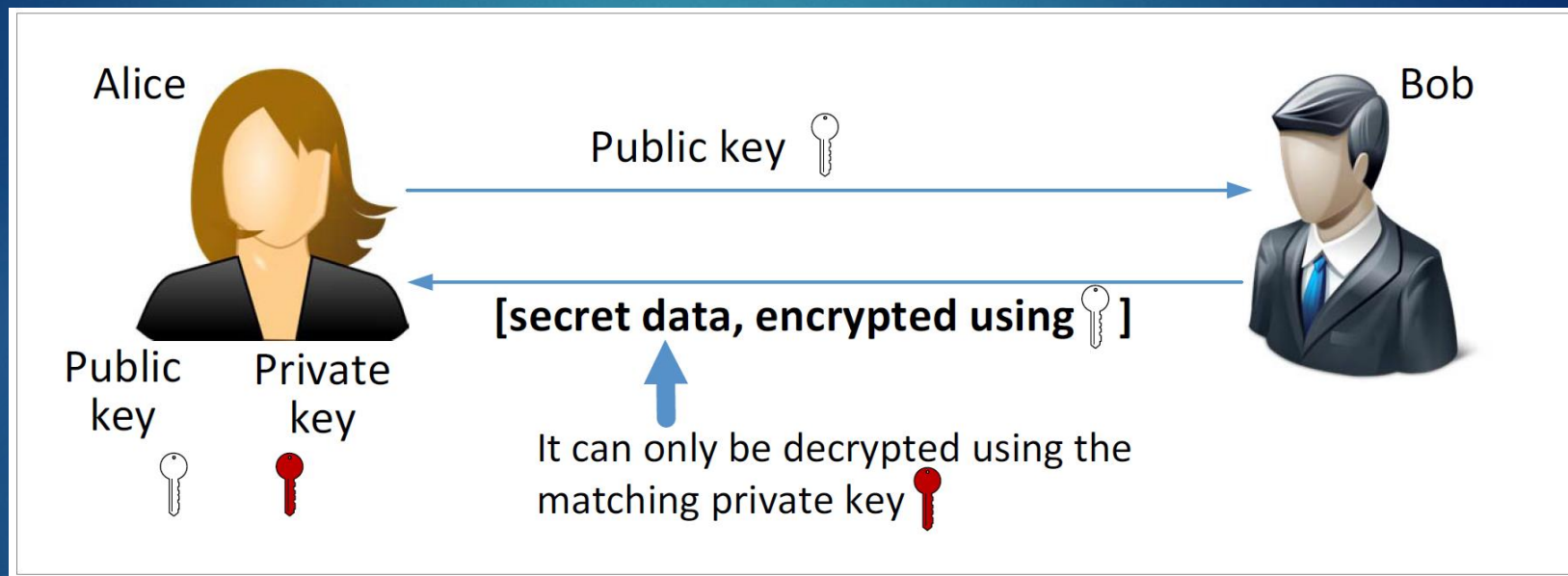


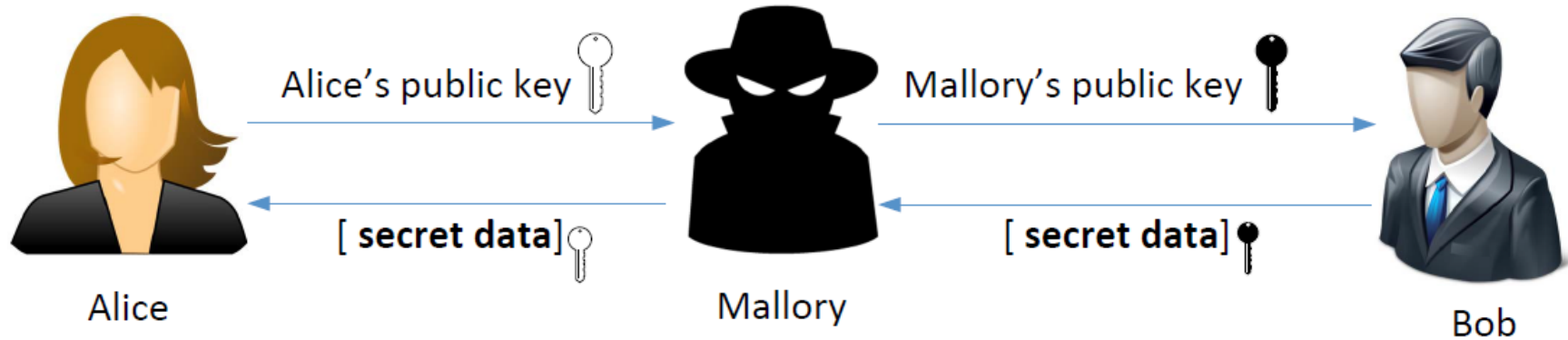
# Infrastructura de chei publice

# Comunicarea secretă folosind perechi cheie publice-cheie privată

2



# Atacul de tip om-la-mijloc (Man-in-the-Middle - MITM) poate apărea



# Care este problema fundamentală?

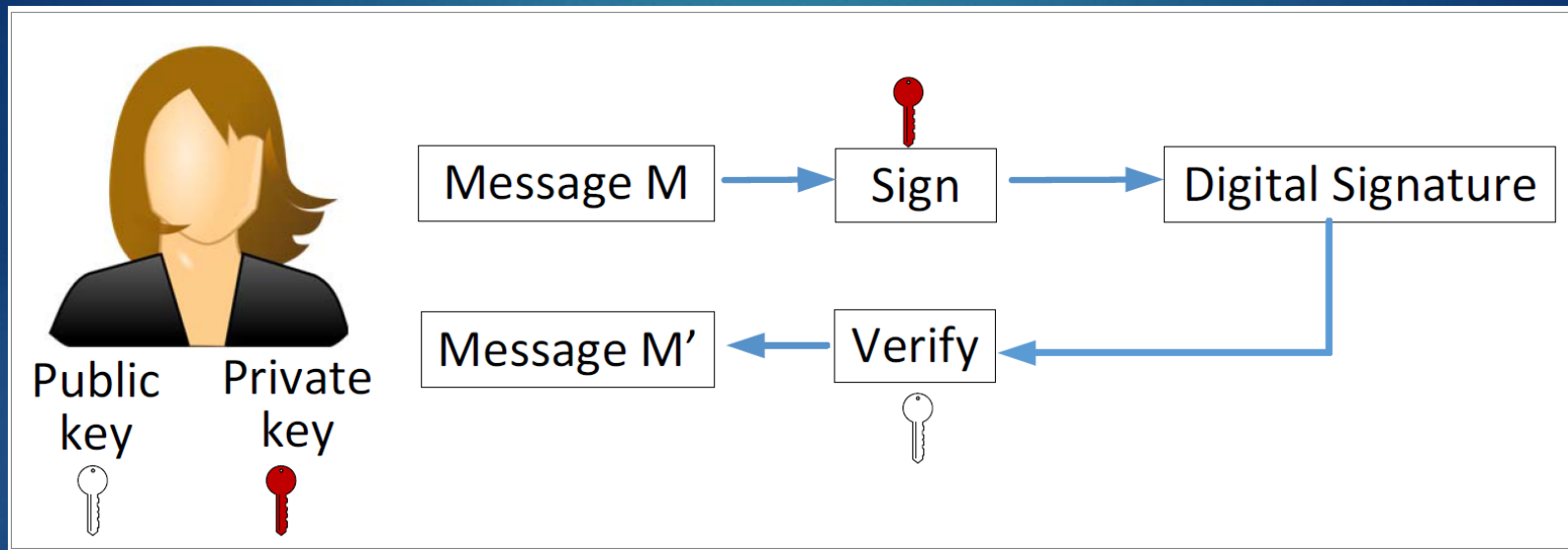
**Problema fundamentală:** Bob nu are cum să știe dacă cheia publică pe care a primit-o aparține lui Alice sau nu.

## Soluția:

- ▶ Să găsească o altă parte de încredere ca să verifice identitatea
- ▶ Să lege o identitate la o cheie publică dintr-un certificat
- ▶ Certificatul nu poate fi falsificat sau modificat (folosește semnătura digitală)

# Semnătura digitală

5



- Dacă nu a fost alterată semnătura,  $M'$  va fi identic cu  $M$
- Doar Alice poate semna (doar ea posedă cheia privată)
- Oricine poate verifica (cheia publică este știută de toți)

# Înfrângerea atacurilor MITM folosind semnătura digitală

- ▶ Alice trebuie să meargă la o **parte de încredere** pentru a obține un certificat.
- ▶ După verificarea identității lui Alice, partea de încredere emite un certificat cu numele lui Alice și cheia ei publică.
- ▶ Alice îi trimite întregul certificat lui Bob.
- ▶ Bob verifică certificatul folosind cheia publică a părții de încredere.
- ▶ Bob cunoaște acum cine este **adevăratul proprietar** al unei chei publice.

# Infrastructura de chei publice

7

- ▶ **Autoritatea de certificate (Certificate Authority - CA):** o parte de încredere, care răspunde de verificarea identității utilizatorilor și de legarea identității verificate de o cheie publică.
- ▶ **Certificat digital:** un document care certifică faptul că cheia publică pe care o conține aparține identității descrise în document.
  - ▶ standard X.509



# Certificat digital

8

- Să obținem certificatul lui paypal

```
$ openssl s_client -showcerts -connect www.paypal.com:443 </dev/null
```

```
-----BEGIN CERTIFICATE-----
MIIHWTCCBkGgAwIBAgIQLNQVEFQ30N5KOSAFavbCfzANBgkqhkiG9w0BAQsFADB3
MQswCQYDVQQGEwJVUzEdMBsGA1UEChMUU3ltYW50ZWNgQ29ycG9yYXRpb24xHzAd
... (omitted) ...
GN/QMQ3a55rjwNQnA3s2WWuHGPaE/jMG17iiL2O/hUdIvLE9+wA+fWrey5//74x1
NeQitYiySDIepHGnng==
-----END CERTIFICATE-----
```

- Salvăm datele de mai sus în fișierul `paypal.pem` și folosim comanda următoare pentru a-l decodifica (urmează)

```
$ openssl x509 -in paypal.pem -text -noout
```



# Exemplu de certificat X.509 (1)

Identitatea CA  
(Symantec)

Proprietarul  
certificatului  
(paypal)

Certificate:

Data:

Serial Number:

2c:d1:95:10:54:37:d0:de:4a:39:20:05:6a:f6:c2:7f

Signature Algorithm: sha256WithRSAEncryption

**Issuer:** C=US, O=Symantec Corporation, OU=Symantec Trust Network,  
CN=Symantec Class 3 EV SSL CA - G3

Validity

Not Before: Feb 2 00:00:00 2016 GMT

Not After : Oct 30 23:59:59 2017 GMT

**Subject:** 1.3.6.1.4.1.311.60.2.1.3=US/  
1.3.6.1.4.1.311.60.2.1.2=Delaware/  
businessCategory=Private Organization/  
serialNumber=3014267, C=US/  
postalCode=95131-2021, ST=California,  
L=San Jose/street=2211 N 1st St,  
O=PayPal, Inc., OU=CDN Support, CN=www.paypal.com

# Exemplu de certificat X.509 (1)

Cheia publică

**Subject Public Key Info:**

Public Key Algorithm: rsaEncryption

Public-Key: (2048 bit)

Modulus:

00:da:43:c8:b3:a6:33:5d:83:c0:63:14:47:fd:6b:22:bd:

bf:4e:a7:43:11:55:eb:20:8b:e4:61:13:ee:de:fe:c6:e2:

... (omitted) ...

7a:15:00:c5:01:69:b5:10:16:a5:85:f8:fd:07:84:9a:c9:

Exponent: 65537 (0x10001)

Semnătura CA

**Signature** Algorithm: sha256WithRSAEncryption

4b:a9:64:20:cc:77:0b:30:ab:69:50:d3:7f:de:dc:7c:e2:fb:93:84:fd:

78:a7:06:e8:14:03:99:c0:e4:4a:ef:c3:5d:15:2a:81:a1:b9:ff:dc:3a:

... (omitted) ...

fb:00:3e:7d:6a:de:cb:9f:ff:ef:8c:65:35:e4:22:b5:88:b2:48:32:1e:

## ► Verificarea subiectului

- Asigură că persoana care solicită un certificat fie deține, fie reprezintă identitatea din câmpul de subiect

## ► Semnarea certificatelor digitale

- CA generează o semnătură digitală pentru certificat folosindu-și cheia proprie
- O dată aplicată semnătura, certificatul nu mai poate fi modificat.
- Semnăturile pot fi verificate de oricine folosind cheia publică a CA.

# Cum să fii o autoritate de certificate

12

- ▶ Urmărim procesul
  - ▶ Cum emite CA un certificat
  - ▶ Cum se obține un certificat de la o CA
  - ▶ Cum să setăm un server de web folosind un certificat

# Setup pentru CA

13

- ▶ Numim CA **ModelCA**
- ▶ E nevoie să facem următoarele pentru setarea **ModelCA**:
  - ▶ Să generăm o pereche cheie publică/cheie privată
  - ▶ Să creăm un certificat X.509 (cine îl va semna?)
  - ▶ Presupunem că **ModelCA** este o CA rădăcină, așa că își va semna propriul certificat (self-signed certificate=certificat semnat de sine).
- ▶ Comanda următoare generează un certificat semnat de sine

```
$ openssl req -x509 -newkey rsa:4096 -sha256 -days 3650  
-keyout modelCA_key.pem -out modelCA_cert.pem
```

# Întrebare (și răspuns)

14

- ▶ **Întrebare** : Dacă certificatul lui ModelCA este semnat de sine, cum îl verificăm?
- ▶ **Răspuns**: Nu avem cum. Doar ne asigurăm că certificatul a fost obținut într-un mod în care avem încredere:
  - ▶ A venit cu sistemul de operare (dacă avem încredere în SO, atunci avem și în certificat.)
  - ▶ A venit cu software (dacă avem încredere în software, atunci avem și în certificat.)
  - ▶ L-am adăugat manual (dacă avem încredere în ce am decis, atunci avem și în certificat.)
  - ▶ Ne-a fost trimis de cineva în care nu avem încredere (nu avem încredere în certificat.)



# Obținerea unui certificat de la CA: Pasul 1

15

- Pasul 1: Generăm o pereche de chei publică/privată

Mărimea cheii RSA

```
$ openssl genrsa -aes128 -out bank_key.pem 2048
```

Criptăm fișierul de  
ieșire cu AES (128-bit)

Conține ambele chei

# Obținerea unui certificat de la CA: Pasul 2

16

- Pasul 2: Generăm o cerere de semnare de certificat (certificate signing request - CSR); e nevoie să furnizăm informație despre identitate

```
$ openssl req -new -key bank_key.pem -out bank.csr -sha256
```

```
$ openssl req -in bank.csr -text -noout
```

Certificate Request:

Data:

Version: 0 (0x0)

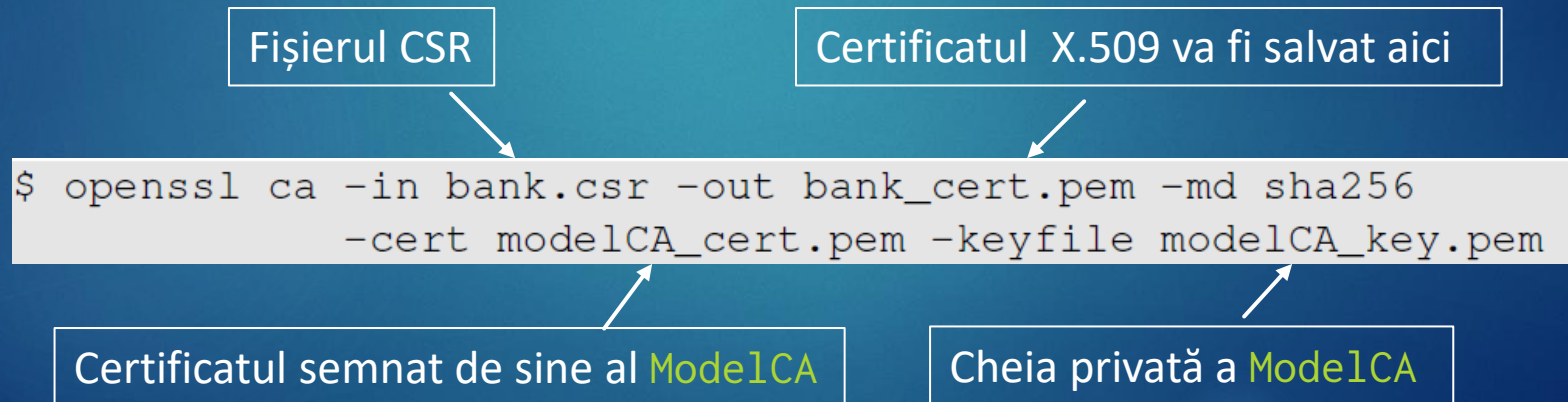
Subject: C=US, ST=New York, L=Syracuse, O=Example Inc,  
**CN=example.com**/emailAddress=email@example.com

CA va verifica această informație despre subiect

# CA: Emiterea unui certificat X.509

17

- ▶ Noi (bank) avem nevoie să trimitem fișierul CSR lui ModelCA.
- ▶ ModelCA va verifica faptul că noi suntem proprietarul real al identității (sau o putem reprezenta) specificate în fișierul CSR.
- ▶ Dacă verificarea are succes, ModelCA va emite un certificat



# Punerea certificatului în serverul de Web

18

- ▶ Vom folosi mai întâi serverul preconstruit în `openssl` pentru a seta un server de web HTTPS

```
$ cp bank.key bank.pem  
$ cat bank.crt >> bank.pem  
$ openssl s_server -cert bank.pem -accept 4433 -www
```

- ▶ Accesăm serverul folosind Firefox (<https://example.com:4433>).  
Primim mesajul de eroare următor. De ce?

example.com:4433 uses an invalid security certificate.

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

(Error code: sec error unknown issuer)

# Răspunsul la întrebarea anterioară

- ▶ Firefox are nevoie să folosească cheia publică a lui **ModelCA** pentru a verifica certificatul
- ▶ Firefox nu are certificatul cu cheia publică a lui **ModelCA**
- ▶ Putem adăuga manual certificatul lui **ModelCA** în Firefox

Alegem `Edit -> Preference -> Advanced -> View Certificates`

Import `ModelCA_cert.pem`

# Setarea Apache pentru HTTPS

20

- Adăugăm următoarea intrare de gazdă virtuală (VirtualHost) în fișierul de configurare al lui Apache:

```
<VirtualHost *:443>
    ServerName example.com
    DocumentRoot /var/www/Example
    DirectoryIndex index.html

    SSLEngine On
    SSLCertificateFile      /etc/apache2/ssl/bank_cert.pem ①
    SSLCertificateKeyFile   /etc/apache2/ssl/bank_key.pem  ②
</VirtualHost>
```

Certificatul  
serverului

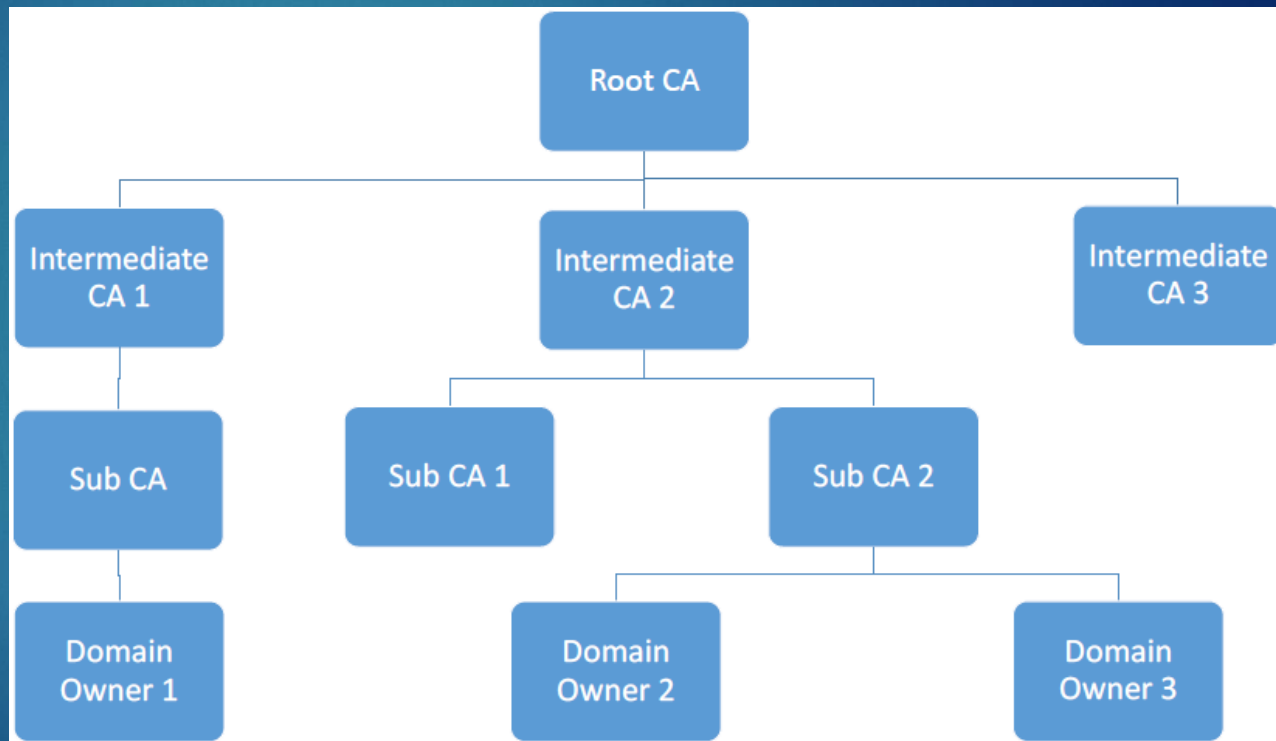
Cheia privată  
a serverului

Observație: Fișierul de configurare al lui Apache se află la:  
`/etc/apache2/sites-available/default-ssl.conf`



# Autorități de certificate rădăcină și intermediare

Există multe în lume și sunt organizate într-o structură ierarhică



# CA-uri rădăcină și certificate semnate de sine

22

- ▶ Cheia publică a unei CA rădăcină este și ea stocată într-un certificat X.509. Este semnată de sine.
- ▶ Certificate semnate de sine: intrările pentru emitent și subiect sunt identice

identice



```
Issuer: C=US, O=VeriSign, Inc., OU=VeriSign Trust Network,  
        OU=(c) 2006 VeriSign, Inc. - For authorized use only,  
        CN=VeriSign Class 3 Public Primary Certification Authority - G5  
Subject: C=US, O=VeriSign, Inc., OU=VeriSign Trust Network,  
        OU=(c) 2006 VeriSign, Inc. - For authorized use only,  
        CN=VeriSign Class 3 Public Primary Certification Authority - G5
```

- ▶ Cum putem avea încredere în ele?
  - ▶ Cheile publice ale CA-rilor rădăcină sunt preinstalate în SO, browser-e și alt software

# CA-uri intermediare și lanțul de încredere

23

```
$ openssl s_client -showcerts -connect www.paypal.com:443
```

```
Certificate chain
```

```
0 s: ... /CN=www.paypal.com
```

```
i: ... /CN=Symantec Class 3 EV SSL CA - G3
```

Certificatul lui Paypal

```
-----BEGIN CERTIFICATE-----
```

```
MIIHWTCCBkGgAwIBAgIQLNQVEFQ30N5KOSAFavbCfzANBgkqhkiG9w0BAQsFADB3
```

...

```
-----END CERTIFICATE-----
```

```
1 s: ... /CN=Symantec Class 3 EV SSL CA - G3
```

```
i: ... /CN=VeriSign Class 3 Public Primary Certification
```

```
Authority - G5
```

Certificatul CA intermediare

```
-----BEGIN CERTIFICATE-----
```

```
MIIFKzCCBBOgAwIBAgIQfuFKb2/v8tN/P61lTTratDANBgkqhkiG9w0BAQsFADCB
```

...

```
-----END CERTIFICATE-----
```

A este  
folosit  
pentru a  
verifica B

A

E nevoie de altceva pentru a verifica A (un certificat de la alt CA intermediar sau CA rădăcină)

# Verificarea manuală a lanțului de certificate

24

- ▶ `paypal.pem`: Salvăm certificatul lui Paypal într-un fișier astfel numit
- ▶ `Symantec-g3.pem`: Salvăm certificatul de la “Symantec Class 3 EV SSL CA – G3”
- ▶ `VeriSign-G5.pem`: Salvăm certificatul lui VeriSign-G5 din browser

Certificatul CA rădăcină



```
$ openssl verify -verbose -CAfile VeriSign-G5.pem  
-untrusted Symantec-G3.pem Paypal.pem  
Paypal.pem: OK
```



Lanțul de certificate

# Crearea certificatelor pentru CA-uri intermediare

- La generarea unui certificat pentru un CA intermediar trebuie să facem ceva anume:

```
$ openssl ca -in modelIntCA.csr -out modelIntCA_cert.pem -md sha256  
-cert modelCA_cert.pem -keyfile modelCA_key.pem  
-extensions v3_ca
```

- Câmpul de extensie din certificat va arata astfel"

```
X509v3 extensions:  
X509v3 Basic Constraints:  
CA:TRUE
```

**TRUE** înseamnă că certificatul se poate folosi pentru a verifica alte certificate, adică proprietarul este o CA. Pentru certificatele non-CA, acest câmp este **FALSE**.

- ▶ Serverul răspunde de trimiterea tuturor certificatelor CA-urilor intermediare necesare pentru verificarea propriului certificat.
- ▶ În Apache, toate certificatele, inclusiv cele ale CA-urilor intermediare sunt puse în fișierul precizat în directivă.

```
<VirtualHost *:443>
    ServerName bank32.com
    DocumentRoot /var/www/html
    DirectoryIndex index.html

    SSLEngine On
    SSLCertificateFile      /home/seed/cert/bank_cert2.pem
    SSLCertificateKeyFile   /home/seed/cert/bank_key.pem
</VirtualHost>
```



# Restartăm Apache

```
// Test the Apache configuration file for errors.  
$ sudo apachectl configtest  
// Enable SSL  
$ sudo a2enmod ssl  
// Enable the sites specified in default-ssl  
$ sudo a2ensite default-ssl  
// Restart Apache  
$ sudo service apache2 restart
```

# CA-uri de încredere în lumea reală

28

- ▶ Nu toate CA-urile de încredere sunt prezente în toate browser-ele.
- ▶ Potrivit W3Techs în martie 2020, IdenTrust avea cea mai mare parte din piață, urmat de DigiCert Group, Sectigo, GoDaddy Group și GlobalSign.
- ▶ Lista CA-urilor suportate de browser poate fi aflată:
  - ▶ **Pentru Chrome:**
    - ▶ Settings -> Show advanced settings -> Manage Certificates
  - ▶ **Pentru Firefox:**
    - ▶ Edit -> Preferences -> Advanced -> Certificates -> View Certificates -> Certificate Manager -> Authorities

# Cum înfrânge PKI atacul MITM

29

- ▶ Presupunem că Alice vrea să viziteze <https://example.com>
- ▶ Atunci când serverul își trimite cheia publică lui Alice, atacatorul interceptează comunicația și face următoarele:
  - Îi înaintează certificatul autentic de la [example.com](https://example.com)
  - Creează un certificat fals
  - Îi trimite propriul certificat lui Alice

# Atacatorul înaintează certificatul autentic

30

- ▶ Atacatorul (Mike) înaintează certificatul autentic
- ▶ Alice trimite serverului un **secret**, cifrat cu cheia publică.
- ▶ **Secretul** este folosit la stabilirea unui canal cifrat între Alice și server
- ▶ Mike nu cunoaște cheia privată corespunzătoare, așa că nu poate afla **secretul**.
- ▶ Tot ce poate face Mike comunicației este să provoace refuzul servirii.
- ▶ **Atacul MITM eșuează.**

# Atacatorul creează un certificat fals

31

- ▶ Atacatorul (Mike) creează un certificat fraudulos pentru domeniul `example.com`.
  - ▶ Mike înlocuiește cheia publică a serverului cu propria cheie publică.
  - ▶ CA-urile de încredere nu vor semna cererea de certificat a lui Mike deoarece el nu deține `example.com`.
  - ▶ Mike își poate semna el însuși certificatul (self-signed certificate).
  - ▶ Browser-ul lui Alice nu va găsi nici un certificat de încredere care să verifice certificatul recepționat și va da următorul avertisment:
- ```
example.com uses an invalid security certificate.  
The certificate is not trusted because it is self-signed.
```
- ▶ **Atacul MITM eșuează** dacă utilizatorul decide să termine conexiunea



# Atacatorul trimite propriul certificat

32



- ▶ Certificatul atacatorului este valid.
- ▶ Browser-ul verifică dacă identitatea din câmpul de subiect se potrivește cu ce dorește Alice.
  - ▶ Există o nepotrivire: **attacker.com**  $\neq$  **example.com**
- ▶ Browser-ul termină protocolul de handshake: **MITM eșuează**



# Emularea unui atac MITM

33

- ▶ Un atac asupra DNS este o cale tipică pentru a realiza MITM
  - ▶ Emulăm atacul asupra DNS schimbând manual fișierul `/etc/hosts` de pe mașina utilizatorului ca să mapăm `example.com` la adresa IP a mașinii atacatorului.
- ▶ Pe mașina atacatorului vom găzdui un sit de web pentru `example.com`.
  - ▶ Folosim certificatul X.509 al atacatorului pentru a seta serverul. Câmpul `Common name` al certificatului conține **attacker32.com**

▶ `example.com` uses an invalid security certificate.  
The certificate is only valid for **attacker32.com**  
(Error code: `ssl_error_bad_cert_domain`)

# Importanța verificării câmpului Common Name

34

- ▶ În timpul TLS/SSL handshake browser-ele realizează două validări importante
  - 1) Verifică dacă certificatul primit este sau nu valid.
  - 2) Verifică dacă subiectul (Common Name) din certificat este același ca hostname al serverului.
- ▶ Lipsa verificării Common Name este o greșeală frecventă în software

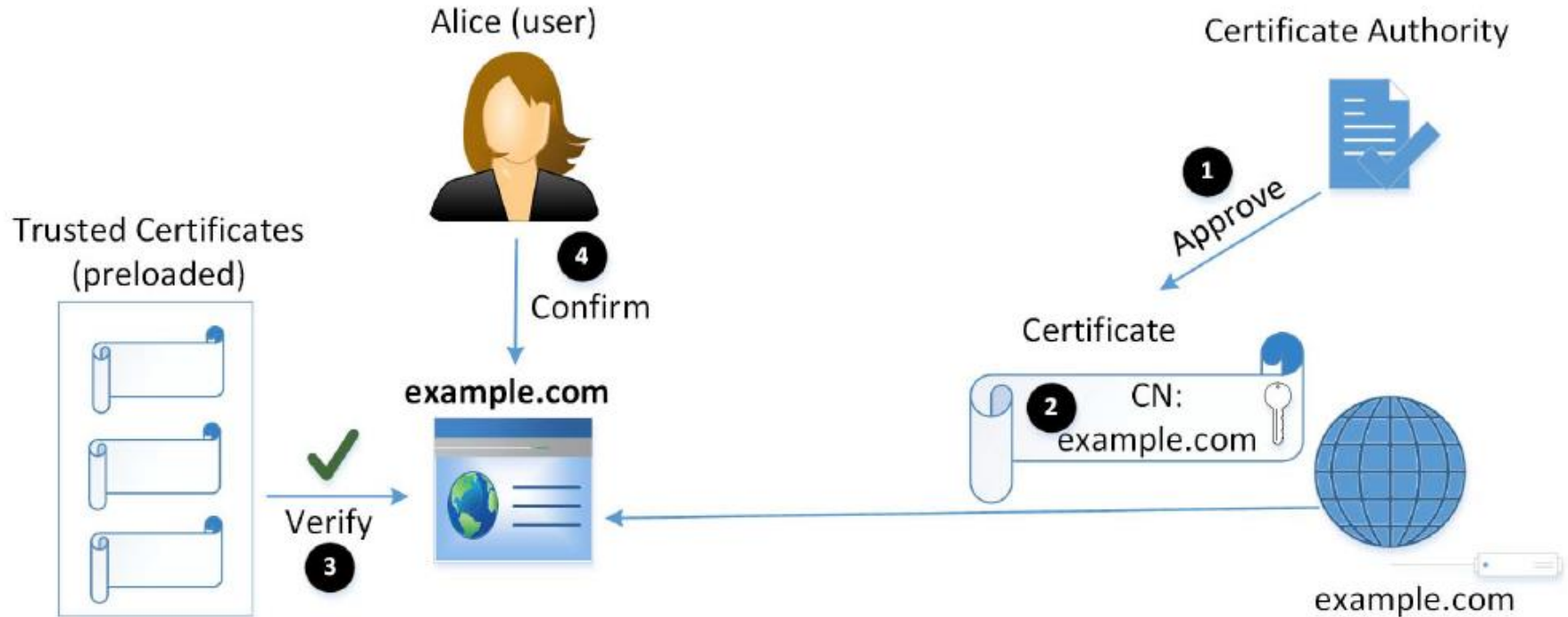
# Man-In-The-Middle Proxy

35

- ▶ Proxy creează un certificat CA semnat de sine care este instalat în browser-ul utilizatorului
- ▶ Se configurează rutarea pe mașina utilizatorului; tot traficul HTTPS care iese este direcționat spre mașina proxy
- ▶ Când un utilizator încearcă să viziteze un sit HTTPS:
  - ▶ Proxy interceptează comunicația
  - ▶ Creează un certificat fals
  - ▶ Browser-ul are deja certificatul proxy în lista sa de încredere și poate verifica toate certificatele false
  - ▶ Proxy devine MITM

# Suprafețe de atac asupra PKI

36



# Atacul asupra procesului de verificare al CA

37

## ► Sarcina CA are două părți:

- Verificarea relației între solicitantul certificatului și informația de subiect din certificat
- Punerea semnăturii digitale pe certificat

## ► Studiu de caz: Comodo Breach [Martie 2011]

- Procesul de aprobare din Europa de sud a fost compromis.
- S-au emis nouă certificate pentru șapte domenii și de aici atacatorul a putut furniza o atestare falsă.
- Unul dintre domeniile afectate (domeniu cheie pentru Firefox browser):  
addons.mozilla.org

# Atacul asupra procesului de semnare al CA

38

- ▶ Dacă este compromisă cheia privată a CA, atacatorii pot semna certificate cu câmpuri de subiect arbitrare
- ▶ **Studiu de caz: the DigiNotar Breach [iunie-iulie 2011]**
  - ▶ DigiNotar: CA comercial de top
  - ▶ Atacatorul a obținut cheia privată a DigiNotar
  - ▶ Au fost emise 531 certificate fără drept.
  - ▶ A fost interceptat trafic destinat subdomeniilor Google: atac MITM.
- ▶ Cum își protejează CA-urile cheia privată
  - ▶ Hardware Security Model (HSM): capabil să genereze și să stocheze chei criptografice; trebuie accesat fizic; nu se poate altera; stocat în seif păzit, asigurat cu pază



# Atacuri asupra **algoritmilor**

39

- ▶ Certificatele digitale depind de două feluri de algoritmi
  - ▶ Funcții de dispersie de unic sens (one-way hash) și semnături digitale
- ▶ **Studiu de caz: Proprietatea de rezistență la coliziune a One-Way Hash**
  - ▶ La CRYPTO2004, Xiaoyun Wang a demonstrat un atac folosind coliziunea împotriva MD5.
  - ▶ În februarie 2017, Google Research a anunțat atacul SHAttered
    - ▶ Atacul a învins proprietatea de rezistență la coliziune a lui of SHA-1
    - ▶ Au fost create două fișiere PDF cu același SHA-1.
- ▶ Contramăsuri: folosirea unui algoritm mai rezistent, d.e. SHA256.

# Atacuri asupra confirmării de la utilizator

40

- ▶ După verificarea certificatului de la server, software client este sigur că certificatul este valid și autentic
- ▶ În plus, software are nevoie să confirme că serverul respectiv este cel cu care utilizatorul dorește să interacționeze.
- ▶ Confirmarea implică două informații
  - ▶ Informație de la / aprobată de utilizator
  - ▶ Câmpul de **Common Name** din certificatul serverului
  - ▶ Există software care nu compară aceste două informații: **defect de securitate**

# Atacuri asupra confirmării de la utilizator

41

## Atac asupra Common Name folosind afișarea incorectă a caracterelor Unicode

- ▶ Zheng a descoperit câteva browser-e care nu afișează corect numele de domeniu dacă acesta conține Unicode.
- ▶ xn-80ak6aa92e.com este codificat cu caractere chirilice. Dar numele de domeniu afișat de unele browser-e este apple.com
- ▶ Atacul:
  - ▶ Se obține un certificat pentru xn-80ak6aa92e.com
  - ▶ Se determină utilizatorul să viziteze xn-80ak6aa92e.com, așa că se realizează potrivirea cu Common name
  - ▶ Browser-ul utilizatorului arată că situl de web este apple.com. **Utilizatorul poate fi păcălit.**
- ▶ Dacă browser-ul i-ar fi spus utilizatorului că domeniul real nu este apple.com, utilizatorul s-ar fi oprit.

# Tipuri de certificate digitale

- ▶ Certificate validate cu domeniul (Domain Validated Certificates - DV)
- ▶ Certificate validate cu organizația (Organizational Validated Certificates - OV)
- ▶ Extended Validated Certificates (EV)

# Certificate validate cu domeniul (Domain Validated - DV)

43

- Cel mai popular tip de certificat
- CA verifică înregistrările de domeniu pentru a verifica dacă domeniul aparține solicitantului.
- Validarea controlului asupra domeniului (Domain Control Validation - DCV) se face pe numele de domeniu din cererea de certificat.
- DCV folosește informații din baza de date WHOIS
- DCV se realizează prin
  - ▶ Email
  - ▶ HTTP
  - ▶ DNS

# Certificate validate cu organizația (OV)

44

- Nu sunt prea populare.
- CA-urile verifică următoarele înainte de a emite certificate OV (**Organization Validated**):
  - ▶ Validarea deținerii sau controlului exclusiv asupra domeniului.
  - ▶ Identitatea și adresa solicitantului.
  - ▶ Legătura solicitantului cu organizația
  - ▶ Adresa organizației.
  - ▶ Înregistrarea WHOIS a organizației.
  - ▶ Verifică numărul de telefon al organizației



# Certificate validate extins (EV)

45


- ▶ CA-urile care emit certificate EV (**Extended Validation**) solicită documente semnate legal din partea autorităților de înregistrare (RA).
- ▶ EV CA validează următoarele informații:
  - ▶ Validarea deținerii sau controlului exclusiv asupra domeniului.
  - ▶ Verifică identitatea, autoritatea, semnătura și legarea de individ.
  - ▶ Verifică adresa fizică a organizației și numărul de telefon.
  - ▶ Verifică existența operațională.
  - ▶ Verifică identitatea legală corespunzătoare a organizației.
- ▶ Certificatul EV costă, dar merită încredere.

# Cum afișează browser-ele tipurile de certificate

46

## Chrome browser

DV/OV Certificate

 Secure | <https://www.microsoft.com/en-us/>

EV Certificate

 PayPal, Inc. [US] | <https://www.paypal.com/us/home>

## Firefox browser

DV/OV Certificate

 <https://www.microsoft.com/en-us/>

EV Certificate

 PayPal, Inc. (US) | <https://www.paypal.com/us/home>