Certificates

Recap: digital signature schemes (Gen, S, V)

(1) From generic OWF (XMSS) long

(2) From DLOG (ECDSA, Schnorr, BLS)

(3) From TDP (RSA) Fast verify

RSA-FDH: $pk = (n, e, H)$, $sk = (n, d, H)$

where $n = p \cdot q$, $e \cdot d = 1 \pmod{\varphi(n)}$, $H : \mathcal{M} \to \mathbb{Z}_n$

$S(sk, m)$: $\sigma \leftarrow H(m)^d \pmod n$

$V(pk, m, \sigma)$: accept iff $[\sigma^e \bmod n] = H(m)$

RSA in practice: PKCS1 v1.5

The hash function $H_{PKCS1} : \mathcal{M} \to \mathbb{Z}_n$

$$H_{PKCS1}(m) = \Big[\ \underbrace{\boxed{01}\boxed{FF}\boxed{FF}\ \cdots\ \boxed{FF}\boxed{00}\boxed{SHA256(m)}}_{\text{RSA modulus size (2048 bits)}}\ \Big]$$

16 bits

So: $\sigma = [H_{PKCS1}(m)^d \bmod n]$

Problem: this hash function is **not full domain.**

$\Rightarrow$ no security analysis.

Fast hash-based signatures

Goal: secure sigs from hash functions (no algebra, post-quantum)

Problem: long sigs.

Step 1: one-time sigs

Secure as long as sig signs only one msg.

The basic scheme (Lamport): for $n=256$-bit msgs

$H: X \to Y$ one-way func. (eg. $X = Y = \{0,1\}^{256}$ for post-quantum)

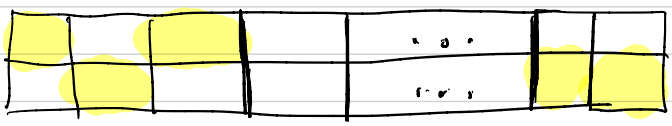
Gen(): choose $\begin{bmatrix} x[0,0], & x[1,0], & \dots, & x[n-1,0] \\ x[0,1], & x[1,1], & \dots, & x[n-1,1] \end{bmatrix} \xleftarrow{R} X^{2n}$

set $y[i,j] \leftarrow H(x[i,j])$ $\quad i=0,\dots,n-1, \quad j=0,1$

output $PK = \begin{pmatrix} y[1,0] & \dots & y[n,0] \\ y[1,1] & \dots & y[n,1] \end{pmatrix}$; $SK = \begin{pmatrix} x[0,0] & \dots & \\ & \dots & x[n-1,1] \end{pmatrix}$

$S(SK, m \in \{0,1\}^n)$: $\sigma := \left(x[0,m[0]], \dots, x[n-1, m[n-1]] \right) \in X^n$

$m = 0101\dots11$



$|\sigma| = 32n = 8$ KB

$|PK| = 16$ KB

$V(PK, m, \sigma)$: accept if $H(\sigma[i]) = y[i, m[i]] \quad \forall i = 0,1,\dots,n-1$

one-time security:

adv. is given PK + one chosen msg attack.

goal: exist. Forgery

Thm: H is one-way func. $\Rightarrow$

Lamport is one-time secure.

Proof: good exercise. $\forall A \; \exists B$

$$1SIG\,Adv[A, Lamport] \leq 2V \cdot OWF\,Adv[B, H]$$

Notes:

(1) Not two time secure! Sgn 0^v, 1^v.

HW: extend to two time security.

(2) Many improvements.

record is Merkle-HORS. (hash-based OTS)

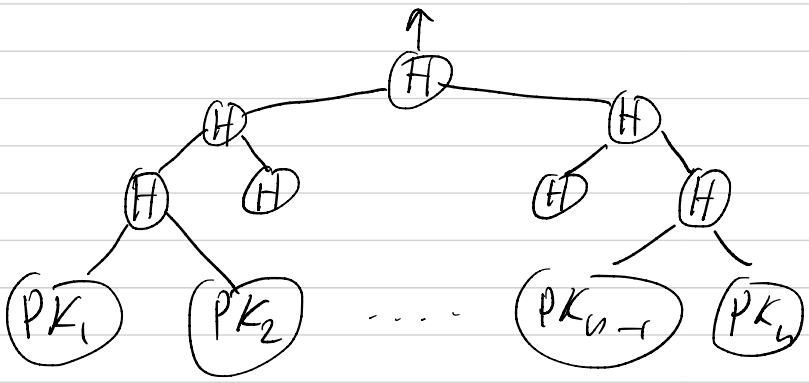
$|PK| = 32$ bytes

$|\sigma| = 1$ KB.

Step 2: one-time sig $\Rightarrow$ many-time sig (e.g., $n = 2^{40}$)

Gen: (1) make n OTS key pairs: $(PK_i, SK_i) \xleftarrow{R} Gen()$ $i = 1, \ldots, n$

(2) Build a merkle tree from n public keys.



(3) Set $PK :=$ root of merkle tree

$SK = \{SK_1, \ldots, SK_n\}$

$S(SK, m)$: sign msg #i with key SK_i.

$$\sigma = (S_{OTS}(SK_i, m), PK_i, \text{Merkle proof for } PK_i)$$

Signer needs to store state: sig counter.

$\Rightarrow$ if leaf used to sign two msgs $\rightarrow$ no security

$V(PK, m, \sigma = (\sigma', PK_i, \pi))$:

$V_{OTS}(PK_i, \sigma') =$ yes & Merkle proof π valid.

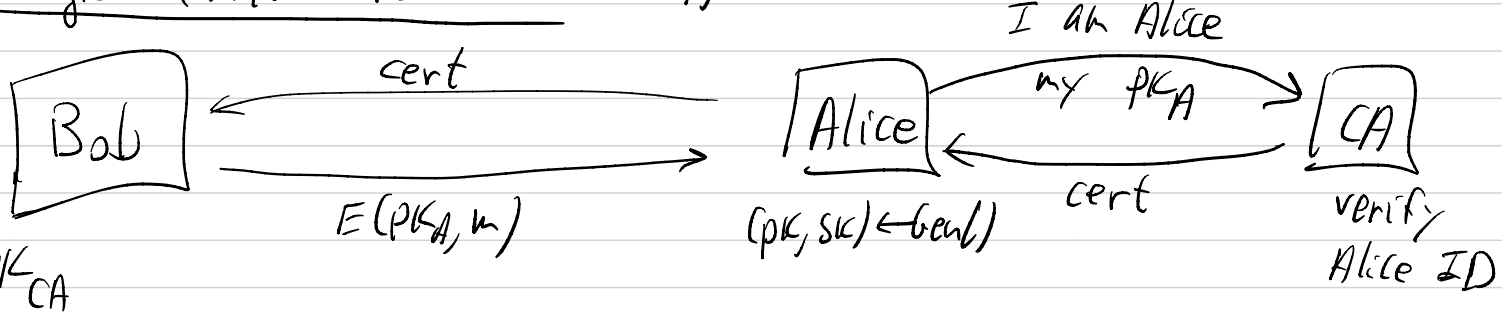
Public key mgnt : certificates

How does Bob obtain Alice's PK_A?

if attacker can fool Bob into using $\hat{PK}_A$

$\Rightarrow$ PcTM & sig. Forgery.

Single domain cert. auth (CA)



cert: binds pub. key to a physical identity

$$\text{cert} = \left[\begin{array}{c|c|c|c|c} \text{X500} \\ \text{issuer} \\ \text{name} & \begin{array}{c}\text{X500} \\ \text{subj-name}\end{array} & PK & \begin{array}{c}\text{validity} \\ \text{period}\end{array} & \cdots \end{array}\right] + \left[\begin{array}{c}\text{CA's} \\ \text{sig on} \\ \text{precert}\end{array}\right]$$

precert

(1) Alice only talks to CA as part of keygen.

(2) CA does not have Alice's SK

(3) Everyone need PK_{CA} to verify cert.

subject Name: www.stanford.edu

Issue Name	
Country or Region	US
Organization	Let's Encrypt
Common Name	R3

) CA

Serial Number	04 0A EB 1E 61 93 DB 9C 61 10 BE CA C8 57 1C 71 46 47
Version	3
Signature Algorithm	SHA-256 with RSA Encryption (1.2.840.113549.1.1.11)
Parameters	None

Not Valid Before	Wednesday, February 3, 2021 at 10:14:31 AM Pacific Standard Time
Not Valid After	Tuesday, May 4, 2021 at 11:14:31 AM Pacific Daylight Time

← exp. date

Public Key Info	
Algorithm	RSA Encryption (1.2.840.113549.1.1.1)
Parameters	None
Public Key	256 bytes : A8 F2 D2 1F 4A 35 ED 8D ...
Exponent	65537
Key Size	2,048 bits
Key Usage	Encrypt, Verify, Wrap, Derive

← RSA pub. key

Signature	256 bytes : 2C 8F BC 72 B7 1D 90 5A ...

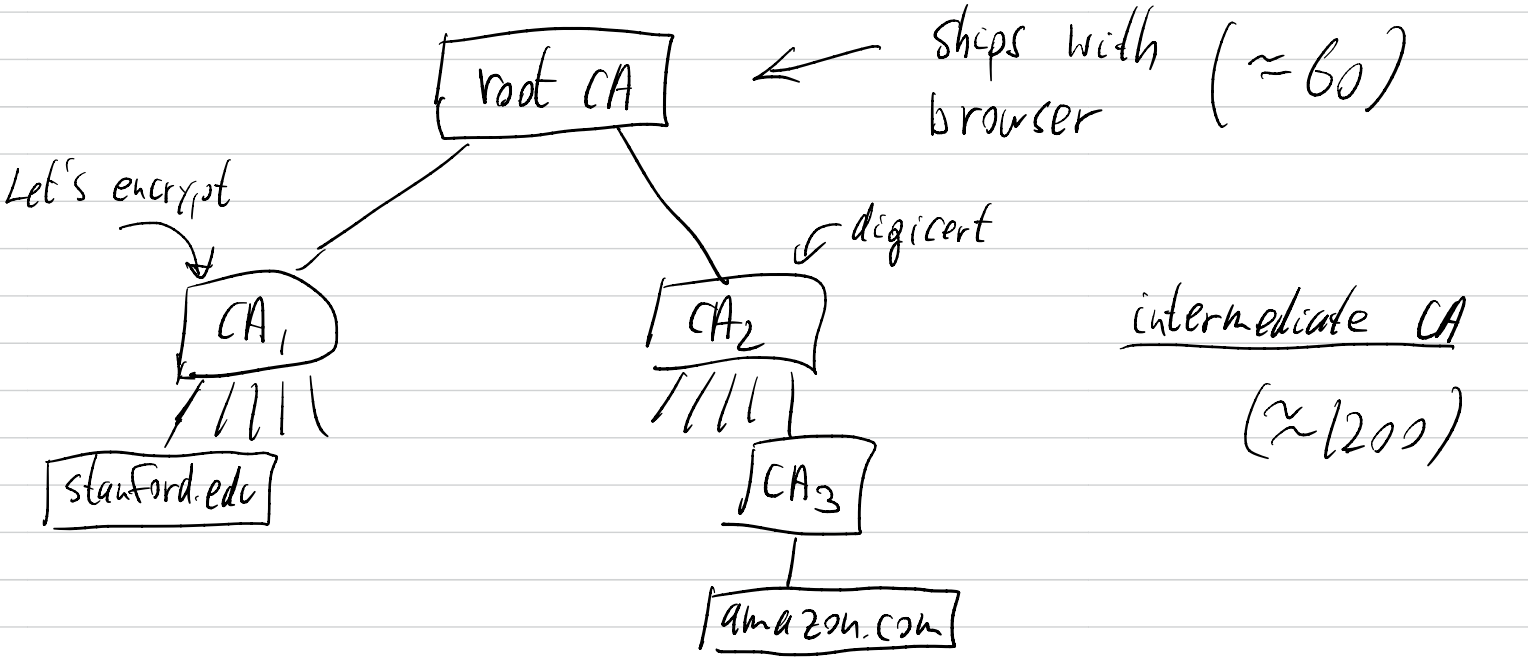
← CA sig.

Extension	Embedded Signed Certificate Timestamp List (1.3.6.1.4.1.11129.2.4.2)
Critical	NO
SCT Version	1
Log Operator	Let's Encrypt
Log Key ID	94 20 BC 1E 8E D5 8D 6C 88 73 1F 82 8B 22 2C 0D D1 DA 4D 5E 6C 4F 94 3D 61 DB 4E 2F 58 4D A2 C2
Timestamp	Wednesday, February 3, 2021 at 11:14:31 AM Pacific Standard Time
Signature Algorithm	SHA-256 ECDSA
Signature	71 bytes : 30 45 02 21 00 BD 83 58 ...
SCT Version	1
Log Operator	Google
Log Key ID	7D 3E F2 F8 8F FF 88 55 68 24 C2 C0 CA 9E 52 89 79 2B C5 0E 78 09 7F 2E 6A 97 68 99 7E 22 F0 D7
Timestamp	Wednesday, February 3, 2021 at 11:14:31 AM Pacific Standard Time
Signature Algorithm	SHA-256 ECDSA
Signature	70 bytes : 30 44 02 20 78 CF 34 79 ...

Extension	Certificate Authority Information Access (1.3.6.1.5.5.7.1.1)
Critical	NO
Method #1	Online Certificate Status Protocol (1.3.6.1.5.5.7.48.1)
URI	http://r3.o.lencr.org
Method #2	CA Issuers (1.3.6.1.5.5.7.48.2)
URI	http://r3.i.lencr.org/

Subject Name	
Common Name	*.google.com
Issuer Name	
Country or Region	US
Organization	Google Trust Services LLC
Common Name	GTS CA 1C3
Serial Number	45 52 E5 F4 0E 97 F8 E2 0A 00 00 00 01 2F 8F DB
Version	3
Signature Algorithm	SHA-256 with RSA Encryption (1.2.840.113549.1.1.11)
Parameters	None
Not Valid Before	Sunday, January 16, 2022 at 6:21:04 PM Pacific Standard Time
Not Valid After	Sunday, April 10, 2022 at 7:21:03 PM Pacific Daylight Time
Public Key Info	
Algorithm	Elliptic Curve Public Key (1.2.840.10045.2.1)
Parameters	Elliptic Curve secp256r1 (1.2.840.10045.3.1.7)
Public Key	65 bytes : 04 B5 22 17 1B 50 1E 1B ...
Key Size	256 bits
Key Usage	Encrypt, Verify, Derive
Signature	256 bytes : 29 4E 32 1B 70 E1 10 32 ...

root CA & intermediate CAs



amazon's cert: $\left[(\text{amazon} \mid PK_A \mid \underset{sig}{CA_3}),\ (CA_3 \mid PK_3 \mid \underset{sig}{CA_2}),\ (CA_2 \mid PK_2 \mid \underset{sig}{root}) \right]$

cert. chain

Typically root CA is offline.

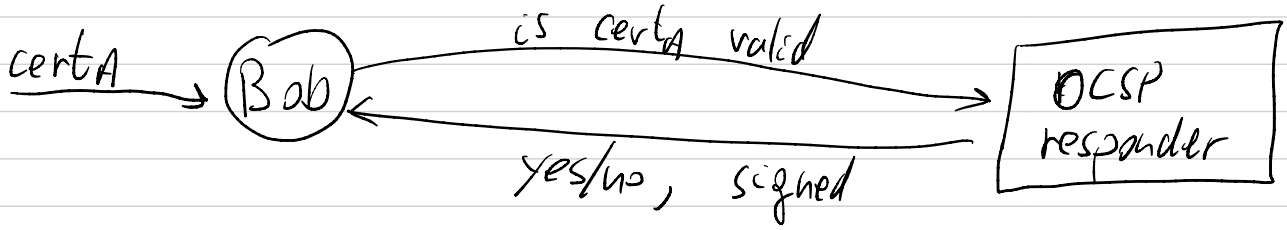
Certificate revocation

Stanford's priv key is stolen ⇒ need to revoke cert.

I. exp. date (one year or shorter ... 3 months)

II. CRL set : cert. revocation list
list of all revoked serial #
shipped by google to chrome

III. OCSP : online cert. status protocol



privacy problem: responder can track Bob

IV. short lived certs (4 days)

- A maintains a cache of 4 certs
- refills every day by contacting CA.
- if revoked, just tell CA to stop issuing certs.

Negligent / misbehaving CAs

2011: Diginotar signed *.google.com, gmail.com, ...
hack → CA untrusted by browsers
→ bankrupt

several times a year ...

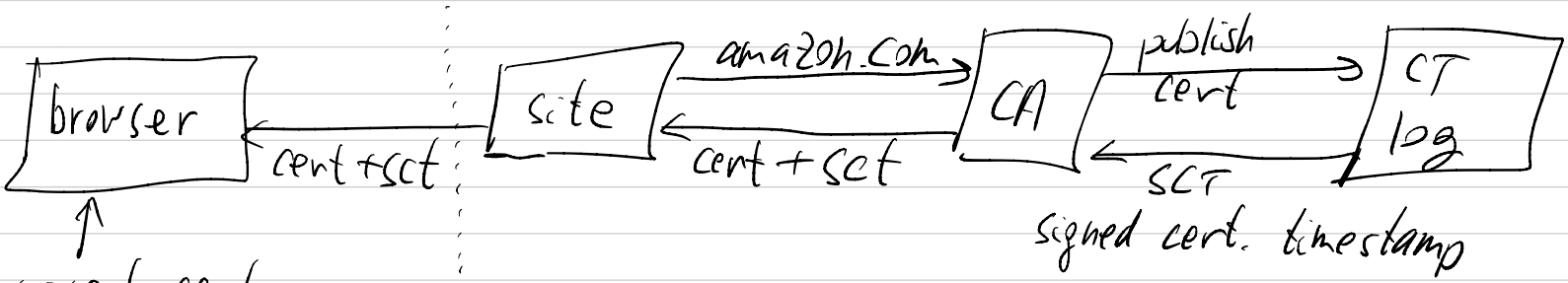
solutions:

(1) Pinning: browser ships with allowable CAs for some domains } pins

ex: gmail.com is pinned to a specific CA.
any chrome instance sees cert for gmail.com
by another CA: - rejects connection
- sends cert to google.

(2) Cert. transparency:

Require CAs to publish log of issued certs.



reject cert w/o SCT

Amazon checks all logs to ensure no invalid certs.