Merkle's Puzzles (1974)

- $0 < i < 2^n = N$
- $X_i, Y_i$ -- random secret keys
- $\text{index}_i$ = random (secret) value
- Puzzle $P_i = \{\text{index}_i, X_i, S\}$
- $S$ -- fixed string, e.g., "Alice to Bob"

$\{P_i \mid 0 < i < 2^n\}$

Pick random $j$, $0 < j < 2^n$

Select $P_j$

Break $Y_j$ by brute force

Obtain $\{\text{index}_j, X_j, S\}$

Look up $\text{index}_j$

Obtain $X_j$

Encrypted communication with $X_j$

Lecture 18

Public Key Certification and Revocation
**CertificationTree / Hierarchy**

Logical tree of CA-s

```
root
  PK_root
CA1
  PK_CA1 SK_root
  CA2
    PK_CA2 SK_CA1
CA3
  PK_CA3 SK_root
    CA4
      PK_CA4 SK_CA3
```

**Hierarchical PKI Example**

CAs

- UCI
- UCSB
- UCSD
- UCR

End users
Hierarchical PKI Example

Upper level CAs

- UCOP
- UCLA
- UCI

CAs

- CSOP
- CSULB
- CSUN
- UCI

End users

gtsudik@uci.edu

Hierarchical PKI Example

Root CA

- State Govt.

Upper level CAs

- Root CA

CAs

- Upper level CAs

End users
Cross Certificate Based PKI

Example

CAs

End users

Cross certificates

UC System

UMass

UTexas
Hybrid PKI example

Note that no cross arrows down or up!

Certificate Paths

Derived from PKI
Certificate Paths

- Verifier must know public key of the first CA
- Other public keys are ‘discovered’ one by one
- All CAs on the path must be (implicitly) trusted by the verifier
X.509 Standard

❖ X.509v3 is the current version
  ▸ ITU standard
  ▸ ISO 9495-2 is the equivalent ISO standard
❖ Defines certificate format, not PKI
❖ Identity and attribute certificates
❖ Supports both hierarchical model and cross certificates
❖ **End users cannot be CAs**

X.509 Service

❖ Assumes a distributed set of servers maintaining a database about certificates

❖ Used in S/MIME, IPSec, SSL/TLS, SET.

❖ RSA, DSA, MD5*, SHA* are most commonly used algorithms
Format:

- version
- serial number
- signature algorithm ID
- issuer name (X.500 Distinguished Name)
- validity period
- subject (user) name (X.500 Distinguished Name)
- subject public key information
- issuer unique identifier (version 2 and 3 only)
- subject unique identifier (version 2 and 3 only)
- extensions (version 3 only)
- signature on the above fields

X.509 Certificate Format
A sample certificate

Certificate:
Data:
  Version: 3 (0x2)
  Serial Number: 28 (0x1c)
  Signature Algorithm: md5WithRSAEncryption
  Issuer: C=US, O=Globus, CN=Globus Certification Authority
  Validity
    Not After: Apr 22 19:21:50 1999 GMT
  Subject: C=US, O=Globus, O=University of Southern California, \
           ou=ISI, CN=bonair.isi.edu
  Subject Public Key Info:
    Public Key Algorithm: rsaEncryption
    RSA Public Key: (1024 bit)
      Modulus (1024 bit):
        00:bf:4c:9b:ae:51:e5:ad:ac:54:4f:12:52:3a:69:
          <snip>
        b4:e1:54:e7:87:57:b7:d0:61
      Exponent: 65537 (0x10001)
  Signature Algorithm: md5WithRSAEncryption
      <snip>

Certificates in Practice

❖ X.509 certificate format is defined in Abstract Syntax Notation 1 (ASN.1)
❖ ASN.1 structure is encoded using the Distinguished Encoding Rules (DER)
❖ A DER-encoded binary sting is typically base-64 encoded to get an ASCII representation
Certificate Revocation Scenario

What if:
❖ Bob's CA goes berserk?
❖ Bob forgets his private key?
❖ Someone steals Bob's private key?
❖ Bob loses his private key?
❖ Bob willingly discloses his private key?
   - Eve can decrypt/sign while Bob's certificate is still valid...
   - Bob reports key loss to CA (or CA finds out somehow)
   - CA issues a Certificate Revocation List (CRL)
     + Distributed in public announcements
     + Published in public databases
   - When verifying Bob's signature or encrypting a message for Bob, Alice's certificate is still valid!
   - IMPORTANT: what about signatures "Bob" generated before he realized his key is lost?
Certificate is a capability!

- Certificate revocation needs to occur when:
  - certificate holder key compromise/loss
  - CA key compromise
  - end of contract (e.g. certificates for employees)

- Certificate Revocation List (CRL) lists certificates that are not yet naturally expired but revoked
- CRL reissued periodically, even if no activity!
- More on revocation later...

Requirements for revocation

- Timeliness
  - Before using a certificate, must check most recent revocation status

- Efficiency
  - Computation
  - Bandwidth and storage
  - Availability

- Security
Types of Revocation

❖ Implicit
- Each certificate is periodically (re-issued
- Alice has a fresh certificate ➔ Alice not revoked
- No need to distribute/publish revocation info

❖ Explicit
- Only revoked certificates are periodically announced
- Alice’s certificate not listed among the revoked ➔ Alice not revoked
- Need to distribute/publish revocation info

Revocation methods

❖ CRL - Certificate Revocation List
  - CRL-DP, indirect CRL, dynamic CRL-DP,
  - delta-CRL, windowed CRL, etc.
  - CRT and other Authenticated Data Structures

❖ OCSP - On-line Certificate Status Protocol

❖ CRS - Certificate Revocation System
CRL

❖ Off-line mechanism

❖ CRL = list of revoked certificates (e.g., SNs) signed by a revocation authority (RA)

❖ RA not always CA that issued the revoked PKC

❖ Periodically issued: daily, weekly, monthly, etc.

Pros & Cons of CRLs

❖ Pros
  - Simple
  - Don’t need secure channels for CRL distribution

❖ Cons
  - Timeliness: “window of vulnerability”
  - CRLs can be huge
  - How to distribute CRLs reliably?
PKI and Revocation

- On January 29 and 30, 2001, VeriSign, Inc. issued two certificates for Authenticode Signing to an individual fraudulently claiming to be an employee of Microsoft Corporation.
- Any code signed by these certificates appears to be legitimately signed by Microsoft.
- Users who try to run code signed with these certificates will generally be presented with a warning dialog, but who wouldn’t trust a valid certificate issued by VeriSign, and claimed to be for Microsoft?
- Certificates were very soon placed in a CRL, but:
  - code that checks signatures for ActiveX controls, Office Macros, and so on, didn’t do any CRL processing.
- According to Microsoft:
  - since the certificates don’t include a CRL Distribution Point (DP), it’s impossible to find and use the CRL!
Certificate Revocation Tree (CRT)

- proposed by P. Kocher (1998)
- based on hash trees
  - hash trees first proposed by R. Merkle in another context in 1979 (one-time signatures)
  - improvement to Lamport-Diffie OTS scheme
  - based on the following idea:
    - A wants to sign a bit of information. A gives B the image \( y = F(x) \)
    - Eventually A reveals the pre-image: \( x \)
    - B checks that: \( y = F(x) \)

CRT contd.

- express ranges of SN of PKC’s as tree leaf labels:
  - E.g., \((5 -- 12)\) means: 5 and 12 are revoked, the others larger than 5 and smaller than 12 are okay
  - Place the hash of the range in the leaf
- response includes the corresponding tree leaf, the necessary hash values along the path to the root, the signed root
- the CA periodically updates the structure and distributes to un-trusted servers called Confirmation Issuers
Example of CRT

query: Is 67 revoked?

Signed root (N_{3,0})

```
N_{2,0} \rightarrow SHA \rightarrow N_{1,0} \rightarrow SHA \rightarrow N_{0,0} \rightarrow \text{SHA (}-\infty \text{ to 7)}
```
```
N_{2,0} \rightarrow SHA \rightarrow N_{1,1} \rightarrow SHA \rightarrow N_{0,1} \rightarrow \text{SHA (7 to 23)}
```
```
N_{2,1} \rightarrow SHA \rightarrow N_{1,2} \rightarrow SHA \rightarrow N_{0,2} \rightarrow \text{SHA (23 to 27)}
```
```
N_{2,1} \rightarrow SHA \rightarrow N_{1,3} \rightarrow SHA \rightarrow N_{0,3} \rightarrow \text{SHA (27 to 37)}
```
```
N_{0,4} \rightarrow \text{SHA (37 to 49)}
```
```
N_{0,5} \rightarrow \text{SHA (49 to 54)}
```
```
N_{0,6} \rightarrow \text{SHA (54 to 88)}
```
```
N_{0,7} \rightarrow \text{SHA (88 to } +\infty )
```

Characteristics of CRT

❖ each response represents a proof

❖ length of proof is: $O(\log n)$
  - Much shorter than CRL which is $O(n)$
  - Where $n$ is # of revoked certificates

❖ only one “real” signature for tree root (can be done off-line)
Explicit Revocation: OCSP

- OCSP = On-line Certificate Status Protocol (RFC 2560) - June 1999
- In place of or, as a supplement to, checking CRLs
- Obtain instantaneous status of a PKC
- OCSP may be used in sensitive, volatile settings, e.g., stock trades, electronic funds transfer, military

OCSP players

1. Cert request
2. Bob
3. Transaction + request
4. OCSP request
5. OCSP response / Error message
6. Transaction response
**OCSP definitive response**

- all definitive responses have to be signed:
  
  * either by issuing CA
  * or by a Trusted Responder (OCSP client trusts the TR’s PKC)
  * or by a CA Authorized Responder which has a special PKC (issued by the CA) saying that it can issue OCSP responses on CA’s behalf

---

**Responses for each certificate**

- **Response format:**
  
  - target PKC SN
  - PKC status:
    * good - positive answer
    * revoked - permanently/temporarily (on-hold)
    * unknown - responder doesn't know about the certificate being requested
  - response validity interval
  - optional extensions
Special Timing Fields

❖ A response contain three timestamps:
  – thisUpdate - time at which the status being indicated is known to be correct
  – nextUpdate - time at or before which newer information will be available
  – producedAt - time at which the OCSP responder signed this response. Useful for response pre-production

Security Considerations

❖ on-line method

❖ DoS vulnerability
  – flood of queries + generating signatures!
  – unsigned responses → false responses
  – pre-computing responses offers some protection against DoS, but…

❖ pre-computing responses allows replay attacks (since no nonce included)
  – but OCSP signing key can be kept off-line
Open questions

❖ Consistency between CRL and OCSP responses
  - possible to have a certificate with two different statuses.
❖ If OCSP is more timely and provides the same information as CRLs, do we still need CRLs?
❖ Which method should come first - OCSP or to CRL?

Implicit Revocation: Certificate Revocation System (CRS)

❖ proposed by Micali (1996)
❖ aims to improve CRL communication costs / size
❖ basic idea: signing a message for every certificate stating its status
❖ use of off-line/on-line signature scheme to reduce update cost
**CRS: creation of a certificate**

- Two new parameters in PKC: $Y_{\text{MAX}}$ and $N$
  
  $$Y_{\text{MAX}} = H^{\text{MAX}}(Y_0)$$
  
  $$N = H(N_0)$$

- $[Y_0, N_0]$ -- per-PKC secrets stored by CA

- $H()$ -- public one-way function

**CRS example: certificate issued for 1 year**

- If still valid, $U_k = Y_{365-i} = H^{365-i}(Y_0)$
- If revoked, $U_k = N_0$

NOTE: $i=0$ at the issuance date