Public Key Infrastructure Fundamentals

Prof. Bart Preneel
COSIC – KU Leuven - Belgium
Firstname.Lastname(at)esat.kuleuven.be
http://homes.esat.kuleuven.be/~preneel
February 2013

Thanks to Paul van Oorschot

Goals

• Understand how public keys can be distributed and revoked on a large scale
• Understand what a CA-based PKI is and what the problems are with their deployment
• Understand how multiple CAs can interoperate depending on their trust relationship

How to establish public keys?

• point-to-point on a trusted channel
  – mail business card, phone
• direct access to a trusted public file (registry or database)
  – authentication trees
• on-line trusted server (bottleneck)
  – OCSP: Online Certificate Status Protocol
• off-line servers and certificates
  – PKI: Public Key Infrastructure
• implicit guarantee of public parameters
  – identity based and self-certified keys

What is a Certificate?

DN: cn=Planckaert, o=VTM, c=BE
Serial #: 8391037
Start: 1/2/13 1:00
End: 1/2/14 0:59
CA DN: o=GLS, c=BE
Public key
Name of issuing CA
CA's digital signature on the certificate

What is a Certificate Revocation List?

DN: cn=CRL2, o=VRS, c=US
Start: 02/04/13 1:02
End: 03/04/13 1:01
Revoked:
191231
123832
923756
CA DN: o=VRS, c=US
Name of issuing CA
CA's digital signature on the CRL

PKI Overview

1. Background:
   Keys and Lifecycle Management

2. PKI components ("puzzle pieces")

3. PKI Architectural View

4. Trust Models
### Background:

**Keys and Lifecycle Management**

#### Encrypting/Signing...

- Alice composes a message for Bob
- Alice logs into her User Profile
- Alice composes a message for Bob

#### Decrypting/Verifying...

- Bob uses the one-time symmetric key to retrieve the message text and signed hash
- Bob logs into his User Profile
- Bob uses the one-time symmetric key to retrieve the message text and signed hash

### Receiving secure e-mail

- Alice sends secure e-mail to Bob
- Alice logs into her User Profile
- Alice composes a message for Bob

### Sending secure e-mail

- Alice encrypts and signs the message
- Alice sends the encrypted message to Bob
- Alice logs into her User Profile
- Alice composes a message for Bob

### Key Lifecycle Management

- Key Generation
- Certificate Issuance
- Certificate Validation
- Key Usage
- Key Expiry
- Key Update

### Fundamental PKI features

- Automated and transparent key and certificate lifecycle management
- Consistent behavior across applications

### PKI provides Unified Security

- E-mail
- Desktop
- Web
- ERP
- VPN
Certification Authority

- Issue certificates for all entities / devices (for multiple applications) from a single CA
  - single system saves h/w, s/w, training, personnel
- Flexible certificate policy / security policy
  - tailor to needs of environment, application or entity (e.g. certificate lifetime, crypto algorithms, keylengths, password rules, ...)

Certificate Repository

- LDAP-compliant directory stores certificates
  - standards-based for interoperability
- Directory products built specifically to address scalability issues
  - X.500 or proprietary schemes to replicate data (scales to millions of users)

Certificate Revocation System

- Automated CRL publishing
  - when certificate revoked, CRL can be automatically published to directory providing near-immediate availability
  - automated CRL checking by application
  - want to avoid applications which require manual end-user actions to check CRLs for each application or certificate usage
Automated Key Update & History

- Users should never even need to know they have their own certificates (password only)
- If key management is not automated or does not provide key history . . .
  - when certificate expires, lose access to all past encrypted data, e-mail, . . .
  - user must request new certificate and repeat entire registration process
- Should replace key, not just new expiry date
- Transparent triggering mechanism, ideally

Key Backup & Recovery

- Enterprise will lose valuable data if keys used to encrypt data are not backed up
  - 20-40% of users forget passwords / year
  - employees leave the organization
- Allows the enterprise to control the backup
  - not reliant on 3rd parties
  - should be configurable to require multiple administrators to authorize access

Support for Non-Repudiation

- Must use separate key pairs for digital signatures and encryption
  - want backup of encryption keys, do not want backup of signature private keys
- Separate key pairs allows lifecycles to be managed independently
- Different policy controls for each key pair
  - security requirements per pair may differ, e.g. valid lifetimes
**Cross-Certification**

- Sufficiently flexible to model existing business relationships
  - includes 1-1 relationships and hierarchies
  - cross-certificate associated with an organization (vs. a service provider)
  - compare to web trust model: trust anyone signed by browser-embedded CAs
- Enterprise manages cross-certification policy & procedures, to reduce business risk
  - cross-certificates created by authorized administrators, transparent to end-user

**Timestamping**

- Legal requirements
- Business requirements related to fixing transactions in time
- Technical requirements related to certificate revocation (non-repudiation)

**Application Software**

- Designed to be enabled to use the PKI (“PKI-ready”)
  - application software
    - (email, file encryption, VPN, web security/SSL, ...)
  - key & certificate lifecycle mgmt
    - (certificate validation, key update, ...)
  - crypto algorithms
    - (symmetric encryption, signature, hash, MAC, key establishment, ...)
PKI Fundamentals

Summary - Essential PKI Components

Much more than a “certificate server” or set of toolkit calls

- Certification Authority
- Revocation system
- Certificate repository ("directory")
- Key backup and recovery system
- Support for non-repudiation
- Automatic key update
- Management of key histories
- Cross-certification
- PKI-ready application software

More info: IETF PKIX Working Group

www.ietf.org

- de facto standards for Internet PKI, X.509-based
- Certificate & CRL Profile [PKIX-1]: RFC 2459
- Certificate Mgmt Protocols [PKIX-CMP, PKIX-3]: RFC 2510

PKI vs. Privilege Management

- Public key certificate binds a public key to an entity
- Establishes who owns a key vs. what privileges that key / owner is granted
- Certificate-processing software (relying party) may implicitly grant privileges
- Privilege Management Infrastructure (PMI) makes privileges explicit
- PMI may utilize PKI as base infrastructure
- example: attribute certificates

Key generation: where?

- CA generates key for user
  - absolute trust
  - need transport of private keys
  - easier management for backup/recovery
- user generates his/her key
  - does user have the expertise? (ok if smart card)
  - need to transport of public keys (integrity channel)
- specialised third party generates keys

PKI vs. Privilege Management

- Public key certificate binds a public key to an entity
- Establishes who owns a key vs. what privileges that key / owner is granted
- Certificate-processing software (relying party) may implicitly grant privileges
- Privilege Management Infrastructure (PMI) makes privileges explicit
- PMI may utilize PKI as base infrastructure
- example: attribute certificates
Trust Models

Hierarchical trust model
- Root CA
- Subordinate CAs
- Relying parties transfer risk to the Root CA
- Root CA "deputizes" subordinate CAs, which issue certificates

Enterprise trust model
- Root CA
- Local CAs
- Relying parties transfer risk to their local CA
- The same local CA issues certificates to these parties
- Qualified relationships between CAs are established
Hierarchical relationships are a special case

Spoke-and-hub model is another special case

All relying parties rely on public keys of same set of CAs

Each of these CAs defines its own community of trust

A relying party trusts the union of these communities

The CA Mess on the web

- 10.8M servers start SSL handshake
- 4.3M use valid certificate chains
- 850 CA certs trustable by Windows or Firefox
- 1.4M unique valid leaf certs
  - 300K signed by one GoDaddy cert
- 80 distinct keys used in multiple CA certs
- several CAs sign the IP address 192.168.1.2 (reserved by RFC 1918)
- 2 leaf certs have 508-bit keys
- Debian OpenSSL bug (2006-2008)
  - resulted in 28K vulnerable certs
  - fortunately only 530 validate
  - only 73 revoked

How can we fix this mess?
CA incidents

- March 2011 – Comodo: 9 fraudulent certs
  - via RA GlobalTrust.it/InstantSSL.it
- Summer 2011 – DigiNotar: 500+ fraudulent certs
  - meet-in-the-middle attack against Google users in Iran (300K unique IPs, 99% from Iran)
  - filed for bankruptcy 20 September 2011
- ( Globalsign) – may have been hacked in 2011
- (Versign) – may have been hacked in 2010
- Bit9, a company that provides software and network security services to the U.S. government and at least 30 Fortune 100 firms lost signing key in February 2013

CA incidents

- Malware signed by key of Government of Malaysia

CA common problem

- Personal trust model
  (and related: “web-of-trust”)
  - all entities are end-users (CAs do not exist)
  - keys are essentially self-guaranteed
  - some end-users may also be introducers
  - end-user imports public keys of others
  CHARACTERISTICS
  - suits individuals, not enterprise/corporations
  - user-centric
  - requires security-aware end-users
  - poor scalability

Trust models & Revocation

- public-key systems are commonly engineered with long-life certificates
- certificates bind a key-pair to identity (and potentially privilege information)
- circumstances change over certificate life
  - keys may become compromised
  - identifying information may change
  - privilege may be withdrawn
- need ability to terminate the binding expressed in the certificate
- revocation: most difficult issue in practice

Revocation options

mechanisms indicating valid certificates
- short-lifetime certificates
mechanisms indicating invalid certificates
- certificate revocation lists - CRLs (v1 X.509)
- CRL fragments (v2 X.509), including...
  - segmented CRLs (CRL distribution points)
  - delta CRLs
  - indirect CRLs
mechanisms providing a proof of status
- status-checking protocols (OCSP, ValiCert)
- iterated hash schemes (Micali)
- certificate revocation trees
CRL: properties

- basic CRL
  - simplicity
  - high communication cost from directory to user
- improved CRL
  - very flexible
  - more complex
  - reduced communication and storage

Online Certificate Status Protocol (OCSP) [RFC 2560]

- on-line query to
  - CA
  - or Trusted Responder
  - or CA designated responder
- containing
  - hash of public key CA
  - hash of public key in certificate
  - certificate serial number

Online Certificate Status Protocol (OCSP)

- signed answer
  - status
    - good: not revoked
    - revoked
    - unknown
  - time
    - thisUpdate
    - nextUpdate
    - producedAt

OCSP: evaluation

- [+] positive and negative information
- [-] need to be on-line
  - risk for denial of service
  - not always possible
- ! OCSP may send you freshly signed but old information

If a browser gets no answer to an OCSP request, it just goes on as if nothing happened (usability is more important than security)


Revocation summary

- established standards for basic revocation
  - v2 CRLs
- more sophisticated solutions may be needed for specific applications
- revocation of higher level public keys is very hard (if not impossible)
  - e.g. requires browser patch
- even after 15 years of PKI history, revocation is problematic in practice

Characterizing questions for trust models

- what are the types/roles of entities involved
- who certifies public keys
- are trust relationships easily created, maintained, updated
- granularity of trust relationships
- ability of particular technology to support existing business models of trust
- how is revocation handled?
  - . . . of end-users . . . of certification authorities
Trust model continuums

- hierarchical
- browser
- enterprise
- personal

[increasing granularity of trust]

- hierarchical
- browser
- personal
- enterprise

[increasing capability to represent B2B trust]

Many other continuums can be formulated.

Trust model summary

Key idea: manageability of trust relationships

Each model has its place --

- personal trust model: okay for security-aware individuals working in small communities
- browser model: simple, large communities, everyone trusts all CAs defined by s/w vendor
- hierarchical model: best given an obvious global root and a grand design methodology
- enterprise trust model: best between peer organizations, where trust flexibility is required
- global PKI will include variety of trust models

Identity-based encryption

- Extra material for information

Identity-Based Encryption (IBE)

- IBE is an old idea
  - Originally proposed by Adi Shamir, S in RSA, in 1984
  - Not possible to build an IBE system based on RSA

- First practical implementation
  - Cocks IMA 2001 and Boneh-Franklin Algorithm Crypto 2001
  - Bilinear Maps (Pairings) on Elliptic Curves
  - Based on well-tested mathematical building blocks
  - Public Key Algorithm used for Key Transport

- The IBE breakthrough is having major impact
  - Now over 400 scientific publications on IBE and Pairing Based Cryptography
  - Major deployments in industry

- Standardization Efforts
  - IBE mathematics is being standardized in IEEE 1363.3
  - IETF S/MIME Informational RFC

IBE Public Keys

… Introduce This Elegance

Public-key Encryption where Identities are used as Public Keys

- IBE Public Key:
  - alice@gmail.com
- RSA Public Key:
  - Exponent=0x10001
  - Modulus=135066410865995522334960321627880590938881479605667027524485143851026510648695338339427150571994417982072821644715513735845197039641917430449458724262934102986438329211037295872567235856943110564073501508187510678560562502055636855294752135096525794167732853390610975054433499811150056977236890927563

How IBE works in practice

Alice sends a Message to Bob

Key Server
  - Master Secret
  - Public Parameters

Requests
  - Private Key
  - Authenticates

1. Requests Private Key for bob@b.com
2. bob@b.com
3. Receives Private Key
4. Bob decrypts with Private Key

Requests
  - private key
  - authenticates

Bob decrypts with Private Key

1. alice@a.com
2. bob@b.com
3. Receives Private Key
4. Bob decrypts with Private Key
How IBE works in practice

Alice sends a Message to Bob

Key Server

Charlie encrypts with bob@b.com

Bob decrypts with Private Key

IBE Public Key Composition

v2 ||
public key definition version

ibc-server.acme.com#1234 ||
server location and public parameter version

week = 252 ||
key validity period

bob@acme.com
e-mail address

IBE Benefits

Dynamic “As Needed” Public and Private Key Generation
- No pre-generation or distribution of certificates
- Dynamic In-Key Recovery – No ADKs
- Allows content, SPAM, and virus scanning at enterprise boundary
- Facilitates archiving in the clear per SEC regulations

Policy in the Public Key
- e.g. Key Validity Period
- No CRLs

Dynamic Groups
- Identities can be groups and roles; no re-issuing keys when group or role changes

Minimal System State
- Master Secret / Public Parameters (~50KB) all you need for disaster recovery
- Server scalability not limited by number of messages

Benefits claimed to lead to:
- High system usability
- Highly scalable architecture
- Low operational impact
- Fully stateless operation

Identity Based Encryption
 Binding of Identity to Key is implicit

IBE summary

- Sounds cool
- Lack of revocation means short-lived keys hence high overhead for recipient
- Key escrow is problematic (definitely for signatures)
  - can be avoided but only with a complex scheme that needs PKI anyway
- How do you know what the system parameters used by people with the address xx@hotmail.com?
  - Can these system parameters be revoked?
PKI Fundamentals

PKI

- Public key cryptography and public keys are essential for large scale secure systems
- PKI as we know today is designed for an off-line world in 1978
- Global PKI is very hard
  - who is authoritative for a given namespace?
  - liability challenge
- Revocation is always hard

- Things are much easier if relying party is the same as issuing party: no certificates are needed