Key Management and Distribution

CSS441: Security and Cryptography

Sirindhorn International Institute of Technology
Thammasat University

Prepared by Steven Gordon on 20 December 2015
css441y15s2l10, Steve/Courses/2015/s2/css441/lectures/key-management-and-distribution.tex, r4295

Contents

Key Management

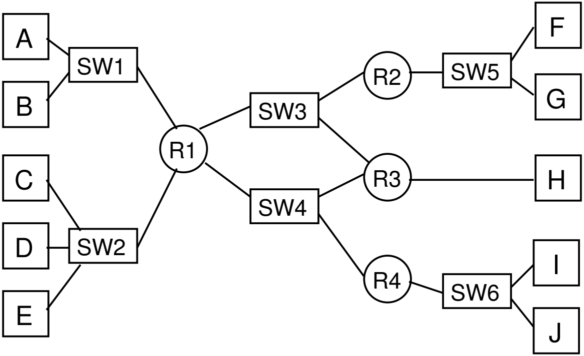
Challenges

- How to share a secret key?
- How to obtain someone else's public key?
- When to change keys?

Assumptions and Principles

- Many users wish to communicate securely across network
- Attacker can intercept any location in network
- Manual interactions between users are undesirable (e.g. physical exchange of keys)
- More times a key is used, greater chance for attacker to discover the key

Where Should Encryption Be Performed?



- Number of keys to be exchanged depends on number of entities wishing to communicate
- Related issue: where to perform encryption
  - Encrypt separately across each link
  - Encrypt only at end-points

Link Encryption vs End-to-End Encryption

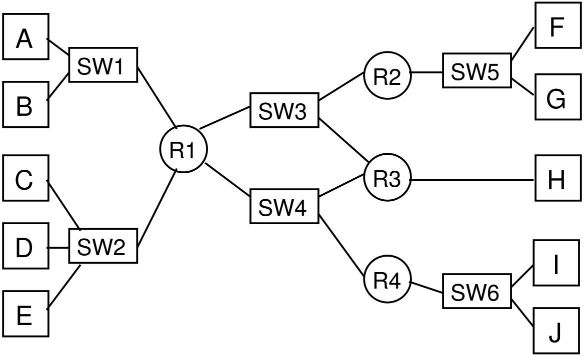
Link Encryption

- Encrypt data over individual links in network
- Each link end-point shares a secret key
- Decrypt/Encrypt at each device in path
- Requires all links/devices to support encryption

End-to-End Encryption

- Encrypt data at network end-points (e.g. hosts or applications)
- Each pair of hosts/applications share a secret key
- Does not rely on intermediate network devices

How Many Keys Need To Be Exchanged?



- Link-level encryption?
- End-to-end encryption between hosts?
- End-to-end encryption between applications?

Exchanging Secret Keys

Option 1: Manual Exchange of All Keys

- All users exchange secret keys with all other users manually (e.g. face-to-face)
- Inconvenient

Option 2: Manual Exchange of Master Keys

- All users exchange master key with trusted, central entity (e.g. Key Distribution Centre)
- Session keys automatically exchanged between users via KDC
- Security and performance bottleneck at KDC

Exchanging Secret Keys

Option 3: Public Key Cryptography to Exchange Secrets

- Use public-key cryptography to securely and automatically exchange secret keys
- Example 1: user A encrypts secret with user B's public key; sends to B
- Example 2: Diffie-Hellman secret key exchange
- Related issue: How to obtain someone else's public key?

Contents

Symmetric Key Distribution using Symmetric Encryption

- Objective: two entities share same secret key
- Principle: change keys frequently
- How to exchange a secret key?
  1. Decentralised Key Distribution: manual distribution of master keys between all entities, automatic distribution of session keys
  2. Key Distribution Centre (KDC): manual distribution of master keys with KDC, automatic distribution of session keys

Key Hierarchy and Lifetimes

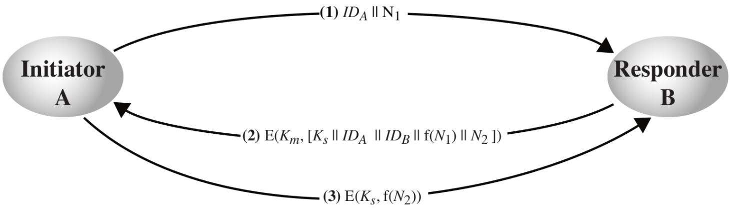
- Master keys used to securely exchange session keys
- Session keys used to securely exchange data
- Change session keys automatically and regularly
- Change master keys manually and seldom
- Session key lifetime:
  - Shorter lifetime is more secure; but increases overhead of exchanges
  - Connection-oriented protocols (e.g. TCP): new session key for each connection
  - Connection-less protocols (e.g. UDP/IP): change after fixed period or certain number of packets sent

Notation

- End-systems: A and B, identified by ID_A and ID_B
- Master key (between A and B): K_m
- Master keys specific to user: K_a, K_b
- Session key (between A and B): K_s
- Nonce values: N_1, N_2
  - Number used only once
  - E.g. time-stamp, counter, random value, function $f()$
  - Must be different for each request
  - Must be difficult for attacker to guess

Decentralised Key Distribution

- Each end-system must manually exchange $n - 1$ master keys (K_m) with others
- Does not rely on trusted-third party

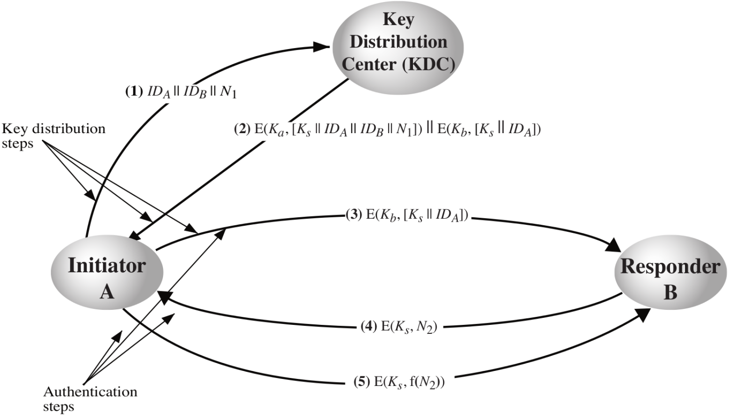


Credit: Figure 14.5 in Stallings, *Cryptography and Network Security*, 5th Ed., Pearson 2011

Using a Key Distribution Centre

- Key Distribution Centre (KDC) is trusted third party
- Users manually exchange master keys with KDC
- Users automatically obtain session key (via KDC) to communicate with other users

Key Distribution with KDC



Credit: Figure 14.3 in Stallings, *Cryptography and Network Security*, 5th Ed., Pearson 2011

Hierarchical Key Control

- Use multiple KDCs in a hierarchy
- E.g. KDC for each LAN (or building); central KDC to exchange keys between hosts in different LANs
- Reduces effort in key distribution; limits damage if local KDC is compromised

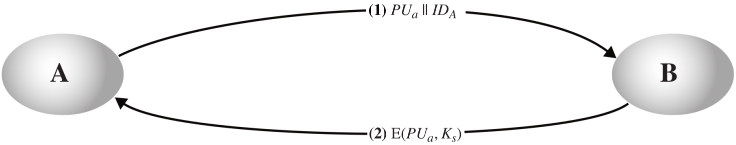
Contents

Symmetric Key Distribution using Asymmetric Encryption

- Asymmetric encryption generally too slow for encrypting large amount of data
- Common application of asymmetric encryption is exchanging secret keys
- Three examples:
  1. Simple Secret Key Distribution
  2. Secret Key Distribution with Confidentiality and Authentication
  3. Hybrid Scheme: Public-Key Distribution of KDC Master Keys

Simple Secret Key Distribution

- Simple: no keys prior to or after communication
- Provides confidentiality for session key
- Subject to man-in-the-middle attack
- Only useful if attacker cannot modify/insert messages

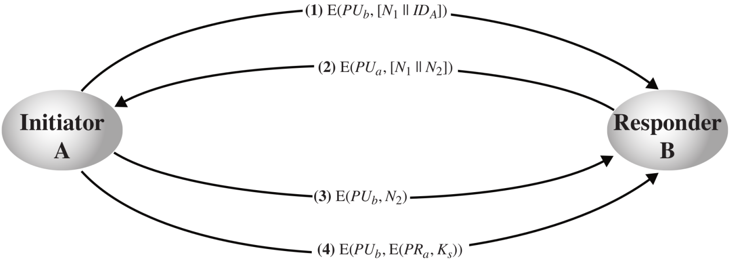


Credit: Figure 14.7 in Stallings, *Cryptography and Network Security*, 5th Ed., Pearson 2011

Man-in-the-Middle Attack

Secret Key Distribution with Confidentiality and Authentication

- Provides both confidentiality and authentication in exchange of secret key



Credit: Figure 14.8 in Stallings, *Cryptography and Network Security*, 5th Ed., Pearson 2011

Hybrid Scheme: Public-Key Distribution of KDC Master Keys

- Use public-key distribution of secret keys when exchanging master keys between end-systems and KDC
- Efficient method of delivering master keys (rather than manual delivery)
- Useful for large networks, widely distributed set of users with single KDC

Contents

Distribution of Public Keys

- By design, public keys are made public
- Issue: how to ensure public key of A actually belongs to A (and not someone pretending to be A)
- Four approaches for distributing public keys
    1. Public announcement
    2. Publicly available directory
    3. Public-key authority
    4. Public-key certificates

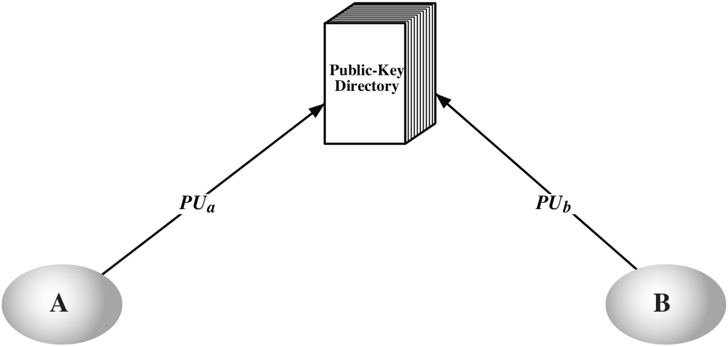
Public Announcements

- Make public key available in open forum: newspaper, email signature, website, conference, ...
- Problem: anyone can announce a key pretending to be another user

Credit: Figure 14.9 in Stallings, *Cryptography and Network Security*, 5th Ed., Pearson 2011

Publicly Available Directory

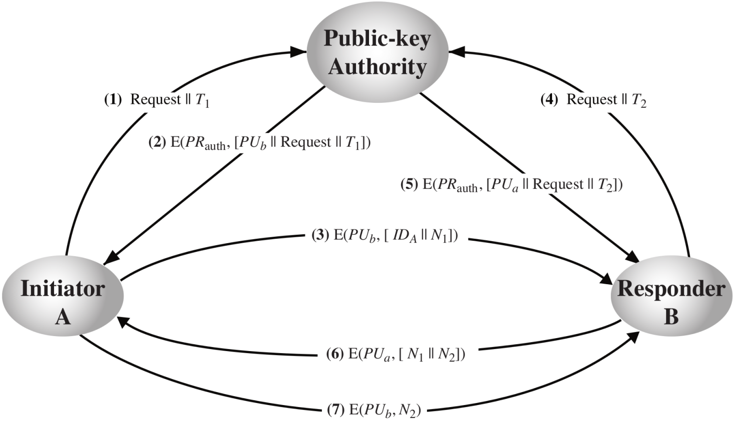
- All users publish keys in central directory
- Users must provide identification when publishing key
- Users can access directory electronically
- Weakness: directory must be secure



Credit: Figure 14.10 in Stallings, *Cryptography and Network Security*, 5th Ed., Pearson 2011

Public-Key Authority

- Specific instance of using publicly available directory
- Assume each user has already security published public-key at authority; each user knows authorities public key



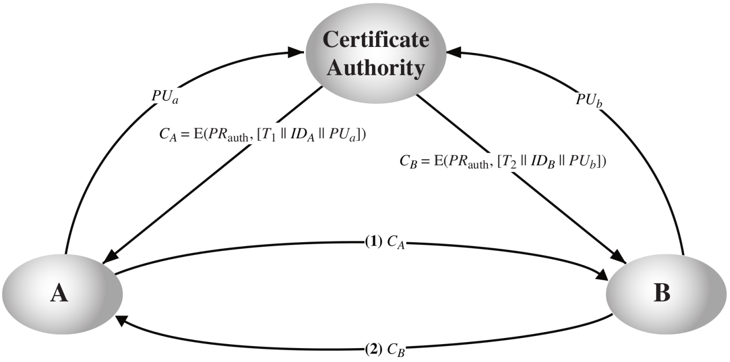
Credit: Figure 14.11 in Stallings, *Cryptography and Network Security*, 5th Ed., Pearson 2011

Public-Key Authority

- First 5 messages are for key exchange; last 2 are authentication of users
- Although 7 messages, public keys obtained from authority can be cached
- Problem: authority can be bottleneck
- Alternative: public-key certificates

Public-Key Certificates

- Assume public keys sent to CA can be authenticated by CA; each user has certificate of CA



Credit: Figure 14.12 in Stallings, *Cryptography and Network Security*, 5th Ed., Pearson 2011

Public Key Certificates

- A certificate is the ID and public-key of a user signed by CA

$$C_A = \mathrm{E}(PR_{auth}, [T||ID_A||PU_a])$$

- Time-stamp T validates currency of certificate (expiration date)
- Common format for certificates is X.509 standard (by ITU)
  - S/MIME (secure email)
  - IP security (network layer security)
  - SSL/TLS (transport layer security)
  - SET (e-commerce)

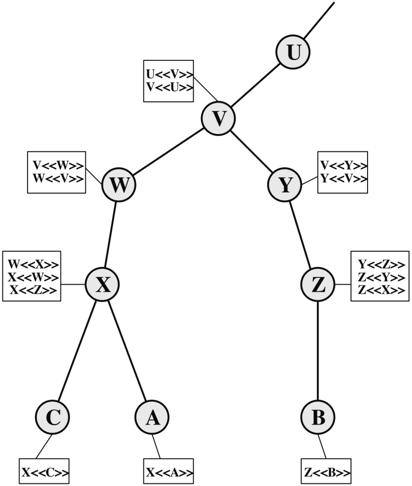
Contents

X.509 Certificates

- Each user has a certificate, although it is created by the Certificate Authority (CA)
- Certificates are stored in a public directory
- Certificate format includes:
  - Version of X.509 certificate
  - Serial number unique to the issuer (CA)
  - Signature algorithm
  - Issuer's name and unique identifier
  - Period of validity
  - Subject's name and unique identifier
  - Subject's public key information: algorithm, parameters, key
  - Signature
- Certificates may be revoked before expiry
  - CA signs a Certificate Revocation List (CRL), which is stored in public directory

Multiple Certificate Authorities

- Multiple CA's can be arranged in hierarchy
- Notation: $Y << X >>$ certificate of X issued by CA Y



Credit: Figure 14.15 in Stallings, *Cryptography and Network Security*, 5th Ed., Pearson 2011