Key management and cryptographic protocols

PV018

Vašek Matyáš

Reminder – relevant topics…

- User authentication and identification
  - Passwords, replay attacks, challenge-response
- Security in communications and networks
  - Authentication in networks
  - Kerberos

Reduction of the problem

- Knowledge of a secret (key) $\Rightarrow$ identity
- For shared-key crypto based on trust in the party the key is shared with
  - Ability to en-/de-crypt or MAC
- For public-key crypto based on trust in the association between the public key and other data
  - Ability to sign or decrypt messages

  - $A \leftarrow B: r_B$
  - $A \rightarrow B: \mathit{cert}_A, r_A, B, S_A(r_A, r_B, B)$
  - $A \leftarrow B: \mathit{cert}_B, A, S_B(r_B, r_A, A)$

Key Management

- Generation
  - Random bit generators (coin tossing, el. noise, etc.)
  - Pseudorandom generators – usual in reality
    - Importance of (statistical) tests
    - Use of good ciphers
- Key storage
- Key distribution
- Key usage
- Key archiving / destroying

…

Key Managements Concepts I.

- Key Certification Center (CA center)
- Key Distribution Center
- Key Escrow
- Key Freshness
- Key Granularity
- Key Material

Key Managements Concepts II.

- Key Notarization
- Key Recovery
- Key Space
- Key Tag
- Trusted Third Party

Classical Fielded Applications

- Symmetric crypto
- Keys at different levels (of security, time of use, etc.). Example (simplified IBM model):
  - Master key – protects terminal keys, in a highly tamper-resistant module
  - Terminal key – protects session keys, stored in a secure (tamper-evident/resistant) memory
  - Session key – protects data in transmission

Use of session (short-term) keys

- To limit volume of ciphertext (under one key) for cryptanalytic attack
- To limit the window of exposure (time and data volume) in the event of key compromise
- To avoid storing large number of distinct keys by creating keys only when actually needed
- To create independence across sessions and/or applications

Protocol

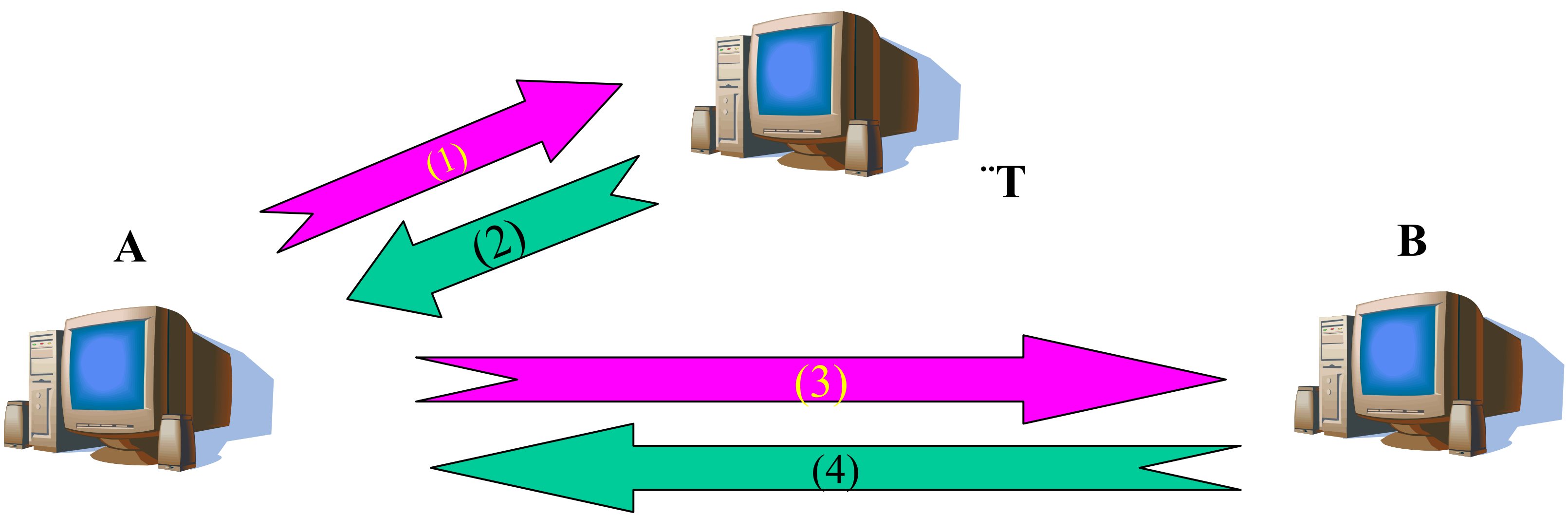
- A multi-party algorithm, defined by a sequence of steps precisely specifying the actions required of two or more parties in order to achieve a specified objective
- Security / cryptography protocols objectives
  - Confidentiality (secrecy), authentication of origin, entity authentication, integrity, key establishment, non-repudiation…

Protocols

- High-level (SSL, IPSEC) & low-level
  - Security functionality point-of-view
  - Network protocol layer point-of-view
    - OSI, TCP/IP
- Single-purpose & multi-purpose
- Standardized & proprietary

Kerberos

- Simplified version of the protocol
  - L – ticket lifetime
  - Def.: $\text{ticket}_B = E_{K_{BT}}(k, \text{“A”}, L)$, $\text{auth} = E_k(\text{“A”}, T_A)$
  - (1) $A \rightarrow T$: ”A”, ”B”, n_A
  - (2) $A \leftarrow T$: ticket_B, $E_{K_{AT}}(k, n_A, L, \text{“B“})$
  - (3) $A \rightarrow B$: ticket_B, auth
  - (4) $A \leftarrow B$: $E_k(T_A)$



Key establishment protocols

- Shared secret becomes available to two or more parties, for subsequent cryptographic use
- **Key transport** – one party (securely) transfers a secret value to other(s)
- **Key agreement** – shared secret is derived by two (or more) parties based on data contributed by, or associated with, each of these, and (ideally) that no party can pre-determine the resulting value

Key establishment concepts

- **Key authentication (implicit)** – assurance to one party that no-one except the specific other party could have gained access to a given key
- **Key confirmation** – assurance to one party that another party actually possess a given key
- **Explicit key authentication** – both above hold

- **Entity authentication** – assurance to one party of the identity of another party actively involved in a protocol

Involvement of trusted parties

- For system setup and/or any protocol run
  - Off-line, on-line, in-line
- Key transport and/or generation
- Trust to keep secrets vs. trust to certify data
- Assumptions of following the course of action prescribed by the protocol, not knowingly collaborating with attackers, etc.

KDC Use – Usual Problems

- Delegation of trust might not be voluntary
- Attacks have to be watched by all parties
  - Key reuse
  - Impersonation of A towards C
  - Impersonation of A towards B

ISO/IEC 9798 – Entity Authentication

- Framework (1), Symmetric (2), Asymm. (3)
- Part 3:
  - Unilateral auth.
    - One-pass – signed sequence number or timestamp
    - Two-pass – challenge-response (random number)
  - Mutual auth.
    - Two-pass – signed sequence numbers or timestamps
    - Three-pass – challenge-response (random number)
    - Two-pass parallel – two unilateral two-pass protocols

Attacker can…

- Record messages
- Replay them later
  - Possibly in different order
  - Some repeatedly
  - Some not at all
- Modify a part of or whole message

Types of attacks on protocols

- Man-in-the-middle
- Replay
- Reflection
- Interleave
- Oracle (chosen-text)
- Forced delay
- …

KE protocol characteristics

- Key freshness
- Key control
  - Can any party control or predict the key value?
- Efficiency
  - Number of message exchanges (passes)
  - Volume of data exchanged
  - Complexity of computation
  - Possibility of pre-computation
- Material pre-distribution (system setup, certificates…)
- Third party involvement
- Non-repudiation

Time-variant parameters (nonces)

- Random numbers (select from a uniform distribution), challenge-response
  - freshness
- Sequence numbers
  - Greater-by-one or only monotonic increase check
  - Counter maintenance, reset policy
- Timestamps
  - Acceptance window
  - Secure, synchronized & distributed time info (clocks)

Types of KE protocols

- Key transport based on symmetric techniques
- Key transport based on asymmetric techniques
- Key agreement based on symmetric techniques
- Key agreement based on asymmetric techniques
- Secret sharing
- Conference keying

Key transport – symmetric techniques

- $A \rightarrow B : E_K(r_A, TVP^*, A^*, B^*)$

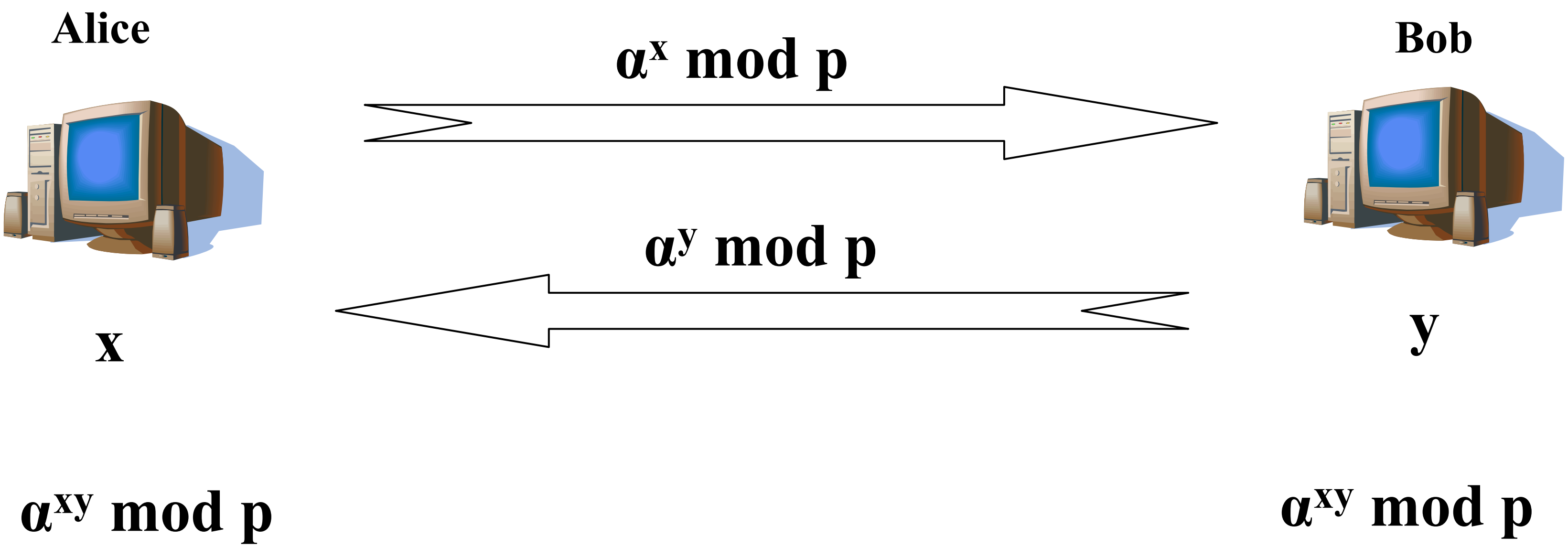
- $A \leftarrow B : n_B$
- $A \rightarrow B : E_K(r_A, n_B, A^*, B^*)$

Shamir's no-key protocol

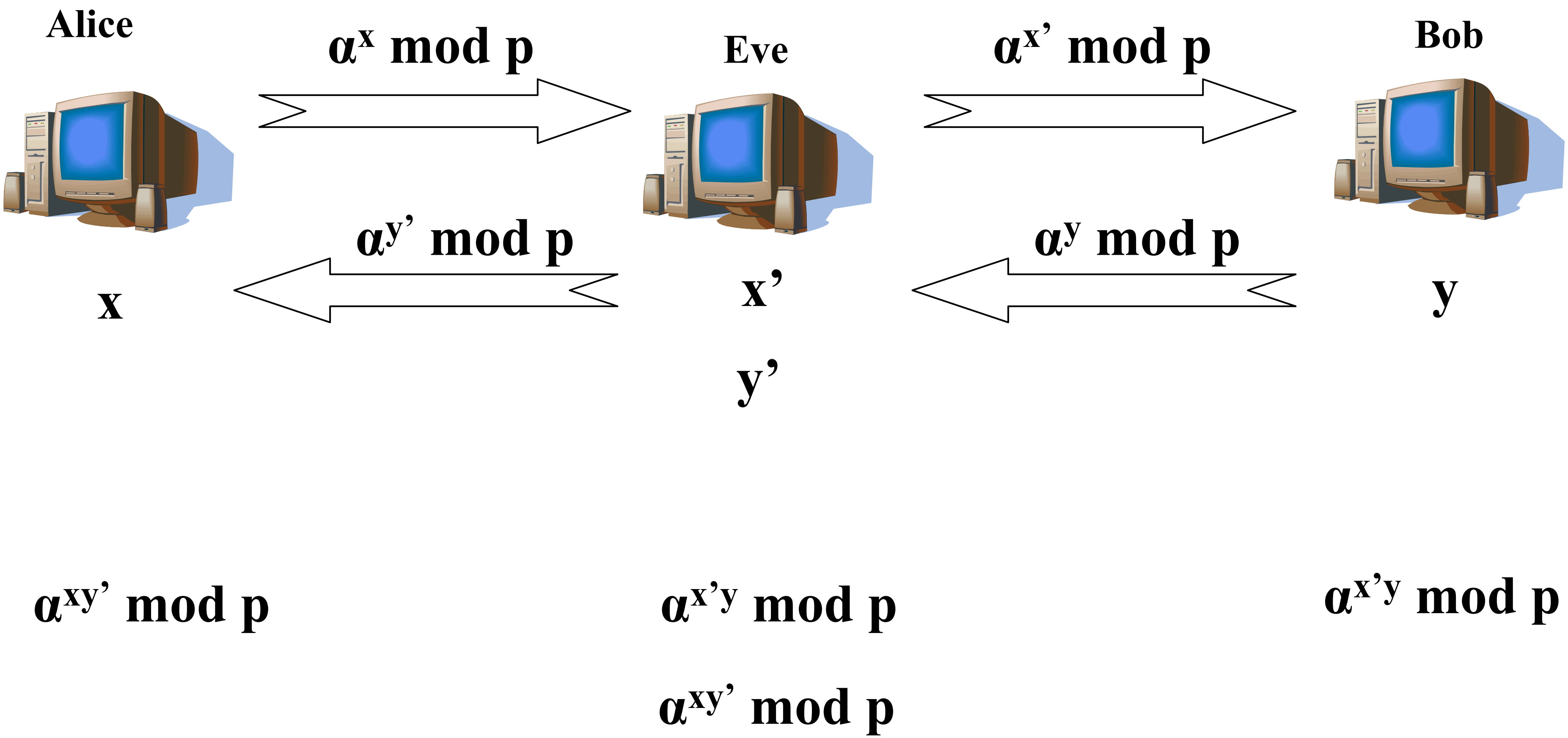
- $A \rightarrow B : E_{K_A}(X)$
- $A \leftarrow B : E_{K_B}(E_{K_A}(X))$
- $A \rightarrow B : E_{K_B}(X)$

- Use of a commutative cipher (not Vernam's)

Diffie-Hellman protocol



Man-in-the-middle attack



The building blocks

- Secure primitives necessary, yet not sufficient
- Playing it safe – precise specification of
  - what shall and shall not be done
  - before, during and after the protocol run
  - with restrictions on use of a given protocol
- Assumptions of critical importance!

Example: ISO/IEC 11770

- Information technology – Security techniques – Key Management
- Part 1: Key management framework
- Part 2: Mechanisms using symmetric techniques
- Part 3: Mechanisms using asymmetric techniques

ISO/IEC 11770-1

1. Scope
2. Normative references
3. Definitions
4. General Disc. of KM
    1. Protection of keys
        1. Crypt. means
        2. Non-crypt. means
        3. Physical means
        4. Organiz. means
    2. Generic Key Life Cycle Model
        1. Transitions between Key States
        2. Transitions, Services and Keys

ISO/IEC 11770-1

5. Concepts of Key M.
    1. Key M. Services
        1. Generate-Key
        2. Register-Key
        3. Create-Key-Certificate
        4. Distribute-Key
        5. Install-Key
        6. Store-Key
        7. Derive-Key
        8. Archive-Key
        9. Revoke-Key
        10. Deregister-Key
        11. Destroy-Key
    2. Support Services
        1. Key M. Facility Services
        2. User-oriented Services
    3. Conceptual Models for Key Distribution
        1. KD between Communicating Entities
        2. KD within One Domain
        3. KD between Domains
7. Specific Service Providers

Annexes (!!!)

ISO/IEC 11770-3

- Secret key agreement (7 mechanisms)
- Secret key transport (6 mechanisms)
- Public key transport
  - Without a TTP (2 mechanisms)
  - Using a CA (1 mechanism ☺)

Related ISO standards

- 7498 – OSI – Security Architecture
- 9798 – Entity Authentication
- 10181 – Security Frameworks for Open Systems

Asymmetric key transport techniques

- Encrypting signed keys
  - $A \rightarrow B: P_B(S_A(B, k, t^*_A))$
  - (* optional) timestamp t_A also authenticates A to B
- Separate signature and encryption
  - $A \rightarrow B: P_B(k, t_A), S_A(B, k, t_A)$
  - Only for signatures without message recovery
- Signing encrypted keys
  - $A \rightarrow B: t_A, P_B(A, k), S_A(B, t_A, P_B(A, k))$

Asymmetric key transport techniques cont'd

- X.509 mutual authentication with key transport
- Def.: $D_A = (t_A, r_A, \text{"B"}, P_B(k_1))$
  $D_B = (t_B, r_B, \text{"A"}, P_A(k_2))$
- Protocol
  - $A \rightarrow B$: $cert_A, D_A, S_A(D_A)$
  - $A \leftarrow B$: $cert_B, D_B, S_B(D_B)$
- Three-pass version with random numbers

Suggested reading this week

- Paper "*Using encryption for authentication in large networks of computers*", R. Needham & M. Schroeder, Comm. ACM, vol. 21, no. 12, pp. 993-999, 1978.

http://lambda.cs.yale.edu/cs422/doc/needham.pdf