Cryptography & Key Exchange Protocols

Faculty of Computer Science & Engineering
HCMC University of Technology
Outline

1. Cryptography-related concepts
2. Key channel for symmetric cryptosystems
3. Perfect encryption
4. Dolev-Yao threat model
5. Protocol “message authentication”
7. Public-key cryptosystems
Cryptography-related concepts

- **Plaintext** is the original content which is readable as textual material. Plaintext needs protecting.
- **Ciphertext** is the result of encryption performed on plaintext using an algorithm. Ciphertext is not readable.
- **Encryption** is the process of turning plaintext into ciphertext, **decryption** is the inverse of the encryption.
- **Cryptosystems** = encryption + decryption algorithms
- Encryption, decryption process needs **keys**
Cryptosystems

Hello,

This content is confidential

Plaintext

Encryption

Ciphertext

Decryption

Key

Key
Cryptography-related concepts

- **Symmetric (shared-/secret-key) cryptosystem**: the same key for (en/de)cryption algorithms
- **Asymmetric (public-key) cryptosystem**: public & private keys for (en/de)cryption algorithms

\[ k_e = k_d \]
\[ k_e \neq k_d \]
Cryptography-related concepts

- (Most popular) Symmetric techniques: DES, AES
  - The same key is used for both encryption and decryption
  - Faster than encryption and decryption in public-key (PK) cryptosystems
  - Less security comparing to encryption and decryption in PK cryptosystems
- Asymmetric techniques: RSA, DSA, Rabin, …
- Hybrid scheme:
  - Asymmetric technique: for the key encryption
  - Symmetric technique: for the data encryption
  - TLS, SSL protocols: how do they work? Homework
SSL protocol

**Phase 1**
- Public key client
- Known information
- Generate random number $R_N$
- Client_hello (crypto information, $R_N$)
- Server certificate
- Demand client certificate
- Check server certificate
- Client certificate
- Check client certificate
- Client certificate (encrypted with Private Key Client)
- Check encrypted client certificate
- Generate random number pre-master-secret $P_{MS}$
- Send $P_{MS}$ (encrypted with public key server)
- Calculate Master-Secret mit $P_{MS}$ $R_N$ $R_N$

**Phase 2**
- Change to encrypted connection with $MS$ as key
- End SSL handshake

**Phase 3**
- Public key server

**Phase 4**
- Private key server
- Known information
- Change to encrypted connection with $MS$ as key
- End SSL handshake
Symmetric encryption techniques

- Most popular symmetric encryption techniques: DES, Tripple DES, AES,
- **DES: Data Encryption Standard**
  - A message is divided into 64-bit blocks
  - Key: 56 bits
  - Brute-force or exhaustive key search attacks (now: some hours).
Symmetric encryption techniques

- **Triple DES**: run the DES algorithm a multiple number of times using different keys
  - Encryption: $c \leftarrow \mathcal{E}_{k_1}(D_{k_2}(\mathcal{E}_{k_1}(m)))$
  - Decryption: $m \leftarrow D_{k_1}(\mathcal{E}_{k_2}(D_{k_1}(c)))$
  - The triple DES can also use three different keys
Symmetric encryption techniques

- AES: Advanced Encryption Standard (Rijndael)
  - Jan 2, 1997, NIST announced the initiation of a new symmetric-key block cipher algorithm, AES, as the new encryption standard to replace the DES
  - Oct 2, 2000: Rijndael was selected
  - Rijndael is designed by two Belgium cryptographers: Daemen and Rijmen

- Rijndael is a block cipher with a variable block size and variable key size
- The key size and the block size can be independently specified to 128, 192 or 256 bits
Asymmetric encryption techniques

- **RSA**: named after 3 inventors Rivest, Shamir và Adleman
  - Two keys: public key and private key
  - Public key is used for encryption.
  - Private key is used for decryption
Digital signatures

- **Digital signatures** is a message signed with a user's private key can be verified by anyone who has access to the user's public key, thereby proving that the user signed it and that the message has not been tampered with.

- Thus:
  - Public key digital signatures provide *authentication* and data *integrity*.
  - A digital signature also provides *non-repudiation*, which means that it prevents the sender from claiming that he or she did not actually send the information.
Cryptography-related concepts

**Simple digital signatures**

- **Private key**
  - Original text
  - Signing

- **Public key**
  - Signed text
  - Verifying

**Memo: Confidential**
- Re: Fiscal Review
  - This quarter's earnings have just come in and...
Cryptography-related concepts

Secure digital signatures

plaintext → hash function → message digest

private key used for signing

digest signed with private key → plaintext + signature
Digital certificates & PKI

- Digital certificates
- PKI (Public Key Infrastructure)

CA (certificate authority)

Alice

Bob
Digital certificates

- Each digital certificate includes the basic elements:
  - Name & URL of CA
  - Public key
  - Owner’s name
  - Valid from – to
- CA is responsible for signing on each digital certificate.
Outline

1. Cryptography-related concepts
2. Key channel for symmetric cryptosystems
3. Perfect encryption
4. Dolev-Yao threat model
5. Protocol “message authentication”
7. Public-key cryptosystems
Key channel for symmetric cryptosystems

Alice → K → Bob

Malice

Trent (TTP)
Key channel for symmetric cryptosystems

- Hybrid scheme:
  - Asymmetric technique: for the key encryption
  - Symmetric technique: for data encryption

- Conventional techniques:
  - Relying on an on-line authentication service
  - This disadvantage limits the scalability of the technique for any open systems applications

- Public-key techniques

- The Quantum Key Distribution Technique
Key channel for symmetric cryptosystems

- The security properties of Key channel for symmetric cryptosystems:
  1. Only Alice & Bob (also TTP) know secret key K.
  2. Alice & Bob ensure that the other know the key K.
  3. Alice & Bob ensure that K is new.
Outline

1. Cryptography-related concepts
2. Key channel for symmetric cryptosystems
3. Perfect encryption
4. Dolev-Yao threat model
5. Protocol “message authentication”
7. Public-key cryptosystems
Perfect encryption

- For a plaintext $M$, a crypto algorithm $A$ and a cryptographic key $K$, the ciphertext $M'$ is calculated as follows:
  \[ M' = A(K,M) = \{M\}_K \]

- Without the key $K$ (in the case of a symmetric cryptosystem), or the matching private key of $K$ (in the case of an asymmetric cryptosystem), the ciphertext $\{M\}_K$ does not provide any cryptanalytic means for finding the plaintext message $M$.

- The ciphertext $\{M\}_K$ and maybe together with some known information about the plaintext $M$ do not provide any cryptanalytic means for finding the key $K$ (in the case of a symmetric cryptosystem), or the matching private key of $K$ (in the case of an asymmetric cryptosystem).
Outline

1. Cryptography-related concepts
2. Key channel for symmetric cryptosystems
3. Perfect encryption
4. Dolev-Yao threat model
5. Protocol “message authentication”
7. Public-key cryptosystems
Dolev-Yao threat model

- Malice (can):
  - can obtain any message passing through the network
  - is a legitimate user of the network, and thus in particular can initiate a conversation with any other user
  - will have the opportunity to become a receiver to any principal
  - can send messages to any principal by impersonating any other principal
Dolev-Yao threat model

- Malice (cannot):
  - cannot guess a random number which is chosen from a sufficiently large space
  - without the correct secret (or private) key, cannot retrieve plaintext from given ciphertext, and cannot create valid ciphertext from given plaintext, wrt. the perfect encryption algorithm
Dolev-Yao threat model

- Malice (cannot):
  - cannot find the private component, i.e., the private key, matching a given public key
  - while he may have control of a large public part of our computing and communication environment, in general, he is not in control of many private areas of the computing environment, such as accessing the memory of a principal's offline computing device
Dolev-Yao threat model

- Suppose that two principals Alice and Bob wish to communicate with each other in a secure manner.
- Suppose also that Alice and Bob have never met before, so they do not already share a secret key between them and do not already know for sure the other party's public key.
- Then how can they communicate securely over completely insecure networks?
Dolev-Yao threat model

**Premise**
- Alice and Trent share key $K_{AT}$; Bob and Trent share key $K_{BT}$.

**Goal**
- Alice and Bob want to establish a new and shared secret key $K$.

1. Alice generates $K$ at random, creates $\{K\}_{K_{AT}}$, and sends to Trent: Alice, Bob, $\{K\}_{K_{AT}}$.

2. Trent finds keys $K_{AT}$, $K_{BT}$, decrypts $\{K\}_{K_{AT}}$ to reveal $K$, creates $\{K\}_{K_{BT}}$ and sends to Bob: Alice, Bob, $\{K\}_{K_{BT}}$.

3. Bob decrypts $\{K\}_{K_{BT}}$ to reveal $K$, forms and sends to Alice: $\{Hello\ Alice, \ I'm \ Bob!\}_{K}$. 
Dolev-Yao threat model

- Problem: K created by Alice is not strong enough
- Bob is unhappy about this
- New protocol: “Session key from Trent”
Dolev-Yao threat model

1. Alice sends to Trent: $Alice, Bob$;

2. Trent finds keys $K_{AT}, K_{BT}$, generates $K$ at random and sends to Alice: $\{K\}_{K_{AT}}, \{K\}_{K_{BT}}$;

3. Alice decrypts $\{K\}_{K_{AT}}$, and sends to Bob: $Trent, Alice, \{K\}_{K_{BT}}$;

4. Bob decrypts $\{K\}_{K_{BT}}$ to reveal $K$, forms and sends to Alice: $\{Hello\ Alice, \ I'm\ Bob!\}_K$. 
Dolev-Yao threat model

- Problem: An attack on protocol "Session key from Trent"

1. Alice sends to Malice("Trent"): Alice, Bob;
2. Malice("Alice") sends to Trent: Alice, Malice;
3. Trent finds keys $K_{AT}, K_{MT}$, generates $K_{AM}$ at random and sends to Alice: $\{K_{AM}\}_{K_{AT}}, \{K_{AM}\}_{K_{MT}}$;
4. Alice decrypts $\{K_{AM}\}_{K_{AT}}$, and sends to Malice("Bob"): Trent, Alice, $\{K_{AM}\}_{K_{MT}}$;
5. Malice("Bob") sends to Alice: $\{Hello\ Alice, \ I'm\ Bob!\}_{K_{AM}}$. 
Dolev-Yao threat model

- "Session key from Trent"
  - Malice must be a legitimate user known to Trent
  - Inside attackers are often more of a threat than outsiders
- Fix: “1. Alice sends to Trent: Alice, \{Bob\}_{K_{AT}},”
  1. Alice sends to Malice("Trent"): Alice, Bob;
  2. Malice("Alice") sends to Trent: Alice, Malice;
  3. Trent finds keys $K_{AT}, K_{MT}$, generates $K_{AM}$ at random and sends to Alice:
     \{K_{AM}\}_{K_{AT}}, \{K_{AM}\}_{K_{MT}};
  4. Alice decrypts \{K_{AM}\}_{K_{AT}}, and sends to Malice("Bob"): Trent, Alice, \{K_{AM}\}_{K_{MT}};
  5. Malice("Bob") sends to Alice: \{Hello Alice, I'm Bob!\}_{K_{AM}}.
Dolev-Yao threat model

But:

1. Alice sends to Trent: Alice, \{Bob\}_{K_{AT}};
2. Malice("Alice") sends to Trent: Alice, \{Malice\}_{K_{AT}};

1. Alice sends to Malice("Trent"): Alice, Bob;

2. Malice("Alice") sends to Trent: Alice, Malice;

3. Trent finds keys \(K_{AT}, K_{MT}\), generates \(K_{AM}\) at random and sends to Alice:
   \(\{K_{AM}\}_{K_{AT}}, \{K_{AM}\}_{K_{MT}}\);

4. Alice decrypts \(\{K_{AM}\}_{K_{AT}},\) and sends to Malice("Bob"): Trent, Alice, \(\{K_{AM}\}_{K_{MT}}\);

5. Malice("Bob") sends to Alice: \{Hello Alice, I'm Bob!\}_{K_{AM}}.
Dolev-Yao threat model

- But making use of old $\{K'\}_{K_{AT}}$ Malice can attack:
  - 2,3. Malice("Trent") sends to Alice: $\{K'\}_{K_{AT}}$, …;

1. Alice sends to Malice("Trent"): Alice, Bob;

2. Malice("Alice") sends to Trent: Alice, Malice;

3. Trent finds keys $K_{AT}, K_{MT}$, generates $K_{AM}$ at random and sends to Alice: $\{K_{AM}\}_{K_{AT}}, \{K_{AM}\}_{K_{MT}}$;

4. Alice decrypts $\{K_{AM}\}_{K_{AT}}$, and sends to Malice("Bob"): Trent, Alice, $\{K_{AM}\}_{K_{MT}}$;

5. Malice("Bob") sends to Alice: $\{Hello \ Alice, \ I'm \ Bob!\}_{K_{AM}}$.  


Dolev-Yao threat model

- Malice is able to alter some protocol messages without being detected
- Thus the protocol needs a security service which can guard against tampering of messages
Outline

1. Cryptography-related concepts
2. Key channel for symmetric cryptosystems
3. Perfect encryption
4. Dolev-Yao threat model
5. Protocol “message authentication”
7. Public-key cryptosystems
Protocol with message authentication

1. Alice sends to Trent: Alice, Bob;

2. Trent finds keys $K_{AT}$, $K_{BT}$, generates $K$ at random and sends to Alice: {Bob, $K$}$_{K_{AT}}$, {Alice, $K$}$_{K_{BT}}$;

3. Alice decrypts {Bob, $K$}$_{K_{AT}}$, checks Bob's identity, and sends to Bob: Trent, {Alice, $K$}$_{K_{BT}}$;

4. Bob decrypts {Alice, $K$}$_{K_{BT}}$, checks Alice's identity, and sends to Alice: {Hello Alice, I'm Bob!}$_{K}$.

See 2.6.3.1 [5] for more details.
Perfect encryption for message authentication service

- Without the key K (in the case of a symmetric cryptosystem), or the matching private key of K (in the case of an asymmetric cryptosystem), the ciphertext $\{M\}_K$ does not provide any cryptanalytic means for finding the plaintext message M.

- The ciphertext $\{M\}_K$ and maybe together with some known information about the plaintext M do not provide any cryptanalytic means for finding the key K (in the case of a symmetric cryptosystem), or the matching private key of K (in the case of an asymmetric cryptosystem).
Perfect encryption for message authentication service

- Without the key $K$, even with the knowledge of the plaintext $M$, it should be impossible for someone to alter $\{M\}_K$ without being detected by the recipient during the time of decryption.
Perfect encryption for message authentication service

- Problem: message replay attack.
- Malice intercepts Alice's request, then:
  1. Alice sends to Malice(“Trent”)
  2. Malice(“Trent”) sends to Alice: \{Bob,K'\}K_{AT}, \{Alice,K'\}K_{BT}

- Two ciphertext blocks containing $K'$ are a replay of old messages which Malice has recorded from a previous run of the protocol (between Alice and Bob)
- This attack will cause Alice & Bob to reuse the old session key $K'$.
- Since $K'$ is old, it may be possible for Malice to have discovered its value (HOW ?? → homework).
Outline

1. Cryptography-related concepts
2. Key channel for symmetric cryptosystems
3. Perfect encryption
4. Dolev-Yao threat model
5. Protocol “message authentication”
7. Public-key cryptosystems
Protocol “challenge-response"

- Symmetric-key Authentication Protocol
- Needham and Schroeder which they published in 1978
- Nonce: a number used once
Protocol “challenge-response”

1. Alice creates $N_A$ at random and sends to Trent: $Alice, Bob, N_A$;

2. Trent generates $K$ at random and sends to Alice: $\{N_A, K, Bob, \{K, Alice\}_K\}_K$;

3. Alice decrypts, checks her nonce $N_A$, checks Bob's ID and sends to Bob: $Trent, \{K, Alice\}_K$;

4. Bob decrypts, checks Alice's ID, creates random $N_B$ and sends to Alice: $\{I'm Bob!N_B\}_K$;

5. Alice sends to Bob: $\{I'm Alice!N_B - 1\}_K$. 
Protocol “challenge-response"

- An attack on the Needham-Schroeder symmetric key authentication protocol:
  - Bob thinks he is sharing a new session key with Alice while actually the key is an old one and may be known to Malice.
Protocol “challenge-response"

1. Alice sends to Malice("Bob"): ...

2. Malice("Alice") sends to Bob: \( \{K', Alice\}_{K_{BT}} \)

3. Bob decrypts, checks Alice's ID and sends to Malice("Alice"): \( \{I'm \ Bob!N_B\}_K \)

4. Malice("Alice") sends to Bob: \( \{I'm \ Alice!N_B - 1\}_K \)
Protocol “challenge-response"

- **Solutions:**
  - More message flows (between Bob & Trent)
  - Timestamps
  - Detailed discussions: 2.6.5
Protocol “Challenge-response” with Timestamps

1. Alice sends to Trent: Alice, Bob
2. Trent sends to Alice: \{Bob, K, T, \{Alice, K, T\}\}_{KBT}^{KAT}
3. Alice checks T and sends to Bob: \{Alice, K, T\}_{KBT}
4. Bob checks T and sends to Alice: \{I’m Bob! N_B\}_K
5. Alice sends to Bob: \{I’m Alice!N_B-1\}_K

- Condition: |Clock − T| < Δt_1 + Δt_2
  - Clock: local clock
  - T: timestamp at Trent
  - Δt_1, Δt_2
Outline

1. Cryptography-related concepts
2. Key channel for symmetric cryptosystems
3. Perfect encryption
4. Dolev-Yao threat model
5. Protocol “message authentication”
7. Public-key cryptosystems
Public-key Cryptosystems

- $K_A, K^{-1}_A$: public & private keys of Alice
- Similarly: $K_B, K^{-1}_B, K_M, K^{-1}_M$
- $\{M\}_{K_A}, \{M\}_{K^{-1}_A}$
Public-key Cryptosystems

**PREMISE**
- Alice's public key is $K_A$,
- Bob's public key is $K_B$,
- Trent's public key is $K_T$.

**GOAL**
Alice and Bob establish a new and shared secret.

1. Alice sends to Trent: *Alice, Bob*;
2. Trent sends to Alice: \{$K_B, Bob$\};
3. Alice verifies Trent's signature on "$K_B, Bob,\" creates her nonce $N_A$ at random, and sends to Bob: \{$N_A, Alice$\}$_{K_B}$;
4. Bob decrypts, checks Alice's ID and sends to Trent: *Bob, Alice*;
5. Trent sends to Bob: \{$K_A, Alice$\};
6. Bob verifies Trent's signature on "$K_A, Alice,\" creates his nonce $N_B$ at random, and sends to Alice: \{$N_A, N_B$\}$_{K_A}$;
7. Alice decrypts, and sends to Bob: \{$N_B$\}$_{K_B}$.
Public-key Cryptosystems

- An attack on public key authentication protocol
  - Found after 17 years
  - Result: Bob thinks he is sharing secrets $N_A$, $N_B$ with Alice while actually sharing them with Malice
  - Method: Malice makes use of Alice as she is trying to establish a connection with him (Alice provides an oracle service)
Public-key Cryptosystems

First run between Alice and Malice

Alice

1–3
\{ N_A, Alice \} K_M

2–3
\{ N_A, Alice \} K_B

2–6
\{ N_A, N_B \} K_A

1–6
\{ N_A, N_B \} K_A

1–7
\{ N_B \} K_M

Second run between Malice("Alice") and Bob

Malice

Bob
Public-key Cryptosystems

- Malice may ask for a session key and Bob may believe that this request is from Alice.
- Then, an example if Bob is a bank, Malice(“Alice”) sends to Bob the following command:

\[
\{NA, NB, "Transfer £1B from my account to Malice's"\}_{KB}
\]
Public-key Cryptosystems

- How to cope with this attack?
  - Homework: see 2.6.6.4, 17.2.3 $\rightarrow$ data integrity
  - This is what we are using nowadays !!

The Needham-Schroeder Public-key Authentication Protocol in Refined Specification

1. Alice Bob : $\{[NA, Alice]KA\}KB$;
2. Bob Alice : $\{NA, [NB]KB\}KA$;
Summary

- Cryptography-related concepts (symmetric/asymmetric techniques, digital signatures, PKI, …)
- Key channel establishment for symmetric cryptosystems
- Perfect encryption
- Dolev-Yao threat model
- Protocol “message authentication”
- Protocol “challenge-response“
- Public-key cryptosystems
Q&A

Question?