2020/07/14 @ Daeyang AI Center 736, Sejong Univ.

Protocols for Authentication and Key Establishment

- A Tutorial Introduction to Authentication and Key Establishment -

Changjun Choi (changjun@pel.sejong.ac.kr)

Protocol Engineering Lab., Sejong University

Contents

1. Introduction
2. Building a Key Establishment Protocol
3. Protocol Architectures
4. Cryptographic Properties
5. Freshness
6. Types of Attack on Protocols
7. Design Principles for Cryptographic Protocols

Introduction

- Overview
  - The newcomer to subject of cryptographic protocols for authentication and key management is likely to be bemused by the sheer variety of techniques and technical background required
  - Even before this stage is reached a more fundamental question needs to be faced
    - Q: “What are these protocols there for at all?”
    - A: It is necessary to provide an understanding of what sets cryptographic protocols apart from other types of protocols
  - This chapter provides necessary background material for those readers who are not already familiar with the topic of cryptographic protocols
    - At the same time, it enables us to start establishing some common concepts and notation that will be used throughout the rest of this book

Building a Key Establishment Protocol

- Protocol design
  - Before designing any protocol the communications architecture must be established
  - Our Scenario has a set of users, any two of whom may wish to establish a <u>new key</u> for use in securing their subsequent communications through cryptography
    - Such a key is known as a ***session key***
  - It is important to understand that successful completion of key establishment is only the beginning of a secure communications session
    - Once an appropriate key has been established its use comes in protecting the real data to be communicated with whatever cryptographic mechanisms are chosen

Building a Key Establishment Protocol

- Protocol design
  - In order to achieve their aim the users interact with an entity called the ***server*** which will also engage in the protocol
  - All users trust the server to execute the protocol faithfully and not to engage in any other activity that will deliberately compromise their security
    - Furthermore, the server is trusted to generate the new key and to do so in such a way that it is sufficiently random to prevent an attacker gaining any useful information about it

Building a Key Establishment Protocol

- Entities of the protocol
  - These are two users whom we denote A and B and the trusted server S
  - The role of S is to generate K_{AB} and transport it to A and B
- Aims of the protocol
  - It is for A and B to establish a new secret key K_{AB} which they can use for secure communications
  - The aims of the protocol can be summarized as follows
    1. At the end of the protocol the value of K_{AB} should be known to both A and B, but to no other parties with the possible exception of S
    2. A and B should know that K_{AB} is newly generated

Building a Key Establishment Protocol

- Protocol scenario & format
  - Scenario of session key establishment protocol
    1. User A contacts S by sending the identities of the two parties who are going to share the new session key
    2. Trusted server S generate the key K_{AB} and send to user A
    3. User A send key K_{AB} and A's identity to user B
  - Different formats for protocol description
    1) Fig. 1.1
       - Visual representation of the flow of messages sent/received between entities in a protocol through **picture**
    2) Protocol 1.1
       - Compact representation of the flow of messages sent/received between entities in a protocol through **notation**

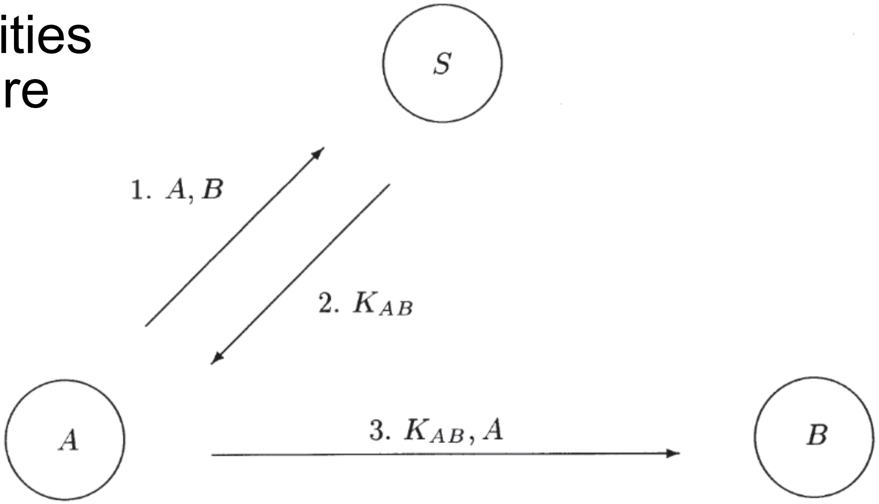


Fig. 1.1. First protocol attempt

1. $A \rightarrow S : A, B$
2. $S \rightarrow A : K_{AB}$
3. $A \rightarrow B : K_{AB}, A$

Protocol 1.1: First protocol attempt in conventional notation

Building a Key Establishment Protocol

- Problems that can occur in a protocol scenario
  - Fig 1.1 and Protocol 1.1 specifies only the messages that are delivered when the protocol is successfully executed
    - In particular there is no description of what happens in the case that a message of the wrong format is received or that no message is received at all
  - Therefore, in this book, trying to explain problem that might occur in a cryptographic protocol through the security assumption

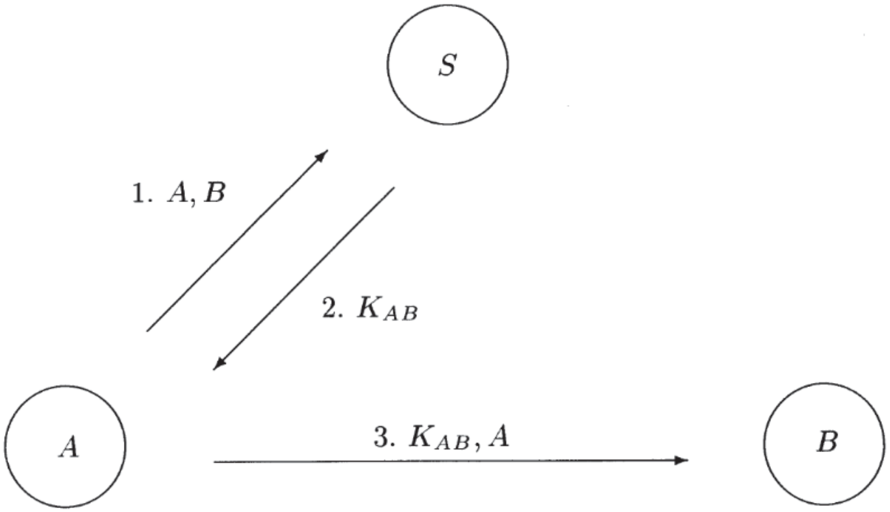
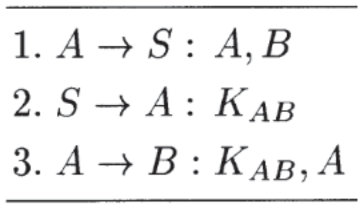


Fig. 1.1. First protocol attempt

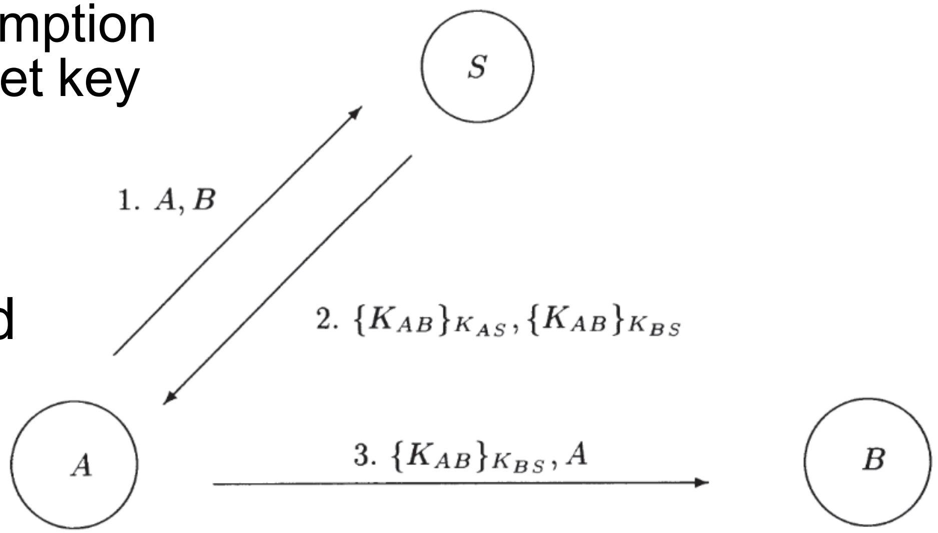


1. $A \rightarrow S : A, B$
2. $S \rightarrow A : K_{AB}$
3. $A \rightarrow B : K_{AB}, A$

Protocol 1.1: First protocol attempt in conventional notation

Building a Key Establishment Protocol

- Confidentiality
  - **Security Assumption 1**
    - *The adversary is able to eavesdrop on all messages sent in a cryptographic protocol*
  - In order to provide confidentiality it is necessary to use a cryptographic algorithm and associated key
    - For now we will simply make the assumption that the server S initially shares a secret key with each user of the system
  - A eavesdropper cannot see K_{AB} since encrypted messages may only be read by the legitimate recipients who have the keys required to decrypt

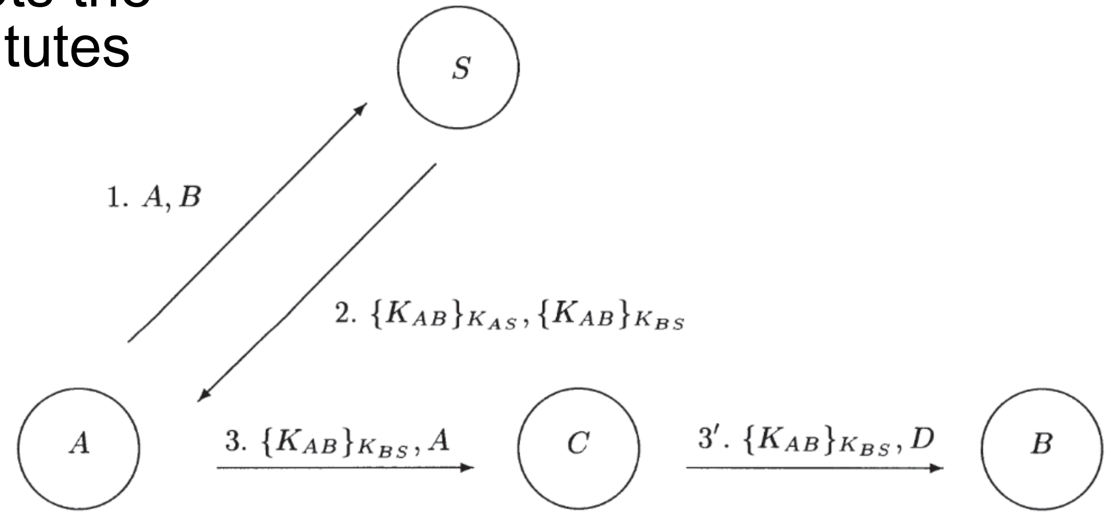


S
1. A, B
2. $\{K_{AB}\}_{K_{AS}}, \{K_{AB}\}_{K_{BS}}$
3. $\{K_{AB}\}_{K_{BS}}, A$
A
B

Fig. 1.2. Second protocol attempt

Building a Key Establishment Protocol

- Authentication
  - **Security Assumption 2**
    - *The adversary is able to alter all messages sent in a cryptographic protocol using any information available*
    - *In addition the adversary can reroute any message to any other principal*
      - *This includes the ability to generate and insert completely new messages*
  - The adversary C simply intercepts the message from A to B and substitutes D's identity for A's
  - The consequence is that B believes that he is sharing the key with D whereas he is in fact sharing it with A



S
1. A, B
2. $\{K_{AB}\}_{K_{AS}}, \{K_{AB}\}_{K_{BS}}$
A
3. $\{K_{AB}\}_{K_{BS}}, A$
C
3'. $\{K_{AB}\}_{K_{BS}}, D$
B

Fig. 1.3. Attack on the second protocol attempt

Building a Key Establishment Protocol

- Authentication
  - Another attack on the protocol does allow C to obtain the session key
    - C alters the message from A to S so that S encrypts the K_{AC} with C's key, K_{CS}, instead of with B's key
      - Since A cannot distinguish between encrypted messages meant for other principals she will not detect the alteration
  - Result of this attack
    - A will believe that the protocol has been successfully completed with B
    - Whereas in fact C knows K_{AC} and so can masquerade as B as well as learn all the information that A sends to B
  - In contrast to the previous attack
    - This one will only succeed if C is a legitimate user known to S

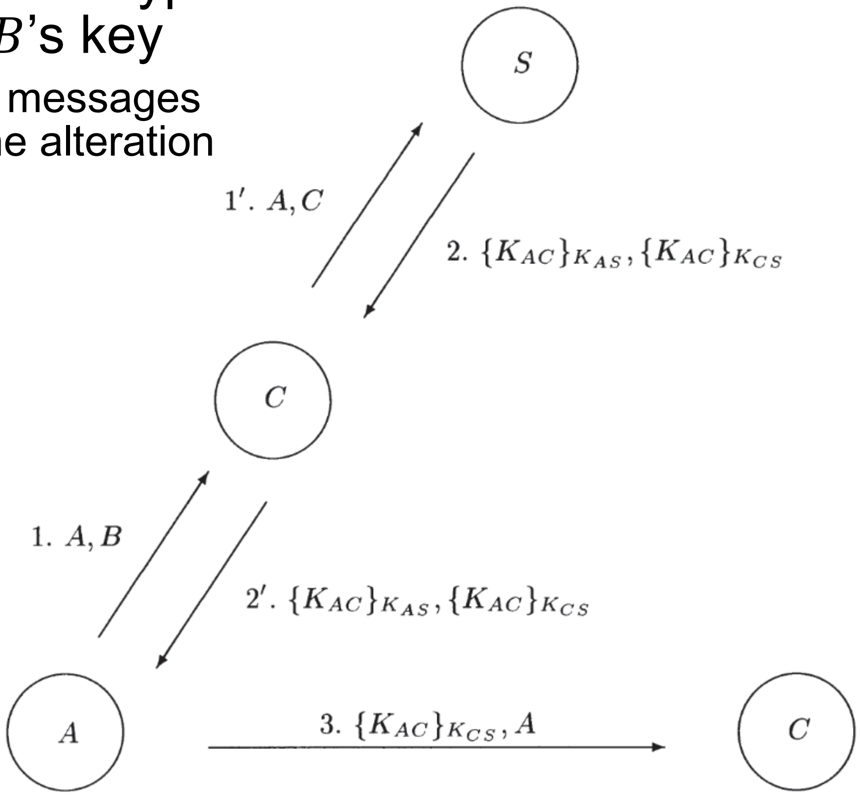

S
1′. A, C
2. $\{K_{AC}\}_{K_{AS}}, \{K_{AC}\}_{K_{CS}}$
C
1. A, B
2′. $\{K_{AC}\}_{K_{AS}}, \{K_{AC}\}_{K_{CS}}$
A
3. $\{K_{AC}\}_{K_{CS}}, A$
C

Fig. 1.4. Alternative attack on second protocol attempt

Building a Key Establishment Protocol

- Authentication
  - **Security Assumption 3**
    - *The adversary may be a legitimate protocol participant (an insider), or an external party (an outsider), or a combination of both*
  - To overcome the attack
    - The identity of the users who are to share K_{AB} need to be bound cryptographically to the value of K_{AB}
    - In other word, the identity of A and B are included in the encrypted messages received from S
    - It is a necessary property of the encryption algorithm used by S that it is not possible to alter the value of the encrypted messages

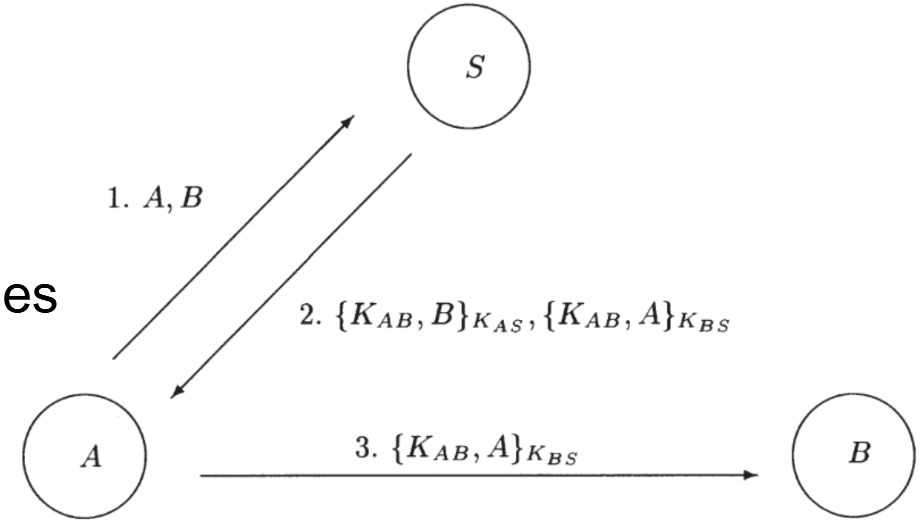

S
1. A, B
2. $\{K_{AB}, B\}_{K_{AS}}, \{K_{AB}, A\}_{K_{BS}}$
A
3. $\{K_{AB}, A\}_{K_{BS}}$
B

Fig. 1.5. Third protocol attempt

Building a Key Establishment Protocol

- Replay
  - Reason that a new key is generated for each session
    1. Session keys are expected to be vulnerable to attack
       - They may be placed in relatively insecure storage and could easily be discarded carelessly after the session is closed
    2. Communications in different sessions should be separated
       - In particular, it should not be possible to replay messages from previous sessions
  - For these reasons a whole class of attacks becomes possible based on the notion that old keys may be replayed in a subsequent session
  - Notice that even if A is careful in the management of session keys used by her, compromise of a session key by B may still allow replay attacks when A communicates with B

Building a Key Establishment Protocol

- Replay
  - **Security Assumption 4**
    - *An adversary is able to obtain the value of the session key K_{AB} used in any sufficiently old previous run of the protocol*
  - Replay attack on protocol
    - C intercepts the message from A to S
      - Indeed S plays no part in the protocol
      - The key K'_{AB} is an old session key used by user A and B in a previous session
    - By Security Assumption 1, 4
      - C can be expected to know the encrypted messages via which K'_{AB} was transported to A and B
      - C can be expected to know the value of K'_{AB}

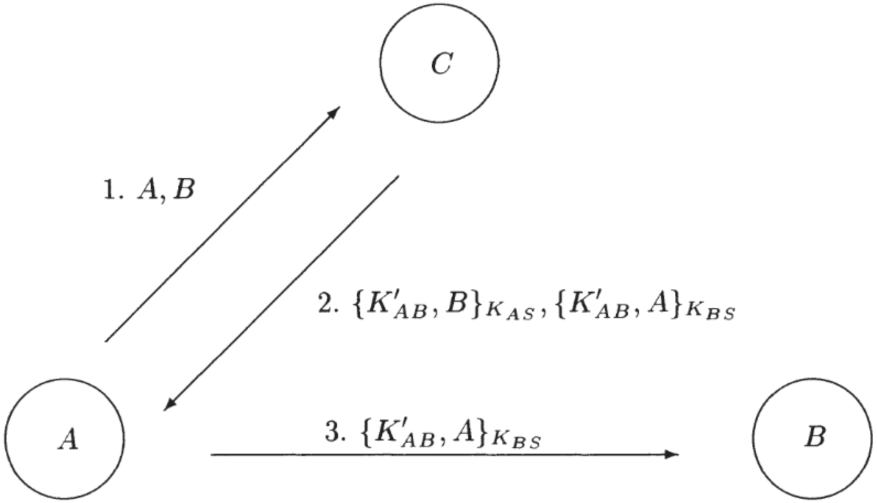

C
1. A, B
2. $\{K'_{AB}, B\}_{K_{AS}}, \{K'_{AB}, A\}_{K_{BS}}$
A
3. $\{K'_{AB}, A\}_{K_{BS}}$
B

Fig. 1.6. Attack on third protocol attempt

- Thus when A completes the protocol with B
  - C is able to decrypt subsequent information encrypted with K'_{AB}
  - C is able to alter messages whose integrity is protected by K'_{AB}

Building a Key Establishment Protocol

- Replay
  - **Definition 1.1**
    - *A nonce is a random value generated by one party and returned to that party to show that a message is newly generated*
  - Generate nonce on protocol
    - User A send nonce N_A to S at the start of the protocol together with the request for a new key
      - If this same value is received with the session key then A can deduce that the key has not been replayed
    - Since B does not directly contact the S
      - It is inconvenient for B to send his own nonce to S to be returned with K_{AB}
    - User B generate a nonce N_B and send this to A protected by K_{AB} itself

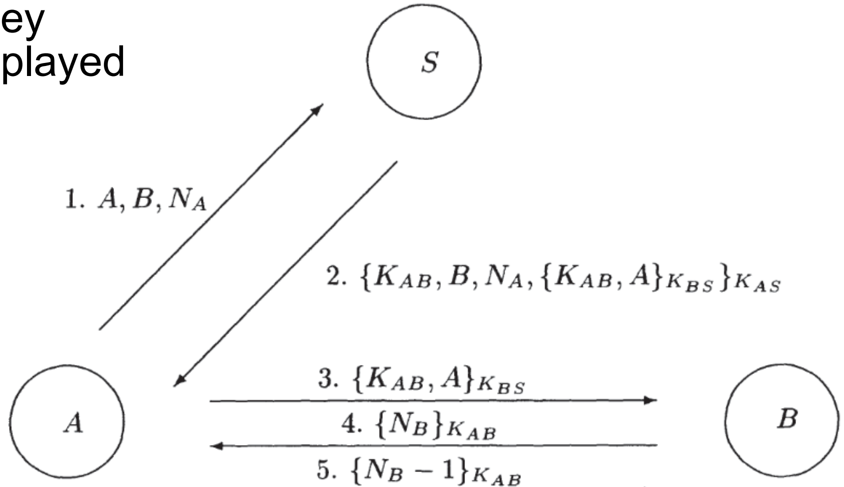

S
1. A, B, N_A
2. $\{K_{AB}, B, N_A, \{K_{AB}, A\}_{K_{BS}}\}_{K_{AS}}$
3. $\{K_{AB}, A\}_{K_{BS}}$
4. $\{N_B\}_{K_{AB}}$
5. $\{N_B - 1\}_{K_{AB}}$
A
B

Fig. 1.7. Fourth protocol attempt (Needham–Schroeder)

Building a Key Establishment Protocol

- Replay
  - The protocol in Fig 1.7
    - Their attack illustrates that there was a flaw in the above argument used to justify the protocol design
      - This can be pinpointed to an assumption that only A will be able to form a correct reply to message 4 from B
  - In the attack in Fig 1.8
    - Since the adversary C can be expected to know the value of an old session key, this assumption is unrealistic
    - C masquerades as A and is thus able to persuade B to use the old key K'_{AB}

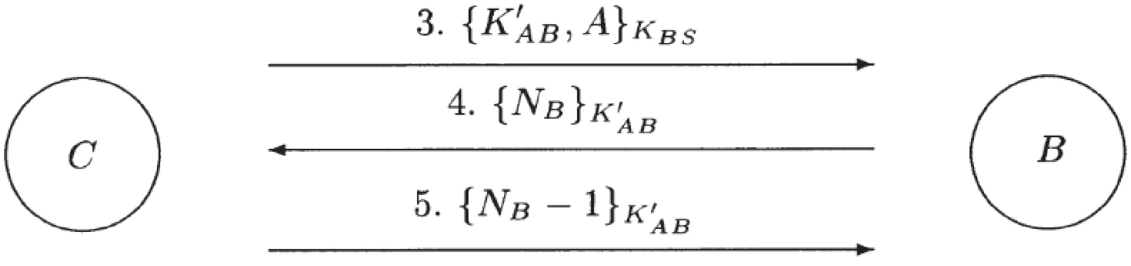


Fig. 1.8. Attack on fourth protocol attempt

Building a Key Establishment Protocol

- Replay
  - In the protocol of Fig 1.9
    - To enable both users to send their nonce value to S
      - The protocol is now initiated by B who sends his nonce, N_B, first to A
    - A adds her nonce N_A, and sends both to S who is now able to return K_{AB} in separate messages for A and B
      - Which can each be verified as fresh by their respective recipients
  - Property of key confirmation
    - It is achieved due to B's use of the key in message 4

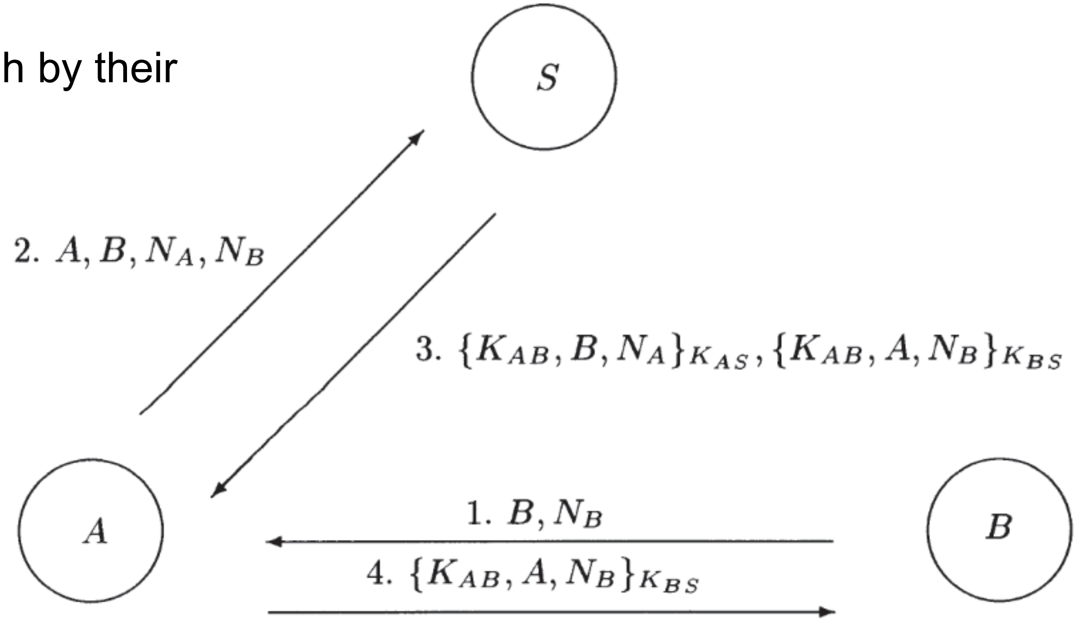

S
2. A, B, N_A, N_B
3. $\{K_{AB}, B, N_A\}_{K_{AS}}, \{K_{AB}, A, N_B\}_{K_{BS}}$
A
1. B, N_B
4. $\{K_{AB}, A, N_B\}_{K_{BS}}$
B

Fig. 1.9. Fifth protocol attempt

Protocol Architecture

- Protocol architecture in Section 1.2
  - Consisted of trusted server which generated the session key, and two other principals
- Consider alternative architecture in Section 1.3
  - There are three features that regard as architectural criteria to classify different protocols
    1. Which keys are already established
    2. How the session key is generated
    3. How many users a protocol is designed to serve

Protocol Architecture

- Existing Cryptographic Keys
  - As a matter of general principle
    - It is not possible to establish an authenticated session key without existing secure channels already being available
    - It is essential either that keys are already shared between different principals or that certified public keys are available
  - Conditions for establishing new session key
    - If two principals wish to establish a new session key there are essentially three possibilities
      1. The principals already share a secret key
      2. The principals possess certified public keys
      3. Each principals shares a key with a trusted server

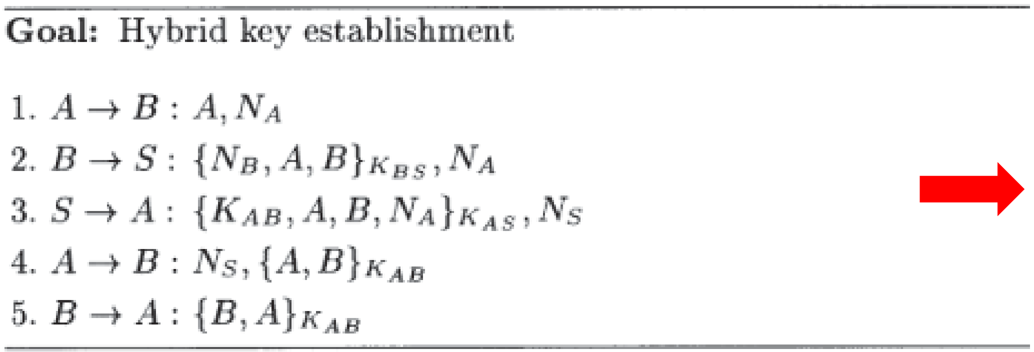
Protocol Architecture

- Method of Session Key Generation
  - There are various ways that may be employed to generate session keys in a *key establishment protocol*
    - **Definition 1.2**
      - *A key transport protocol is a key establishment protocol in which one of the principals generates the key and this key is then transferred to all protocol users*
    - **Definition 1.3**
      - *A key agreement protocol is a key establishment protocol in which the session key is a function of inputs* <u>*by all protocol users*</u>
      - → A protocol that generates a session key using information exchanged with each other through interactions between users
    - **Definition 1.4**
      - *A hybrid protocol is a key establishment protocol in which the session key is a function of inputs by more than one principal, but* <u>*not by all protocol users*</u>
      - *This means that the protocol is a key agreement protocol from the viewpoint of some users, and a key transport protocol from the viewpoint of others*

Protocol Architecture

- Example of hybrid protocol
  - The Initial statement
    - S: K_{AS}, K_{BS}, N_S
    - A: K_{AS}, A, N_A
    - B: K_{BS}, B, N_B
  - Session key is calculated as follow
    - $K_{AB} = f(N_B, N_S)$

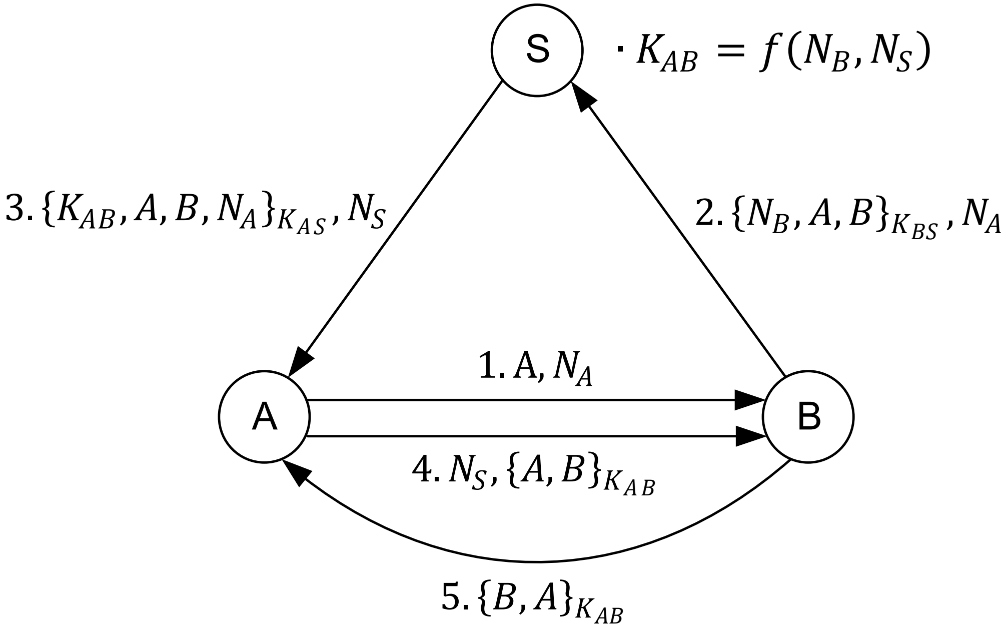
- Viewpoint of each users
  - From A's viewpoint, it looks like a *key transport protocol*
  - From B's viewpoint, this is like a *key agreement protocol* because B has input to the key



Goal: Hybrid key establishment

1. $A \rightarrow B : A, N_A$
2. $B \rightarrow S : \{N_B, A, B\}_{K_{BS}}, N_A$
3. $S \rightarrow A : \{K_{AB}, A, B, N_A\}_{K_{AS}}, N_S$
4. $A \rightarrow B : N_S, \{A, B\}_{K_{AB}}$
5. $B \rightarrow A : \{B, A\}_{K_{AB}}$

Protocol 1.2: A protocol in an unusual class


S
$\cdot K_{AB} = f(N_B, N_S)$
$3. \{K_{AB}, A, B, N_A\}_{K_{AS}}, N_S$
$2. \{N_B, A, B\}_{K_{BS}}, N_A$
$1. \text{A}, N_A$
A
B
$4. N_S, \{A, B\}_{K_{AB}}$
$5. \{B, A\}_{K_{AB}}$

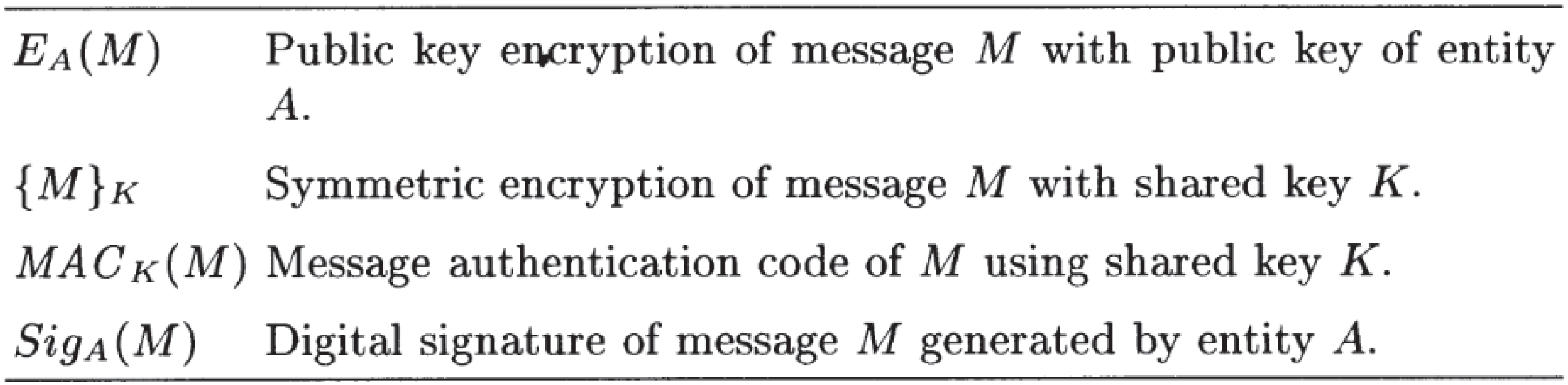
Cryptographic Properties

- Understanding property of cryptographic
  - Highlight the importance of distinguish the different properties that may be provided by cryptographic algorithms
  - There are 4 fundamental objectives
    - Confidentiality
      - It means that data is encrypted with keys shared between the parties to make the actual data inaccessible
    - Data integrity
      - It ensures that data has not been altered by unauthorized entities
    - Data origin authentication
      - It guarantees the origin of data
      - Since altering the data must alter its origin, we may say that data origin authentication implies data integrity
    - Non-repudiation
      - It ensures that entities cannot deny sending data that they have committed
      - This is typically provided using a digital signature mechanism

Cryptographic Properties

- Definitions for the cryptographic mechanisms
  - Consider cryptographic mechanisms that are typically used to provide the main cryptographic services
  - The definitions in Table 1.1
    - It is informal, but rely on an intuitive understanding of what it means for a computation to be easy or difficult

Table 1.1. Summary of notation for cryptographic algorithms



$E_A(M)$	Public key encryption of message M with public key of entity A.
$\{M\}_K$	Symmetric encryption of message M with shared key K.
$MAC_K(M)$	Message authentication code of M using shared key K.
$Sig_A(M)$	Digital signature of message M generated by entity A.

Freshness

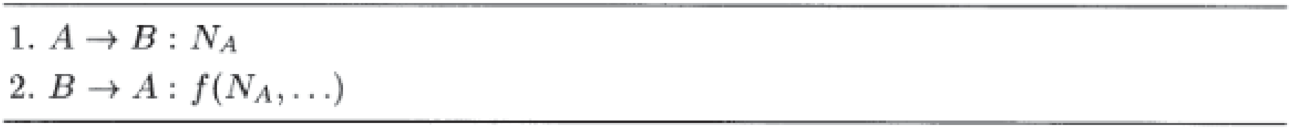
- Properties that determines if the key is new
  - Protocols designed to achieve authentication in real time need to ensure that messages sent are not replays
    - It need to ensure that message elements are new, or fresh, is a very common protocol requirement
  - A freshness value must have the property that it can be guaranteed not to have been used before
  - There are 3 common types of freshness
    - Timestamp
    - Nonce
    - Counter

Freshness

- Timestamp
  - Definition
    - It is a sequence of characters or encoded information that identifying when a certain event occurred
    - Time value used to generate session key
      - Usually giving date and time of day, sometimes accurate to a small fraction of a second
  - The sender adds the current time to the message when it is sent
    - This is checked by the recipient when the message is received by comparing with the local time
    - If the received timestamp is within an acceptable window of the current time, then the message is regarded as fresh
  - The difficulty of using timestamp
    - It is that synchronized time clocks are required and must be maintained securely

Freshness

- Nonce
  - Definition
    - It is an arbitrary random number used only once in a cryptographic communication
  - Use of a nonce
    - The recipient (A), of the message generates a nonce (N_A), and passes it to the sender (B)
    - The N_A is returned with the message after processing with some cryptographic function (f)



1. $A \rightarrow B : N_A$
2. $B \rightarrow A : f(N_A, \ldots)$

Protocol 1.3: Use of a nonce (random challenge)

  - Attention must be paid to the quality of random numbers produced
    - Since if the nonce to be used is predictable a valid reply can be obtained in advance and later replayed

Freshness

- Counter
    - Definition
        - A counter is a number that increases by 1
        - The sender and recipient maintain a synchronized counter
            - Counter value is sent with the message and then incremented
    - Disadvantage of counters
        - It is that state information must be maintained for each potential communication partner
        - If a counter value is not synchronized with the receiver, then preplay attacks are possible

Types of Attack on Protocols

- Overview
  - Purpose of this section
    - It is to summarize the attacks that can occur in secure protocol
  - Notes of caution
    1. For many protocols the list will not be complete
       - The ways in which the adversary may interact with one or more protocol runs are infinite
       - There are almost bound to be attacking possibilities that we have omitted
       - What is really required is confidence that it meets its security objectives given a known list of assumptions
       - On the other hand, we should not underestimate the usefulness of a list of typical weaknesses to check against
    2. Different protocols have different objectives
       - In our examples above the stated aim was to derive securely a new session key
       - However, some protocols may have no material to transfer confidentially, being concerned only with real time authentication
       - Whether or not a protocol achieves particular goals depends on what attacks are deemed possible

Types of Attack on Protocols

- Table 1.3 Types of protocol attack

Name	Description
Eavesdropping	• The adversary captures the information sent in the protocol.
Modification	• The adversary alters the information sent in the protocol.
Replay	• The adversary records information seen in the protocol and then sends it to the same, or a different, principal, possibly during a later protocol run.
Preplay	• The adversary engages in a run of the protocol prior to a run by the legitimate principals.
Reflection	• The adversary sends protocol messages back to the principal who sent them.
Denial of Service	• The adversary prevents or hinders legitimate principals from completing the protocol.
Typing Attacks	• The adversary replaces a (normally encrypted) protocol message field of one type with a (normally encrypted) message field of another type.
Cryptanalysis	• The adversary gains some useful leverage from the protocol to help in cryptanalysis.
Certificate Manipulation	• The adversary chooses or modifies certificate information to attack one or more protocol runs.
Protocol Interaction	• The adversary chooses a new protocol to interact with a known protocol.

Design Principles for Cryptographic Protocols

- Rules of thumb
  - Definition
    - A procedure or standard that can be easily learned and applied based on actual experience rather than scientific or mathematical theory
  - Abadi and Needham have proposed a set of principles intended to act as "rules of thumb" for protocol designers
    - They were derived from observation of the most common errors that have been found in published protocols

Design Principles for Cryptographic Protocols

- Rules of thumb

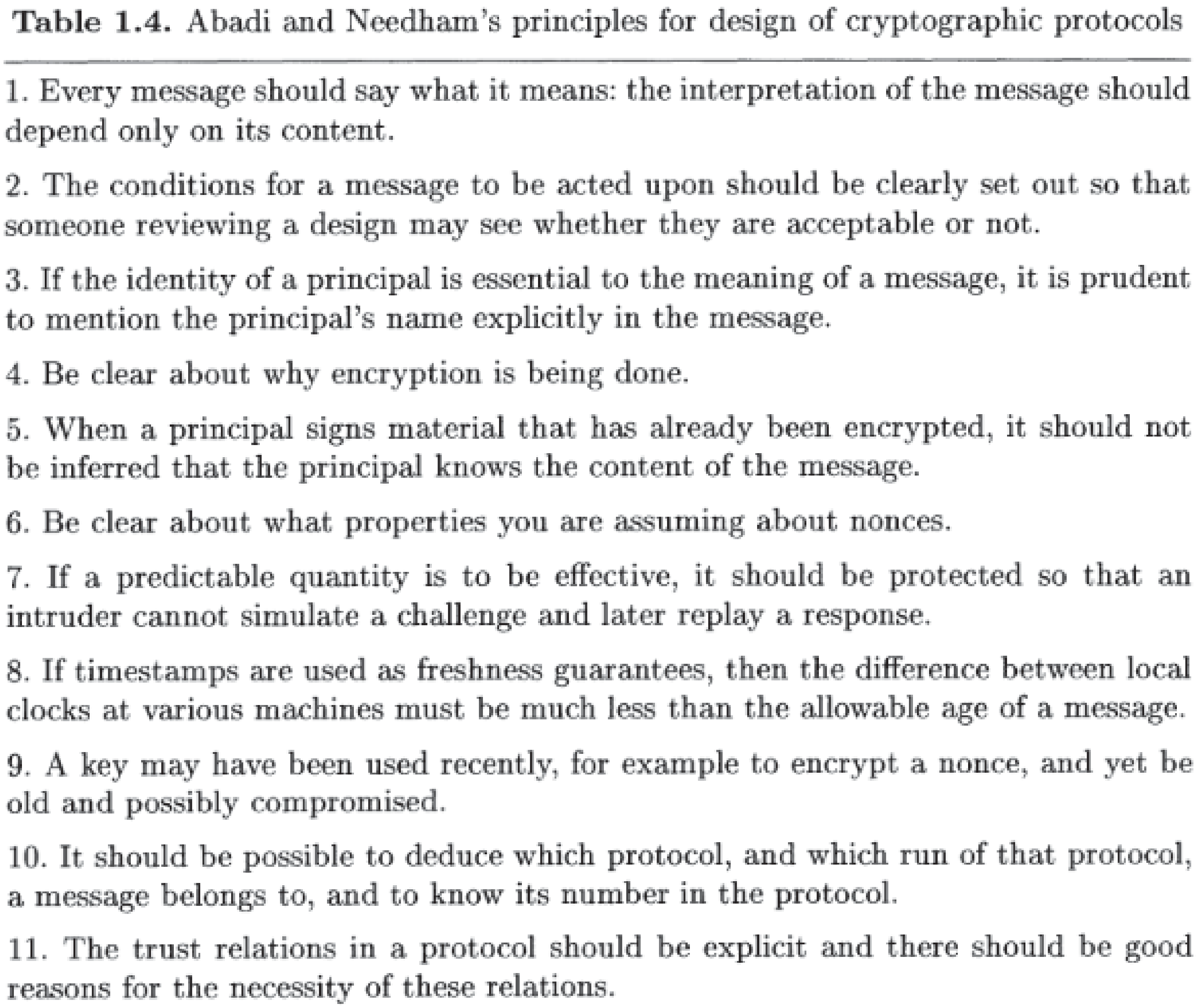


Table 1.4. Abadi and Needham's principles for design of cryptographic protocols

1. Every message should say what it means: the interpretation of the message should depend only on its content.

2. The conditions for a message to be acted upon should be clearly set out so that someone reviewing a design may see whether they are acceptable or not.

3. If the identity of a principal is essential to the meaning of a message, it is prudent to mention the principal's name explicitly in the message.

4. Be clear about why encryption is being done.

5. When a principal signs material that has already been encrypted, it should not be inferred that the principal knows the content of the message.

6. Be clear about what properties you are assuming about nonces.

7. If a predictable quantity is to be effective, it should be protected so that an intruder cannot simulate a challenge and later replay a response.

8. If timestamps are used as freshness guarantees, then the difference between local clocks at various machines must be much less than the allowable age of a message.

9. A key may have been used recently, for example to encrypt a nonce, and yet be old and possibly compromised.

10. It should be possible to deduce which protocol, and which run of that protocol, a message belongs to, and to know its number in the protocol.

11. The trust relations in a protocol should be explicit and there should be good reasons for the necessity of these relations.

Thanks!

최 창 준 (changjun@pel.sejong.ac.kr)