

Cryptographic protocols – introduction

Martin Stanek

Department of Computer Science
Comenius University
`stanek@dcs.fmph.uniba.sk`

Cryptology 1 (2021/22)

Content

Introduction

Some protocols and basic notions

- Diffie-Hellman protocol

- Interlock protocol

- Dolev-Yao model

- Freshness – nonces and timestamps

Basic protocols and attacks

- Needham-Schroeder protocol

- WMF protocol

Attacks

- Replay attacks and symmetry – WMF, NSPK

- Implementation issues (Otway-Rees)

- Denning-Sacco protocol

Introduction

- ▶ cryptographic protocols
 - ▶ goals: secrecy, authentication, integrity, anonymity, unlinkability, ...
 - ▶ environment: untrusted channels, dishonest participants
- ▶ our focus: authentication and key agreement (session-key)
- ▶ session-key
 - ▶ less data for cryptanalysis
 - ▶ logical separation of data from different sessions
 - ▶ using symmetric constructions for confidentiality and authenticity
- ▶ IPsec (IKE), TLS (handshake), SSH, WPA3 (SAE/Dragonfly), Noise Protocol Framework, ...
- ▶ prerequisite for secure communication
- ▶ various proposals (requirements, capabilities, environment)
- ▶ other protocols (not discussed here):
 - ▶ voting, money, private information retrieval, multiparty computation, etc.

Diffie-Hellman protocol

- ▶ two principals A, B
- ▶ shared group G of prime order q with generator g
 - ▶ public, known to everyone (e.g. an attacker)
- ▶ goal: key agreement
- ▶ DH protocol:
 1. $A \rightarrow B: g^a$, for random $a \in \mathbb{Z}_q$
 2. $B \rightarrow A: g^b$, for random $b \in \mathbb{Z}_q$
 - ▶ A computes $K = (g^b)^a = g^{ab}$, and B computes $K = (g^a)^b = g^{ab}$
 - ▶ the shared secret can be used to derive a symmetric key(s)
- ▶ passive adversary: CDH problem $g^a, g^b \rightarrow g^{ab}$

MITM attack

- ▶ active adversary in DH protocol
 - ▶ can intercept and change messages
- ▶ man-in-the-middle attack (M is an attacker):
 1. $A \rightarrow M(B): g^a$
 2. $M(A) \rightarrow B: g^x$
 3. $B \rightarrow M(A): g^b$
 4. $M(B) \rightarrow A: g^y$
 - ▶ A computes $K_A = g^{ay}$, B computes $K_B = g^{xb}$
 - ▶ M can compute $K_A = (g^a)^y = g^{ay}$ as well as $K_B = (g^b)^x = g^{bx}$
 - ▶ M can “translate” messages between A and B (or create his/her own)
- ▶ Can M enforce $K_A = K_B$ in the MITM attack?
 - ▶ if not, M should be there till the end or “simulate” a connection error
 - ▶ $g^x = g^y = 1 \Rightarrow K_A = K_B = 1$

Fixing DH protocol

- ▶ authenticate messages in the protocol
- ▶ additional assumptions – PKI (distribution of authentic public keys), preshared secrets, etc.
- ▶ DH in various forms is a base for majority of key agreement protocols
- ▶ Station-to-Station protocol:
 1. $A \rightarrow B: g^a$
 2. $B \rightarrow A: g^b, \text{Cert}_B, E_K(\text{Sig}_B(g^b, g^a))$
 3. $A \rightarrow B: \text{Cert}_A, E_K(\text{Sig}_A(g^a, g^b))$
 - ▶ key agreement and authentication of participants
 - ▶ the shared secret $K = g^{ab}$
 - ▶ Sig_U denotes signature produced by user U
 - ▶ certificates contain public keys for verifying signatures
 - ▶ E is symmetric encryption and “proves” the possession of K

Interlock protocol

- ▶ idea: let's force the MITM attacker to be “active” in the communication
- ▶ scenarios (possible MITM attack):
 - ▶ after unauthenticated DH protocol
 - ▶ after unauthenticated distribution of key using asymmetric encryption
- ▶ A/B wants to send a message m_A/m_B to B/A
- ▶ both encrypt their message and exchange halves of the ciphertexts (c_A/c_B), and then the other halves:
 1. $A \rightarrow B: c_{A1}$ (first half of c_A)
 2. $B \rightarrow A: c_{B1}$ (first half of c_B)
 3. $A \rightarrow B: c_{A2}$ (second half of c_A)
 4. $B \rightarrow A: c_{B2}$ (second half of c_B)
- ▶ a half of the ciphertext should be useless for the recipient
 - ▶ e.g. even/odd bits, encryption combined with MAC, ...

Interlock protocol (2)

- ▶ assume MITM attacker M and keys K_A and K_B used for $A \leftrightarrow M$ and $M \leftrightarrow B$ communications, respectively
- ▶ M can send the original message to A or B but not both
- ▶ example (sending m_B to A)
 1. $A \rightarrow M(B): c_{A1}$ (first half of c_A)
 2. $M(A) \rightarrow B: c'_{A1}$ (first half of c'_A , for some made-up m'_A)
 3. $B \rightarrow M(A): c_{B1}$ (first half of c_B)
 4. $M(A) \rightarrow B: c'_{A2}$ (second half of c'_A)
 5. $B \rightarrow M(A): c_{B2}$ (second half of c_B , M can decrypt m_B)
 6. $M(B) \rightarrow A: c'_{B1}$ (first half of c'_B , M encrypts m_B with K_A)
 7. $A \rightarrow M(B): c_{A2}$ (second half of c_A , M can decrypt m_A)
 8. $M(B) \rightarrow A: c'_{B2}$ (second half of c'_B , **A decrypts m_B**)
- ▶ Can we detect made-up messages?
 - ▶ phones – reading aloud the messages from interlock protocol or session-key checksum (voice synthesis ?)

Dolev-Yao model

- ▶ the adversary controls the network completely
 - ▶ eavesdrop, forge, delete, inject, replay, redirect messages
 - ▶ perform any computation with data (and keys) learned or possessed
- ▶ very strong (but appropriate) model
- ▶ protocol secure in DY model will be secure also in a weaker model
- ▶ sometimes weaker model is assumed in practice:
 - ▶ verification SMS sent to a mobile phone
- ▶ we will assume DY model in this lecture

Assumptions

- ▶ ideal cryptography:
 - ▶ perfect encryption, signatures, hash functions, message authentication codes, random number generators etc.
 - ▶ even more, e.g. encrypted messages cannot be manipulated without detection, no info about message without a key, tuples with two or three messages cannot be confused etc.
- ▶ flawless implementation (see history of problems in SSL/TLS)
 - ▶ instantiation of crypto algorithms (e.g. oracle padding attacks (POODLE), combination with compression (CRIME, BREACH))
 - ▶ getting implementation right (e.g. Heartbleed, export versions of algorithms, timing attacks, Bleichenbacher's attack)
- ▶ even then the analysis is non-trivial

What is wrong with this protocol?

- ▶ A generates a session-key K and sends it encrypted and signed to B
- ▶ one-way authentication and key distribution
 1. $A \rightarrow B: E_B(A, B, K), \text{Sig}_A(E_B(A, B, K))$
 - ▶ assumptions: A knows the public key of B (for asymmetric encryption), B knows the public key of A (for signature verification)
 - ▶ B verifies the signature and decrypts K
- ▶ the problem: replay attack
 - ▶ after K leaks the attacker can replay the message
 - ▶ B is tricked into using K as a good key for communication with A

What is wrong with this protocol?

- ▶ A generates a session-key K and sends it encrypted and signed to B
- ▶ one-way authentication and key distribution
 1. $A \rightarrow B: E_B(A, B, K), \text{Sig}_A(E_B(A, B, K))$
 - ▶ assumptions: A knows the public key of B (for asymmetric encryption), B knows the public key of A (for signature verification)
 - ▶ B verifies the signature and decrypts K
- ▶ the problem: replay attack
 - ▶ after K leaks the attacker can replay the message
 - ▶ B is tricked into using K as a good key for communication with A

Freshness of messages

- ▶ prevention of replay attacks: nonces and timestamps
- ▶ nonce
 - ▶ usually sufficiently long random string/number (i.e. unpredictable); (sometimes “unique” is sufficient)
 - ▶ used just for a particular instance of the protocol
 - ▶ unlikely to be present in some previous instances of the protocol
 - ▶ usually the confidentiality is not needed
 - ▶ examples: SSL/TLS, IKEv2
- ▶ timestamp
 - ▶ sufficiently precise time information included into a message
 - ▶ somewhat synchronized clocks are required
 - ▶ clock manipulation should be prevented
 - ▶ example: Kerberos

Needham-Schroeder protocol

- ▶ the protocol uses trusted third party – server S
 - ▶ S shares symmetric keys with participants (K_U with participant U)
- ▶ participants A and B
- ▶ goals: authentication and distribution of a session-key K_{AB}
- ▶ assumptions: N_A/N_B nonces generated by A/B ,
- ▶ the protocol:
 1. $A \rightarrow S: A, B, N_A$
 2. $S \rightarrow A: \{N_A, K_{AB}, B, \{K_{AB}, A\}_{K_B}\}_{K_A}$
 3. $A \rightarrow B: \{K_{AB}, A\}_{K_B}$
 4. $B \rightarrow A: \{N_B\}_{K_{AB}}$
 5. $A \rightarrow B: \{N_B - 1\}_{K_{AB}}$
- ▶ positives: S involved only once, stateless, ...
- ▶ insecure!

Needham-Schroeder protocol

- ▶ the protocol uses trusted third party – server S
 - ▶ S shares symmetric keys with participants (K_U with participant U)
- ▶ participants A and B
- ▶ goals: authentication and distribution of a session-key K_{AB}
- ▶ assumptions: N_A/N_B nonces generated by A/B ,
- ▶ the protocol:
 1. $A \rightarrow S: A, B, N_A$
 2. $S \rightarrow A: \{N_A, K_{AB}, B, \{K_{AB}, A\}_{K_B}\}_{K_A}$
 3. $A \rightarrow B: \{K_{AB}, A\}_{K_B}$
 4. $B \rightarrow A: \{N_B\}_{K_{AB}}$
 5. $A \rightarrow B: \{N_B - 1\}_{K_{AB}}$
- ▶ positives: S involved only once, stateless, ...
- ▶ insecure!

Attacking Needham-Schroeder protocol

- ▶ attack found by Denning and Sacco
- ▶ weakness: B cannot verify the freshness of K_{AB}
- ▶ the attacker can force B to accept a compromised K_{AB} (by cryptanalysis or by leak)
- ▶ the attack (M knows K_{AB} and thus (s)he can finish the protocol):
 3. $M(A) \rightarrow B: \{K_{AB}, A\}_{K_B}$ (replay of old message)
 4. $B \rightarrow M(A): \{N'_B\}_{K_{AB}}$
 5. $M(A) \rightarrow B: \{N'_B - 1\}_{K_{AB}}$
- ▶ How to fix the protocol?
 - ▶ e.g. A requests N_B from B at the beginning

Modified Wide Mouth Frog protocol

- ▶ participants A and B , trusted third party (server S)
- ▶ timestamps (T_U generated by U)
- ▶ S shares symmetric keys with participants
- ▶ goals: one-way authentication and distribution of the session-key K
- ▶ the protocol
 1. $A \rightarrow S: A, \{T_A, B, K\}_{K_A}$
 2. $S \rightarrow B: \{T_S, A, B, K\}_{K_B}$
- ▶ Original WMF:
 1. $A \rightarrow S: A, \{T_A, B, K\}_{K_A}$
 2. $S \rightarrow B: \{T_S, A, K\}_{K_B}$
 - ▶ Can you find a weakness?

Modified Wide Mouth Frog protocol

- ▶ participants A and B , trusted third party (server S)
- ▶ timestamps (T_U generated by U)
- ▶ S shares symmetric keys with participants
- ▶ goals: one-way authentication and distribution of the session-key K
- ▶ the protocol
 1. $A \rightarrow S: A, \{T_A, B, K\}_{K_A}$
 2. $S \rightarrow B: \{T_S, A, B, K\}_{K_B}$
- ▶ Original WMF:
 1. $A \rightarrow S: A, \{T_A, B, K\}_{K_A}$
 2. $S \rightarrow B: \{T_S, A, K\}_{K_B}$
 - ▶ Can you find a weakness?

Attacks

- ▶ examples:
 - ▶ Needham-Schroeder public-key protocol (1978) – attack, Lowe (1995)
 - ▶ various weaknesses in real-world protocols: PPTP, SSL/TLS, ...
 - ▶ WPA 4-way handshake (802.11i, 2004) - attack, Vanhoef (2017)
- ▶ Weaknesses/attack types
 - ▶ replay attacks
 - ▶ imprecise description
 - ▶ implementation issues
 - ▶ symmetry of messages
 - ▶ variable length of objects
 - ▶ interaction of protocols, etc.
- ▶ usually a combination of weaknesses and attack techniques

Replay attack

- ▶ “classic” example: Needham-Schroeder protocol
- ▶ Wide mouth frog protocol
 1. $A \rightarrow T : A, \{T_A, B, K\}_{K_A}$
 2. $T \rightarrow B : \{T_T, A, K\}_{K_B}$
- ▶ notation and assumptions:
 - ▶ T – trusted third party / server
 - ▶ K_A / K_B – symmetric key shared between A and T or B and T
 - ▶ T_A / T_T – time-stamp produced by A and T , respectively
- ▶ objectives:
 - ▶ distribution of session-key K
 - ▶ authentication of A (B is authenticated after the use of K)

Attacking WMF

- ▶ attack – repeating messages, employing their symmetry, and using T as an oracle:

$$1. A \rightarrow T : \quad A, \{T_A, B, K\}_{K_A}$$

$$2. T \rightarrow B : \quad \{T_T, A, K\}_{K_B}$$

$$3. E(B) \rightarrow T : \quad B, \{T_T, A, K\}_{K_B}$$

$$4. T \rightarrow E(A) : \quad \{T'_T, B, K\}_{K_A}$$

$$5. E(A) \rightarrow T : \quad A, \{T'_T, B, K\}_{K_A}$$

$$6. T \rightarrow E(B) : \quad \{T''_T, A, K\}_{K_B}$$

...

refreshing the time-stamp

after obtaining K (leak, cryptanalysis):

$$1'. E(A) \rightarrow T : \quad A, \{T^*_T, B, K\}_{K_A}$$

$$2'. T \rightarrow B : \quad \{T^{**}_T, A, K\}_{K_B}$$

- ▶ fix: break the symmetry, e.g. add sender's identifier (or message number) into the second message

Attacking WMF

- ▶ attack – repeating messages, employing their symmetry, and using T as an oracle:

$$1. A \rightarrow T : \quad A, \{T_A, B, K\}_{K_A}$$

$$2. T \rightarrow B : \quad \{T_T, A, K\}_{K_B}$$

$$3. E(B) \rightarrow T : \quad B, \{T_T, A, K\}_{K_B}$$

$$4. T \rightarrow E(A) : \quad \{T'_T, B, K\}_{K_A}$$

$$5. E(A) \rightarrow T : \quad A, \{T'_T, B, K\}_{K_A}$$

$$6. T \rightarrow E(B) : \quad \{T''_T, A, K\}_{K_B}$$

...

refreshing the time-stamp

after obtaining K (leak, cryptanalysis):

$$1'. E(A) \rightarrow T : \quad A, \{T^*_T, B, K\}_{K_A}$$

$$2'. T \rightarrow B : \quad \{T^{**}_T, A, K\}_{K_B}$$

- ▶ fix: break the symmetry, e.g. add sender's identifier (or message number) into the second message

- ▶ Needham-Schroeder public-key protocol (1978)
 1. $A \rightarrow B : \{A, N_A\}_{K_B}$
 2. $B \rightarrow A : \{N_A, N_B\}_{K_A}$
 3. $A \rightarrow B : \{N_B\}_{K_B}$
- ▶ notation and assumptions:
 - ▶ K_A / K_B – A 's / B 's public key
 - ▶ N_A / N_B – nonce produced by A / B
- ▶ objectives:
 - ▶ mutual (two-way) authentication of A and B
 - ▶ N_A and N_B can be used for session-key construction

Attacking NSPK

- ▶ attack – after initial message from A , E starts a session with B pretending to be A (both instances complete successfully):

1. $A \rightarrow E : \quad \{A, N_A\}_{K_E}$
- 1'. $E(A) \rightarrow B : \quad \{A, N_A\}_{K_B}$
- 2'. $B \rightarrow E(A) : \quad \{N_A, N_B\}_{K_A}$
2. $E \rightarrow A : \quad \{N_A, N_B\}_{K_A}$
3. $A \rightarrow E : \quad \{N_B\}_{K_E}$
- 3'. $E(A) \rightarrow B : \quad \{N_B\}_{K_B}$

- ▶ fix: e.g. adding an identifier of B into the second message:

1. $A \rightarrow B : \{A, N_A\}_{K_B}$
2. $B \rightarrow A : \{N_A, N_B, B\}_{K_A}$
3. $A \rightarrow B : \{N_B\}_{K_B}$

Attacking NSPK

- ▶ attack – after initial message from A , E starts a session with B pretending to be A (both instances complete successfully):

1. $A \rightarrow E : \{A, N_A\}_{K_E}$
- 1'. $E(A) \rightarrow B : \{A, N_A\}_{K_B}$
- 2'. $B \rightarrow E(A) : \{N_A, N_B\}_{K_A}$
2. $E \rightarrow A : \{N_A, N_B\}_{K_A}$
3. $A \rightarrow E : \{N_B\}_{K_E}$
- 3'. $E(A) \rightarrow B : \{N_B\}_{K_B}$

- ▶ fix: e.g. adding an identifier of B into the second message:

1. $A \rightarrow B : \{A, N_A\}_{K_B}$
2. $B \rightarrow A : \{N_A, N_B, B\}_{K_A}$
3. $A \rightarrow B : \{N_B\}_{K_B}$

Otway-Rees protocol

- ▶ Otway, Rees (1987)

1. $A \rightarrow B : M, A, B, \{N_A, M, A, B\}_{K_A}$
2. $B \rightarrow T : M, A, B, \{N_A, M, A, B\}_{K_A}, \{N_B, M, A, B\}_{K_B}$
3. $T \rightarrow B : M, \{N_A, K\}_{K_A}, \{N_B, K\}_{K_B}$
4. $B \rightarrow A : M, \{N_A, K\}_{K_A}$

- ▶ notation and assumptions:

- ▶ T – trusted server
- ▶ K_A / K_B – symmetric key shared between A / B and T
- ▶ N_A / N_B – nonce produced by A / B
- ▶ M – randomly chosen identifier of this protocol run

- ▶ objectives:

- ▶ distribution of session-key K
- ▶ authentication of A (B is authenticated after the use of K)

Attacking Otway-Rees protocol – attack 1

- ▶ implementation issue
- ▶ attack: improper block cipher mode – ECB:
 - ▶ let $|N_B|$ be a multiply of block length
 - ▶ encrypted nonce can be replaced in $\{N_B, K\}_{K_B}$
 - ▶ result: old session-key can be forced for use in a new session

Attacking Otway-Rees protocol – attack 2

again an implementation issue

attack: improper block cipher mode – CBC:

- ▶ assumption: the plaintext N_B, M, A, B fits into 3 blocks:

$$P_1 = N_B, \quad P_2 = M, \quad P_3 = A, B.$$

- ▶ CBC: random IV , encrypted and prepended as the first block of ciphertext
- ▶ attacker E starts the first protocol instance with B :
 1. $E \rightarrow B$: $M', E, B, \{N_E, M', E, B\}_{K_E}$
 2. $B \rightarrow E(T)$: $M', E, B, \{N_E, M', E, B\}_{K_E}, \{N'_B, M', E, B\}_{K_B}$
- ▶ $\{N'_B, M', E, B\}_{K_B} = \{IV'\}_{K_B}, C'_1, C'_2, C'_3$

...attack 2 continues

- ▶ E starts the second instance with B , pretending to be A :

$$1'. E(A) \rightarrow B: M, A, B, \{N_E, M, E, B\}_{K_E}$$

$$2'. B \rightarrow E(T): M, A, B, \{N_E, M, E, B\}_{K_E}, \{N_B, M, A, B\}_{K_B}$$

- ▶ let $\{N_B, M, A, B\}_{K_B} = \{IV\}_{K_B}, C_1, C_2, C_3$

- ▶ E modifies the intercepted message and sends to T :

$$3'. E(B) \rightarrow T: S, E, B, \{N_E, S, E, B\}_{K_E}, X$$

where $X = \{IV\}_{K_B}, C_1, C'_2, C'_3$

...attack 2 continues

- ▶ decrypting X :

$$\begin{aligned} D_{K_B}(X) &= N_B, C_1 \oplus D_{K_B}(C'_2), C'_2 \oplus D_{K_B}(C'_3) \\ &= N_B, C_1 \oplus M' \oplus C'_1, E, B. \end{aligned}$$

- ▶ E sets $S = C_1 \oplus M' \oplus C'_1$
- ▶ $3'$ is a legitimate message from T 's point of view

4'. $T \rightarrow E(B) : S, \{N_E, K\}_{K_E}, \{N_B, K\}_{K_B}$

5'. $E(T) \rightarrow B : M, \{N_E, K\}_{K_E}, \{N_B, K\}_{K_B}$

6'. $B \rightarrow E(A) : M, \{N_E, K\}_{K_E}$

- ▶ result: B thinks (s)he communicates with A ; E knows the key K

fix: add some redundant data into encrypted message (e.g. hash); use MAC, authenticated encryption etc.

...attack 2 continues

- ▶ decrypting X :

$$\begin{aligned} D_{K_B}(X) &= N_B, C_1 \oplus D_{K_B}(C'_2), C'_2 \oplus D_{K_B}(C'_3) \\ &= N_B, C_1 \oplus M' \oplus C'_1, E, B. \end{aligned}$$

- ▶ E sets $S = C_1 \oplus M' \oplus C'_1$
- ▶ $3'$ is a legitimate message from T 's point of view

$$4'. T \rightarrow E(B) : S, \{N_E, K\}_{K_E}, \{N_B, K\}_{K_B}$$

$$5'. E(T) \rightarrow B : M, \{N_E, K\}_{K_E}, \{N_B, K\}_{K_B}$$

$$6'. B \rightarrow E(A) : M, \{N_E, K\}_{K_E}$$

- ▶ result: B thinks (s)he communicates with A ; E knows the key K

fix: add some redundant data into encrypted message (e.g. hash); use MAC, authenticated encryption etc.

Imprecise description of protocol – attack 3

- ▶ let's assume that T does not check the consistence of plaintext and encrypted data

- ▶ attack:

1'. $A \rightarrow B :$ $M, A, B, \{N_A, M, A, B\}_{K_A}$
2'. $B \rightarrow E(T) :$ $M, A, B, \{N_A, M, A, B\}_{K_A}, \{N_B, M, A, B\}_{K_B}$
3'. $E \rightarrow T :$ $M, A, E, \{N_A, M, A, B\}_{K_A}, \{N_E, M, A, B\}_{K_E}$
4'. $T \rightarrow E :$ $M, \{N_A, K\}_{K_A}, \{N_E, K\}_{K_E}$
5'. $E(B) \rightarrow A :$ $M, \{N_A, K\}_{K_A}$

- ▶ result:

- ▶ A assumes to be communicating with B
- ▶ E impersonates B and E knows the session-key K

Improper length of objects – attack 4

▶ let $|K| = |M, A, B|$

1'. $A \rightarrow E(B) : M, A, B, \{N_A, M, A, B\}_{K_A}$

4'. $E(B) \rightarrow A : M, \{N_A, M, A, B\}_{K_A}$

▶ result:

▶ A assumes the communication with B

▶ E impersonates B and E knows the “session-key” (regardless of mode used for encryption)

▶ general observation: messages should be bounded to the particular step of the protocol

Symmetry of messages

- ▶ examples: NSPK protocol, WMF protocol
- ▶ usually multiple simultaneous protocol instances
- ▶ protocol for mutual authentication:
 1. $A \rightarrow B : N_A$
 2. $B \rightarrow A : \{N_A, N_B\}_K$
 3. $A \rightarrow B : N_B$
- ▶ notation and assumptions:
 - ▶ K – symmetric key shared between A and B
 - ▶ N_A / N_B – nonce generated by A and B , respectively

Attack employing the symmetry

1. $A \rightarrow E(B) : N_A$
 - 1'. $E(B) \rightarrow A : N_A$
 - 2'. $A \rightarrow E(B) : \{N_A, N'_A\}_K$
 2. $E(B) \rightarrow A : \{N_A, N'_A\}_K$
 3. $A \rightarrow E(B) : N'_A$
-
- 3'. $E(B) \rightarrow A : N'_A$

- ▶ result: A believes that (s)he communicates with B
- ▶ fix:
 - ▶ restrict the number of parallel runs or keeping track of recent nonces (not a good solution)
 - ▶ break the symmetry, e.g. insert participant's identifier into encrypted message

Denning-Sacco protocol

- ▶ Denning, Sacco (1981)

1. $A \rightarrow T : A, B$
2. $T \rightarrow A : C_A, C_B$
3. $A \rightarrow B : C_A, C_B, \{\{K, T_A\}_{K_A^{-1}}\}_{K_B}$

- ▶ notation and assumption:

- ▶ C_A / C_B – public-key certificate of A / B
- ▶ T_A – time-stamp produced by A
- ▶ K – session-key generated by A
- ▶ $\{X\}_{K_A^{-1}}$ – message X signed by A

- ▶ objectives:

- ▶ distribution of session-key K
- ▶ one-way authentication of A

Attacking Denning-Sacco protocol

Abadi (1994):

1. $A \rightarrow T :$ A, E
2. $T \rightarrow A :$ C_A, C_E
3. $A \rightarrow E :$ $C_A, C_E, \{\{K, T_A\}_{K_A^{-1}}\}_{K_E}$
- 3'. $E(A) \rightarrow B :$ $C_A, C_B, \{\{K, T_A\}_{K_A^{-1}}\}_{K_B}$

- ▶ result: E authenticates as A for B with known session-key K
- ▶ fix: e.g. insert recipient identifier into the signed data

Attacking Denning-Sacco protocol

Abadi (1994):

1. $A \rightarrow T :$ A, E
2. $T \rightarrow A :$ C_A, C_E
3. $A \rightarrow E :$ $C_A, C_E, \{\{K, T_A\}_{K_A^{-1}}\}_{K_E}$
- 3'. $E(A) \rightarrow B :$ $C_A, C_B, \{\{K, T_A\}_{K_A^{-1}}\}_{K_B}$

- ▶ result: E authenticates as A for B with known session-key K
- ▶ fix: e.g. insert recipient identifier into the signed data

Protection of predictable data

- ▶ requesting a current time:
 1. $A \rightarrow S : A, N_A$
 2. $S \rightarrow A : \{T_S, N_A\}_{K_A}$
- ▶ if N_A is predictable:
 1. $E(A) \rightarrow S : A, N_A$
 2. $S \rightarrow E(A) : \{T_S, N_A\}_{K_A}$ (this can be use as a reply for A's request later)
- ▶ fix (doesn't work if N_A is a constant):
 1. $A \rightarrow S : A, \{N_A\}_{K_A}$
 2. $S \rightarrow A : \{T_S, \{N_A\}_{K_A}\}_{K_A}$

Protection of predictable data

- ▶ requesting a current time:
 1. $A \rightarrow S : A, N_A$
 2. $S \rightarrow A : \{T_S, N_A\}_{K_A}$
- ▶ if N_A is predictable:
 1. $E(A) \rightarrow S : A, N_A$
 2. $S \rightarrow E(A) : \{T_S, N_A\}_{K_A}$ (this can be use as a reply for A's request later)
- ▶ fix (doesn't work if N_A is a constant):
 1. $A \rightarrow S : A, \{N_A\}_{K_A}$
 2. $S \rightarrow A : \{T_S, \{N_A\}_{K_A}\}_{K_A}$

Formal methods for protocol security?

- ▶ formal methods and tools for reasoning about the security of cryptographic protocols
 - ▶ ProVerif, Scyther, OFMC, Tamarin, Verifpal ...
 - ▶ ...they help to increase our trust in protocol's security
 - ▶ what is modeled?
 - ▶ the implementation can change everything
- ▶ various protocols were analyzed formally – with some vulnerabilities found later (WPA 4-way handshake, TLS 1.3, ...)