Classical Protocols

Design and Verification of Security Protocols and Security Ceremonies

Programa de Pós-Graduacão em Ciências da Computacão
Dr. Jean Everson Martina

March-June 2018

UNIVERSIDADE FEDERAL
DE SANTA CATARINA

Protocols to See Today!

- Otway-Rees;

Protocols to See Today!

- Otway-Rees;
- Woo–Lam;

Protocols to See Today!

- Otway-Rees;
- Woo–Lam;
- Neuman–Stubblebine;

Protocols to See Today!

- Otway-Rees;
- Woo–Lam;
- Neuman–Stubblebine;
- Wide-Mouth Frog protocol;

Protocols to See Today!

- Otway-Rees;
- Woo–Lam;
- Neuman–Stubblebine;
- Wide-Mouth Frog protocol;
- Yahalom.

Otway-Rees Protocol

- By Dave Otway and Owen Rees in 1987;

Otway-Rees Protocol

- By Dave Otway and Owen Rees in 1987;
- A protocol for efficient mutual authentication (via a mutually trusted third party);

Otway-Rees Protocol

- By Dave Otway and Owen Rees in 1987;
- A protocol for efficient mutual authentication (via a mutually trusted third party);
- It assures both principal parties of the timeliness of the interaction without the use of clocks or double encipherment.

Otway-Rees Protocol

1. $A \rightarrow B : M, A, B, \{N_A, M, A, B\}_{K_{AS}}$

Otway-Rees Protocol

1. $A \rightarrow B : M, A, B, \{N_A, M, A, B\}_{K_{AS}}$
2. $B \rightarrow S : M, A, B, \{N_A, M, A, B\}_{K_{AS}}, \{N_B, M, A, B\}_{K_{BS}}$

Otway-Rees Protocol

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

Otway-Rees Protocol

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

Otway-Rees Protocol - Questions

- What are the assumptions?

Otway-Rees Protocol - Questions

- What are the assumptions?
- What seems to be the goal?

Otway-Rees Protocol - Questions

- What are the assumptions?
- What seems to be the goal?
- What might the principals believe after each step?

Otway-Rees Protocol - Questions

- What are the assumptions?
- What seems to be the goal?
- What might the principals believe after each step?
- What is missing?

Otway-Rees Protocol - Questions

- What are the assumptions?
- What seems to be the goal?
- What might the principals believe after each step?
- What is missing?
- Is it secure?

Two Attacks on Otway-Rees Protocol

- A malicious intruder can arrange for A and B to end up with different keys:

Two Attacks on Otway-Rees Protocol

- A malicious intruder can arrange for A and B to end up with different keys:
  1. After step 3, B has received K_{AB};

Two Attacks on Otway-Rees Protocol

- A malicious intruder can arrange for A and B to end up with different keys:
  1. After step 3, B has received K_{AB};
  2. An intruder then intercepts the fourth message;

Two Attacks on Otway-Rees Protocol

- A malicious intruder can arrange for A and B to end up with different keys:
  1. After step 3, B has received K_{AB};
  2. An intruder then intercepts the fourth message;
  3. The intruder resends message 2, so S generates a new key K'_{AB}, sent to B;

Two Attacks on Otway-Rees Protocol

- A malicious intruder can arrange for A and B to end up with different keys:
  1. After step 3, B has received K_{AB};
  2. An intruder then intercepts the fourth message;
  3. The intruder resends message 2, so S generates a new key K'_{AB}, sent to B;
  4. The intruder intercepts this message too, but sends to A the message $M, \{N_a, K_{AB}\}_{K_{AS}}$;

Two Attacks on Otway-Rees Protocol

- A malicious intruder can arrange for A and B to end up with different keys:
  1. After step 3, B has received K_{AB};
  2. An intruder then intercepts the fourth message;
  3. The intruder resends message 2, so S generates a new key K'_{AB}, sent to B;
  4. The intruder intercepts this message too, but sends to A the message $M, \{N_a, K_{AB}\}_{K_{AS}}$;
  5. A has K'_{AB}, while B has K_{AB}.

Two Attacks on Otway-Rees Protocol

- A malicious intruder can arrange for A and B to end up with different keys:
  1. After step 3, B has received K_{AB};
  2. An intruder then intercepts the fourth message;
  3. The intruder resends message 2, so S generates a new key K'_{AB}, sent to B;
  4. The intruder intercepts this message too, but sends to A the message $M, \{N_a, K_{AB}\}_{K_{AS}}$;
  5. A has K'_{AB}, while B has K_{AB}.
- Another problem: although the server tells B that A used a nonce, B doesn't know if this was a replay of an old message.

Woo–Lam Protocol

- Woo–Lam refers to various computer network authentication protocols designed by Simon S. Lam and Thomas Woo;

Woo–Lam Protocol

- Woo–Lam refers to various computer network authentication protocols designed by Simon S. Lam and Thomas Woo;
- The most common is the one published in 1992;

Woo–Lam Protocol

- Woo–Lam refers to various computer network authentication protocols designed by Simon S. Lam and Thomas Woo;
- The most common is the one published in 1992;
- The protocols enable two communicating parties to authenticate each other and to exchange session keys;

Woo–Lam Protocol

- Woo–Lam refers to various computer network authentication protocols designed by Simon S. Lam and Thomas Woo;
- The most common is the one published in 1992;
- The protocols enable two communicating parties to authenticate each other and to exchange session keys;
- It involves the use of a trusted key distribution center (KDC) to negotiate between the parties;

Woo–Lam Protocol

- Woo–Lam refers to various computer network authentication protocols designed by Simon S. Lam and Thomas Woo;
- The most common is the one published in 1992;
- The protocols enable two communicating parties to authenticate each other and to exchange session keys;
- It involves the use of a trusted key distribution center (KDC) to negotiate between the parties;
- Both symmetric-key and public-key variants have been described.

Woo–Lam Protocol

1. $A \rightarrow B : A$

Woo–Lam Protocol

1. $A \rightarrow B : A$
2. $B \rightarrow A : NB$

Woo–Lam Protocol

1. $A \rightarrow B : A$
2. $B \rightarrow A : NB$
3. $A \rightarrow B : \{NB\}_{K_{AS}}$

Woo–Lam Protocol

1. $A \rightarrow B : A$
2. $B \rightarrow A : NB$
3. $A \rightarrow B : \{NB\}_{K_{AS}}$
4. $B \rightarrow S : \{A, \{NB\}_{K_{AS}}\}_{K_{BS}}$

Woo–Lam Protocol

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

Woo–Lam Protocol - Questions

- What are the assumptions?

Woo–Lam Protocol - Questions

- What are the assumptions?
- What seems to be the goal?

Woo–Lam Protocol - Questions

- What are the assumptions?
- What seems to be the goal?
- What might the principals believe after each step?

Woo–Lam Protocol - Questions

- What are the assumptions?
- What seems to be the goal?
- What might the principals believe after each step?
- What is missing?

Woo–Lam Protocol - Questions

- What are the assumptions?
- What seems to be the goal?
- What might the principals believe after each step?
- What is missing?
- Is it secure?

Attack on Woo–Lam Protocol

- Two simultaneous inbound authentication attempts initiated by an attacker C, where C is also considered as any other regular participant;

Attack on Woo–Lam Protocol

- Two simultaneous inbound authentication attempts initiated by an attacker C, where C is also considered as any other regular participant;
- C pretends to be A in one and retains its own identity C for the other;

Attack on Woo–Lam Protocol

- Two simultaneous inbound authentication attempts initiated by an attacker C, where C is also considered as any other regular participant;
- C pretends to be A in one and retains its own identity C for the other;
- C obtains nonces from B for both runs and encrypts the nonce NB intended for A with its own server key and returns it to B, retaining its original identity;

Attack on Woo–Lam Protocol

- Two simultaneous inbound authentication attempts initiated by an attacker C, where C is also considered as any other regular participant;
- C pretends to be A in one and retains its own identity C for the other;
- C obtains nonces from B for both runs and encrypts the nonce NB intended for A with its own server key and returns it to B, retaining its original identity;
- When the nonce is returned by the server, it leads B to believe that it has authenticated A, whereas A has not even participated in either of the runs.
- The attack is complete.

Neuman–Stubblebine Protocol

- Created by Neuman and Stubblebine in 1993;

Neuman–Stubblebine Protocol

- Created by Neuman and Stubblebine in 1993;
- It allows individuals communicating over such a network to prove their identity to each other;

Neuman–Stubblebine Protocol

- Created by Neuman and Stubblebine in 1993;
- It allows individuals communicating over such a network to prove their identity to each other;
- This protocol utilizes time stamps, but does not depend on synchronized clocks;

Neuman–Stubblebine Protocol

- Created by Neuman and Stubblebine in 1993;
- It allows individuals communicating over such a network to prove their identity to each other;
- This protocol utilizes time stamps, but does not depend on synchronized clocks;
- It has an Establishment phase and a Communication phase.

Neuman–Stubblebine Protocol - Establishment

1. $A \rightarrow B : A, N_A$

Neuman–Stubblebine Protocol - Establishment

1. $A \rightarrow B : A, N_A$
2. $B \rightarrow S : B, N_B, \{A, N_A, T_B\}_{K_{BS}}$

Neuman–Stubblebine Protocol - Establishment

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

Neuman–Stubblebine Protocol - Establishment

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

Neuman–Stubblebine Protocol - Communication

1. $A \rightarrow B : \{A, K_{AB}, T_B\}_{K_{BS}}, N'_A$

Neuman–Stubblebine Protocol - Communication

1. $A \rightarrow B : \{A, K_{AB}, T_B\}_{K_{BS}}, N'_A$
2. $B \rightarrow A : N'_B, \{N'_A\}_{K_{AB}}$

Neuman–Stubblebine Protocol - Communication

1. $A \rightarrow B : \{A, K_{AB}, T_B\}_{K_{BS}}, N'_A$
2. $B \rightarrow A : N'_B, \{N'_A\}_{K_{AB}}$
3. $A \rightarrow B : \{N'_B\}_{K_{AB}}$

Neuman–Stubblebine Protocol - Questions

- What are the assumptions?

Neuman–Stubblebine Protocol - Questions

- What are the assumptions?
- What seems to be the goal?

Neuman–Stubblebine Protocol - Questions

- What are the assumptions?
- What seems to be the goal?
- What might the principals believe after each step?

Neuman–Stubblebine Protocol - Questions

- What are the assumptions?
- What seems to be the goal?
- What might the principals believe after each step?
- What is missing?

Neuman–Stubblebine Protocol - Questions

- What are the assumptions?
- What seems to be the goal?
- What might the principals believe after each step?
- What is missing?
- Is it secure?

Attacks on Neuman–Stubblebine Protocol

- Weidenbach Attack:
- The server can be used by the attacker to generate an arbitrary number of messages $\{A, K_{AB}, T_B\}_{K_{BS}}$. As the attacker knows that the only thing that changes is the key K_{AB} he can make the Server to generate material for known-plain text attacks;

Attacks on Neuman–Stubblebine Protocol

- Paradox Attack:
- While B sends message (2) to S, C intercepts the ciphertext A, N x, TbKbs and the nonce N b generated by B . C ignores the message (3) (bypasses Step (2) and Step (3)) and sends A, N x, ToKbs together with NbN x as the message (4) to B. Because both A, Nx, TbKbs and A, Kab, TbKbs have the same format B cannot distinguish one from the other;

Attacks on Neuman–Stubblebine Protocol

- Oracle Attack:
- C first intercepts the ticket in Establishment. Although C cannot decrypt, C may send the intercepted ticket and a forged nonce, N'_C, to B;

Attacks on Neuman–Stubblebine Protocol

- Oracle Attack:
- C first intercepts the ticket in Establishment. Although C cannot decrypt, C may send the intercepted ticket and a forged nonce, N'_C, to B;
- Upon receiving the message, B verifies the ticket. If it is valid, B responds $\{N'_C\}_{K_{AB}}$ and a new nonce, N'_B, to A;

Attacks on Neuman–Stubblebine Protocol

- Oracle Attack:
- C first intercepts the ticket in Establishment. Although C cannot decrypt, C may send the intercepted ticket and a forged nonce, N'_C, to B;
- Upon receiving the message, B verifies the ticket. If it is valid, B responds $\{N'_C\}_{K_{AB}}$ and a new nonce, N'_B, to A;
- Once C intercepts this message, he uses B as an oracle and starts a new session with B;

Attacks on Neuman–Stubblebine Protocol

- C sends the nonce N'_B, he just received coupled the same ticket to B. Upon receiving the message, B sends back $\{N'_C\}_{K_{AB}}$ and a new nonce, N''_B, to A;

Attacks on Neuman–Stubblebine Protocol

- C sends the nonce N'_B, he just received coupled the same ticket to B. Upon receiving the message, B sends back $\{N'_C\}_{K_{AB}}$ and a new nonce, N''_B, to A;
- X can intercept it and get the encrypted nonce $\{N'_B\}_{K_{AB}}$;

Attacks on Neuman–Stubblebine Protocol

- C sends the nonce N'_B, he just received coupled the same ticket to B. Upon receiving the message, B sends back $\{N'_C\}_{K_{AB}}$ and a new nonce, N''_B, to A;
- X can intercept it and get the encrypted nonce $\{N'_B\}_{K_{AB}}$;
- Finally, C successfully passes the first authentication session of B by sending the $\{N'_B\}_{K_{AB}}$ back to B.

Wide-Mouth Frog protocol Protocol

- The protocol was first described under the name "The Wide-mouthed-frog Protocol" in the paper "A Logic of Authentication" (1990), which introduced BAN Logic;

Wide-Mouth Frog protocol Protocol

- The protocol was first described under the name "The Wide-mouthed-frog Protocol" in the paper "A Logic of Authentication" (1990), which introduced BAN Logic;
- The paper gives no rationale for the protocol's whimsical name;

Wide-Mouth Frog protocol Protocol

- The protocol was first described under the name "The Wide-mouthed-frog Protocol" in the paper "A Logic of Authentication" (1990), which introduced BAN Logic;
- The paper gives no rationale for the protocol's whimsical name;
- It allows individuals communicating over a network to prove their identity to each other while also preventing eavesdropping or replay attacks, and provides for detection of modification and the prevention of unauthorized reading.

Wide-Mouth Frog protocol Protocol

1. $A \rightarrow S : A, \{T_A, B, K_{AB}\}_{K_{AS}}$

Wide-Mouth Frog protocol Protocol

1. $A \rightarrow S : A, \{T_A, B, K_{AB}\}_{K_{AS}}$
2. $S \rightarrow B : \{T_S, A, K_{AB}\}_{K_{BS}}$

Wide-Mouth Frog protocol Protocol

1. $A \rightarrow S : A, \{T_A, B, K_{AB}\}_{K_{AS}}$
2. $S \rightarrow B : \{T_S, A, K_{AB}\}_{K_{BS}}$

To prevent active attacks, some form of authenticated encryption (or message authentication) must be used.

Wide-Mouth Frog protocol - Questions

- What are the assumptions?

Wide-Mouth Frog protocol - Questions

- What are the assumptions?
- What seems to be the goal?

Wide-Mouth Frog protocol - Questions

- What are the assumptions?
- What seems to be the goal?
- What might the principals believe after each step?

Wide-Mouth Frog protocol - Questions

- What are the assumptions?
- What seems to be the goal?
- What might the principals believe after each step?
- What is missing?

Wide-Mouth Frog protocol - Questions

- What are the assumptions?
- What seems to be the goal?
- What might the principals believe after each step?
- What is missing?
- Is it secure?

Problems on Wide-Mouth Frog protocol

- A global clock is required;

Problems on Wide-Mouth Frog protocol

- A global clock is required;
- The server S has access to all keys;

Problems on Wide-Mouth Frog protocol

- A global clock is required;
- The server S has access to all keys;
- The value of the session key K_{AB} is completely determined by A, who must be competent enough to generate good keys;

Problems on Wide-Mouth Frog protocol

- A global clock is required;
- The server S has access to all keys;
- The value of the session key K_{AB} is completely determined by A, who must be competent enough to generate good keys;
- It can replay messages within the period when the timestamp is valid. A is not assured that B exists;

Problems on Wide-Mouth Frog protocol

- A global clock is required;
- The server S has access to all keys;
- The value of the session key K_{AB} is completely determined by A, who must be competent enough to generate good keys;
- It can replay messages within the period when the timestamp is valid. A is not assured that B exists;
- The protocol is stateful. This is usually undesired because it requires more functionality and capability from the server. For example, S must be able to deal with situations in which B is unavailable.

Attack on Wide-Mouth Frog protocol

- The value of the session key K_{AB} is completely determined by an untrusted peer in the protocol;

Attack on Wide-Mouth Frog protocol

- The value of the session key K_{AB} is completely determined by an untrusted peer in the protocol;
- A Man-in-the-middle attack is trivial;

Attack on Wide-Mouth Frog protocol

- The value of the session key K_{AB} is completely determined by an untrusted peer in the protocol;
- A Man-in-the-middle attack is trivial;
- C can deliberately reuse keys to defeat the protocols goals.

Yahalom Protocol

- Unpublished protocol. It appears on BAN Logic paper as personal communication by the authors with Yahalom;

Yahalom Protocol

- Unpublished protocol. It appears on BAN Logic paper as personal communication by the authors with Yahalom;
- Yahalom uses a trusted arbitrator to distribute a shared key between two people;

Yahalom Protocol

- Unpublished protocol. It appears on BAN Logic paper as personal communication by the authors with Yahalom;
- Yahalom uses a trusted arbitrator to distribute a shared key between two people;
- This protocol can be considered as an improved version of Wide Mouth Frog protocol.

Yahalom Protocol

1. $A \rightarrow B : A, N_A$

Yahalom Protocol

1. $A \rightarrow B : A, N_A$
2. $B \rightarrow S : B, \{A, N_A, N_B\}_{K_{BS}}$

Yahalom Protocol

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

Yahalom Protocol

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

Yahalom Protocol - Questions

- What are the assumptions?

Yahalom Protocol - Questions

- What are the assumptions?
- What seems to be the goal?

Yahalom Protocol - Questions

- What are the assumptions?
- What seems to be the goal?
- What might the principals believe after each step?

Yahalom Protocol - Questions

- What are the assumptions?
- What seems to be the goal?
- What might the principals believe after each step?
- What is missing?

Yahalom Protocol - Questions

- What are the assumptions?
- What seems to be the goal?
- What might the principals believe after each step?
- What is missing?
- Is it secure?

Attacks on Yahalom Protocol

- Bob completed his protocol execution believing he was communicating with Alice, but it actually was not so;

Attacks on Yahalom Protocol

- Bob completed his protocol execution believing he was communicating with Alice, but it actually was not so;
- Because the first encrypted chunk in the fourth message does not include the terms used for proving the freshness of the session key, such as NB, the encrypted chunk could be a replayed message.

Discussion

- What the classical protocols tell us in terms of designs?

Discussion

- What the classical protocols tell us in terms of designs?
- What the classical protocols tell us in terms of flaws?

Discussion

- What the classical protocols tell us in terms of designs?
- What the classical protocols tell us in terms of flaws?
- How most of these flaws were discovered?

Questions????

UNIVERSIDADE FEDERAL
DE SANTA CATARINA

This work is licensed under the Creative Commons Attribution 4.0 International License. To view a copy of this license, visit http://creativecommons.org/licenses/by/4.0/.

UNIVERSIDADE FEDERAL
DE SANTA CATARINA