Cryptography and Network Security, PART V: Proof Theory

Timo Karvi

11.2013

Before 1990, there were no formal proofs of key agreement protocols. Proof methods for cryptographical primitives had, however, started to develop in the 1980's. One of breakthroughs at that time, 1982, was the public key system of Goldwasser and Micali. They developed the concept of semantic security and proved formally that their system is correct. After 1990, the proof theory of protocols started to developed quickly:

- Bellare and Rogaway I, 1993.
- Bellare and Rogaway II, 1995.
- Bellare, Pointcheval, Rogaway, 2000.
- Canetti and Krawczyk, 2001.
- LaMacchia, Lauter and Mityagin, 2006.

Introduction II

We present the model of Bellare, Pointcheval and Rogaway. It is reasonably effective but at the same time straightforward, and it is easy after this model to familiarize oneself with later models. The basic game theoretic approach is the same in all the models.

Our presentation follows the book Kim-Kwang Raymond Choo: *Secure Key Establishment*, Springer 2008.

Adversarial Powers I

- Adversary $\mathcal{A}$ is a probalististic, polynomial-time machine that is in control of all communications between a fixed set of protocol participants.
- $\mathcal{A}$ interacts with a set of oracles $\Pi^i_{U_u,U_v}$, where $\Pi^i_{U_u,U_v}$ is the i'th instantiation of a protocol participants U_u and U_v. The participants wish to establish a common secret session key.
- $\mathcal{A}$ controls the channels via the queries to the targeted oracles. The possible queries are:
  - **Send**(U_u, U_v, i, m): $\Pi^i_{U_u,U_v}$ responds according to the protocol specification. Also reactions *accept* and *reject m* are delivered to $\mathcal{A}$.
  - **Session − Key − Reveal**(U_u, U_v, i): Any oracle $\Pi^i_{U_u,U_v}$, upon receiving this query, will send the session keys it possesses to $\mathcal{A}$.
  - **Session − State − Reveal**(U_u, U_v, i): The oracle $\Pi^i_{U_u,U_v}$ will send all its internal data to $\mathcal{A}$ except long-term secret parameters. However, the oracle answers only if it has no session keys.

Adversarial Powers II

- **Corrupt**(U_u, K_E): This query allows $\mathcal{A}$ to corrupt U_u and will thereby learn the complete internal state of U_u. K_E is the session key possessed by the participant.
- **Test**(U_u, U_v, i): If $\Pi^i_{U_u,U_v}$ has a session key, then when the query arrives, the oracle chooses a random bit b and sends to $\mathcal{A}$ either a random key or the actual session key. $\mathcal{A}$ succeeds if it can guess the bit b.

Definition of Security I

Security is defined using the game $\mathcal{G}$, played between $\mathcal{A}$ and a collection of oracles $\Pi^i_{U_u,U_v}$. Before defining the game, we need one oracle concept:

Definition

Oracle $\Pi^i_{U_u,U_v}$ is fresh or holds a fresh session key at the end of execution, if and only if

1. *$\Pi^i_{U_u,U_v}$ has terminated successfully with or without a partner oracle $\Pi^i_{U_v,U_u}$.*
2. *Both $\Pi^i_{U_u,U_v}$ and $\Pi^i_{U_v,U_u}$ have not been sent a* **Reveal** *query.*
3. *U_u and U_v have not been sent a* **Corrupt** *query.*

$\mathcal{A}$ runs the game $\mathcal{G}$, whose setting is as follows:

Definition of Security II

Stage 1: $\mathcal{A}$ can send oracle queries.

Stage 2: At some point during $\mathcal{G}$, $\mathcal{A}$ will choose a *fresh* session and send a **Test** query to a *fresh oracle* in the test session.

Stage 3: $\mathcal{A}$ continues to make oracle queries, but it cannot make **Corrupt** or **Session-Key-Reveal** queries.

Stage 4: Eventually, $\mathcal{A}$ terminates $\mathcal{G}$ and outputs a bit b'.

The advantage of $\mathcal{A}$ in the game is defined by the formula

$$\mathrm{Adv}^{\mathcal{A}} = |2 \cdot \mathrm{Prob}[b = b'] - 1|.$$

BPR 200 Model I

- Bellare, Pointcheval and Rogaway 2000.
- In this model it is assumed that sessions have identifiers (SIDs). Every message contains a SID which shows the session the message belongs to. Protocol designers can construct SIDs as they choose. But the way SIDs are constructed can have an impact on the security of the protocol in this model.
- An oracle who has terminated successfully (or *accepted* as the term is called) will hold the associated session key, a SID and a partner identifier.

BPR 200 Model II

Definition

Two oracles $\Pi^i_{A,B}$ *and* $\Pi^j_{B,A}$ *are partners, iff both oracles*

1. *have accepted the same session key with the same SID,*
2. *have agreed on the same set of principals (i.e. initiator and responder), and*
3. *no other oracles have terminated successfully with the same SID.*

BPR 200 Model III

Definition

A protocol is secure in the BPR 2000 model, if

a) Key Establishment: *For all probabilistic polynomial time adversaries $\mathcal{A}$, the advantage of $\mathcal{A}$, $\mathrm{Adv}^{\mathcal{A}}$ in game $\mathcal{G}$ is negligible.*

b) Entity authentication goal: *The probability of any probabilistic polynomial time adversary violating entity authentication is negligible.*

3PKD Protocol I

We present the basic version of the 3PKD protocol and show an attack against it. Then we present a modified protocol and prove that it is secure in the BPR 2000 model. The goal of the protocol are:

- to distribute a session key between two communicating partners with the help of a server;
- no forward secrecy;
- no mutual authentication;
- concurrent executions possible.

3PKD Protocol II

The protocol:

$$
\begin{array}{lll}
1. & A \longrightarrow B: & R_A \\
2. & B \longrightarrow S: & R_A,\ R_B \\
3a. & S \longrightarrow A: & \{SK_{AB}\}_{K^E_{AS}},\ \left[A, B, R_A, \{SK_{AB}\}_{K^E_{AS}}\right]_{K^{MAC}_{AS}} \\
3b. & S \longrightarrow B: & \{SK_{AB}\}_{K^E_{BS}},\ \left[A, B, R_A, \{SK_{AB}\}_{K^E_{BS}}\right]_{K^{MAC}_{BS}}
\end{array}
$$

However, there is an attack against this protocol:

$$
\begin{array}{lll}
1. & A \longrightarrow I_B: & R_A \\
1'. & I_A \longrightarrow B: & R_E \\
2'. & B \longrightarrow I_S: & R_E,\ R_B \\
2. & I_B \longrightarrow S: & R_A,\ R_B \\
3a. & S \longrightarrow A: & \{SK_{AB}\}_{K^E_{AS}},\ \left[A, B, R_A, \{SK_{AB}\}_{K^E_{AS}}\right]_{K^{MAC}_{AS}} \\
3b. & S \longrightarrow B: & \{SK_{AB}\}_{K^E_{BS}},\ \left[A, B, R_A, \{SK_{AB}\}_{K^E_{BS}}\right]_{K^{MAC}_{BS}}
\end{array}
$$

After A and B have terminated successfully and accepted the session key SK_{AB}, $\mathcal{A}$ sends a Reveal query to A and obtains the key SK_{AB}. Now $\mathcal{A}$ can send a Test query to B, because the protocol run B has done is still fresh. Because the adversary now knows the session key, he has a complete knowledge of the answer B is sending back. So $\mathrm{Adv}^{\mathcal{A}} = 1$.

3PKD Protocol IV

In order to get more familiar with the game, we present all the queries used in the attack:

Query	**Response**
SendClient$(A, B, i, *)$	R_A
SendClient(B, A, j, R_E)	$R_E,\ R_B$
SendServer$(A, B, s, (R_A, R_B))$	$(\alpha_{A,i}, \beta_{A,i}),\ (\alpha_{B,j}, \beta_{B,j})$
SendClient$(A, B, i, (\alpha_{A,i}))$	Accept$_{A,i}$
SendClient$(B, A, j, (\alpha_{B,j}))$	Accept$_{B,j}$
Reveal(A, B, i)	$SK_{A,B,i}$

Improved 3PKD Protocol I

$$
\begin{array}{lll}
1. & A \longrightarrow B: & R_A \\
2. & B \longrightarrow S: & R_A,\ R_B \\
3a. & S \longrightarrow A: & \{SK_{AB}\}_{K_{AS}^{E}},\ \left[A, B, R_A, R_B, \{SK_{AB}\}_{K_{AS}^{E}}\right]_{K_{AS}^{MAC}}, R_B \\
3b. & S \longrightarrow B: & \{SK_{AB}\}_{K_{BS}^{E}},\ \left[A, B, R_A, R_B, \{SK_{AB}\}_{K_{BS}^{E}}\right]_{K_{BS}^{MAC}}
\end{array}
$$

Principles of a Security Proof I

- The correctness is proved by finding a reduction to the security of the encryption scheme and the message authentication scheme.
- The assumption is that the encryption and message authentication schemes are secure. Then we make an assumtion that $\mathcal{A}$ has a non-negligible advantage in the game against the protocol.
- We show that this advantage leads to a non-negligible advantage in the game against the encryption and message authentication schemes.
- We define again the advantage functions. Let $\Omega = (\mathcal{K}, E, D)$ be an encryption scheme and C_{K_A} and C_{K_B} two challengers with secret keys K_A and K_B. (C_{K_A} can also be called *an encryption oracle*.)
- Let I_Ω be a single eavesdropper testing the scheme Ω.
- Let M_Ω be *a multiple eavesdropper*. This means that when M_Ω sends challenges m_1 and $m2$, he will receive two encrypted messages back; one of the messages encrypted by two different keys.

Principles of a Security Proof II

- Let I_Ω choose two plaintexts m_0, m_1. C_{K_A} chooses $b \in \{0, 1\}$ randomly and calculates $c = E_{K_A}(m_b)$. Let $I_\Omega(c) = b'$ denote the bit I_Ω has guessed. Then

$$\mathrm{Adv}^{I_\Omega} \; = \; 2 \times \mathrm{Prob}(b = b') - 1.$$

- Consider next M_Ω. M_Ω sends plaintext messages m_0 and m_1 to two different challengers who choose the same random bit and send $C_A = E_{K_A}(m_b)$ and $C_A = E_{K_B}(m_b)$ back to M_Ω. M_Ω then calculates a bit b'. The advantage function is now the same as in the case of I_Ω:

$$\mathrm{Adv}^{M_\Omega} \; = \; 2 \times \mathrm{Prob}(b = b') - 1.$$

Principles of a Security Proof III

Lemma

Suppose $\mathrm{Adv}^{I_\Omega} = \varepsilon$. *Then* $\mathrm{Adv}^{M_\Omega} \leq 2 \times \varepsilon$.

Proof. Check Bellare, Boldyreva, Micali: *Public-key Encryption in a Multi-User Setting: Security Proofs and Improvements*, EUROCRYPT 2000, LNCS 1807, pp. 259-274.

Secure MAC under ACMA

- The first part of the proof deals with the MAC function and we define the MAC security under an adaptive chosen message attack (ACMA).
- Suppose a challenger has chosen a MAC key k. An adversary chooses a message m, and using a polynomial time random MAC forger calculates a tag T for m. Then he sends (m, T) to the challenger to be verified.

Definition

MAC is secure under ACMA, if

$$\mathrm{Prob}[\mathrm{Verification}(m, T) = \mathrm{Valid}]$$

is negligible.

Proof of Authentication in 3PKD I

- Assume that at some stage $\mathcal{A}$ makes $\mathrm{SendClient}(B, A, j, (\alpha_{B,j}, \beta_{B,j}))$ query to some fresh oracle $\Pi^{j}_{B,A}$ who accepts.
- Assume furthermore that the MAC tag $\beta_{B,j}$ in the query was not previously output by a fresh oracle. Hence Prob[ForgerySuccess] is non-negligible.
- We construct an adaptive MAC forger $\mathcal{F}$ against the security of the message authentication scheme using as a helper $\mathcal{A}$. The attack game proceeds as follows:

Stage 1. $\mathcal{F}$ is provided permanent access to the MAC oracle $\mathcal{O}_{k'}$ associated with the MAC key k' throughout the game.

- $\mathcal{F}$ randomly chooses a principal U', where $U' \in \{U_1, \cdots, U_n\}$. U' is $\mathcal{F}$'s guess that $\mathcal{A}$ will choose U' for the attack.
- $\mathcal{F}$ randomly generates the list of MAC keys for principals $\{U_1, \cdots, U_n\} \backslash \{U'\}$.
- $\mathcal{F}$ randomly generates the list of encryption keys for the principals $U_1, \cdots, U_n$.
- n is polynomial with respect to the security parameter of the MAC.

Proof of Authentication in 3PKD III

Stage 2.

- $\mathcal{F}$ runs $\mathcal{A}$ and answers all oracle queries from $\mathcal{A}$ as required using the keys chosen in Stage 1 and $\mathcal{O}_{k'}$.
- In addition, $\mathcal{F}$ records all the MAC tags it receives from $\mathcal{O}_{k'}$.
- If, during its execution, $\mathcal{A}$ makes an oracle query that includes a forged MAC digest for U', then $\mathcal{F}$ outputs the MAC forgery as its own and halts.
- Otherwise, $\mathcal{F}$ halts when $\mathcal{A}$ halts.

The random choice of U' by $\mathcal{F}$ means that the probability that U' is the participant for whom $\mathcal{A}$ generates a forgery (if $\mathcal{A}$ generates any forgery at all) is at least $1/n$. Hence the success probability of $\mathcal{F}$ is

$$\mathrm{Prob}(\mathrm{ForgerySuccess})^{\mathcal{F}}) \geq \mathrm{Prob}[\mathrm{MACforgery}]/n.$$

Proof of Authentication in 3PKD IV

It was assumes that the MAC scheme is secure so this leads to a contradiction. It follows that

$$\mathrm{Prob}[\mathrm{MACforgery}] \leq n \cdot \mathrm{Prob}(\mathrm{ForgerySuccess})^{\mathcal{F}}),$$

and both probabilities are negligible. $\square$