

Verification of “Ping-Pong” Protocols in Quadratic Time

Heiko Stamer

University of Kassel
Department of Mathematics/Computer Science
Heinrich-Plett-Straße 40, 34132 Kassel, Germany

`stamer@theory.informatik.uni-kassel.de`

76F7 3011 329D 27DB 8D7C 3F97 4F58 4EB8 FB2B E14F

3. Jahrestagung “Sicherheit – Schutz und Zuverlässigkeit”
Fachbereich Sicherheit der Gesellschaft für Informatik
Otto-von-Guericke Universität Magdeburg, 20.-22. Februar 2006

Outline

Introduction

Dolev-Yao Model

“Ping-Pong” Protocols

Verification in Quadratic Time

Conclusion

Cryptographic protocols are sequential exchanges of messages between participating entities (principals) whose purpose is to ensure *protection or security goals*.

Protection goals are required properties to ensure the “security” of an application or a communication system:

- Authentication of principals
- Confidentiality/Secrecy and integrity of transmitted messages or distributed keys
- Robustness, non-repudiation, anonymity, ...

Notation for protocol steps: $A \rightarrow B : m$

Honest principal names: A, B, C, \dots

Impersonator or Adversary: I, Z

Nonces: n_A, n_B, \dots (fresh/random numbers)

Cryptographic keys: $K_{AB}, PK_A, SK_A, \dots$

Primitives: $\{\cdot\}_K$ (encryption), $[\cdot, \cdot]$ (pairing), ...

Introduction ▶ Motivation

“Finally, protocols such as those developed here are prone to extremely subtle errors that are unlikely to be detected in normal operation. The need for techniques to verify the correctness of such protocols is great, and we encourage those interested in such problems to consider this area.”

Roger M. Needham, Michael D. Schroeder (1978)

Flawed protocol: Needham-Schroeder Public-Key Protocol, 1978

Protocol goal: Authentication of two principals, e.g. A and B

Impersonator executes two concurrent sessions 1 and 2:

Short protocol version without authentication server:

1. $A \rightarrow B : \{n_A, A\}_{PK_B}$
2. $B \rightarrow A : \{n_A, n_B\}_{PK_A}$
3. $A \rightarrow B : \{n_B\}_{PK_B}$

- 1.(1) $A \rightarrow I : \{n_A, A\}_{PK_I}$
- 1.(2) $I(A) \rightarrow B : \{n_A, A\}_{PK_B}$
- 2.(2) $B \rightarrow I(A) : \{n_A, n_B\}_{PK_A}$
- 2.(1) $I \rightarrow A : \{n_A, n_B\}_{PK_A}$
- 3.(1) $A \rightarrow I : \{n_B\}_{PK_I}$
- 3.(2) $I(A) \rightarrow B : \{n_B\}_{PK_B}$

This attack was **found only in 1995** by Gavin Lowe.

D. Dolev, A.C. Yao: On the Security of Public Key Protocols
IEEE Transactions on Inf. Theory, 29(2), pp. 198–208, 1983.

- It was the first paper that provides a formal model:
 - Precise language for the description of protocols
 - Formal specification of the desired protection goals
 - Execution model, e.g. behavior/capabilities of the adversary
- Protocol restrictions are mostly on honest participants and the protection goal, i.e. the adversarial model is still quite general
- Further important results of the paper:
 - Security characterization for the Cascade Protocols
 - Polynomial-time algorithm to decide the security problem for a more general subclass of protocols (Name-Stamp Protocols)

Dolev-Yao Model ▶ Motivating Examples

Protection goal: Secrecy of the transmitted message m

Example (Secrecy violated)

- | | | | | | | | |
|----|-------------------|---------------------|----------------|----------------|-------------------|---|----------------|
| | 1.(1) | $A \rightarrow Z/B$ | : | $\{m\}_{PK_B}$ | | | |
| 1. | $A \rightarrow B$ | : | $\{m\}_{PK_B}$ | 1.(2) | $Z \rightarrow B$ | : | $\{m\}_{PK_B}$ |
| 2. | $B \rightarrow A$ | : | $\{m\}_{PK_A}$ | 2.(2) | $B \rightarrow Z$ | : | $\{m\}_{PK_Z}$ |
| | 2.(1) | $Z \rightarrow A$ | : | $\{m\}_{PK_A}$ | | | |

Example (Secrecy kept)

- | | | | |
|----|-------------------|---|-------------------|
| 1. | $A \rightarrow B$ | : | $\{m, A\}_{PK_B}$ |
| 2. | $B \rightarrow A$ | : | $\{m, B\}_{PK_A}$ |

Example (Secrecy violated)

1. $A \rightarrow B$: $\{\{m\}_{PK_B}, A\}_{PK_B}$
2. $B \rightarrow A$: $\{\{m\}_{PK_A}, B\}_{PK_A}$

- 1.(1) $A \rightarrow B$: $\{\{m\}_{PK_B}, A\}_{PK_B}$
- 2.(1) $B \rightarrow Z/A$: $\{\{m\}_{PK_A}, B\}_{PK_A}$
- 1.(2) $Z \rightarrow A$: $\{\{\{m\}_{PK_A}, B\}_{PK_A}, Z\}_{PK_A}$
- 2.(2) $A \rightarrow Z$: $\{\{\{m\}_{PK_A}, B\}_{PK_Z}, A\}_{PK_Z}$
- 1.(3) $Z \rightarrow A$: $\{\{m\}_{PK_A}, Z\}_{PK_A}$
- 2.(3) $A \rightarrow Z$: $\{\{m\}_{PK_Z}, A\}_{PK_Z}$

Dolev-Yao Model ▶ Basic Assumptions

- 1 Perfect public key cryptosystem (E_X, D_X) and infrastructure:
 - One-way functions E_X are “unbreakable”; only X knows D_X
 - Commuting operators, i.e. $\forall X : E_X D_X = D_X E_X = 1$
 - Secure public directory contains (X, E_X) for all users X
- 2 Two-party protocols (no trusted third party is necessary)
 - Honest parties are stateless (exception: protocol step pointer)
- 3 Uniform format of the exchanged protocol messages
- 4 Behavior of the adversary (active saboteur):
 - He can obtain and intercept any message passing through the network, i.e. he acts as the network.
 - He is a legitimate user of the network (or has corrupted some participants), and thus he can initiate concurrent protocol instances with any other user immediately.
 - He can maintain state information, i.e. a record of all transmitted messages and protocol sessions.
- 5 Protection goal: Secrecy of the input for honest principals

“Ping-Pong” Protocols ▶ Overview

- Very simple protocols (may have less practical relevance)
- Dolev and Yao distinguish between two classes of protocols:

Cascade Protocols are protocols in which the users can apply only the public key **encryption-decryption operators** (E_X, D_X) on messages. However, they might be applied in several layers and in a different order for each distinct protocol step.

Name-Stamp Protocols are protocols in which the users are additionally allowed to **append** (i_X) , **delete** (d) , and **check** (d_X) **names** on messages. Obviously, these protocols are a generalization of the former ones.

Formally, these operators are mappings from $\{0, 1\}^*$ into $\{0, 1\}^*$. They are supposed to be compatible with respect to composition.

“Ping-Pong” Protocols ▶ Notations

Let Σ be finite alphabet of operator symbols and Σ^* be the set of all strings over Σ including the empty word λ .

Operators in “Ping-Pong” protocols:

$$\Sigma = \{d\} \cup \{E_X, D_X, i_X, d_X \mid X \text{ is a user name}\}$$

Based on the properties of the public key cryptosystem and the name-stamps we get the following reduction/rewriting system \mathcal{R} :

$$E_X D_X \rightarrow \lambda$$

$$D_X E_X \rightarrow \lambda$$

$$d_X i_X \rightarrow \lambda$$

$$d i_X \rightarrow \lambda$$

Observation: If $a, b, c \in \Sigma$, $ab \rightarrow_{\mathcal{R}} \lambda$ and $bc \rightarrow_{\mathcal{R}} \lambda$ then $a = c$.

Proof: $c(m) = ab(c(m)) = abc(m) = a(bc(m)) = a(m)$ for all messages m .

Hence the reduction process has the Church-Rosser property.

Reduced form (unique normal form) of $\alpha \in \Sigma^*$ is denoted by $\bar{\alpha}$.

“Ping-Pong” Protocols ▶ Notations

Restricted operator set for a user X :

$$\Sigma_X = \{d, D_X\} \cup \{E_Y, i_Y, d_Y \mid Y \text{ is a user name}\}$$

Definition (Dolev, Even, and Karp)

A “Ping-Pong” protocol $\mathfrak{P}(S, R)$ between the initiator S and the responder R is a sequence of operator-words $\alpha_1, \alpha_2, \dots, \alpha_\ell$, such that $\alpha_i \in \Sigma_S^*$ if i is odd and $\alpha_i \in \Sigma_R^*$ if it is even.

Operator-words of the adversary: $\Delta = (\Sigma_Z \cup \{\alpha_i[X, Y] \mid 1 \leq i \leq \ell, X \text{ and } Y \text{ are different users}\})^*$

Definition (Dolev, Even, and Karp)

Let $\alpha_1[S, R]$ be the first operator-word of $\mathfrak{P}(S, R)$ and $Z \notin \{S, R\}$ an adversary. \mathfrak{P} is **insecure**, if there exists a word $\gamma \in \Delta$ such that $\overline{\gamma\alpha_1} = \lambda$.

It is sufficient to consider only $\overline{\gamma\alpha_1} = \lambda$ instead of $\overline{\gamma\alpha_i\alpha_{i-1}\cdots\alpha_1} = \lambda$. Further, it is sound to consider two honest users S, R and the adversary Z .

“Ping-Pong” Protocols ▶ Example

Example (Secrecy violated)

1. $A \rightarrow B$: $\{\{m\}_{PK_B}, A\}_{PK_B}$ $\alpha_1[A, B] = E_B i_A E_B$
2. $B \rightarrow A$: $\{\{m\}_{PK_A}, B\}_{PK_A}$ $\alpha_2[A, B] = E_A i_B E_A D_B d_A D_B$

$$\gamma' = \underbrace{D_Z d_A D_Z}_{Z} \underbrace{E_Z i_A E_Z D_A d_Z D_A}_{A(3)} \underbrace{E_A i_Z d D_Z d_A D_Z}_{Z(3)} \underbrace{E_Z i_A E_Z D_A d_Z D_A}_{A(2)} \underbrace{E_A i_Z}_{Z(2)}$$

$$\overline{\gamma' \alpha_2 \alpha_1} = \lambda$$

$$\gamma = \gamma' \alpha_2 \quad \overline{\gamma \alpha_1} = \lambda$$

Theorem (Dolev, Even, and Karp)

For “Ping-Pong” protocols there exists a security checking algorithm whose inputs are the generic cancellation rules and the protocol. The time-complexity is $O(n^3)$, where n is the length of the input.

“Ping-Pong” Protocols ▶ Generic Idea of Dolev, Even, and Karp

Generic construction: $O(n^3)$ -time and $O(n^3)$ -space

Let L be the context-free language of all possible operator-words which can be reduced by the cancellation rules \mathcal{R} to the empty word λ :

$$L = \{w \in \Sigma^* \mid \bar{w} = \lambda\}$$

Let L_Z be the regular language of all possible attacks on $\mathfrak{P}(S, R)$:

$$L_Z = \{\gamma\alpha_1[S, R] \mid \gamma \in \Delta\}$$

Essentially, the security problem for “Ping-Pong” protocols (w.r.t. the secrecy of m) is the question whether or not the intersection of a regular language and a context-free language is non-empty, i.e. $L \cap L_Z \stackrel{?}{=} \emptyset$.

Remark

The original algorithm of Dolev, Even, and Karp has time-complexity $O(n^3 m)$ and space-complexity $O(n^2 m)$, where m is the size of the context-free grammar G such that $L = L(G)$ and n is the number of states of the NFA A such that $L_Z = L(A)$.

Verification in Quadratic Time ▶ Observations

Observation

The language L (w.r.t. the classical cancellation rules \mathcal{R}) is generated by the **unambiguous** context-free grammar $\mathcal{G} = (\{A, B\}, \Sigma, A, P)$ with the productions

$$P = \left\{ \begin{array}{l} A \rightarrow AB \mid \lambda \\ B \rightarrow D_X A E_X \mid E_X A D_X \mid d_X A i_X \mid d A i_X \quad \text{for all } X \in \{S, R, Z\} \end{array} \right\}.$$

Obviously, we have $L(\mathcal{G}) = L$. The unambiguousness (i.e. unique left- or right-derivation for each word from $L(\mathcal{G})$) follows from the distinctness of the terminal symbols in all productions of type $B \rightarrow \alpha A \beta$ ($\alpha, \beta \in \Sigma$).

Observation

The algorithm of Esparza, Rossmanith, and Schwoon (2000) solves the decision problem $L(\mathcal{G}) \cap R \stackrel{?}{=} \emptyset$ in quadratic time, if \mathcal{G} is an unambiguous context-free grammar and R a regular language.

Verification in Quadratic Time ▶ Algorithm

Fixed: Cancellation rules (rewriting system \mathcal{R}) and Σ

Input: Two-party “Ping-Pong” protocol $\mathfrak{P}(S, R)$

Output: Answer to the question of whether or not $\mathfrak{P}(S, R)$ is insecure w.r.t. the secrecy of transmitted messages

- 1 Construct the simplified NFA $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ for L_Z
- 2 Construct the grammars \mathcal{G} and \mathcal{G}' for the context-free language L
- 3 Compute the NFA $\mathcal{A}' = (Q, \Sigma', \delta', q_0, F)$ for $\text{pre}^*(L_Z)$
- 4 Check whether the starting symbol A of \mathcal{G}' is accepted by \mathcal{A}'

Verification in Quadratic Time ▶ Algorithm

Fixed: Cancellation rules (rewriting system \mathcal{R}) and Σ

Input: Two-party “Ping-Pong” protocol $\mathfrak{P}(S, R)$

Output: Answer to the question of whether or not $\mathfrak{P}(S, R)$ is insecure w.r.t. the secrecy of transmitted messages

- 1 Construct the simplified NFA $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ for L_Z
 - (a) Set the initial state to q_0 and the final states to $F = \{q_1\}$.
 - (b) For the suffix α_1 insert the path $(q_0, \alpha_1[S, R], q_1)$ into δ .
 - (c) For each operator $\sigma \in \Sigma_Z$ insert a loop (q_0, σ, q_0) into δ .
 - (d) For $1 \leq i \leq \ell$ and $X, Y \in \{S, R, Z\}$ where $X \neq Y$ insert the path $(q_0, \alpha_i[X, Y], q_0)$ into δ .
- 2 Construct the grammars \mathcal{G} and \mathcal{G}' for the context-free language L
- 3 Compute the NFA $\mathcal{A}' = (Q, \Sigma', \delta', q_0, F)$ for $\text{pre}^*(L_Z)$
- 4 Check whether the starting symbol A of \mathcal{G}' is accepted by \mathcal{A}'

Verification in Quadratic Time ▶ Algorithm

Fixed: Cancellation rules (rewriting system \mathcal{R}) and Σ

Input: Two-party “Ping-Pong” protocol $\mathfrak{P}(S, R)$

Output: Answer to the question of whether or not $\mathfrak{P}(S, R)$ is insecure w.r.t. the secrecy of transmitted messages

- 1 Construct the simplified NFA $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ for L_Z
- 2 Construct the grammars \mathcal{G} and \mathcal{G}' for the context-free language L
 - (a) Construct the unambiguous context-free grammar \mathcal{G}
 - (b) Obtain \mathcal{G}' by extended Chomsky-normalization of \mathcal{G}
- 3 Compute the NFA $\mathcal{A}' = (Q, \Sigma', \delta', q_0, F)$ for $\text{pre}^*(L_Z)$
- 4 Check whether the starting symbol A of \mathcal{G}' is accepted by \mathcal{A}'

Verification in Quadratic Time ▶ Algorithm

Fixed: Cancellation rules (rewriting system \mathcal{R}) and Σ

Input: Two-party “Ping-Pong” protocol $\mathfrak{P}(S, R)$

Output: Answer to the question of whether or not $\mathfrak{P}(S, R)$ is insecure w.r.t. the secrecy of transmitted messages

- 1 Construct the simplified NFA $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ for L_Z
- 2 Construct the grammars \mathcal{G} and \mathcal{G}' for the context-free language L
- 3 Compute the NFA $\mathcal{A}' = (Q, \Sigma', \delta', q_0, F)$ for $\text{pre}^*(L_Z)$
 - $\text{pre}^*(L_Z) = \{w \in \Sigma'^* \mid \exists v \in L_Z : w \Rightarrow_{\mathcal{G}}^* v\}$ is the set of predecessors of L_Z w.r.t. the derivation relation induced by \mathcal{G}' .
 - For a regular set L_Z and a context-free grammar \mathcal{G}' the language $\text{pre}^*(L_Z)$ is always regular (Book and Otto, 1985).
 - For unambiguous grammars \mathcal{G}' the algorithm of Esparza, Rossmanith, and Schwoon computes \mathcal{A}' in time $\mathcal{O}(|P'| |Q|^2)$.
- 4 Check whether the starting symbol A of \mathcal{G}' is accepted by \mathcal{A}'

Verification in Quadratic Time ▶ Algorithm

Fixed: Cancellation rules (rewriting system \mathcal{R}) and Σ

Input: Two-party “Ping-Pong” protocol $\mathfrak{P}(S, R)$

Output: Answer to the question of whether or not $\mathfrak{P}(S, R)$ is insecure w.r.t. the secrecy of transmitted messages

- 1 Construct the simplified NFA $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ for L_Z
- 2 Construct the grammars \mathcal{G} and \mathcal{G}' for the context-free language L
- 3 Compute the NFA $\mathcal{A}' = (Q, \Sigma', \delta', q_0, F)$ for $\text{pre}^*(L_Z)$
- 4 Check whether the starting symbol A of \mathcal{G}' is accepted by \mathcal{A}'
 - The protocol $\mathfrak{P}(S, R)$ is **insecure**, if and only if $A \in \text{pre}^*(L_Z)$.

Verification in Quadratic Time ▶ Analysis and Results

Analysis of the time and space complexity:

Let $n = \sum_{i=1}^{\ell} |\alpha_i|$ be the size of the input (the protocol $\mathfrak{P}(S, R)$).

- 1 The NFA \mathcal{A} can be constructed in time $\mathcal{O}(n)$ and space $\mathcal{O}(n)$.
- 2 For fixed cancellation rules this step is time-/space-neutral.
- 3 The algorithm of Esparza, Rossmanith, and Schwoon (2002) computes in time $\mathcal{O}(|P'|n^2)$ and space $\mathcal{O}(|P'|n^2)$ the NFA \mathcal{A}' .
- 4 Checking whether the starting symbol of the grammar \mathcal{G}' is accepted by \mathcal{A}' takes linear time and constant space.

Corollary

The presented algorithm decides the security problem for two-party “Ping-Pong” protocols (classical cancellation rules \mathcal{R}) in quadratic time.

Verification in Quadratic Time ▶ Arbitrary Cancellation Rules

Question: Does there always exist an unambiguous context-free grammar for each arbitrary set of cancellation rules? **Yes!**

- In general it is undecidable whether a context-free grammar has an equivalent unambiguous context-free grammar.
- However, cancellation rules are homogeneous SRSs of degree 2 (special kind of monadic SRS). Since the elements of Σ act like operators the corresponding SRS is always Church-Rosser.
- Hence one can construct a deterministic PDA for the language L and thus a corresponding unambiguous LR(1)-grammar for L .

Theorem

For two-party “Ping-Pong” protocols (with arbitrary but fixed cancellation rules) there exists an algorithm that decides the security problem (w.r.t. the secrecy of messages) in quadratic time.

Conclusion

Comparison with the original algorithm:

	Dolev, Even, Karp (1982)	presented algorithm
fixed cancellation rules	$\mathcal{O}(n^3)$ -time, $\mathcal{O}(n^2)$ -space	$\mathcal{O}(n^2)$ -time, $\mathcal{O}(n^2)$ -space
cancellation rules as input	$\mathcal{O}(n^3)$ -time, $\mathcal{O}(n^2)$ -space	$\mathcal{O}(n^3)$ -time, $\mathcal{O}(n^3)$ -space

Concluding remarks:

- Our algorithm provides a small improvement (linear factor) in the running time, if the cancellation rules are fixed.
- In general, this kind of security checking approach assumes that Σ is free from any relations other than those implied by the predefined cancellation rules.
- “Ping-Pong” protocols have a very limited “expressive power”, i.e. real-world protocols are often not describable.

References



Danny Dolev and Andrew C. Yao.

On the Security of Public Key Protocols.

IEEE Transactions on Information Theory, 29(2):198–208, 1983.



Danny Dolev, Shimon Even, and Richard M. Karp.

On the Security of Ping-Pong Protocols.

Information and Control, 55(1–3):57–68, 1982.



Ronald V. Book and Friedrich Otto.

Cancellation Rules and Extended Word Problems.

Information Processing Letters, 20:5–11, 1985.



Javier Esparza, Peter Rossmanith, and Stefan Schwoon.

A Uniform Framework for Problems on Context-Free Grammars.

Bulletin of the EATCS, 72:169–177, 2000.



Ronald V. Book and Friedrich Otto.

String-Rewriting Systems.

Springer Texts and Monographs in Computer Science, 1993.