OUKA
The University of Osaka
Institutional Knowledge Archive

Title	Polynomial Time Verification Methods for the Security of Cryptographic Protocols
Author(s)	Watanabe, Hajime
Citation	大阪大学, 1997, 博士論文
Version Type	VoR
URL	https://doi.org/10.11501/3129221
rights	
Note	

The University of Osaka Institutional Knowledge Archive : OUKA

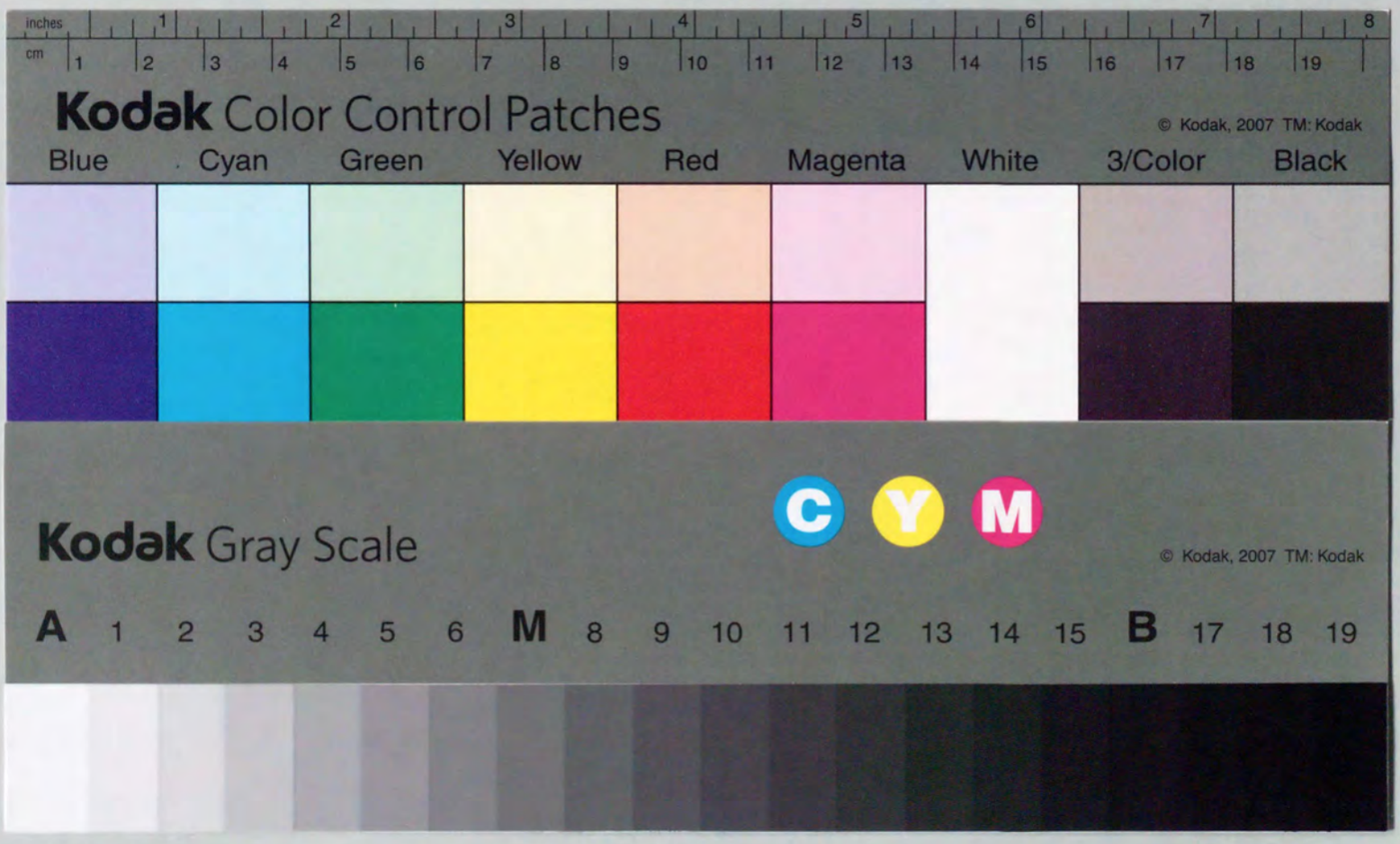
https://ir.library.osaka-u.ac.jp/

The University of Osaka

Polynomial Time Verification Methods for the Security of Cryptographic Protocols

Hajime Watanabe

December 1996



Polynomial Time Verification Methods for the Security of Cryptographic Protocols

by

Hajime Watanabe

December 1996

Dissertation submitted to Graduate School of Engineering Science of
Osaka University in partial fulfillment of the requirements for
the degree of Doctor of Philosophy in Engineering

Abstract

These days, it becomes popular that people use computer networks and many network services are provided. In computer networks, messages are transmitted over communication lines, and it is not difficult to eavesdrop the lines. Moreover, injecting false messages or replays of previous messages to the lines is also not difficult.

To prevent the unauthorized disclosure of data, the cryptographic system which consists of encryption and decryption is used. To prevent the unauthorized modification of data, various basic protocols, say a digital signature, are proposed. Also, by using these basic techniques, protocols for many activities, say an election and a bidding, are proposed. These protocols use cryptographic systems and/or one-way functions, and are called cryptographic protocols.

Because of a protocol's defect, a cryptographic protocol may not be secure even though the cryptographic systems used in the protocol are assumed to be secure. For this reason, when we discuss the security of a protocol itself, the cryptographic systems used in the protocol are assumed to be secure. Under this assumption, the cryptographic protocol is said to be secure if it is impossible for enemies to violate the intention of the protocol, for example, to disclose or modify protected data, where it is supposed that the enemies can use any information which they can get from networks by wiretapping, and use any operation which is not prohibited in the protocol. In most cases, it is not clear whether or not a proposed cryptographic protocol is secure, and the security of very few cryptographic protocols except for some protocols are verified in their proposal.

For such security problems of cryptographic protocols, the decision problem has been formalized as a unification problem of two terms in term rewriting system. In this formalization, the two terms are not unifiable if and only if the protocol is secure. In general, it is known that the decision problem is undecidable. Moreover, there is no even semidecision procedure to assure the security of cryptographic protocols which are secure in the above sense. Our group has presented a sufficient condition under which the decision problem is decidable, and for the security problems which satisfy the sufficient condition, our group also presented a polynomial time decision algorithm. But it is not clear whether or not the decision algorithm can decide the security in practical time since the worst case time complexity of this algorithm is $O(n^8)$.

In the first part of this dissertation, the implementation and evaluation of the algorithm are described. To implement this algorithm, the detail of the algorithm is designed since the algorithm was presented in abstract way. The algorithm is also

refined to speed up and to save space in average case since it is surmised that the original algorithm is not efficient in average case. To evaluate the usefulness of the algorithm, the security problems of several cryptographic protocols which satisfy the sufficient condition are considered, and those security problems are solved by using the implemented system. The system can decide the security in a few minutes or less for simple protocols and in two days or less for comparatively complex protocols. From these results, it is confirmed that this system can decide the security in practical time and we conclude that this system is useful to decide the security of cryptographic protocols.

There are some cases that the security problems of cryptographic protocols proposed so far do not satisfy the sufficient condition. Many of those security problems do not satisfy one subcondition (right-linearity) of the sufficient condition but satisfy any other subconditions. In the second part of this dissertation, a polynomial time verification method for security problems including the problems which do not satisfy right-linearity is proposed. Since this method is not a decision algorithm, not all the security of such problems can be decided by this verification method. To verify such security problems, first, the decision problem is modified to the problem with sorts (types) of terms. In this decision problem, we introduce three transformations of a formal description of a security problem. They transform the original description to the description which satisfies right-linearity grammatically. And we show the relations between the security of the original description and that of transformed one. By using these transformations as subroutines, we present a new polynomial time verification method for security problems including the problems which do not satisfy right-linearity. As described above, this method cannot decide all the security of such problems, but stops in polynomial time. A security problem for the network authentication protocol, Kerberos, which is used in the Internet is one of the security problems that newly can be assured the security by this new method.

Consequently, we conclude that, by using the verification methods described in this dissertation, we can analyze the security of cryptographic protocols more formally.

Acknowledgments

I am deeply indebted to many people for the advice, feedback and support they gave to me in the course of this work. I would especially like to thank Professor emeritus Tadao Kasami, currently Professor of Nara Institute of Science and Technology for his invaluable support, discussions and encouragement throughout the work.

I am grateful to my supervisor Professor Kenichi Taniguchi for his invaluable suggestions and discussions on the work. I am also obliged to Professor Nobuki Tokura and Professor Mamoru Fujii for their helpful comments and suggestions. I would like to thank Professor Seishi Nishikawa, Professor Kenichi Hagihara and Professor Tohru Kikuno for their valuable comments.

I am extremely thankful to Associate Professor Toru Fujiwara for his invaluable discussions and great support throughout the work. I also thank to Mr. Takashi Muramatsu for his helpful discussions.

I would like to thank Professor Minoru Ito, Professor Hiroyuki Seki and Associate Professor Toyoo Takata of Nara Institute of Science and Technology, and Lecturer Masahiro Higuchi for their kind and helpful supports. I am thankful to Research Associates Yasunori Ishihara, Ryuichi Nakanishi and Yuichi Kaji of Nara Institute of Science and Technology for their valuable support. I am also grateful to Ms. Machiko Uehara for her kind support.

Finaly, I would like to thank all the members of Kasami Laboratory of Osaka University and Kasami Laboratory of Nara Institute of Science and Technology.

List of Publications

Journal Papers

[1] H. Watanabe, T. Fujiwara and T. Kasami, "A System for Deciding the Security of Cryptographic Protocols," IEICE Transactions on Fundamentals, Vol. E76-A, No. 1, pp. 96–103, Jan. 1993.

[2] H. Watanabe, T. Fujiwara and T. Kasami, "An Improved Method for Formal Security Verification of Cryptographic Protocols," IEICE Transactions on Fundamentals, Vol. E78-A, No. 7, pp. 1089–1096, July 1996.

Workshops

[3] H. Watanabe, T. Fujiwara, T. Takata and T. Kasami, "On a System for Deciding the Security of Cryptographic Protocols," Proceedings of SCIS92, SCIS92-14A, April 1992 (In Japanese).

[4] H. Watanabe, T. Fujiwara, T. Takata and T. Kasami, "On the Security Verification of the Authentiction Protocol Kerberos," Proceedings of SCIS94, SCIS94-1A, Jan. 1994 (In Japanese).

Contents

Chapter 1

Introduction

These days, it becomes popular that people use computer networks and many network services are provided. In computer networks, messages are transmitted over communication lines, and it is not difficult to eavesdrop the lines. Moreover, injecting false messages or replays of previous messages to the lines is also not difficult[2].

To prevent the unauthorized disclosure of data, the cryptographic system which consists of encryption and decryption is used. To prevent the unauthorized modification of data, various basic protocols, say digital signature protocols, are proposed. Also, by using these basic techniques, protocols for many activities, say an election and a bidding, are proposed. These protocols use cryptographic systems and/or one-way functions, and are called cryptographic protocols.

Because of a protocol's defect, a cryptographic protocol may not be secure even though the cryptographic systems used in the protocol are assumed to be secure. For this reason, when we discuss the security of a protocol itself, the cryptographic systems used in the protocol are assumed to be secure. Under this assumption, the cryptographic protocol is said to be secure if it is impossible for enemies to violate the intention of the protocol, for example, to disclose or modify protected data, where it is supposed that the enemies can use any information which they can get from networks by wiretapping, and use any operation which is not prohibited in the protocol[3, 4, 6, 7]. But the security of very few cryptographic protocols except for some protocols, say [11], are verified formally in their proposal. In most cases, it is not clear whether or not a proposed cryptographic protocol is secure. It has been pointed out that the designers of a network (or those of cryptographic protocols) were not necessarily honest. Hence, the users of cryptographic protocols desire to assure the security of them.

For such security problems of cryptographic protocols, various kinds of research have been done. The decision problem of the security problems has been discussed and has been formalized as a unification problem of two terms in term rewriting system[3, 4, 6, 7]. In this formalization, the two terms are not unifiable if and only if the protocol is secure. In general, it is known that the decision problem is undecidable[6], that is, there is no systematic way to decide the security of cryptographic protocols.

Several approaches to verify the security of cryptographic protocols have been proposed. If a protocol is insecure, it is theoretically possible to find security flaws

by enumerating and checking all the information which active enemies can obtain. By using this approach, some systems to support finding security flaws have been implemented, say Interrogator[16] and NRL Protocol Analyzer[12]. But as mentioned in [15], this approach is not efficient and this approach cannot be used to assure the security of cryptographic protocols. If a cryptographic protocol is secure in the above sense, it must be pointed out that the protocol is secure in some way, but even a semidecision procedure to assure the security does not exist.

Another approach based on modal logics, BAN logic[1], has been proposed to decide the security of authentication protocols. This approach is similar to what have been developed for the analysis of evolution of knowledge and belief in distributed systems. BAN logic can show whether or not an authentication protocol is secure. But as pointed out in [15], BAN logic has a problem in deciding the security of authentication protocols since active attacks like impersonating are not considered.

In [3], the decision problem in very simple and restricted cases is discussed and therefore, the class of protocols which can be decided the security by this method is too limited.

In [6, 7], our group has presented a sufficient condition under which the decision problem is decidable, and our group also has presented a polynomial time (of the length of the description of a security problem) decision algorithm for the security problems of cryptographic protocols which satisfy the sufficient condition. The class of protocols which can be decided the security by this algorithm is much larger than that considered in [3].

But it is not clear whether or not the decision algorithm proposed in [6, 7] can decide the security in practical time since this algorithm needs $O(n^8)$ time in worst case, where n represents the length of the description of a security problem. In this dissertation, the decision algorithm proposed in [6, 7] is implemented and evaluated. To implement this algorithm, the detail of the algorithm is designed since the algorithm was presented in abstract way. The algorithm is also refined to speed up and to save space in average case since it is surmised that the original algorithm is not efficient in average case. To evaluate the usefulness of the algorithm, the security problems of several cryptographic protocols which satisfy the sufficient condition are considered, and those security problems are solved by using the implemented system. As the result of modification and refinement of the original algorithm, the system can decide the security in a few minutes or less for simple protocols and in two days or less for comparatively complex protocols. From these results, it is confirmed that this system can decide the security in practical time and we conclude that this system is useful to decide the security of cryptographic protocols.

There are some cases that the security problems of cryptographic protocols proposed so far do not satisfy the sufficient condition. The sufficient condition consists of three subconditions. Many of those security problems do not satisfy one subcondition, especially one property of the subcondition called right-linearity, but satisfy any other subconditions of the sufficient condition. In this dissertation, a polynomial time verification method for security problems including the problems which do not satisfy right-linearity is proposed. Since this method is not a decision algorithm, not all the security of such problems can be decided by this new method. To verify such security problems, first, the decision problem is modified to the problem with sorts (types). By this modification, it becomes natural to treat the substitutions for variables whose domains are restricted. We introduce three transformations of a formal description of a security problem. They transform the original description to the description which satisfies right-linearity grammatically. And we show the relations between the security of the original description and that of transformed one. By using these transformations as subroutines, we present a new polynomial time verification method for security problems including the problems which do not satisfy right-linearity. As described above, this method cannot decide all the security of such problems, but stops in polynomial time. One of the security problems that newly can be assured the security by this new method is a security problem for the network authentication protocol, Kerberos, which is used in the Internet. This fact shows that this new method is more useful than the decision algorithm proposed in [6, 7].

Consequently, we conclude that, by using the verification methods described in this dissertation, we can analyze the security of cryptographic protocols more formally.

In Chapter 2, first, a brief review of the decision problem for the security of cryptographic protocols formalized in [6, 7] is given. Next, the detail and the refinement about the decision algorithm proposed in [6, 7] is discussed. And in this chapter, a system which implements the detailed algorithm on a UNIX workstation is presented. To evaluate the usefulness of the system, the security for several protocols are decided by the system.

In Chapter 3, first, the decision problem formalized in [6, 7, 21] is modified. Next, it is shown that the decision algorithm is not applicable for security problems of some proposed cryptographic protocols by giving an example of such security problems, that is, a security problem for the authentication protocol Kerberos. And we propose a new verification method for security problems including the problems which do not satisfy one subcondition of the sufficient condition. To confirm the usefulness of the new method, the security problem for Kerberos is verified.

Chapter 4 concludes the results of this dissertation and discusses future research subjects.

Chapter 2

Implementation of Polynomial Time Decision Algorithm for Security of Cryptographic Protocols and Its Evaluation

2.1 Introduction

For the security of cryptographic protocols, the decision problem has been discussed and has been formalized as a unification problem of two terms in term rewriting system[3, 4, 6, 7]. In this formalization, the two terms are not unifiable if and only if the protocol is secure. It is shown that, in general, the decision problem is undecidable[6]. In [6, 7], our group has presented a sufficient condition under which the decision problem is decidable, and also has presented a polynomial time algorithm for the security of cryptographic protocols which satisfy the sufficient condition.

But it is not clear whether or not the decision algorithm proposed in [6, 7] can decide the security in practical time since this algorithm needs $O(n^8)$ time in worst case, where n represents the length of the description of a security problem.

In this chapter, the decision algorithm proposed in [6, 7] is implemented and evaluated. In the algorithm, a term is represented as a string and the above unification problem of two terms is reduced to the non-emptiness problem of the intersection of two regular sets. The intersection is empty if and only if the two terms are not unifiable, that is, the protocol is secure. We show the way to construct non-deterministic finite automata (NFA) which accept those two regular sets. Each NFA is constructed by constructing an initial NFA and adding edges according to axioms in the formal description of the security problem to the initial NFA. To decide whether or not a new edge should be added for a state pair, the path labels between the states of NFA at that time are examined. We can speed up this in avarage case by dividing edges into groups on their labels to reduce the number of checks. We develop a compact representation of NFAs to save space and reduce the computation time for deciding the non-emptiness of intersection of two regular sets.

To evaluate the usefulness of the algorithm, the security problems of several cryptographic protocols which satisfy the sufficient condition are considered. These protocols are digital signatures[10], a bidding protocol[17], a secret communication protocol[18] and an authentication protocol[20]. Those security problems are solved by using the implemented system. As the result of modification and refinement of the original algorithm, the system can decide the security in a few minutes or less for simple protocols, digital signatures and an authentication protocol, and in two days or less for comparatively complex protocols, a bidding protocol and a secret communication protocol. From these results, it is confirmed that this system can decide the security in practical time and we conclude that this system is useful to decide the security of cryptographic protocols.

This chapter is organized as follows. In Section 2.2, a brief review of the security problem of cryptographic protocols formalized in [4, 6, 7] is given. In Section 2.3, a sufficient condition under which the security problem is decidable is shown. In Section 2.4, we go into detail about our decision algorithm proposed in [6]. In Section 2.5, a decision system which implements the algorithm on a UNIX workstation is shown. In Section 2.6, the decision experiments for several protocols are summarized.

2.2 Security Problem

We follow the definition of the security of cryptographic protocols formalized in [6, 7].

For a given cryptographic protocol, the security problem is defined by 5-tuple $(\boldsymbol{F}, \boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O}, \boldsymbol{G})$, where $\boldsymbol{F}, \boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O}$ and $\boldsymbol{G}$ are defined as follows:

(1) $\boldsymbol{F}$ is a finite set of function symbols which contains constant symbols. $\boldsymbol{F}$ contains the symbols of operations of the protocol, i.e. function symbols of encryption and decryption, constant symbols such as messages and keys, and so on.

For $f \in \boldsymbol{F}$, let $a(f)$ denote the number of arguments of f. For a subset $\boldsymbol{F}' \subseteq \boldsymbol{F}$, let $T_{VAR}(\boldsymbol{F}')$ be the infinite set of terms on $\boldsymbol{F}'$ with variables and let $T(\boldsymbol{F}')$ be the set of terms without variables.

(2) $\boldsymbol{A}$ is a finite set of axioms by which we represent the properties of the functions of $\boldsymbol{F}$. We assume that enemies can use no knowledge on the properties of functions other than the axioms in $\boldsymbol{A}$. The congruence relation defined by $\boldsymbol{A}$ is denoted by $\overset{\equiv}{}_{A}$.

(3) $\boldsymbol{I}$ is a finite subset of $T(\boldsymbol{F})$. $\boldsymbol{I}$ consists of those terms which correspond to information that enemies are assumed to acquire, e.g. their own keys and/or information through wiretapping.

(4) $\boldsymbol{O}$ is a subset of $\boldsymbol{F}$. $\boldsymbol{O}$ consists of the symbols of those operations which are assumed to be available for the enemies.

Let $Q(\boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O})$ be the set of terms representing such information that the enemies can acquire by executing any combination of operations provided by the system on the information corresponding to terms in $\boldsymbol{I}$ and by drawing inference from axiom in $\boldsymbol{A}$. $Q(\boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O})$ is defined formally as follows:

1. $\boldsymbol{I} \subset Q(\boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O})$,
2. for $f \in \boldsymbol{O}$ and $t_i \in Q(\boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O})$ with $1 \leq i \leq a(f)$, $f(t_1, t_2, \cdots, t_{a(f)}) \in Q(\boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O})$, and
3. for $t_1 \in Q(\boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O})$ and $t_2 \in T(\boldsymbol{F})$ such that $t_1 \overset{\equiv}{}_{A} t_2$, $t_2 \in Q(\boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O})$.

(5) $\boldsymbol{G} = \{g_1, g_2\}$ for terms $g_1, g_2 \in T_{VAR}(\boldsymbol{F})$. It represents enemies' goal. Let the set of all the variables which occur at least once in g_1 or g_2 be $\{X_1, X_2, \cdots, X_h\}$. For h terms $t_1, t_2, \cdots, t_h$ in $Q(\boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O})$, consider the following condition:

Condition: Let g_1' and g_2' be the terms obtained from g_1 and g_2 by substituting t_i for each variable X_i, respectively, then it holds that $g_1' \overset{\equiv}{}_{A} g_2'$.

The goal of enemies is to know at least one set of the "values" of $t_1, t_2, \cdots, t_h$, which satisfies the condition.

The protocol described by $(\boldsymbol{F}, \boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O}, \boldsymbol{G})$ is defined to be secure if and only if there exist no h terms in $Q(\boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O})$, which satisfy the condition.

We may consider several $(\boldsymbol{F}, \boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O}, \boldsymbol{G})$'s for the security of one protocols. In the following, we discuss about the decision problem whether or not a protocol is secure for its environment in the sense of the definition above. Several decision examples will be shown in Section 2.6.

2.3 A Sufficient Condition under which Security Problem is Decidable

The security problem is decidable if the following conditions 1 to 3 hold [6].

Condition 1: $\boldsymbol{F}$ is partitioned into two sets $\boldsymbol{F}_1$ and $\boldsymbol{F}_2$, and $\boldsymbol{A}$ is also partitioned into two sets $\boldsymbol{A}_1$ and $\boldsymbol{A}_2$ in the following way:

1. Each function f in $\boldsymbol{F}_1$ is defined in terms of function in $\boldsymbol{F}_2$ by a unique axiom in $\boldsymbol{A}_1$ as follows:

For each $f \in \boldsymbol{F}_1$, there is exactly one axiom in $\boldsymbol{A}_1$ of the form $f(X_1, X_2, \cdots, X_{a(f)}) == t$, where $X_1, X_2, \cdots, X_{a(f)}$ are distinct variables and $t \in T_{VAR}(\boldsymbol{F}_2)$. All the variable X_i with $1 \leq i \leq a(f)$ also occurs in its right-hand side.

2. Properties of functions in $\boldsymbol{F}_2$ are defined by axioms in $\boldsymbol{A}_2$. The left-hand and right-hand sides of an axiom in $\boldsymbol{A}_2$ are terms in $T_{VAR}(\boldsymbol{F}_2)$ and any variable in the right-hand side of the axiom in $\boldsymbol{A}_2$ also occurs in its left-hand side. By regarding axioms in $\boldsymbol{A}_2$ as rewriting rules from its left-hand side to the right-hand side, $\boldsymbol{A}_2$ has the finite termination property and the Church-Rosser property[9] on $T_{VAR}(\boldsymbol{F}_2)$, that is, if $t_1 \overset{\equiv}{}_{A_2} t_2$, t_1 and t_2 can be rewritten to the same term by the rewriting rules.

3. In the right-hand side of each axiom in $\boldsymbol{A}(\boldsymbol{A}_1$ and $\boldsymbol{A}_2)$, each variable occurs exactly once in the axiom (**right-linearity**).

$\boldsymbol{F}_1$ is the set of function symbols which corresponds to operations available in the protocol and these operations are defined by the axioms in $\boldsymbol{A}_1$. Functions in $\boldsymbol{F}_2$ are the primitive functions which are used to define operations, and the axioms in $\boldsymbol{A}_2$ represent their properties.

Condition 2: g_1 and g_2 have no common variables. For any variable X in g_1 (or g_2), X occurs only once in g_1(or g_2).

To describe Condition 3, we need some definitions. Let N_+ and N_+^* be the set of positive integers and the set of strings of positive integers, respectively. Let λ be the empty string in N_+^*. For $t \in T_{VAR}(\boldsymbol{F}_2)$, we define the set of "occurrences", denoted $Occ(t) \subseteq N_+^*$, of subterms of t and the subterm at occurrence v, denoted t/v, for $v \in Occ(t)$ as follows:

1. If t is a constant in $\boldsymbol{F}_2$ or a variable, then $Occ(t) \triangleq \{\lambda\}$, and $t/\lambda \triangleq t$.

2. If $t = f(t_1, t_2, \cdots, t_{a(f)})$, then $Occ(t) \triangleq \{\lambda\} \cup \{iv | 1 \leq i \leq a(f), v \in Occ(t_i)\}$, $t/\lambda \triangleq t$ and $t/iv \triangleq t_i/v$.

For a term $t \in T_{VAR}(\boldsymbol{F}_2)$ and an occurrence $v \in Occ(t)$, we say that v is a leaf occurrence of t, if and only if the subterm t/v is a constant in $\boldsymbol{F}_2$ or a variable. For a term t, consider a leaf occurrence v. For any $v_1 \in Occ(t)$ which is not a prefix of v, if t/v_1 is a variable or has no variable, then we call v a trunk of t.

Condition 3: Each axiom, $t_L == t_R$, in $\boldsymbol{A}_2$, satisfies either 1 or 2.

1. t_R is a variable X or is $f(X)$ for $f \in \boldsymbol{F}_2$ and variable X. The variable X occurs in t_L only once but $t_L \neq X$ and the occurrence of X, denoted v_L, is a trunk of t_L.

2. t_R is a constant in $\boldsymbol{F}_2$ and t_L has a trunk v_L such that t_L/v_L is a constant in $\boldsymbol{F}_2$ or a variable which occurs exactly once in t_L.

2.4 Polynomial Time Decision Algorithm for the Security Problem

2.4.1 Preliminaries

In this section, we restrict the form of axioms in $\boldsymbol{A}_2$ for simplicity. For each axiom in $\boldsymbol{A}_2$, its right-hand side is a variable and the variable occurs exactly once in its left-hand side. Other variables in its left-hand side occur at most twice. To describe the algorithm, we need some definitions.

Definition of $\boldsymbol{O}'$: Let $\boldsymbol{O}'$ be the smallest set of terms such that

1. for each axiom $t_L == t_R$ in $\boldsymbol{A}_1$, $t_R \in \boldsymbol{O}'$, and

2. for $f \in \boldsymbol{F}_2$ and distinct variables $X_1, X_2, \cdots, X_{a(f)}$, $f(X_1, X_2, \cdots, X_{a(f)}) \in \boldsymbol{O}'$.

Definition of T_0, T_1, T_{1*}: For $t \in T_{VAR}(\boldsymbol{F})$, let $\overline{t}$ denote the normal form of t which is obtained by rewriting t with axioms in $\boldsymbol{A}_2$ as term rewriting rules. Define

$$\begin{aligned} T_0 &= \{\overline{t} \mid t \in \boldsymbol{I}\} \cup \boldsymbol{O}', \\ T_1 &= \{\overline{g_1}, \overline{g_2}\} \cup T_0 , \\ T_{1*} &= \{t \mid t \text{ is a subterm of a term in } T_1\}. \end{aligned}$$

We also define that $T_{1*}^2 = \{(t_1, t_2) \mid t_1, t_2 \in T_{1*}\}$.

Definition of $\Sigma, L(t)$: We define an alphabet Σ as follows:

$$\begin{aligned} \Sigma \triangleq \; & \{t | t \text{ is a constant in } \boldsymbol{F}_2\} \cup \{f_{t_1, t_2, \cdots, t_{i-1}, *, t_{i+1}, \cdots, t_{a(f)}} | \\ & f(t_1, t_2, \cdots, t_{a(f)}) \text{ is in } T_{1*} \text{ or a subterm of left-hand and} \\ & \text{right-hand sides of axioms in } \boldsymbol{A}_2, 1 \leq i \leq a(f)\}. \end{aligned}$$

Let S be a symbol other than those in Σ. We define a string on $\Sigma \cup \{S\}$, denoted $st(t, v)$, for $t \in T_{1*} \cup \{$subterms of left-hand and right-hand sides of axioms in $\boldsymbol{A}_2\}$ and a leaf occurrence v of t, as follows:

1. If t is a constant in $\boldsymbol{F}_2$, then $st(t, \lambda) = t$.

2. If t is a variable, then $st(t, \lambda) = S$.

3. If $t = f(t_1, t_2, \cdots, t_{a(f)})$ for $f \in \boldsymbol{F}_2$ and $t_1, t_2, \cdots, t_{a(f)} \in T_{VAR}(\boldsymbol{F}_2)$, and $v = iv_1$, then $st(t, v) = f_{t_1,\cdots,t_{i-1},*,t_{i+1},\cdots,t_{a(f)}} \cdot st(t_i, v_1)$.

For $t \in T_{1*}$, let $L(t) = L_T(t) \cup L_N(t)\{\alpha|\alpha \in LU_N\}^*\{\alpha|\alpha \in LU_T\}$, where

$$\begin{aligned} L_T(t) &= \{st(t,v)|t/v \text{ is a constant}\}, \\ LU_T &= \bigcup_{t_2 \in T_0} L_T(t_2), \\ L_N(t) &= \{\alpha|\text{For a leaf occurrence } v \text{ of } t, st(t,v) = \alpha S\}, \\ LU_N &= \bigcup_{t_1 \in T_0} L_N(t_1). \end{aligned}$$

Let Q' be the set of terms such that

1. $\{t|t \in T_0$ and there is a leaf occurrence v of t such that t/v is a constant $\} \subset Q'$,
2. Any term obtained from $t \in T_0$ by substituting $t' \in Q'$ for one variable in t is in Q'.

Then for any string α in $L(t)$ which contains a variable, there is a term t' which is obtained from t by substituting $t'' \in Q'$ for one variable in t and its leaf occurrence v such that $st(t', v) = \alpha$ if we extend the definition of $st(t', v)$ to any term $t' \in Q'$ naturally.

Definition of ψ_f, ψ_*, ψ_t: Functions $\psi_f : \Sigma \to \boldsymbol{F}_2$, $\psi_* : \Sigma \to N_+ \cup \{0\}$, and $\psi_t : \Sigma \times N_+ \to T_{VAR}(\boldsymbol{F}_2)$ are defined as follows:

1. For $b = f_{t_1,t_2,\cdots,t_{i-1},*,t_{i+1},\cdots,t_{a(f)}} \in \Sigma$,

   $$\psi_f(b) \triangleq f, \psi_*(b) \triangleq i, \text{ and } \psi_t(b, j) \triangleq t_j, \text{ for } 1 \le j \le i-1, i+1 \le j \le a(f).$$

2. If t is a constant in $\boldsymbol{F}_2$, then $\psi_f(t) = t, \psi_*(t) = 0$.

Definition of $RR(P)$: For a subset P of T^2_{1*}, let $RR(P)$ be the smallest set of rewriting rules defined as follows:

For an axiom in $\boldsymbol{A}_2$, $t_L == X$, consider the unique leaf occurrence $t_L/v_L = X$ of t_L. Let $st(t_L, v_L) = b_1 b_2 \cdots b_q S$. For any $\alpha = a_1 a_2 \cdots a_q$ satisfying 1 and 2, $\alpha \Rightarrow \lambda \in RR(P)$.

1. $\psi_f(a_i) = \psi_f(b_i)$, $\psi_*(a_i) = \psi_*(b_i)$ for $1 \le i \le q$.
2. For b_i with $1 \le i \le q$ and j with $1 \le j \le a(\psi_f(b_i))$ and $j \ne \psi_*(b_i)$, if $\psi_t(b_i, j)$ is a term which has no variable, then $\psi_t(b_i, j) = \psi_t(a_i, j)$. Otherwise ($\psi_t(b_i, j)$ is a variable), if there are i' and j' such that $\psi_t(b_i, j)$ and $\psi_t(b_{i'}, j')$ are the same variable, then $(\psi_t(a_i, j), \psi_t(a_{i'}, j')) \in P$.

Definition of $L_R(L(t), RR(P))$: $L_R(L(t), RR(P))$ is the set of strings in $L(t)$ and those obtained from $L(t)$ by applying rewriting rules in $RR(P)$, that is, $L_R(L(t), RR(P))$ is defined as the smallest set of strings as follows:

1. $L(t) \subset L_R(L(t), RR(P))$ and
2. for $\alpha_1\alpha_2\alpha_3 \in L_R(L(t), RR(P))$ and $\alpha_2 \Rightarrow \lambda \in RR(P)$, $\alpha_1\alpha_3 \in L_R(L(t), RR(P))$.

Definition of $EQSYMB^2(P)$: For the subset P of T^2_{1*}, let $EQSYMB^2(P)$ be the smallest set such that

1. for $a \in \Sigma$, $(a, a) \in EQSYMB^2(P)$,
2. for $a_1, a_2 \in \Sigma$, $(a_1, a_2) \in EQSYMB^2(P)$ if
   (a) $\psi_f(a_1)$ and $\psi_f(a_2)$ are the same function (not constant) in $\boldsymbol{F}_2$,
   (b) $\psi_*(a_1) = \psi_*(a_2)$, and
   (c) for all j with $1 \le j \le a(\psi_f(a_1))$ and $j \ne \psi_*(a_1)$, $(\psi_t(a_1, j), \psi_t(a_2, j)) \in P$.

Definition of $E^2(P, t_1, t_2)$: For a subset P of T^2_{1*}, and two terms $t_1, t_2 \in T_{1*}$, the predicate $E^2(P, t_1, t_2)$ is true if and only if there are strings of the same length, $a_{11}a_{12}\cdots a_{1r} \in L_R(L(t_1), RR(P))$ and $a_{12}a_{22}\cdots a_{2r} \in L_R(L(t_2), RR(P))$, such that $(a_{1j}, a_{2j}) \in EQSYMB^2(P)$, for $1 \le j \le r$.

2.4.2 Decision Algorithm

It is shown in [6] that, to decide whether or not a given cryptographic protocol is secure, it is sufficient to get subsets of T^2_{1*}, $P_0, P_1, \cdots$, where

$$\begin{aligned} P_0 = &\ \{(t,t)|t \in T^2_{1*}\}, \\ P_{i+1} = &\ P_i \cup \{(t_1, t_2)|(t_1, t_2) \notin P_i \text{ and } E^2(P_i, t_1, t_2) = \text{True}\}, \end{aligned}$$

successively, until $P_i = P_{i+1}$ holds, and then check whether $(\overline{g_1}, \overline{g_2})$ is in P_i or not. The protocol is "Secure" if and only if $(\overline{g_1}, \overline{g_2})$ is not in P_i. Since $L_R(L(t_j), RR(P_i))$ with $j = 1, 2$ are regular languages[6, 7], we can evaluate whether $E^2(P_i, t_1, t_2)$ is true or not in the same way of deciding whether the intersection of the two regular languages is empty or not.

The decision algorithm for the security problem of cryptographic protocols is described as follows:

Algorithm for Solving the Security Problem

INPUT: The formal description of the protocol $(\boldsymbol{F}, \boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O}, \boldsymbol{G})$
OUTPUT: The answer, "**Secure**" or "**Insecure**"

STEP 0

1. Compute T_{1*} and $P = \{(t_1, t_1) | t_1 \in T_{1*}\}$.
2. For each $t \in T_{1*}$, construct a nondeterministic finite automaton (NFA) $M_0(t)$ which accepts $L(t)$.

STEP 1

1. Compute $RR(P)$.
2. For each $t \in T_{1*}$, construct an NFA which accepts $L_R(L(t), RR(P))$ from $M_0(t)$ and $RR(P)$. And replace $M_0(t)$ by the resultant NFA.

STEP 2

1. Compute $EQSYMB^2(P)$.
2. For each $(t_1, t_2) \in T^2_{1*}$, which is not in P, construct an NFA $M_1^2(t_1, t_2)$ which accepts a nonempty language if and only if the predicate $E^2(P, t_1, t_2)$ is true.
3. Compute

$$P' = \{(t_1, t_2) | (t_1, t_2) \notin P \text{ and } M_1^2(t_1, t_2) \text{ accepts a nonempty language}\}.$$

STEP 3

If $P' = \emptyset$ then goto **STEP 4**, else replace P by $P \cup P'$, and goto **STEP 1**.

STEP 4

If $(\overline{g_1}, \overline{g_2}) \notin P$, then output "**Secure**", else output "**Insecure**".

2.5 Implementation of Decision Algorithm

We have developed a decision system for the security problem of cryptographic protocols on UNIX Workstation with language C. The number of lines of the source program is about 3000. In this section, we describe the implementation of the algorithm. For other parts of the algorithm not described below, the implementation is straightforward.

Construction of NFA, $M_0(t)$, which Accepts $L(t)$ [STEP 0–2]

Instead of constructing a state transition diagram of $M_0(t)$ for each $t \in T_{1*}$, we construct one state transition diagram, denoted δ, as follows:

1. For $t \in T_1$, consider a tree representation of term t as shown in Fig. 2.1(a). Let nodes of the tree expressions of terms in T_1 and s_F be the states of the state transition diagram. s_F is the only final state.
2. Let each edge of a tree be directed in the direction from root to leaf. If directed edge e is from v labeled f and the term expressed by the subtree whose root is v is $f(t_1, t_2, \cdots, t_{a(f)})$, then let the label of e be $f_{t_1,t_2,\cdots,t_{i-1},*,t_{i+1},\cdots,t_{a(f)}}$(see Fig. 2.1(b)).
3. For each leaf of trees whose label is a constant c, then add a directed edge labeled c from the leaf to s_F. For each leaf whose label is a variable, add directed edges labeled ε from the leaf to each root of terms $t \in T_0$.

Since $t \in T_{1*}$ is a subterm of T_1, there is a state, $s_{init}(t)$, which corresponds to the root of the tree for t. In the state transition diagram δ, regard $s_{init}(t)$ as the initial state. Then the resulting NFA determined by δ and $s_{init}(t)$ accepts $L(t)$.

Procedure to Update $M_0(t)$[STEP 1–2]

For each path on a state transition diagram, let the label of the path be the string obtained by concatenating labels of edges on the path. For example, if all labels of edges are ε then the label of the path is λ, and if the labels of edges are a, ε, b then the label of the path is ab.

Update NFA $M_0(t)$ to accept the language $L_R(L(t), RR(P))$ as follows:

For each reduction rule $w \Rightarrow \lambda$ in $RR(P)$ and all paths labeled w, repeat adding an edge labeled ε which connects each ends of w on δ.

Let $RR_1(P)$ be the set of reduction rules which are in $RR(P)$ before the latest execution of STEP 1–2 (if this is the first turn then $RR_1(P) = \emptyset$) and let $RR_0(P)$ be the set of reduction rules added by the execution of STEP 1–1 of this turn.

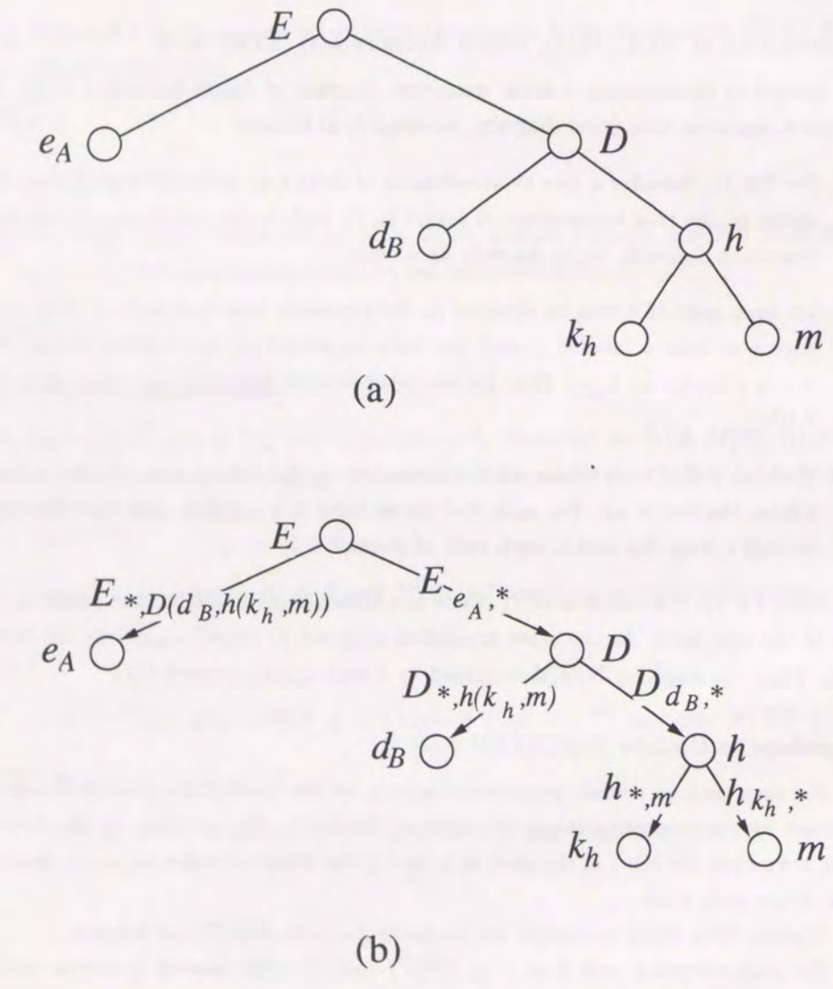


Fig. 2.1 Tree expression of a term $t = E(e_A, D(d_B, h(k_h, m)))$.

Then, if we can add edges to the state transition diagram, we can do it by using rules in $RR_0(P)$ only. First, deal with those paths which have already existed and $RR_0(P)$, and next, deal with those paths which contain edges added newly and $RR_0(P) \cup RR_1(P)$.

To describe the procedure, we introduce some definitions. On the state transition diagram at each time, let $ST^+(s, w)$ be the set of states reachable from the state s via a path labeled w, and let $ST^-(s, w)$ be the set of those states from which there is a path labeled w to the state s. $ST^+(s, \lambda)$ (or $ST^-(s, \lambda)$) contains s itself.

Procedure to Update δ

INPUT: $\delta, RR_0(P), RR_1(P)$
OUTPUT: The new δ

1. Add all edges of δ to a queue of edges, denoted $QE0$, and let another queue of edges, denoted $QE1$, be empty.

2. If $QE0$ is empty then go to 3, else repeat (a) and (b) during it is not empty.

   (a) Take out the first edge e from $QE0$. Suppose that e is an edge labeled a from s_1 to s_2.

   (b) For the edge e and each reduction rule $RR_0(P)$ such that $a_1a_2 \Rightarrow \lambda$ ($a_i \in \Sigma, 1 \leq i \leq 2$), execute the following procedure.

      i. If $a = a_1$, then get the sets of states $K_B = ST^-(s_1, \lambda)$ and $K_E = ST^+(s_2, a_2)$ and for any pair of states, $s_B \in K_B$ and $s_E \in K_E$, such that there is no edge labeled ε from s_B to s_E, add the edge labeled ε to δ and enqueue the edge to $QE1$.

      ii. If $a = a_2$, get two sets of states $K_B = ST^-(s_1, a_1)$ and $K_E = ST^+(s_2, \varepsilon)$, and add edges to δ and $QE1$ like above.

      iii. If $a = \varepsilon$, then get two sets of states, $K_B = ST^-(s_1, a_1)$ and $K_E = ST^+(s_2, a_2)$, and add edges to δ and $QE1$ like above.

3. If $QE1$ is empty, then output δ and terminate, else repeat (a) and (b) until $QE1$ becomes empty.

   (a) Take out the first edge e of $QE1$. Suppose that e is an edge labeled a from s_1 to s_2.

   (b) For the edge e and each reduction rule in $RR_0(P) \cup RR_1(P)$ such that $a_1a_2 \Rightarrow \lambda$ ($a_i \in \Sigma, 1 \leq i \leq 2$), execute 2(b)i–2(b)iii.

Procedure to Construct M_1^2 [STEP 2–2]

Let the set of pairs of states in δ be denoted K^2. Construct the following state transition diagram δ^2 on K^2. For each state in K^2, (s_1, s_2), and each element in $EQSYMB^2(P)$, (a_1, a_2), add an edge from each state of a subset of K^2, $ST^-(s_1, a_1) \times ST^-(s_2, a_2)$ to (s_1, s_2).

Procedure to Get P' [STEP 3–1]

Use the state transition diagram δ^2.

1. Let the set of pairs of states, P', be empty.
2. By tracing edges of δ^2, get the set of those states, denoted K_P, from which the final state, $(s_F, s_F) \in K_2$, is reachable on δ^2.
3. For $(t_1, t_2) \notin P$, if $(s_{init}(t_1), s_{init}(t_2)) \in K_P$, then add (t_1, t_2) to P'.

2.6 Experimental Results and Evaluation

2.6.1 Digital Signature Protocols

For two digital signature protocols in [10], we consider some security problems and decide their security.

In both protocols, a hash function, denoted h, and encryption and decryption functions of a public key cryptosystem, denoted E and D, respectively, are used. We assume that E and D have the property represented by the following axioms.

$$E(e_X, D(d_X, Y)) == Y,$$
$$D(d_X, E(e_X, Y)) == Y,$$

where e_X is a public encryption key of user X and d_X is a decryption key of user X only known to X.

Digital Signature Protocol (1)

The protocol is shown in Fig. 2.2. A public key k_h is used as the first argument of hash function h. When user A sends a message M and A's signature of M to user B, they do the following operations:

- User A generates A's signature $D(d_A, h(k_h, M))$ for M. Then A sends the pair of the message and the signature to B.

- User B gets the pair. B computes the result obtained by applying the function h with the public key k_h to the message and the result obtained by applying the function E with A's public key e_A to the signature. B accepts the pair if and only if they are the same.

For the security of the protocol shown in Fig. 2.2, we consider the following two problems:

a-1 Suppose that A sent a message M and A's signature $D(d_A, h(k_h, M))$ for M to B. Whether or not a enemy B and/or another user C can forge A's signature $D(d_A, h(k_h, M'))$ for a message M' $(M' \neq M)$ chosen by the enemy?

a-2 Whether or not a enemy B and/or another user C can obtain the pair, a message Z and the signature $D(d_A, h(k_h, Z))$?

For example, the formal description of **a-1** is as follows:

$$\begin{aligned}
\boldsymbol{F} &= \{h, E, D, M, M', k_h, d_A, d_B, d_C, \\
&\quad e_A, e_B, e_C, A, B, C\}, \\
\boldsymbol{A} &= \{E(e_X, D(d_X, Y)) == Y, \\
&\quad D(d_X, E(e_X, Y)) == Y| \\
&\quad Y \in \{A, B, C\}\}, \\
\boldsymbol{I} &= \{M, M', D(d_A, h(k_h, M)), k_h, \\
&\quad d_B, d_C, e_A, e_B, e_C, A, B, C\}, \\
\boldsymbol{O} &= \{h(X, Y), E(X, Y), D(X, Y)\}, \\
\boldsymbol{G} &= \{E(e_A, X), h(k_h, M')\}.
\end{aligned}$$

Digital Signature Protocol (2)

For the protocol shown in Fig.2.3, we consider the following two security problems:

b-1 Suppose that A sent a pair, a message M and A's signature for M to B. Whether or not a enemy B and/or another user C can forge A's signature $D(d_A, M')$ for a message M' $(M' \neq M)$ chosen by the enemy?

b-2 Whether or not a enemy B and/or another user C can make a pair, a message X and the signature $h(k_h, E(e_A, Z))$?

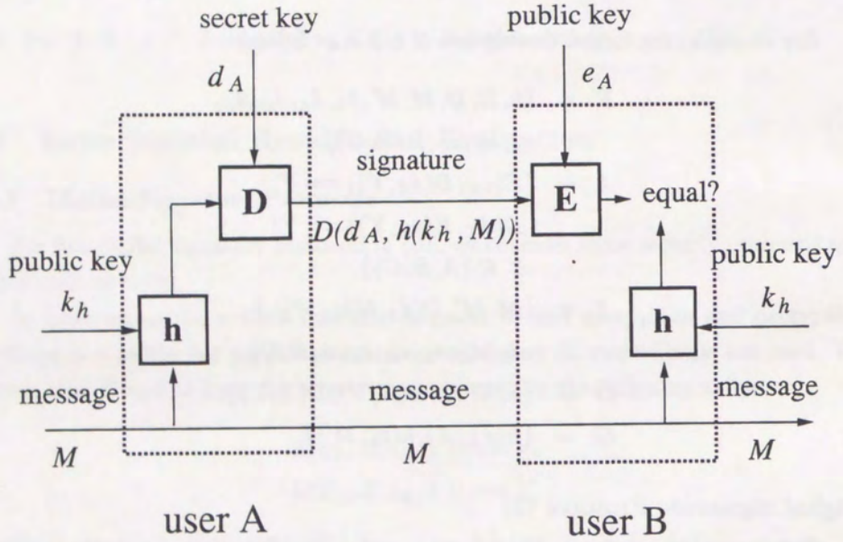


Fig. 2.2 Digital signature protocol (1).

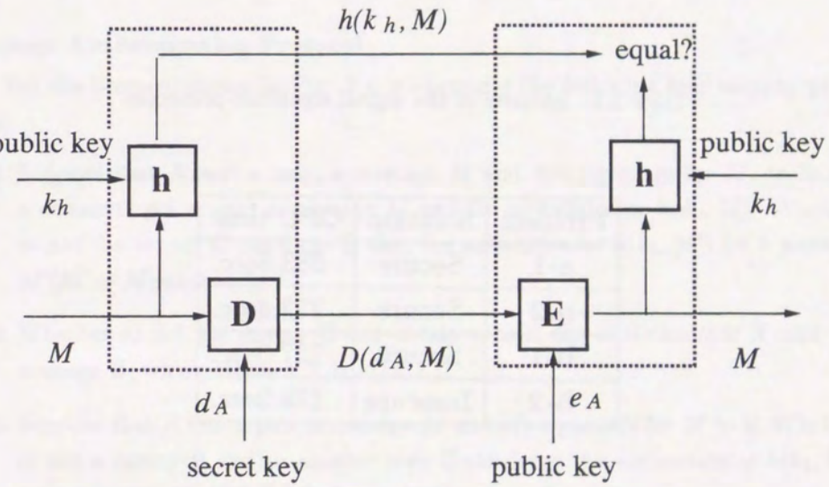


Fig. 2.3 Digital signature protocol (2).

Table 2.1 Security of the digital signature protocols.

Problem	Solution	CPU time
a-1	Secure	383.8sec.
a-2	Secure	272.4sec.
b-1	Secure	737.0sec.
b-2	Insecure	178.2sec.

Decision Results of the Problems

The decision results of the problems and the CPU times to get the conclusions are listed in Table 2.1. The execution is done by using a workstation, Solbourne Series 5/600.

The solution of **b-2** is "Insecure" because there is a pair such as M for Z and $h(k_h, E(e_A, M))$ for X, which satisfies the goal by the enemy, as shown in [19]. But as mentioned in [10], it is possible to detect the unlawful acts of the enemy because the message which B had rarely makes sense.

2.6.2 Message Authentication Protocol

For a protocol for message authentication [10] shown in Fig. 2.4, we consider some security problems and decide their security.

Message Authentication Protocol

For the protocol shown in Fig. 2.4, we consider the following four security problems:

c-1 Suppose that A sent a pair, a message M and A's signature for M, to B and a enemy C got a pair, a message M and its authenticator $h(k_h, M)$. Whether or not the enemy C can forge B that the authenticator $h(k_h, M')$ for a message $M'(M' \neq M)$ chosen by C?

c-2 Whether or not the enemy C can obtain a pair, the authenticator X and the message Z, which satisfies $X = h(k_h, Z)$?

c-3 Suppose that A sent a pair, a message M and A's signature for M to B. Whether or not a enemy B and/or another user C can forge the authenticator $h(k_h, M')$ for a message M' $(M' \neq M)$ chosen by the enemy?

c-4 Whether or not a enemy B and/or another user C can obtain a pair, a message Z and the authenticator X, which satisfies $X = h(k_h, Z)$?

In **c-3** and **c-4**, we consider whether or not the protocol can be used as a digital signature protocol.

Decision Results of the Problems

The decision results of the above problems and the CPU times to get the conclusions are listed in Table 2.2.

The solutions of **c-3** and **c-4** are "Insecure" because the protocol is not for digital signature. Therefore, the results do not matter in the case that this protocol is used for message authentication.

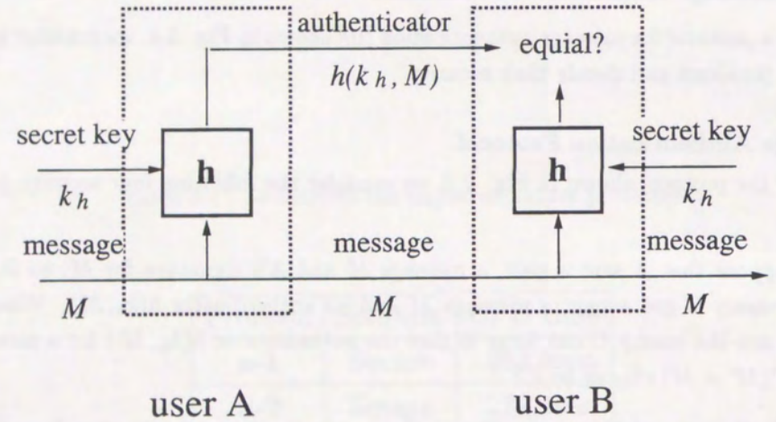


Fig. 2.4 Message authentication protocol.

Table 2.2 Security of the message authentication protocol.

Problem	Solution	CPU time
c-1	Secure	2.2sec.
c-2	Secure	10.0sec.
c-3	Insecure	10.0sec.
c-4	Insecure	2.0sec.

2.6.3 Authentication Protocol for Electronic Contracts

For an authentication protocol for electronic contracts [20] shown in Fig. 2.5, we consider some security problems and decide their security.

In Fig. 2.5, c_1 and c_2 denote hash functions, E and D denote encryption and decryption functions on a public key cryptosystem, respectively, and E and D have the property which can realize the authentication. For the details of the protocol, see [20].

For this protocol, we consider the following five security problems:

d-1 After B sent his electronic tally, W'_B, whether or not a enemy C can impersonate A and send the electronic seal of A, W_A?

d-2 Whether or not a enemy C can forge a tally of B, W'_B?

d-3 After A received a tally W'_B of B, whether or not A can forge the electronic seal of B, W_B, without sending his seal, W_A?

d-4 After B received a message m from A, whether or not B can forge the seal of A, W_A, without sending his seal W_B?

d-5 When a enemy C knew the secret key of A, s_A, whether or not C can send the seal of A, W_A, after he receives a tally of B, W'_B?

Decision Results of the Problems

The decision results of the problems and the CPU times to get the conclusions are listed in Table 2.3.

The solution for **d-5** is "Insecure" because the enemy knew the secret key of A. But in [20], proposers of this protocol suppose that the enemy cannot get any secret key, and they also propose the protocol of secret key distribution (we have verified that this key distribution protocol is also "Secure"). Therefore, the result does not matter for the security of this protocol.

In addition, we have decided the security of cryptographic protocols proposed in [17, 18]. And we confirmed that each protocol is "Secure" with respect to the proposer's intention.

2.7 Conclusions

In this chapter, a system for deciding the security of cryptographic protocols is shown. This system can decide the security in polynomial time. Although there is a protocol (the unsigned bidding proposed in [17]) which takes 2 days to decide its

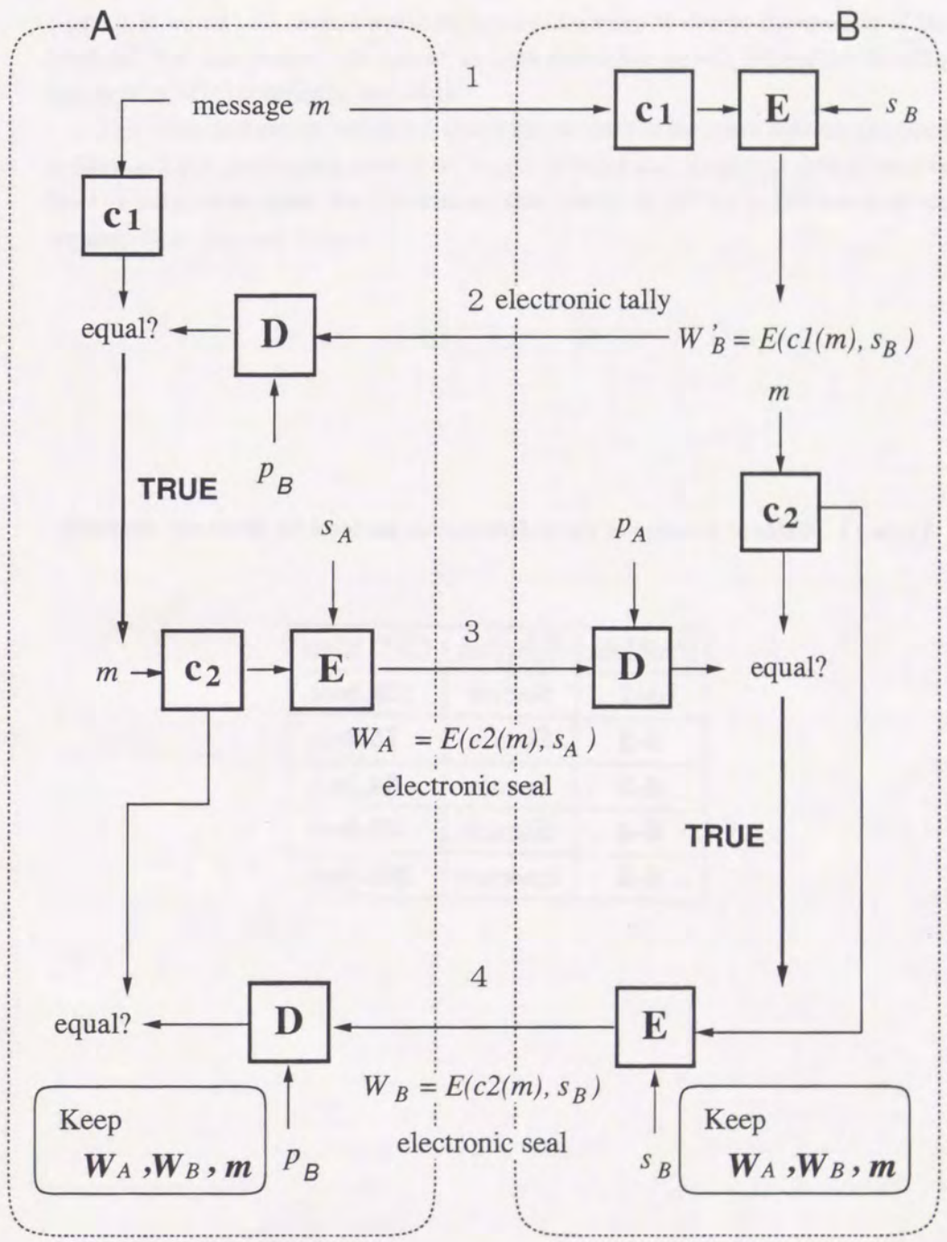


Fig. 2.5 Authentication protocol for electronic contracts.

Table 2.3 Table 3: Security of the authentication protocol for electronic contracts.

Problem	Solution	CPU time
d-1	Secure	139.8sec.
d-2	Secure	75.8sec.
d-3	Secure	84.2sec.
d-4	Secure	43.6sec.
d-5	Insecure	391.3sec.

security, it would take several weeks for a trained person to decide the security of the protocol. For that reason, the system we have developed is very helpful for deciding the security of cryptographic protocols.

This is the first system which can assure the security in the sense that we described in Section 2.2 in polynomial time of the length of input and which can detect security flaws in polynomial time. For this reason, this system must be a useful assistant for cryptographic protocol designers.

Chapter 3

A Polynomial Time Verification Method for Security of a Wider Class of Cryptographic Protocols

3.1 Introduction

In the previous chapter, we implemented the decision algorithm proposed in [6, 7] and evaluated the usefulness of the algorithm by verifying the securities of several cryptographic protocols. As a result, it was shown that this algorithm is very useful but there are some cases that the security problems of cryptographic protocols do not satisfy the sufficient condition under which the decision problem is decidable. The sufficient condition consists of three subconditions. Many of those security problems do not satisfy one subcondition, especially the condition 1.3 called right-linearity, but satisfy any other subconditions of the sufficient condition.

In this chapter, a polynomial time verification method for security problems including the problems which do not satisfy right-linearity is proposed. To verify the security of such problems, first, the decision problem is modified to the problem with sorts (types). By this modification, it becomes natural to express the operations which check the sorts of the inputs and, only if the sorts are valid, output some informations. If the domain of a variable which violates right-linearity in an axiom is finite, the names of users for example, we can transform the description of such problem to the following description which satisfies right-linearity. The new description is obtained from original one by replacing the axiom with axioms which are obtained by substituting the values in the domain to the variable. We prove that the original description and the transformed description are equivalent on the security by showing that the set of values which the enemies can get in the original description and that in the transformed one are the same, that is, the original description is secure if and only if the transformed description is secure. If the size of the domain is too large, it is not practical to apply this transformation to get a formal description of a security problem which satisfies right-linearity from a description which does not satisfy right-linearity. We show a condition under which it is enough to consider only a few representative elements in

the domain. When this condition is satisfied in the original description, for the original description and the transformed description which can be obtained by adding a new constant to the domain of the original description, we prove that these two descriptions are equivalent on the security by showing that the set of values which the enemies can get in the transformed description is greater than that in the original description but the values which the enemies want to get are not in the difference of two sets. By using this result, we show the way to decide the size of the domain which is sufficient to verify the security. For example, when the goal of the enemies is to get the secret message and operations which are not prohibited in the protocol are the same for all users, the description of the problem satisfies this condition. In this case, as the user's names, it is sufficient to consider one user's name except for those who send or receive the secret message legally, and we can use the first transformation practically to get the description which satisfies right-linearity.

We also show a transformation of a description which is obtained by replacing each occurrence of the variable which violates right-linearity in the right-hand side of axioms as an occurrence of different variables. For example, the term $h_1(X, Y, h_2(Y, h_3(X, Y)))$ is transformed to $h_1(X_1, Y_1, h_2(Y_2, h_3(X_2, Y_3)))$. In this transformation, since the set of values which the enemies can get in the transformed description includes the set of values which the enemies can get in the original description, the original description is secure if the transformed description is secure. This is not an equivalent transformation but, by using this transformation, we can always get a description which satisfies right-linearity grammatically.

By using these transformations as subroutines, we present a new polynomial time verification method for security problems including the problems which do not satisfy right-linearity.

For an application of this new method, we consider a basic security problem of a network authentication protocol, Kerberos, "whether or not an enemy client process and an enemy server process can obtain the session key between a client and a server." The decision algorithm proposed in [6, 7] is not applicable to the security problem. And we show that Kerberos is secure for that problem in the above sense. This result shows that this new verification method is more useful than the decision algorithm. The security of Kerberos was also discussed in [1] but active attacks like impersonating are not considered in this method.

This chapter is organized as follows. In Section 3.2, a brief review of the generalized security problem of cryptographic protocols is given. In this section, the authentication protocol Kerberos is described briefly, and one of the security problems for Kerberos is considered. Section 3.3 explains our new verification method. In Section 3.4, for an application of our new method, the security problem for Kerberos is summarized and

we show the usefulness of this method.

3.2 Generalized Security Problem

3.2.1 Definition of Generalized Security Problem

A formal description of the security problem of cryptographic protocols has been presented in [6] and [7]. In this chapter, we slightly modify it by introducing the sorts (or data types).

For a given cryptographic protocol, the security problem is defined by 5-tuple $(\boldsymbol{F}, \boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O}, \boldsymbol{G})$, where $\boldsymbol{F}, \boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O}$ and $\boldsymbol{G}$ are defined as follows:

1. $\boldsymbol{F}$ is a finite set of function symbols which contains constant symbols. $\boldsymbol{F}$ consists of the symbols of operations of the protocol, i.e. function symbols of encryption and decryption, constant symbols such as messages and keys, and so on.

   For $f \in \boldsymbol{F}$, let $a(f)$ denote the number of arguments of f. We assume that there is a rank function $rank$, which assigns $rank(f) = (\boldsymbol{u}; S^f)$ to each function symbol $f \in \boldsymbol{F}$, where $\boldsymbol{u}$ is an $a(f)$-tuple of sorts and S^f is a sort. The i-th component of $\boldsymbol{u}$ with $1 \leq i \leq a(f)$ denotes the sort of the i-th argument of f and S^f is the sort of returned value of f. Note that each constant a also has rank of the form $(\lambda; S^a)$ where λ is the 0-tuple. Let the sort of a term be the sort of the out-most function of the term. Hereafter, we abbreviate a rank of the form $((S_1^f, \ldots, S_n^f); S^f)$ by $(S_1^f, \ldots, S_n^f; S^f)$. For $\boldsymbol{F}$, the infinite set $T_{VAR}(\boldsymbol{F})$ of terms on $\boldsymbol{F}$ and the sort of terms except for variables are defined as follows:

   (a) For a constant $c \in \boldsymbol{F}, c \in T_{VAR}(\boldsymbol{F})$. The sort of the term c is that of the constant function c.

   (b) A variable X is in $T_{VAR}(\boldsymbol{F})$.

   (c) For $f \in \boldsymbol{F}$ and $t_1, \ldots, t_{a(f)} \in T_{VAR}(\boldsymbol{F})$ such that t_i with $1 \leq i \leq a(f)$ is a variable or a term of sort S_i^f, $f(t_1, \ldots, t_{a(f)}) \in T_{VAR}(\boldsymbol{F})$. The sort of the term $f(t_1, \ldots, t_{a(f)})$ is defined as that of f.

   Let $T(\boldsymbol{F})$ be the set of terms in $T_{VAR}(\boldsymbol{F})$ without variables.

2. $\boldsymbol{A}$ is a finite set of axioms by which we define the properties of the functions of $\boldsymbol{F}$. We assume that enemies can use no knowledge on the properties of functions other than the axioms in $\boldsymbol{A}$. The congruence relation defined by $\boldsymbol{A}$ is denoted by $\overset{\equiv}{A}$. If no-hand side of an axiom is a variable, then the both-hand sides have the same sort.

3. $\boldsymbol{I}$ is a finite subset of $T(\boldsymbol{F})$. $\boldsymbol{I}$ consists of terms which correspond to information that enemies know, e.g. their own keys and information obtained through wiretapping.

4. $\boldsymbol{O}$ is a subset of $\boldsymbol{F}$. $\boldsymbol{O}$ consists of the symbols of those operations which are not prohibited to the enemies.

   Let $Q(\boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O})$ be the set of terms representing such information that the enemies can obtain by executing any combination of operations provided by the system on the information corresponding to terms in $\boldsymbol{I}$ and by drawing inference from axiom in $\boldsymbol{A}$. $Q(\boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O})$ is defined formally as follows:

   (a) $\boldsymbol{I} \subset Q(\boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O})$.

   (b) For $f \in \boldsymbol{O}$ and $t_i \in Q(\boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O})$ whose sort is S_i^f with $1 \leq i \leq a(f)$, $f(t_1, t_2, \cdots, t_{a(f)}) \in Q(\boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O})$.

   (c) For $t_1 \in Q(\boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O})$ and $t_2 \in T(\boldsymbol{F})$ such that $t_1 \stackrel{\equiv}{A} t_2$, $t_2 \in Q(\boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O})$.

5. $\boldsymbol{G} = \{g_1, g_2\}$ is a pair of terms $g_1, g_2 \in T_{VAR}(\boldsymbol{F})$. It represents enemies' goal in the following sense:

   Let the set of all the variables that occur at least once in g_1 or g_2 be $\{X_1, X_2, \cdots, X_h\}$ and suppose that the sort of terms which can be substituted for X_i with $1 \leq i \leq h$ is S_i. Consider the following $\boldsymbol{O}$-Restricted Unification Problem(ORUP):

   ORUP: The problem to decide whether or not there exist h terms, $t_i \in Q(\boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O})$ of sort S_i such that, for the substitution $\sigma \stackrel{\triangle}{=} \{X_1 \mapsto t_1, \ldots, X_h \mapsto t_h\}$, $g_1\sigma \stackrel{\equiv}{A} g_2\sigma$, where $\{X_1 \mapsto t_1, \ldots, X_h \mapsto t_h\}$ denotes substituting t_i for X_i in g_1 and g_2 with $1 \leq i \leq h$. □

   The goal of enemies is to obtain at least one solution of ORUP, that is, to obtain at least one substitution σ which satisfies $g_1\sigma \stackrel{\equiv}{A} g_2\sigma$.

The answer of the security problem for the protocol described by $(\boldsymbol{F}, \boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O}, \boldsymbol{G})$ is defined to be secure if and only if there exist no h terms in $Q(\boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O})$ which satisfy the condition. For simplicity, we say $(\boldsymbol{F}, \boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O}, \boldsymbol{G})$ is secure (or not secure), if the answer of it is secure (or not secure).

We may consider several $(\boldsymbol{F}, \boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O}, \boldsymbol{G})$'s for the security of one protocol.

3.2.2 Example: A Security Problem of Kerberos Protocol

In this section, we first describe the network authentication protocol Kerberos briefly. The aim of this protocol is to provide secure communication between a client

process and a server process by having a common session key between them. In detail, see [13].

In Kerberos protocol, there are two special kinds of servers, called Kerberos Server and Ticket-Granting Server (TGS). For simplicity, we consider the simplest case where only one Kerberos Server and one TGS exist.

Kerberos Server holds all secret keys of servers, clients and TGS. Let tgs be the identifier of TGS, let $K(s)$, $K(c)$ and $K(tgs)$ be a server s's, a client c's and tgs's secret key, and let $Ks(c, s)$ be the session key for the communication between c and s, respectively. Kerberos server creates a session key between a client and TGS, and a ticket by which TGS can authenticate the client. TGS creates a session key between a client and a server, and a ticket by which the server can authenticate the client.

Let $C2$ and $C3$ denote functions concatenate 2 and 3 elements, respectively. Let E and D denote encryption and decryption functions on symmetric key cryptosystem, respectively. $E(K, M)$ means the ciphertext of plain text M encrypted by key K and $D(K, C)$ means the decrypted text of ciphertext C by key K.

In Kerberos protocol, a client c and a server s can get a common session key each other by executing the following procedure (see Figure 3.1):

1. Client c sends its identifier c and the identifier of TGS tgs to the Kerberos server to request issuing a ticket for TGS.

2. Kerberos server creates a session key between the client c and the TGS, $Ks(c, tgs)$, and a ticket T_{c-tgs} for TGS where

   $$T_{c-tgs} = E(K(tgs), C3(c, tgs, Ks(c, tgs))).$$

   Then encrypt them by client c's secret key $K(c)$ and returns it to the client c.

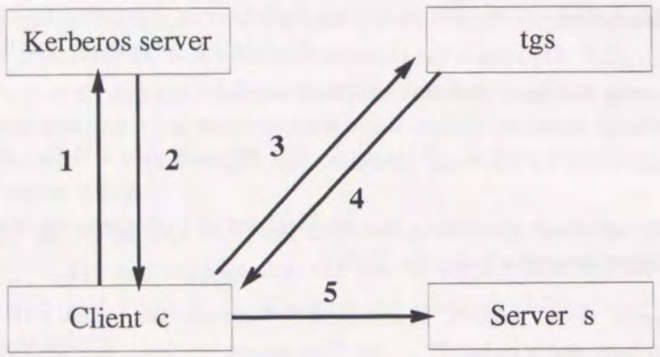
3. The client c gets T_{c-tgs} and $Ks(c, tgs)$ by decrypting the data sent by Kerberos server with his secret key. By using the key $Ks(c, tgs)$, the client make his authenticator,

   $$A_{c-tgs} = E(Ks(c, tgs), c),$$

   and send the server's identifier s, T_{c-tgs} and A_{c-tgs} to TGS.

4. TGS confirms the validity of T_{c-tgs} and A_{c-tgs} received from the client c. Note that TGS can obtain $Ks(c, tgs)$ from T_{c-tgs} by using TGS's secret key. Only when they are valid, that is, when the names of client in T_{c-tgs} and A_{c-tgs} are the same, TGS creates a session key between the client c and the server s, denoted $Ks(c, s)$, and a ticket for the server s,

   $$T_{c-s} = E(K(s), C3(c, s, Ks(c, s))).$$



1. c, tgs
2. $E(K(c), C2(Ks(c,tgs), E(K(tgs), C3(c,tgs,Ks(c,tgs)))))$
3. s, $E(K(tgs), C3(c,tgs,Ks(c,tgs)))$,$E(Ks(c,tgs),c)$
4. $E(Ks(c,tgs), C2(Ks(c,s), E(K(s), C3(c,s,Ks(c,s)))))$
5. $E(K(s), C3(c,s,Ks(c,s)))$, $E(Ks(c,s),c)$

Fig. 3.1 Kerberos protocol.

Then, TGS encrypts them by session key $Ks(c,tgs)$ and returns $E(Ks(c,tgs), C2(Ks(c,s), T_{c-s}))$ to the client c.

5. The client c gets T_{c-s} and $Ks(c,s)$ by using the session key $Ks(c,tgs)$, then, sends T_{c-s} and c's authenticator ,

$$A_{c-s} = E(Ks(c,s), c),$$

to the server s.

6. The server s confirms the validity of T_{c-s} and A_{c-s} received from the client c and gets the secret session key between the client c and the server s by decrypting T_{c-s} with his own secret key.

We omit timing information like a timestamp, for simplicity, since it is secure with properly introduced restrictions on timing if the protocol is secure without them and we are mainly concerned with the core of the security problem.

A Security Problem of Kerberos

Consider the following security problem of Kerberos protocol when a client c_1 and a server s_1 get a common session key $Ks(c_1,s_1)$ by using Kerberos protocol. The problem is whether or not an enemy client process, e_c, and an enemy server process, e_s, can obtain the session key between a client, c_1, and a server, s_1, under the condition that the enemy can use any information from unprotected channels and operations not prohibited in the protocol.

Let n_s and n_c be the numbers of servers and clients which are not enemy, respectively. We assume that n_c and n_s are constants and that one session key has been held by each pair of a client process and a server process. We consider the case that e_c and e_s have wiretapped all sessions concerned between all server processes and all client processes.

Then, the above security problem is described as follows:

$$\begin{aligned} \boldsymbol{F} \;=\; & \{E, D, C2, C3, P21, P22, P31, P32, P33, OPK, OPT, Auth, K, Ks \\ & tgs, e_c, e_s\} \cup \{c_i | 1 \le i \le n_c\} \cup \{s_j | 1 \le j \le n_s\}. \end{aligned}$$

We use three sorts, S_C, S_S and S_M in this description. The sort of functions which denote the names of client processes is S_C. The sort of functions which denote the names of server processes is S_S. The sorts of the remaining functions are S_M. Sorts S_C and S_S are subsorts of S_M, that is, a function t of sort S_C (or S_S) is also of sort S_M. In other words, the sort of every term in $T_{VAR}(\boldsymbol{F})$ is S_M or its subsort. In Kerberos protocol, the Kerberos server and the TGS can check the sorts of data sent

by a process and if at least one of the sorts are not valid, these server do not reply to the process.

The meaning and the rank function of each operation are summarized in Tables 3.1 and 3.2, respectively.

$$
\begin{aligned}
\boldsymbol{A} &= \{E(X,D(X,Y)) == Y, D(X,E(X,Y)) == Y,\\
&\quad P21(C2(X,Y)) == X, P22(C2(X,Y)) == Y,\\
&\quad P31(C3(X,Y,Z)) == X, P32(C3(X,Y,Z)) == Y,\\
&\quad P33(C3(X,Y,Z)) == Z, Auth(X,X,Y) == Y,\\
&\quad OPK(X) == E(K(X), C2(Ks(X,tgs), E(K(tgs), C3(X,tgs,Ks(X,tgs))))),\\
&\quad OPT(W,X,Y,Z) == Auth(P31(D(K(tgs),Y)), D(P33(D(K(tgs),Y)),Z),\\
&\qquad E(P33(D(K(tgs),Y)), C2(Ks(W,X), E(K(X), C3(P31(D(K(tgs),Y)),\\
&\qquad X, Ks(W,X))))))\},\\
\boldsymbol{I} &= \{tgs, c_i, s_j, e_c, e_s, K(e_c), K(e_s),\\
&\quad E(K(c_i), C2(Ks(c_i,tgs), E(K(tgs), C3(c_i,tgs,Ks(c_i,tgs))))),\\
&\quad E(K(tgs), C3(c_i,tgs,Ks(c_i,tgs))), E(Ks(c_i,tgs),c),\\
&\quad E(Ks(c_i,tgs), C2(Ks(c_i,s_j), E(K(s_j), C3(c_i,s_j,K(c_i,s_j))))),\\
&\quad E(K(s_j), C3(c_i,s_j,Ks(c_i,s_j))), E(Ks(c_i,s_j),c_i)\},\\
\boldsymbol{O} &= \{E, D, C2, C3, P21, P22, P31, P32, P33, OPK, OPT, Auth\},\\
\boldsymbol{G} &= \{X, Ks(c_1,s_1)\}.
\end{aligned}
$$

In the previous chapter, a sufficient condition is shown under which a security problem is decidable in polynomial time. An input formal description of a security problem to the system described in the previous chapter is required to satisfy the sufficient condition. The system is not applicable to the security problem for Kerberos since the form of axioms for OPK and OPT do not satisfy the following subcondition

Subcondition: Let $\bar{\boldsymbol{O}}$ denote the subset of function symbols in $\boldsymbol{O}$ whose unique axiom has the form $f(X_1,\ldots,X_{a(f)}) == r$, where X_i with $1 \le i \le a(f)$ are distinct variables, and for $f \in \bar{\boldsymbol{O}}$, let A_f denote the axiom of f. Each variable X_i with $1 \le i \le a(f)$ occurs in its right-hand side r exactly once (right-linearity).

3.3 Polynomial Time Verification Method

In the security problem for Kerberos, the numbers n_c and n_s depend on the network, and these values may be very large. Here, we prove that it is enough to consider

Table 3.1 Functions used in Kerberos protocol.

E	The encryption function.
D	The decryption function.
C_2, C_3	Concatenate functions for two and three elements, respectively.
$P21, P22$	Projection functions for $C2$ which satisfy the third and forth axioms in $\boldsymbol{A}$, respectively.
$P31, P32, P33$	Projection functions for $C3$ which satisfy the fifth, sixth and seventh axioms in $\boldsymbol{A}$, respectively.
OPK	A function which corresponds to the operation in steps 1 and 2.
OPT	A function which corresponds to the operation in steps 3 and 4.
$Auth$	A function which decides the validity of information sent in step 3.

Table 3.2 The *rank* functions of function symbols.

function symbol	*rank* of the function
E, D	$(S_M, S_M; S_M)$
$C2$	$(S_M, S_M; S_M)$
$C3$	$(S_M, S_M, S_M; S_M)$
$P21, P22$	$(S_M; S_M)$
$P31, P32, P33$	$(S_M; S_M)$
OPK	$(S_C; S_M)$
OPT	$(S_C, S_S, S_M, S_M; S_M)$
$Auth$	$(S_M, S_M, S_M; S_M)$
K	$(S_M; S_M)$
Ks	$(S_M, S_M; S_M)$
tgs	$(\lambda; S_M)$
c_i, e_c	$(\lambda; S_C)$
s_j, e_s	$(\lambda; S_S)$

the case where $n_c = 1$ and $n_s = 1$. The similar problem occurs for many protocols. Therefore, we consider this problem in a more general framework.

Consider a security problem $P \triangleq (\boldsymbol{F}, \boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O}, \boldsymbol{G})$. For a constant a in $\boldsymbol{F}$, introduce a new constant a' of sort S^a and consider the following security problem $P_+ \triangleq (\boldsymbol{F}_+, \boldsymbol{A}, \boldsymbol{I}_+, \boldsymbol{O}, \boldsymbol{G})$, where

$$\begin{aligned} \boldsymbol{F}_+ &= \boldsymbol{F} \cup \{a'\}, \\ \boldsymbol{I}_+ &= \boldsymbol{I} \cup \{\varphi(t) | t \in \boldsymbol{I}, t \text{ contains } a \text{ in its subterm.}\}, \end{aligned}$$

and $\varphi(t)$ is the term obtained from t by replacing every a in t with a'. For P_+ and P, the following theorem holds in general.

Theorem 1: P_+ is secure *iff* P is secure.
Proof. See Appendix A. □

Corollary 1: The security problem for Kerberos with any n_c and n_s is secure *iff* the problem with $n_c = 1$ and $n_s = 1$ is secure.

For a security problem which does not satisfy the Subcondition (right-linearity) explained in the previous section but satisfies other subconditions of the sufficient condition, there exist many cases where the security of the protocol can be verified by using the following two theorems.

Let $\dot{\boldsymbol{O}}$ denote the subset of function symbols in $\boldsymbol{O}$ whose unique axiom has the form $f(X_1, \ldots, X_{a(f)}) == r$, where X_i with $1 \leq i \leq a(f)$ are distinct variables but this axiom does not satisfy the right-linearity, and for $f \in \dot{\boldsymbol{O}}$, let A_f denote the axiom of f. Consider a formal description $P \triangleq (\boldsymbol{F}, \boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O}, \boldsymbol{G})$ such that

1. the right-hand side of A_f does not include f in its subterm and
2. there is no occurrence of f in every term in $\boldsymbol{I}$ and $\boldsymbol{G}$.

For $f \in \dot{\boldsymbol{O}}$, let $f(X_1, \ldots, X_{a(f)}) == r$ be the unique axiom in which f appears. Let D_i be the set of terms in $T(\boldsymbol{F})$ with sort S_i^f with $1 \leq i \leq a(f)$. Without loss of generality, we can assume that there is a non-negative integer l with $l \leq a(f)$ such that, for any i with $1 \leq i \leq l$, all of the following three conditions hold, and for $l < i \leq a(f)$, at least one of the conditions does not hold:(1) X_i appears more than once in r, (2) the number of element in D_i is finite, and (3) $D_i \subseteq \boldsymbol{I}$.

Now we consider the following transformed formal description of a security problem $\hat{P} \triangleq (\hat{\boldsymbol{F}}, \hat{\boldsymbol{A}}, \hat{\boldsymbol{I}}, \hat{\boldsymbol{O}}, \hat{\boldsymbol{G}})$:

1. $\hat{\boldsymbol{F}}$ is the set of function symbols obtained by replacing function f with function symbols $f_{a_1,\ldots,a_l}$ where $a_i \in D_i$.

2. $\hat{\boldsymbol{A}}$ is the set of axioms defined as follows: If $l = a(f)$, $\hat{\boldsymbol{A}} = \boldsymbol{A} - \{f(X_1, \ldots, X_{a(f)}) == r\}$. Otherwise, replace the axiom $f(X_1, \ldots, X_{a(f)}) == r$ with the following axioms: for all substitutions $\sigma \triangleq \{X_1 \mapsto a_1, \ldots, X_l \mapsto a_l\}$ with $a_i \in D_i$,

$$f_{a_1,\ldots,a_l}(X_{l+1}, \ldots, X_{a(f)}) == r\sigma.$$

3. $\hat{\boldsymbol{I}} \triangleq \boldsymbol{I} \cup \{r\sigma | \sigma = \{X_1 \mapsto a_1, \ldots, X_l \mapsto a_l\}, a_i \in D_i, 1 \leq i \leq l\}$ if $l = a(f)$. Otherwise, $\hat{\boldsymbol{I}} \triangleq \boldsymbol{I}$.

4. $\hat{\boldsymbol{O}} \triangleq \boldsymbol{O} - \{f\}$ if $l = a(f)$. Otherwise, let $\hat{\boldsymbol{O}}$ be the set of function symbols obtained from $\boldsymbol{O}$ by replacing f with all functions $f_{a_1,\ldots,a_l}$ where $a_i \in D_i$.

5. $\hat{\boldsymbol{G}} \triangleq \boldsymbol{G}$.

For P and $\hat{P}$, the following theorem obviously holds:

Theorem 2: P is secure *iff* $\hat{P}$ is secure.

If there is no axiom in $\hat{\boldsymbol{A}}$ which does not satisfy the right-linearity, then by applying the system described in the previous chapter to the transformed problem, we can verify the security.

In the security problem of Kerberos described in the previous section, this theorem can be used to remove the non-right-linearity of A_{OPK} and A_{OPT}. But there still remains an axiom A_{OPT} which does not satisfy the right-linearity.

For a formal description of a security problem $P \triangleq (\boldsymbol{F}, \boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O}, \boldsymbol{G})$, let $\dot{\boldsymbol{O}}$ be the set of function symbols described in this subsection. And for each $f \in \dot{\boldsymbol{O}}$, let l_i denote the number of occurrences of X_i in the right-hand side of A_f with $1 \leq i \leq a(f)$. Then, introduce a new function f' of arity $\sum_{i=1}^{a(f)} l_i$ for f and transform the security problem P to $P' \triangleq (\boldsymbol{F}', \boldsymbol{A}', \boldsymbol{I}', \boldsymbol{O}', \boldsymbol{G}')$ in the following way:

1. For each $f \in \dot{\boldsymbol{O}}$, replace each f with f' in $\boldsymbol{F}$. Let $\boldsymbol{F}'$ denote new $\boldsymbol{F}$.
2. For each $f \in \dot{\boldsymbol{O}}$, replace each f with f' in $\boldsymbol{O}$. Let $\boldsymbol{O}'$ denote new $\boldsymbol{O}$.
3. For each f' with $f \in \boldsymbol{O}$, add an axiom of f', instead of f's axiom, to $\boldsymbol{A}$ as follows: To describe the new axioms, we need some definitions. Let N_+ and N_+^* be the set of positive integers and the set of strings of positive integers, respectively. Let λ be the empty string in N_+^*. For $t \in T_{VAR}(\boldsymbol{F})$, we define the set of "occurrences" of subterms of t, denoted $Occ(t) \subseteq N_+^*$, and the subterm at occurrence v, denoted t/v, for $v \in Occ(t)$ as follows:

   (a) If t is a constant in $\boldsymbol{F}$ or a variable, then $Occ(t) \triangleq \{\lambda\}$, and $t/\lambda \triangleq t$.

   (b) If $t = f(t_1, t_2, \cdots, t_{a(f)})$, then $Occ(t) \triangleq \{\lambda\} \cup \{iv | 1 \leq i \leq a(f), v \in Occ(t_i)\}$, $t/\lambda \triangleq t$ and $t/iv \triangleq t_i/v$.

The left-hand side of f''s axiom is

$$f'(X_{11}, \ldots, X_{1l_1}, \ldots, X_{a(f)1}, \ldots, X_{a(f)l_{a(f)}}).$$

Suppose that the occurrences of X_i in r with $1 \leq i \leq a(f)$ are $O_{i1}, \ldots, O_{il_i}$, then the right-hand side of f''s axiom be the form obtained from r by replacing r/O_{ij} with a variable X_{ij} where $1 \leq j \leq l_i$. For example, if f's axiom is

$$f(X, Y) == h_1(X, Y, h_2(Y, h_3(X, Y))),$$

where X and Y are variables, then f''s axiom is

$$f'(X_1, X_2, Y_1, Y_2, Y_3) == h_1(X_1, Y_1, h_2(Y_2, h_3(X_2, Y_3))).$$

For a term $t \in T_{VAR}(\boldsymbol{F})$, let $\psi(t)$ be the term obtained from t by replacing every subterm of the form $f(t_1, \ldots, t_{a(f)})$ with

$$f'(\underbrace{t_1, \ldots, t_1}_{l_1}, \ldots, \underbrace{t_{a(f)}, \ldots, t_{a(f)}}_{l_{a(f)}}).$$

4. $\boldsymbol{I}' = \{\psi(t) | t \in \boldsymbol{I}\}$.

5. $\boldsymbol{G}' = \{\psi(g_1), \psi(g_2)\}$.

For any P and its transformed description P', the following theorem holds.

Theorem 3: If P' is secure, then P is also secure.

Proof. See Appendix A. □

Let ($\boldsymbol{F}$,$\boldsymbol{A}$,$\boldsymbol{I}$,$\boldsymbol{O}$,$\boldsymbol{G}$) be the formal description of a security problem by applying Theorem 2 and $\boldsymbol{A}$ contains axioms which do not satisfy the right-linearity. In this case, run the system described in the previous chapter for the transformed formal description ($\boldsymbol{F}'$,$\boldsymbol{A}'$,$\boldsymbol{I}'$,$\boldsymbol{O}'$,$\boldsymbol{G}'$) and decide the security. If the output is secure, which implies that ($\boldsymbol{F}$,$\boldsymbol{A}$,$\boldsymbol{I}$,$\boldsymbol{O}$,$\boldsymbol{G}$) is also secure, then output "Secure", otherwise output "This system cannot decide the security" and halt. In our problem about Kerberos, this theorem is used to remove the non-right-linearity of A_{OPT} (for further detail, see the next section).

3.4 Verification Example: Verification for Security of Kerberos Protocol

From Corollary 1, it is sufficient to consider the case where there exist one server process, denoted s_1, and one client process, denoted c_1, in deciding the security of Kerberos.

By using theorems in the last section, transform the problem as follows: By Theorem 2, instead of considering the function OPK and its axiom, add the terms (to $\boldsymbol{I}$) obtained by substituting a client's names c_1 and an enemy client's name e_c for X of the right-hand side of A_{OPK}, respectively. For the function OPT and its axiom, the set of terms which can be substituted for W and X are $\{c_1, e_c\}$ and $\{s_1, e_s\}$, respectively. By Theorem 2, instead of the function symbol OPT and its axiom, it is sufficient to introduce the new function symbols OPT_{c_1,s_1}, OPT_{e_c,s_1}, OPT_{c_1,e_s} and OPT_{e_c,e_s} and their axioms:

$$\begin{aligned} OPT_{c_1,s_1}(Y,Z) \;==\; & Auth(P31(D(K(tgs),Y)), D(P33(D(K(tgs),Y)),Z), \\ & E(P33(D(K(tgs),Y)), C2(Ks(c_1,s_1), E(K(s_1), C3(P31(D(K(tgs),Y)), \\ & s_1, Ks(c_1,s_1))))))), \end{aligned} \tag{3.1}$$

$$\begin{aligned} OPT_{e_c,s_1}(Y,Z) \;==\; & Auth(P31(D(K(tgs),Y)), D(P33(D(K(tgs),Y)),Z), \\ & E(P33(D(K(tgs),Y)), C2(Ks(e_c,s_1), E(K(s_1), C3(P31(D(K(tgs),Y)), \\ & s_1, Ks(e_c,s_1))))))), \end{aligned} \tag{3.2}$$

$$\begin{aligned} OPT_{c_1,e_s}(Y,Z) \;==\; & Auth(P31(D(K(tgs),Y)), D(P33(D(K(tgs),Y)),Z), \\ & E(P33(D(K(tgs),Y)), C2(Ks(c_1,e_s), E(K(e_s), C3(P31(D(K(tgs),Y)), \\ & e_s, Ks(c_1,e_s))))))), \text{ and} \end{aligned} \tag{3.3}$$

$$\begin{aligned} OPT_{e_c,e_s}(Y,Z) \;==\; & Auth(P31(D(K(tgs),Y)), D(P33(D(K(tgs),Y)),Z), \\ & E(P33(D(K(tgs),Y)), C2(Ks(e_c,e_s), E(K(e_s), C3(P31(D(K(tgs),Y)), \\ & e_s, Ks(e_c,e_s))))))). \end{aligned} \tag{3.4}$$

Moreover, instead of using these four function symbols, introduce the new function symbols, OPT'_{c_1,s_1}, OPT'_{e_c,s_1}, OPT'_{c_1,e_s} and OPT'_{e_c,e_s} of arity 2, in the same manner as the way to construct the transformed description in Theorem 3.

After reconstructing the formal description in Section 3, the transformed formal description for this problem is as follows:

$$\begin{aligned} \boldsymbol{F} \;=\; & \{E, D, C2, C3, P21, P22, P31, P32, P33, Auth, K, \\ & OPT'_{c_1,s_1}, OPT'_{e_c,s_1}, OPT'_{c_1,e_s}, OPT'_{e_c,e_s}, tgs, c_1, s_1, e_c, e_s, Ks\}, \\ \boldsymbol{A} \;=\; & \{E(X, D(X,Y)) == Y, D(X, E(X,Y)) == Y, \\ & P21(C2(X,Y)) == X, P22(C2(X,Y)) == Y, \end{aligned}$$

$$\begin{aligned} & P31(C3(X,Y,Z)) == X, P32(C3(X,Y,Z)) == Y, \\ & P33(C3(X,Y,Z)) == Z, Auth(X,X,Y) == Y, \\ & \text{and the four axioms of } OPT'_{c_1,s_1}, OPT'_{e_c,s_1}, OPT'_{c_1,e_s} \text{ and } OPT'_{e_c,e_s} \\ & \text{obtained from Eq.(3.1)-(3.4)}\}, \\ \boldsymbol{I} \;=\; & \{tgs, c_1, s_1, e_c, e_s, K(e_c), K(e_s), \\ & E(K(c), C2(Ks(c,tgs), E(K(tgs), C3(c, tgs, Ks(c,tgs))))), \\ & E(K(e_c), C2(Ks(e_c,tgs), E(K(tgs), C3(e_c, tgs, Ks(e_c,tgs))))), \\ & E(K(tgs), C3(c, tgs, Ks(c,tgs))), E(Ks(c,tgs), c), \\ & E(Ks(c_1,tgs), C2(Ks(c_1,s_1), E(K(s_1), C3(c_1, s_1, Ks(c_1,s_1))))), \\ & E(Ks(e_c,tgs), C2(Ks(e_c,s_1), E(K(e_c), C3(e_c, s_1, Ks(e_c,s_1))))), \\ & E(K(s_1), C3(c_1, s_1, Ks(c_1,s_1))), E(Ks(c_1,s_1), c_1)\}, \\ \boldsymbol{O} \;=\; & \{E, D, C2, C3, P21, P22, P31, P32, P33, Auth, \\ & OPT'_{c_1,s_1}, OPT'_{e_c,s_1}, OPT'_{c_1,e_s}, OPT'_{e_c,e_s}\}, \\ \boldsymbol{G} \;=\; & \{X, Ks(c_1,s_1)\}. \end{aligned}$$

By using the system constructed in the previous chapter, we have decided the security and the answer is "Secure". It takes about 8.5 hours to decide the security by using DEC AlphaServer 2100 4/275 (202.9 SPECint 92). Consequently, we conclude that the enemy processes e_c and e_s cannot obtain $Ks(c_1,s_1)$.

3.5 Conclusions

In this chapter, we introduce some transformations of a formal description of a security problem and prove some theorems on the relation between the security of the original description and that of transformed one. By using the theorems, we present a useful approach to decide security problems which do not satisfy some of the conditions described in the previous chapter. This method can assure the security for the security problem in a class which properly includes the class the above algorithm can assure. For this reason, this method gives cryptographic protocol designers great helps.

For an application of this new method, we considered a basic security problem of Kerberos protocol, and have confirmed its security.

Chapter 4

Conclusions

In this dissertation, polynomial time verification methods for the security of cryptographic protocols were described.

In Chapter 2, the implementation and the evaluation of an algorithm proposed in [6, 7] to decide the security of cryptographic protocols was shown. For the security problems of cryptographic protocols which satisfy the sufficient condition[6, 7] under which the algorithm is applicable, the system could solve the security problems in a few minutes or less for simple protocols and in two days or less for comparatively complex protocols. By these result, it was confirmed that this system could decide the security in practical time. Moreover, this is the first system which can assure the security of cryptographic protocols and this is the first system which stops in polynomial time.

There are some cases that the security problems of cryptographic protocols proposed so far do not satisfy the sufficient condition. Many of those security problems do not satisfy one subcondition called right-linearity, but satisfy any other subconditions of the sufficient condition. In Chapter 3, we proposed a polynomial time verification method for such security problems. Not all the security of such problems can be decided by this new method, that is, this verification method has the cases that outputs "This system cannot decide the security" for such security problems but this method can always stop in polynomial time. In the security problems that newly can be assured the security by this new method, there exists a security problem for the network authentication protocol, Kerberos, which is used in the Internet. By this fact, it is shown that this new method is more useful than the decision algorithm proposed in [6, 7].

Consequently, we concluded that, by using the system and the verification method described in this dissertation, we can analyze the security of cryptographic protocols more formally.

The future research subjects are as follows: The form of axioms in $\boldsymbol{A}_1$ is very restricted, that is, only the form $f(x_1, \cdots, x_{a(f)})$ is allowed by the sufficient condition. One subject is, by relaxing the restriction of the form of left-hand side of axioms in $\boldsymbol{A}_1$, to get the new sufficient condition to make the problem decidable, and, by modifying the procedures related to $\boldsymbol{A}_1$ of the decision algorithm, to construct a new decision algorithm under the new sufficient condition. Another subject is to output how the

enemy get to their goal information concretely when the system or the verification method have decided that the security of a cryptographic protocol is insecure. This information will be a help to detect the fault of this insecure protocol.

Appendix A

Proofs of Theorems

Proof of Theorem 1

To prove Theorem 1, we show the following lemmas first:

Lemma 1: For two terms t_1 and t_2 in $T(\boldsymbol{F}_+)$ such that $t_1 \stackrel{\equiv}{A} t_2$, $\varphi^{-1}(t_1) \stackrel{\equiv}{A} \varphi^{-1}(t_2)$, where $\varphi^{-1}(t_+)$ is the term obtained from t_+ by replacing every a' in t_+ with a.

Proof. The set $\boldsymbol{A}$ of axioms in P_+ is the same as that in P, and no axiom in $\boldsymbol{A}$ includes a'. For this reason, whenever $t_1 \stackrel{\equiv}{A} t_2$ holds, $\varphi^{-1}(t_1) \stackrel{\equiv}{A} \varphi^{-1}(t_2)$ also holds. □

Lemma 2: For each $t_+ \in Q(\boldsymbol{A}, \boldsymbol{I}_+, \boldsymbol{O})$, there exists one term $t \in Q(\boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O})$ such that $\varphi^{-1}(t_+) = t$.

Proof. We prove this lemma by induction.

For a formal description $(\boldsymbol{F}', \boldsymbol{A}', \boldsymbol{I}', \boldsymbol{O}', \boldsymbol{G}')$, let $Q_i(\boldsymbol{A}', \boldsymbol{I}', \boldsymbol{O}')$ be the set of terms defined as follows:

1. $Q_0(\boldsymbol{A}', \boldsymbol{I}', \boldsymbol{O}') = \boldsymbol{I}'$,
2. $Q_{i+1} = Q_i \cup \{f(t_1, \cdots, t_{a(f)}) |$ for $f \in \boldsymbol{O}'$ and $t_i \in Q_i(\boldsymbol{A}', \boldsymbol{I}', \boldsymbol{O}')$ whose sort is S_i^f with $1 \leq i \leq a(f)\}$ $\cup \{t_2 |$ for $t_1 \in Q_i(\boldsymbol{A}', \boldsymbol{I}', \boldsymbol{O}')$ and $t_2 \in T(\boldsymbol{F}')$ such that $t_1 \stackrel{\equiv}{A'} t_2\}$.

We prove this lemma by showing that, for each $t_+ \in Q_i(\boldsymbol{A}, \boldsymbol{I}_+, \boldsymbol{O})$, there exists one term $t \in Q_i(\boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O})$ such that $\varphi^{-1}(t_+) = t$.

For each $t_+ \in Q_0(\boldsymbol{A}, \boldsymbol{I}_+, \boldsymbol{O})$, the statement holds obviously. Assume that the statement holds for $i = k$,

Case 1: Consider $f(t_1, \ldots, t_{a(f)}) \in Q_{k+1}(\boldsymbol{A}, \boldsymbol{I}_+, \boldsymbol{O})$ with $t_1, \ldots, t_{a(f)} \in Q_k(\boldsymbol{A}, \boldsymbol{I}_+, \boldsymbol{O})$. Since $\varphi^{-1}(f(t_1, \ldots, t_{a(f)})) = f(\varphi^{-1}(t_1), \ldots, \varphi^{-1}(t_{a(f)}))$ and each $\varphi^{-1}(t_j)$ with $1 \leq j \leq a(f)$ is in $Q_k(\boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O})$ by inductive hypothesis, we have that

$$f(\varphi^{-1}(t_1), \ldots, \varphi^{-1}(t_{a(f)})) \in Q_{k+1}(\boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O}).$$

Case 2: Consider $t_{2+} \in Q_{k+1}(\boldsymbol{A}, \boldsymbol{I}_+, \boldsymbol{O})$ such that there is a term $t_{1+} \in Q_k(\boldsymbol{A}, \boldsymbol{I}_+, \boldsymbol{O})$ with $t_{1+} \stackrel{\equiv}{A} t_{2+}$. There exists $t_1 \in Q_k(\boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O})$ such that $\varphi^{-1}(t_{1+}) = t_1$ by inductive hypothesis. By Lemma 1, $\varphi^{-1}(t_{2+}) \stackrel{\equiv}{A} t_1$ and $\varphi^{-1}(t_{2+}) \in Q_{k+1}(\boldsymbol{A}, \boldsymbol{I}, \boldsymbol{O})$ holds. □

Proof of Theorem 1. Since $Q(\boldsymbol{A},\boldsymbol{I},\boldsymbol{O}) \subset Q(\boldsymbol{A},\boldsymbol{I}_+,\boldsymbol{O})$ holds by the definitions of P and P_+, if P is not secure, then P_+ is not secure.

For g_1 and g_2 in $\boldsymbol{G}$ and a substitution $\sigma \triangleq \{X_1 \mapsto t_1, \ldots, X_h \mapsto t_h\}$, suppose that $\varphi(g_1)\sigma \overset{\equiv}{}_A \varphi(g_2)\sigma$ holds, where t_i is in $Q(\boldsymbol{A},\boldsymbol{I}_+,\boldsymbol{O})$ with $1 \leq i \leq h$.

The set of variables of a term t and that of $\varphi(t)$ are the same. By the definition of φ and φ^{-1},

$$\varphi^{-1}(\varphi(g_1)\sigma) = g_1\sigma_{\varphi^{-1}},$$

where $\sigma_{\varphi^{-1}}$ denotes the substitution $\{X_1 \mapsto \varphi^{-1}(t_1), \ldots, X_h \mapsto \varphi^{-1}(t_h)\}$,and

$$\varphi^{-1}(\varphi(g_2)\sigma) = g_2\sigma_{\varphi^{-1}}$$

holds. Since each t_i with $1 \leq i \leq h$ is in $Q(\boldsymbol{A},\boldsymbol{I}_+,\boldsymbol{O})$, by Lemma 2, each $\varphi^{-1}(t_i)$ is in $Q(\boldsymbol{A},\boldsymbol{I},\boldsymbol{O})$. By Lemma 1, we have that

$$\varphi^{-1}(\varphi(g_1)\sigma) \overset{\equiv}{}_A \varphi^{-1}(\varphi(g_2)\sigma).$$

Consequently, if P_+ is not secure, then P is not secure. □

Proof of Theorem 3

To prove Theorem 3, we show that the following lemmas hold:

Lemma 3: For two terms t_1 and t_2 in $T(\boldsymbol{F})$ such that $t_1 \overset{\equiv}{}_A t_2$, $\psi(t_1) \overset{\equiv}{}_{A'} \psi(t_2)$.

Proof. By the definitions of $\boldsymbol{A}$ and $\boldsymbol{A}'$, whenever an axiom A in $\boldsymbol{A}$ is applicable to a term $t \in T(\boldsymbol{F})$, the axiom A' in $\boldsymbol{A}'$ corresponding to A is applicable to the term $\psi(t)$. And for the resultant term t' of applying A to t, the resultant term of applying A' to $\psi(t)$ is $\psi(t')$. Hence, the statement holds. □

Lemma 4: For each $t \in Q(\boldsymbol{A},\boldsymbol{I},\boldsymbol{O})$, $\psi(t) \in Q(\boldsymbol{A}',\boldsymbol{I}',\boldsymbol{O}')$.

Proof. We prove the lemma by induction. Let Q_i be the set of terms defined in the proof of Lemma 2. We prove this lemma by showing that

$$\psi(t) \in Q_i(\boldsymbol{A}',\boldsymbol{I}',\boldsymbol{O}') \text{ for each } t \in Q_i(\boldsymbol{A},\boldsymbol{I},\boldsymbol{O}).$$

In two formal descriptions P and P', for $t \in Q_0(\boldsymbol{A},\boldsymbol{I},\boldsymbol{O})$, obviously $\psi(t) \in Q_0(\boldsymbol{A}',\boldsymbol{I}',\boldsymbol{O}')$(basis). Assume that the statement holds for k. Consider $f(t_1,\ldots,t_{a(f)}) \in Q_{k+1}(\boldsymbol{A},\boldsymbol{I},\boldsymbol{O})$ with $t_1,\ldots,t_{a(f)} \in Q_k(\boldsymbol{A},\boldsymbol{I},\boldsymbol{O})$.

Case 1: In the case that $f \notin \dot{\boldsymbol{O}}$, since $\psi(f(t_1,\ldots,t_{a(f)})) = f(\psi(t_1),\ldots,\psi(t_{a(f)}))$ and each $\psi(t_i) \in Q_k(\boldsymbol{A}',\boldsymbol{I}',\boldsymbol{O}')$ with $1 \leq i \leq a(f)$, we have that $\psi(f(t_1,\ldots,t_{a(f)})) \in Q_{k+1}(\boldsymbol{A}',\boldsymbol{I}',\boldsymbol{O}')$.

In the other case that $f \in \dot{\boldsymbol{O}}$, $\psi(f(t_1,\cdots,t_{a(f)}))$ is

$$f'(\underbrace{\psi(t_1),\ldots,\psi(t_1)}_{l_1},\ldots,\underbrace{\psi(t_{a(f)}),\ldots,\psi(t_{a(f)})}_{l_{a(f)}})$$

by the definition of f and f'. Since each $\psi(t_i) \in Q_k(\boldsymbol{A}',\boldsymbol{I}',\boldsymbol{O}')$ with $1 \leq i \leq a(f)$, all arguments of f' are in $Q_k(\boldsymbol{A}',\boldsymbol{I}',\boldsymbol{O}')$ and $f' \in \boldsymbol{O}'$,

$$f'(\underbrace{\psi(t_1),\ldots,\psi(t_1)}_{l_1},\ldots,\underbrace{\psi(t_{a(f)}),\ldots,\psi(t_{a(f)})}_{l_{a(f)}})$$

in $Q_{k+1}(\boldsymbol{A}',\boldsymbol{I}',\boldsymbol{O}')$. Consequently,

$$\psi(f(t_1,\ldots,t_{a(f)})) \in Q_{k+1}(\boldsymbol{A}',\boldsymbol{I}',\boldsymbol{O}').$$

Case 2: For $t_k \in Q_k(\boldsymbol{A},\boldsymbol{I},\boldsymbol{O})$ and $t \in T(\boldsymbol{F})$ such that $t_k \overset{\equiv}{}_A t$, $t \in Q_{k+1}(\boldsymbol{A},\boldsymbol{I},\boldsymbol{O})$ by assumption. By Lemma 3, $\psi(t_k) \overset{\equiv}{}_{A'} \psi(t)$ holds. Since $\psi(t) \in T(\boldsymbol{F}')$, $\psi(t) \in Q_{k+1}(\boldsymbol{A}',\boldsymbol{I}',\boldsymbol{O}')$. □

Proof of Theorem 3. We prove the contraposition. Like the proof of Theorem 1, for g_1 and g_2 in $\boldsymbol{G}$ and a substitution σ, suppose that

$$g_1\sigma \overset{\equiv}{}_A g_2\sigma.$$

By the definition of ψ,

$$\psi(g_1)\sigma = \psi(g_1)\sigma_\psi,$$

where σ_ψ denotes the substitution $\{X_1 \mapsto \psi(t_1), \ldots, X_h \mapsto \psi(t_h)\}$,and

$$\psi(g_2)\sigma = \psi(g_2)\sigma_\psi,$$

hold. By Lemma 4, each $\psi(t_i)$ with $1 \leq i \leq h$ are in $Q(\boldsymbol{A}',\boldsymbol{I}',\boldsymbol{O}')$. By Lemma 3,

$$\psi(g_1)\sigma_\psi \overset{\equiv}{}_{A'} \psi(g_2)\sigma_\psi$$

that is,

$$\psi(g_1)\sigma \overset{\equiv}{}_{A'} \psi(g_2)\sigma.$$

Consequently, if P is not secure, then P' is not secure. □

References

[1] M. Burrows, M. Abadi, and R. Needham, "A logic of authentication," ACM Trans. Computer Syst., vol.8, no.1, pp.18–36, Feb. 1990.

[2] D.E. Denning, "Cryptography and Data Security," Addison-Wesley, 1982.

[3] D. Dolev, S. Even, and R.M. Karp, "On the security of ping-pong protocols," Inf. & Control, vol.55, no.1–3, pp.57–68, Dec. 1982.

[4] T. Fujiwara, A. Kitai, K Taniguchi, and T. Kasami, "On decision problem of the security for cryptographic protocols," IECE Technical report, AL82–78, March 1984 (In Japanese).

[5] T. Fujiwara, T. Kasami, and S. Yamamura, "A formal verification for the security of a digital signature protocol," IECE Technical report, AL81–90, Jan. 1982 (In Japanese).

[6] T. Fujiwara, K. Taniguchi, and T. Kasami, "Decision problem of the security for cryptographic protocols," IECE Trans. vol.J69–D, no.6, pp.984–992, June 1986 (In Japanese).

[7] T. Fujiwara, K. Taniguchi, and T. Kasami, "Decision problem of the security for cryptographic protocols," Discrete Algorithms and Complexity, pp.263–286, Academic Press, 1987.

[8] DOV M. Gabbay, C.J. Hogger, and J.A. Robinson, "Handbook of Logic in Artificial Intelligence and Logic Programming," vol.1, Oxford Science Publications, 1993.

[9] G. Huet, "Confluent reductions: abstract properties and applications to term rewriting systems," Proc. IEEE 18th Ann. Symp. on Foundation of Computer Science, pp.30–45, 1977.

[10] K. Koyama, "Fast and Secure Digital Signature Using Public-Key Cryptosystems," IECE Trans. vol.J67–D, no.3, pp.305–312, March 1984 (In Japanese).

[11] T. Kasami, S. Yamamura, and K. Mori, "A key management scheme for end-to-end encryption and a formal verification of its security," IECE Trans. vol.J65–D, no.6, pp.695–702, June 1982 (In Japanese).

[12] R. Kemmerer, C.A. Meadows, and J.K. Millen, "Three system for cryptographic protocol analysis," J. Cryptology, vol.7, no.2, pp.79-130, July 1994.

[13] J. Kohl and C. Neuman, "The Kerberos network authentication service (V5)," RFC 1510, Sep. 1993.

[14] C.A. Meadows, "Applying Formal Methods to the Analysis of a Key Management Protocol," J. of Computer Security, vol.1, no.1, pp.5–35, IOS Press Jan. 1992.

[15] C.A. Meadows, "Formal verification of cryptographic protocols: a survey," Pre-Proc. ASIACRYPTO'94, pp.117–130, Nov. 1994.

[16] J.K. Millen, S.C. Clark, and S.B. Freedman, "The Interrogator: protocol security analysis," IEEE Trans. Software Eng., vol.13, no.2, pp.274–288, Feb. 1987.

[17] M. Sumida, T. Takata, T. Fujiwara. and T. Kasami, "A Protocol for Bidding," Proc. SCIS91, 12C, Jan. 1991 (In Japanese).

[18] S. Susaki, T. Matsumoto and H. Imai, "A Note on Untraceable Communication Network," Proc. SCIS91, 12B, Jan. 1991 (In Japanese).

[19] S. Tsujii, S. Aoki, M. Kanda, M. Manbo, T. Ikeda and J. Chao, "A study on information security system with AI Approach," Proc. SCIS91, 19A, Jan. 1991 (In Japanese).

[20] K. Takaragi, T. Shiraishi, and R. Sasaki, "Authentication Method for Electronic Contracts with IC Card Key Management," Trans. IEE Japan vol.107-C, no.1, pp.46–53, Jan. 1987 (In Japanese).

[21] H. Watanabe, T. Fujiwara, and T. Kasami, "A system for deciding the security of cryptographic protocols," IEICE Trans. Fundamentals, vol.E76–A, no.1, pp.96–103, Jan. 1993.