Journal of Information Security and Applications 36 (2017) 101–111

Contents lists available at ScienceDirect

Journal of Information Security and Applications

journal homepage: www.elsevier.com/locate/jisa

Model checking the *i*KP electronic payment protocols☆

Kazuhiro Ogata

Japan Advanced Institute of Science and Technology (JAIST), 1-1 Asahidai, Nomi, Ishikawa 923–1292, Japan

ARTICLE INFO

Article history:

MSC:
00-01
99-00

Keywords:
Electronic commerce
Maude
Model checking
Observational transition systems (OTSs)
Rewriting

ABSTRACT

Much progress of formal verification techniques has been made and therefore many formal verification case studies of systems, such as security protocols, have been conducted and reported. Two major formal verification techniques are model checking and interactive theorem proving. Almost all case studies, however, use either model checking or interactive theorem proving. Even though both techniques are used, two different specifications dedicated to model checking and interactive theorem proving, respectively, are used. It would be desirable to make it possible to use one specification for one system for both model checking and interactive theorem proving. Observational transition systems (OTSs) have been proposed so that they can be written as equational theory specifications and used for interactive theorem proving. OTSs can also be written as rewrite theory specifications, which can be used for model checking, but it would take time to model check OTSs written as rewrite theory specifications in the existing techniques.

There are two main contributions described in this article: (1) to propose an effective way to write OTSs as rewrite theory specifications, and (2) to conduct a case study in which an electronic commerce protocol has been model checked. Contribution (1) can be regarded as an effective way to translate equational theory specifications of OTSs for interactive theorem proving into rewrite theory specifications of OTSs for model checking. Moreover, since rewrite theory specifications of OTSs can be used for interactive theorem proving as well, contribution (1) may lead to the desirable situation aforementioned.

1. Introduction

Much progress of formal verification techniques has been made. The major formal verification techniques are model checking and theorem proving. Many model checkers and interactive theorem provers have been developed. Among model checkers are a New Symbolic Model Verifier (NuSMV) [1], Spin [2], Symbolic Analysis Laboratory (SAL) [3], Process Analysis Tool (PAT) [4] and Alloy [5], while among interactive theorem provers are Prototype Verification System (PVS) [6], Larch Prover [7], Higher-Order Logic (HOL) [8], Isabelle/HOL [9] and Coq [10]. Many case studies have been conducted, in which systems, such as security protocols, have been formally verified, most of which use either model checking or interactive theorem proving. Even if both model checking and interactive theorem proving is used, two different specifications dedicated to model checking and interactive theorem proving, respectively, are used [11,12]. It would be desirable to make it possible to use one specification for one system for both model checking and interactive theorem proving.

*i*KP (*i*-Key-Protocol, $i = 1, 2, 3$) [13,14] is a family of electronic payment protocols (which are security protocols) and has affected the design of the Secure Electronic Transactions (SET) protocol[1] [15]. Sellers, buyers and acquirers (credit card processing banks) participate in the protocols. While Ogata, et al. were trying to verify that 2KP and 3KP enjoy a security property with an interactive theorem prover, they happened to find counterexamples showing that 2KP and 3KP do not enjoy the property [16,17]. The property is called the payment agreement property, which is that whenever an acquirer authorizes a payment, both the buyer and seller concerned agree on it. Note that 1KP does not enjoy the property by definition. Although the experience shows that interactive theorem proving could be used to find counterexamples, the way to find the counterexamples is ad hoc as well as time-consuming. It took a couple of weeks to find the counterexamples.

We have then conducted a case study so as to confirm that the counterexamples can be systematically found by a model checker and model checking can help systems verification with interactive

☆ This work was partially supported by JSPS Kakenhi Grant Numbers 26540024 and 26240008.

E-mail address: ogata@jaist.ac.jp

[1] The SET specification books were available at http://www.setco.org/set_specifications.html but the website disappeared. The books, however, are currently available at some websites, such as https://en.wikipedia.org/wiki/Secure_Electronic_Transaction.

http://dx.doi.org/10.1016/j.jisa.2017.08.006
2214-2126/

theorem proving. For these purposes, we used essentially the same way to model 2KP and 3KP as that used for interactive theorem proving [16]. If one model of one system can be used for both interactive theorem proving and model checking, we can conveniently use model checking as follows: (1) to confirm that there exists a counterexample showing that the system does not enjoy a property before we try to verify that the system enjoys the property with interactive theorem proving and (2) to filter out inappropriate lemmas from conjectured ones used for interactive theorem proving. The reason why we still would like to use interactive theorem proving is because we often need to use interactive theorem proving to verify that systems whose (reachable) state spaces are unbounded enjoy properties.

We used Maude [18] as a model checker. Maude is a rewriting logic-based specification/programming language, one direct successor of OBJ3 [19], the most famous algebraic specification language mainly designed by Goguen. Rewriting logic is a logic that can be used for formally specifying and analyzing concurrent and distributed systems. It has as a sub-logic membership equational logic, which is a variant of order-sorted algebras that are the main logic of OBJ3 and then makes it possible to use flexible data structures, such as associative and/or commutative binary constructors. One characteristic of Maude is to allow inductively defined data structures to be used in specifications that are model checked and to be able to deal with unbounded-state (even unbounded-reachable-state) systems as model checking targets.

We initially expected that the state explosion would prevent Maude from finding the counterexamples because *i*KP is more complex than typical authentication protocols, such as Needham–Schroeder Public-Key (NSPK) authentication protocol [20]. On the contrary, Maude quickly found the counterexamples because the counterexamples are at a shallow position (at depth 4) from a given initial state, which is an example of "Small World Hypothesis" [5]. Since the models used of 2KP and 3KP have unbounded number of reachable states, however, most existing model checkers may not even accept the models.

There are several reasons for which we used Maude. Among them are as follows: (1) Maude can accept state machines that have unbounded number of (even reachable) states, (2) inductive data types can be freely used, (3) message exchanges can be naturally described as rewrite theory specifications in Maude, and (4) the execution speed is reasonably fast (comparable to SPIN) [21]. These characteristics, especially (1), make it possible to find the counterexamples. The case study indicates that even when a system to be model checked is large, Maude can find a counterexample showing that the system does not enjoy a property if the counterexample is at a shallow position from a given initial state.

In the case study, 2KP and 3KP were modeled as observational transition systems (OTSs) [22,23]. OTSs are mathematical models of systems and have been initially introduced for interactive theorem proving. In the case study, however, we came up with an effective way to write OTSs as rewrite theory specifications in Maude. One lesson we learned from the case study is that our way to model security protocols for interactive theorem proving, together with Maude, can also be effectively used for model checking. Therefore, we do not need to make different models for different purposes, namely interactive theorem proving and model checking. Moreover, since rewrite theory specifications of OTSs can be used for interactive theorem proving [24], the proposed technique in the present article may lead to the desirable situation that one specification (not only one model) for one system can be used for both model checking and interactive theorem proving, which does not require to translate one specification into some other for one system.

Although a way to specify and model checking authentication protocols in Maude is described in [25], it is worth reporting on the case study for several reasons. Among them are as follows. (1) Our way to model protocols is different from that described in [25]. The way described in [25] is tailored for model checking. (2) Security properties treated in [25] are only secrecy properties, which are different from the payment agreement property. (3) We modeled, specified and analyzed *i*KP, which is more complex than NSPK analyzed in [25].

Initiate	$B \to S:\ \mathrm{ID}_B$
Invoice	$S \to B:\ \mathrm{Clear}, [_{2,3}\mathrm{Sig}_S]$
Payment	$B \to S:\ \mathrm{EncSlip}, [_3\mathrm{Sig}_B]$
Auth-Request	$S \to A:\ \mathrm{Clear}, \mathrm{EncSlip}, [_{2,3}\mathrm{Sig}_S], [_3\mathrm{Sig}_B]$
Auth-Response	$A \to S:\ \mathrm{RESPCODE}, \mathrm{Sig}_A$
Confirm	$S \to B:\ \mathrm{RESPCODE}, \mathrm{Sig}_A$

Fig. 1. The *i*KP protocols.

The rest of the article is organized as follows. We describe *i*KP and OTSs in Sections 2 and 3, respectively. Section 4 introduces part of Maude that is closely related to this article. We propose an effective way to write OTSs as rewrite theory specifications in Maude and describe how to model check that OTSs enjoy invariant properties with Maude in Section 5. We report on the case study in Sections 6 and 7. We mention some related work in Section 8 and finally conclude the article in Section 9.

2. The *i*KP payment protocols

Fig. 1 shows the three *i*KP protocols from which quantities that are irrelevant to the payment agreement property are hidden. The protocols are the same as those described in [17]. Parts enclosed by $[_{2,3}\ldots]$ and $[_3\ldots]$ are ignored for 1KP and 1KP/2KP, respectively. The main difference between 1, 2 and 3KP is the increasing use of digital signatures as more of the parties involved possess a private/public key-pair.

B, *S* and *A* stand for a buyer, a seller and an acquirer, respectively. An acquirer is a credit card processing bank and is also called an acquiring bank. Each acquirer *A* has a private key K_A that enables signing and decryption. In this article, for brevity, we assume that its public counterpart K_A^{-1} that enables signature verification and encryption is securely conveyed to every buyer and seller participating in the protocols. Each seller *S* in 2KP/3KP and each buyer *B* in 3KP has a private/public key-pair $(\mathrm{K}_S, \mathrm{K}_S^{-1})$ and $(\mathrm{K}_B, \mathrm{K}_B^{-1})$, respectively. We also assume that each seller's public key is securely conveyed to every acquirer and buyer in 2KP/3KP, and that each buyer's public key is securely conveyed to every acquirer and seller in 3KP.

Cryptographic primitives used in *i*KP are as follows:

- $\mathcal{H}(\cdot)$: A one-way hash function.
- $\mathcal{H}_k(K, \cdot)$: A keyed one-way hash function; the first argument *K* is the key.
- $\mathcal{E}_X(\cdot)$: A public-key encryption function with K_X^{-1}.
- $\mathcal{S}_X(\cdot)$: A public-key signing function with K_X.

Basic values occurring in *i*KP are as follows:

- PRICE: Amount and currency.
- NONCE_S: Seller's nonce (random number) used for payment replay protection.
- ID_S: Seller ID.
- R_B: Random number chosen by *B* to form ID_B.
- BAN: Buyer's Account Number, such as a credit card number.
- RESPCODE: Response from the clearing network: YES/NO or authorization code.

Composite values used in *i*KP are as follows:

- ID_B: A buyer pseudo-ID $\mathcal{H}_k(\mathrm{R}_B, \mathrm{BAN})$.