Automatic Verification of the STS Authentication Protocol with Crème

Masahiro Nakano[1], Kazuhiro Ogata[2,1], Masaki Nakamura[1] and Kokichi Futatsugi[1]
[1]Graduate School of Information Science
Japan Advanced Institute of Science and Technology (JAIST)
[2]NEC Software Hokuriku, Ltd.

Abstract: We describe a case study in which the Crème verification tool has been used to verify automatically that the STS authentication protocol has two desired invariant properties.

1. Introduction

Crème[4] is an automatic invariant verification tool for a class of behavioral specifications, i.e. algebraic specifications of transition systems (OTSs[5] are used as transition systems), which are written in equations. The method used in Crème is fixpoint computation of a monotone function made from preconditions of predicates. The method is one of the standard techniques for automatic (invariant) verification, but Crème has the characteristic such that term rewriting is mainly used as the basic mechanism.

The STS protocol[2] is an authentication protocol with public-key cryptography and uses the Diffie-Hellman (DH) key-exchange method. Nobody doubts that security protocols are a key to success of safe communication on the Internet. A subject of developing them is subtle errors that are extremely difficult to reveal by test, which implies that they are worth analyzing formally. We report on a case study in which Crème can verify automatically that the protocol has two desired invariant properties: 1) the intruder cannot glean the symmetric key K_{pq} exchanged by two principals p and q unless p or q is the intruder, and 2) K_{pq} can be properly exchanged by p and q after the three message exchanges. Properties 1) and 2) are called the key-secrecy property and the key-exchange property respectively.

2. Crème

Crème[4] is an automatic theorem prover for transition systems, which can be infinite states. The current implementation of Crème can prove that such transition systems, which are specified in an algebraic specification language, have invariant properties. A precondition of a (state) predicate P with respect to a transition system is a condition such that an arbitrary transition changes an arbitrary state satisfying the condition to a state satisfying P. Crème is based on fixpoint computation of a monotone function that computes preconditions of predicates. Such method is also used in existing automatic theorem provers such as STeP[3]. But, we often cannot obtain a fixpoint of such a monotone function in only a straightforward way. Therefore, we have devised a method to find invariant candidates, which are used as lemmas, so that a fixpoint can be effectively computed. We call the method *Generalization*. Given a predicate P, Generalization generates a set Φ of predicates which imply P, and tries to prove that each predicate in Φ is invariant. If one of them is proved, it is used instead of P for fixpoint computation. Since Generalization can be implemented with Associative-Commutative rewriting (AC rewriting) and can be used for specifications including user-defined data types, it is suited to theorem provers for algebraic specifications. Given an algebraic specification of a transition system and a predicate, the current implementation of Crème tries to automatically prove the predicate to be invariant to the transition system. It draws a graph such that each node denotes a predicate (which is the given predicate or a generated one on the fly) and each edge denotes a relation between two predicates (e,g, one is used as a lemma to prove the other). Such a graph can help human users understand the proof of the given predicate and/or find out a reason why the given predicate cannot be proved.

3. The STS Protocol

The protocol can be written:

$$\begin{array}{ll}\mathrm{Msg1} & p \to q:\ \mathrm{PA}, \mathrm{EX}_p \\ \mathrm{Msg2} & q \to p:\ \mathrm{EX}_q, \mathrm{Cert}_q, \{\{\mathrm{EX}_p, \mathrm{EX}_q\}_{\mathrm{S}_q}\}_{\mathrm{K}_{pq}} \\ \mathrm{Msg3} & p \to q:\ \mathrm{Cert}_p, \{\{\mathrm{EX}_p, \mathrm{EX}_q\}_{\mathrm{S}_p}\}_{\mathrm{K}_{pq}}\end{array}$$

PA is a DH parameter, a pair of a large prime number n and a natural number g such that $g < n$. Given a DH parameter PA, EX_x is $g^{r_x} \bmod n$ called a component, where x is p or q, r_x is a random number chosen by a principal x, and K_{pq} is $g^{r_p r_q} \bmod n$. K_{pq} is used as a symmetric key for p and q to communicate securely with each other and the purpose of the protocol is for p and q to exchange K_{pq} securely. S_x and P_x are the private and public keys given to x. Given a message M and a (public or symmetric) key K, $\{M\}_K$ is the message encrypted with K; if K is a private key, $\{M\}_K$ is a signed message. We suppose that there exists a trustable certificate authority T; T issues Cert_x for every principal x and a DH parameter PA, where Cert_x is $\{x, \mathrm{P}_x, \mathrm{PA}\}_{\mathrm{S_T}}$. We take into account two more message exchanges as

$$\mathrm{Msg4}\quad p \to q:\ \{secret\}_{\mathrm{K}_{pq}} \qquad \mathrm{Msg5}\quad q \to p:\ \{secret\}_{\mathrm{K}_{pq}}$$

These two message exchanges correspond to that p and q communicate securely with each other using K_{pq} after K_{pq} has been exchanged, where *secret* denotes the contents of the communication. We suppose that Msg4's and Msg5's are only sent by trustable principals. Cipher texts in Msg2's and Msg3's are called ciphers and cipher texts in Msg4's and Msg5's encrypted communications.

4. Verification of the STS Protocol

4.1 Modeling

The way of modeling the protocol is the same as that described in [5]. When modeling the protocol, we take into account the (most general) intruder. In addition to the actions specified by the protocol, the intruder gleans as many data values as possible and fakes messages based on them, provided that the cryptosystem used is unbreakable. The intruder gleans from the network three kinds of data values: components, signatures and ciphers. Three operators *cco*, *csi* and *cci* are used to denote collections of those values. Given a network *nw* (i.e. a collection of messages sent), $cco(nw)$, $csi(nw)$ and $cci(nw)$ are the collections of components, signatures and ciphers gleaned by the intruder from the network. An operator *ccm* is also used to denote a collection of encrypted communications; given a network *nw*, $ccm(nw)$ is the collection of encrypted communications that

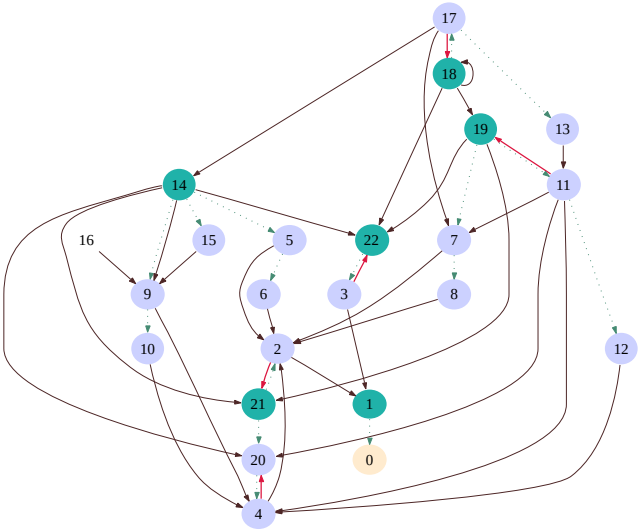

Figure 1. The proof structure of the key-secrecy property

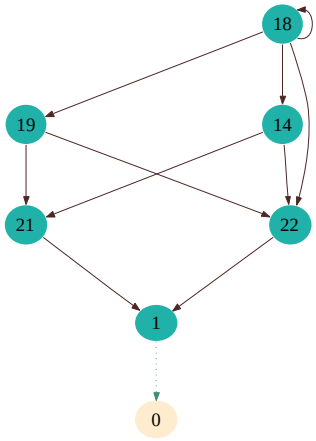

Figure 2. The minimized proof structure of the key-secrecy property

has been sent by trustable principals and that can be decrypted by the intruder.

Let *Network* be a type of networks (i.e. collections of messages) and void be the empty network. The protocol is modeled as the OTS $\mathcal{S}_{\mathrm{STS}}$ such that 1) $\mathcal{O}_{\mathrm{STS}}$ consists of $nw : \Upsilon \rightarrow \mathit{Network}$, 2) $\mathcal{I}_{\mathrm{STS}}$ is $\{\upsilon \in \Upsilon \mid nw(\upsilon) = \mathrm{void}\}$, and 3) $\mathcal{T}_{\mathrm{STS}}$ consists of 13 parameterized transitions; five of them formalize sending messages exactly following the protocol and the remaining the intruder's faking messages: two transitions for faking Msg1's, four ones for faking Msg2's and two ones for faking Msg3's. Given a state υ, $nw(\upsilon)$ is the collection of messages sent until the state.

Let P, M, Crt, Cmp and Sig be types of principals, messages, certificates, components and signatures. In this paper we describe the transition for sending Msg3's exactly following the protocol and a transition for faking Msg3's. The former is $\mathit{mes3}_{p,q,m1,m2} : \Upsilon \rightarrow \Upsilon$ and the latter $\mathit{fkm31}_{p,crt,cmp,sig} : \Upsilon \rightarrow \Upsilon$ for $p, q \in P$, $m1, m2 \in M$, $\mathit{crt} \in \mathit{Crt}$, $\mathit{cmp} \in \mathit{Cmp}$ and $\mathit{sig} \in \mathit{Sig}$. The effective condition of $\mathit{mes3}_{p,q,m1,m2}$ is that p has sent q a valid Msg1 $m1$ and $m2$ is a valid Msg2 that seems to have been sent by q in reply to $m1$; if the condition holds in a state υ, a valid Msg3 is put into the network $nw(\upsilon)$ in reply to $m2$. The effective condition of $\mathit{fkm31}_{p,crt,cmp,sig}$ is that a component cmp and a signature sig are available to the intruder; we suppose that every certificate is freely available to the intruder; if the condition holds in a state υ, an Msg3 (whose destination is p) faked from crt, cmp and sig is put into the network $nw(\upsilon)$.

$\mathcal{S}_{\mathrm{STS}}$ is written in Maude, an algebraic specification/programming language.

4.2 Verification

In this paper, we describe the verification that the protocol has the key-secrecy property. The property can be expressed as $EC \in \mathit{ccm}(nw(S)) \Rightarrow \mathit{for}(EC) = \mathit{intruder}$, where $\mathit{for}(ec)$ denotes the principal for which an encrypted communication ec is created. Let P_0 be this formula. All we have to do is to prove P_0 invariant to $\mathcal{S}_{\mathrm{STS}}$.

Given the specification of $\mathcal{S}_{\mathrm{STS}}$ and P_0, Crème tries to prove P_0 invariant to $\mathcal{S}_{\mathrm{STS}}$ by discovering a set of predicates, which imply P_0 and are a fixpoint of a monotone function F_{STS} that computes preconditions of predicates with respect to $\mathcal{S}_{\mathrm{STS}}$. The proof is successfully done and the graph shown in Figure 2. The six dark nodes that are discovered by Crème are a fixpoint of F_{STS}. In addition to the six predicates, Crème discovers other predicates in the course of the proof. Figure 1 shows the graph in which all the predicates (21 in all) discovered are included. Except for one predicate P_x, all the predicates that disappear in Figure 2 are logically implied by one of the six predicates shown in Figure 2. P_x was introduced as part of a precondition of P_y during the proof, but since Crème found P_x unnecessary for the proof, P_x was deleted.

Crème can also prove the key-exchange property in the similar way. Crème gets 62 additional lemmas, and 8 of them are needed to prove finally.

5. Conclusion

Crème can fully automatically prove the key-secrecy property and the key-exchange property of the STS protocol as well as the NSLPK protocol[4]. [1] proposes a specialized method of proving secrecy properties of authentication protocols automatically. But, Crème can be used to prove other properties such as the the key-exchange property of the STS protocol and the agreement property of the NSLPK protocol. Besides, our way of modeling and verification is not specialized for authentication protocols but can be used for general distributed systems. Nevertheless, Crème can prove the both properties automatically.

References

[1] V. Cortier, J. Millen, and H. Rueß. Proving secrecy is easy enough. In *14th IEEE Computer Security Foundations Workshop*. IEEE Computer Society, 2001.

[2] W. Diffie, P. C. van Oorschot, and M. J. Wiener. Authentication and authenticated key exchanges. *Designs, Codes and Cryptography*, 2:107–125, 1992.

[3] Z. Manna, N. Bjorner, A. Browne, E. Y. Chang, M. Colon, L. de Alfaro, H. Devarajan, A. Kapur, J. Lee, H. Sipma, and T. E. Uribe. Step: The stanford temporal prover. In *TAPSOFT*, pages 793–794, 1995.

[4] M. Nakano, K. Ogata, M. Nakamura, and K. Futatsugi. Automating invariant verification of behavioral specifications. submitted for publication, 2005.

[5] K. Ogata and K. Futatsugi. Proof scores in the OTS/CafeOBJ method. In *6th FMOODS*, volume 2884 of *LNCS*, pages 170–184. Springer, 2003.