MODELLING REAL TIME AUTHENTICATION PROTOCOLS USING ALGEBRAIC SPECIFICATION TECHNIQUES – THE CASE OF TESLA PROTOCOL

Iakovos Ouranos[1] Petros Stefaneas[2] Kostas Barlas[2] Stefanos Demertzis[2]
George Koletsos[1] Panayiotis Frangos[1]

[(1)]School of Electrical & Computer Engineering
[(2)]School of Applied Mathematical & Physical Sciences,
National Technical University of Athens, Greece
iouranos @central.ntua.gr

Keywords: Algebraic Specification and Verification, CafeOBJ, Authentication, TESLA protocol

1. INTRODUCTION

We present how to model real time authentication protocols with CafeOBJ algebraic specification language [1]. The protocols are specified as Timed Observational Transition Systems or TOTSs [2]. Based on this specification, we can verify desirable properties of the protocols through induction and/or case analysis, thanks to the CafeOBJ system and its support to the interactive theorem proving. The method we apply is a combination of [2] and [3]. As a case study we model TESLA protocol [4], which is used for the source authentication in multicast communication settings, and verify a safety property. The paper is organized as follows. Section 2 describes informally the TESLA protocol while section 3 presents how to model it with CafeOBJ. Section 4 concludes the paper.

2. THE TESLA PROTOCOL

The Timed Efficient Stream Loss-tolerant Authentication (TESLA) broadcast authentication protocol is distinguished from other types of cryptographic protocols in both its key management scheme and its use of timing. Basic TESLA that is the simplest version of the protocol informally works as follows. An initial authentication is achieved using a public key signature. The subsequent messages are authenticated using Message Authentication Codes (MACs) linked back to the initial signature. In message *n-1*, the sender *S* generates a key k_n , and transmits $f(k_n)$, where *f* is a suitable cryptographic hash function, to the receivers *R*, as a commitment to that key. In message *n*, *S* sends a data packet m_n, authenticated using a MAC with key k_n. The key itself is revealed in message *n+1*. Each receiver checks that the received k_n corresponds to the commitment received in message *n-1*, verifies the MAC in message *n*, and then accepts the data packet m_n as authentic. Message *n* also contains a commitment to the next key k_{n+1}, authenticated by the MAC, thus allowing a chain of authentications [5]. The messages exchanged are as follows:

Init Message: $R \rightarrow S$: n_R
Reply Message $S \rightarrow R$: $\{f(k_1), n_R\}_{SK(S)}$
Msg_1: $S \rightarrow R$: $d_1, f(k_2)$, MAC($k_1, d_1, f(k_2)$)
Msg_n: $S \rightarrow R$: $d_n, f(k_{n+1}), k_{n-1}$, MAC($k_n, d_n, f(k_{n+1}), k_{n-1}$)

where n_R is a nonce generated by the receiver to ensure freshness and d_1, d_n the data transmitted. The protocol requires an important time synchronization assumption, the *security condition*: the receiver will not accept message *n* if it arrives after the sender might have sent message *n+1*, otherwise an intruder can capture message *n+1*, and use the key k_n from within it to fake a message *n*. Thus the agents' clocks need to be loosely synchronized.

3. MODELLING TESLA USING CAFEOBJ

We have modeled TESLA as a Timed Observational Transition System in CafeOBJ. The specification consists of sorts (or types), operators on the sorts, and equations that define operators. The specification is executable. The visible sorts corresponding to the basic data types and the related operators are as follows:

- sort **Agent** denotes agents; constant **enemy** denotes the enemy,
- sort **SKey** denote the secret key used for the encryption of the initial packet,
- sort **Key** denotes the key used for MACs,
- sort **Prf** denotes the pseudorandom function *f*; given a key *k*, *f(k)* returns the commitment for the key, while operator *k* returns the argument of *f(k)*.
- sort **Nonce** denotes nonces. Given agents p1, p2 and random number r, n(p1,p2,r) denotes a nonce generated by agent p1 to authenticate agent p2, where r makes the nonce globally unique and unguessable.
- sort **Cipher** denotes the ciphertexts encrypted with sender's private key.
- sorts **Mac1**, **Mac2** denote the message authentication codes of messages *m1* and *mn* correspondingly.

The four operators to denote the four kinds of messages are **im**, **rm**, **m1** and **mn** which are declared as:

op im : Agent Agent Agent Nonce → Msg
op rm : Agent Agent Agent Cipher → Msg
op m1 : Agent Agent Agent Prf Mac1 → Msg
op mn : Agent Agent Agent Prf Key Mac2 Nat → Msg

where **Msg** denotes messages. We mention the indexing of each message **mn**. The network is modeled as a multiset of messages, which is used as the storage that the intruder can use. The enemy tries to glean seven kinds of values from the network, which are *Nonces*, *Ciphertexts*, *Pseudorandom function values*, *Message Authentication Codes* of two kinds and *Keys*.

The state space of the protocol is declared as the hidden sort **Tesla**. The specification consists of seven observations and twelve parameterized transitions. The four transitions formalize sending messages exactly following the protocol and the remaining the enemy's faking messages. In addition there is a time advancing transition rule *tick* that advances the master clock. Each transition is executed between a lower and an upper bound. We assume that the sender sends messages at discrete time units, while an enemy can send messages whenever he wants.

Based on the specification we have proved the following invariant property

At any reachable state, if a key can be obtained by the enemy, then either the key belongs to the enemy or it has been revealed as part of a message.

The above property is denoted by operator inv1.

inv1(T, K) =
K \in keys(nw(T)) implies
(p(K) = enemy) or (s s i(K)) * d1 <= now(T) .

where **T** denotes any reachable state of the protocol, **K** the key, **i(K)** is the index of the key, $\mathbf{d_1}$ is the time delay of sending action, **s** denotes the successor index, **\in** is the membership operator and **now(T)** denotes the current time.

The proof is done by induction on the number of transitions applied. The methodology [2] includes case analysis and appropriate lemma discovery.

We have used two additional lemmas to prove the above property.

4. CONCLUSIONS

We have formally specified and verified TESLA protocol using the TOTS/CafeOBJ method, to show its application to the modeling of real time authentication protocols. The protocol has also been modeled and verified in [5], [6], and [7]. CafeOBJ provides a flexible human computer interaction mechanism in good balance such that humans make proof plans and machines conduct tedious and detailed computations, and proof scores have flexible structure.

REFERENCES

[1] Diaconescu, R. and Futatsugi, K. (1998): CafeOBJ Report. AMAST Series in Computing, 6. World Scientific.

[2] Ogata, K. and Futatsugi, K. (2007):Modeling and Verification of Real-Time Systems Based on Equations. Science of Computer Programming, doi:10.1016/j.scico.2006.10.011.

[3] Ogata, K. and Futatsugi, K. (2006): Some Tips on Writing Proof Scores in the OTS/CafeOBJ Method. In K. Futatsugi et al. (Eds.).Goguen Festschrift, vol. LNCS 4060, 596-615.

[4] Perrig, A., Canetti, R., Tygar, J.D., and Song, D.(2000): Efficient Authentication and Signing of Multicast Streams over Lossy Channels. In Proc. of IEEE Security and Privacy Symposium, 56-73.

[5] Broadfoot, P. and Lowe, G. (2002): Analysing a Stream Authentication Protocol using Model Checking. In Proc. of ESORICS'02, vol. LNCS 2502, 146-161.

[6] Archer, M. (2002): Proving Correctness of the Basic TESLA Multicast Stream Authentication Protocol with TAME. In Proc. of WITS 2002.

[7] Lomuscio, A., Raimondi, F., and Wozna, B.(2006): Verification of the TESLA protocol with MCMAS-X. In Proceedings of CS&P, International Workshop on Concurrency, Specification and Programming.