Analysis of an E-voting Protocol using the Inductive Method

Najmeh Miramirkhani[1], Hamid Reza Mahrooghi[1], Rasool Jalili[1]

[1] Sharif University of Technology,Tehran, Iran
{miramirkhani@ce., mahrooghi@ce., jalili@ }sharif.edu

Abstract. E-voting protocols require providing several security properties and it is well known that only formal methods can provide a proof that a given protocol meets its requirements. These methods aim at presenting a mathematical proof, to verify if a protocol satisfies desired properties. The Inductive Method, which is based on theorem proving, is used to analyze various classical and real-world applied security protocols. Unlike the model checking approach which typically has some restrictions on the size of the system, theorem proving attempts to prove and guarantee properties of a protocol when an unlimited number of agents are engaged in the protocol execution. In this paper, we present a model for a well-known e-voting protocol, FOO'92, using the Inductive Method. This is somehow a complex task, as typical e-voting protocols may employ some advanced cryptographic primitives such as blind signature and commitment as well as broadcast communications. Along with modeling of these two primitives, we extend the Inductive Method to support broadcast communications and finally the protocol is modeled based on these extensions.

Keywords: Formal Verification, Security Property, Theorem Proving, Inductive Method, E-voting Protocols.

1 Introduction

It appeared that a large portion of the protocols described in academic literature is subject to attacks [2]. Formal analysis is one of the approaches that help to detect and correct found vulnerabilities and It can be carried out using model checking or theorem proving.
Model checking is not limited to finite transition systems, but since it suffers from the problem of state explosion, there is usually some constraints on the size of the under-study system. Although these restrictions lead to analyze a protocol under a limited number of agents (three or four agents), but verification algorithms are usually automatic and always terminate. Verification methods based on model-checking are more suitable for finding attacks than proving security properties, because based on a bounded number of agents we have no guarantee that the protocol will be secure under more agents. Theorem proving approach aims to establish that a security property is a logical consequence of the premises which describe the system. It has not the limitations of the model checking approach, but because of the undecidability problem, the theorem proving tools are not usually automatic and require user interactions.
Analysis of e-voting protocols have been carried out using various approaches [5], [6]. Model-checking is one of the most well-known and conventional formal approach. For example, formal analysis of FOO'92 using applied-pi calculus [6] , LYSA-calculus [7], ACP process

algebra [3] and mCRL2 [8]). In this paper, we use a pure theorem proving approach for the first time to model a well-known e-voting protocol. Our method is the Inductive Method which is one of the notable works in the verification of systems relying on theorem proving [1].The capability of this method in terms of verifying some complex industrial protocols such as SET, motivated us to candidate this framework for modeling of e-voting protocols. To this end, we extend the framework in some ways to model the well-known e-voting protocol FOO'92.

2 Inductive Method

The Inductive Method, based on the mathematical induction concept, introduced by Paulson [1] for verifying various protocols [4], [9]. The basic concepts in this method are *event* and *trace*. As an example of event, when an agent *A* sends a message *M* to an agent *B*, the event "*says A B M*" occurs in the network. Using the concept of *event*, the network traffic can be modeled through various events occurring in the network. A trace is a list of events which models a history of the network (a possible execution). A security protocol is modeled as a set of all possible traces which is defined inductively. Such a definition specifies how to extend a trace belonging to the set of traces using a new event. Accordingly, security properties can be proved by induction on traces.

Various roles in the protocol are formalized using the *agent* datatype. It consists of *Server* role which is the trusted third party required in most symmetric cryptographic protocols, *Spy* that is an adversary in the environment, and *Friend i* which indicates an unlimited number of agents indexed by the natural number *i*. *msg* datatype models any kind of messages exchanged in the network. It contains recursive constructors for defining encrypted, hashed, or concatenated messages. The *event* datatype represents different events which could occur during the protocol execution. For example, agents send messages to each other(*Says*), they may receive some messages(*Gets*), or an agent makes a note of a message for its subsequent use(*Notes*).

Each kind of cryptographic keys is formalized by a function. A set called *symKeys* is defined to represent symmetric encryption. The keys which belong to this set has the same encryption and decryption keys. Asymmetric encryption are implemented using functions which assign each agent a private or public key. Using an axiom which states that no private key equals any public key, asymmetric keys become disjoint from symmetric keys.

The *initialState* function assigns each agent his long-term keys before the protocol execution. In order to operate on messages, the some operators such as *synth* and *analz* are defined. Function *analz* recursively extracts all the obtainable information from message set H. Function *synth*, synthesizes all possible messages by encrypting and concatenating members of a message-set H, recursively.

Dynamic knowledge of agents is modeled by the *knows* function. The function inspects a given trace and extracts any messages that the given agent can know from that trace.

The implemented threat model of the Inductive Method is an extension of the Dolev-Yao model. The *Spy* besides having the capabilities of a Dolev-Yao attacker, can compromise with other agents.

3 E-voting protocols

E-voting refers to any voting process which benefits from electronic means. One of the well known e-voting protocols is the FOO'92 scheme introduced by Fujioka, Okamoto, and Ohta [3].

Three types of agents participate in the FOO'92 protocol: voter (V), authentication authority (A), and counting authority (C). The steps of the protocol are as follows:

1. V → A : V, σv(χ(ξ(v, r), b))
2. A → V : σA(χ(ξ(v, r), b))
3. V → C: σA(ξ(v, r))
4. C → : l, σA(ξ(v, r))
5. V → C: l, r

The protocol consists of three phases: registration, casting and tallying. In the registration phase(steps 1 and 2), voter uses blind signature schema χ and blinding factor *b* in order to preserve his privacy. Blind signature is a mathematical scheme that hides the contents of a message from the third party's view who signs it, namely he signs the message with closed eyes. Then the resultant signed blinded message could be uncovered by removing the blinding factor to obtain the original message signed by that party. In this phase, *A* checks that only illegible voters can vote.

In the casting phase(steps 3) using the commitment schema ξ and commitment key *r*, *V* commits to his vote and sends the committed vote to *C*. he keeps the key private until the end of the voting process.

In the tallying phase(steps 4 and 5), the counting authority decides to publish the results in the bulletin board so that every one can read it. Then *V* reveals his key (*r*) and the authority can decrypt the committed vote and merge it to the results of the election.

4 Modeling of the FOO'92 Protocol

We first formalize blind signature, commitment bit and broadcast communication and employ it for modeling of the FOO'92 protocol.

Blind signature and commitment can be considered as symmetric encryption. Each agents has its own keys and since the keys are random strings, no two agents have the same keys. So an injective function is defined to assign each agent, a key. Blinding factor and commitment factor of compromised agents are available to the *Spy* before the protocol execution. So, *initialState* of agents is changed and related theorems is proved and updated. In addition an axiom is defined which states that a singed message could be obtained from a signed blinded message. Blinding factors and commitment keys are also became disjoint by an axiom.

We introduced broadcast communication to the Inductive Method by adding the new event *Broadcast*. By occurring this event in the network, all the agents will know the broadcasted message. Hence the *knows* function which computes dynamic knowledge of agents in a trace, is consequently changed. The above formalization and related theorems are not brought here due to lack of space.

Management of phases: Distinguishing between phases of the protocol is crucial to achieve security goals such as privacy. We define three deadlines to specify the condition of reaching the end of each protocol phase. Then using the formalization of current time of a trace (*CT* function), it could be verified if a given trace is in a specific phase.

Protocol Agents: In FOO'92, *A* needs to access all of the public keys of voters and his own private key. *C* requires to have *A*'s public key to ensure that a vote is actually signed by him. The two authorities (*A* and *C*) could engage in the protocol using the knowledge of a friendly agent but since we have only one instance of these authorities, we should fix each of them to be represented by a specific *Friend*. Moreover, in our attempt to model the protocol we assume that they do not compromise.

Vote Value: In view of the fact that each voter has limited options to select as his/her vote, a vote value could be guessed by the adversary. Guessable numbers are modeled separately from unguessable numbers such as nonces and fresh values by the *Number* constructor. We employ this type of messages to model a vote value.

Model of the protocol: In the following figure, the model of the protocol is presented. Traces of the protocol could be constructed recursively by these rules. if the premises *P* of rule *R* is satisfied, the rule *R:* $\frac{P}{Q}$ could be used to extend the trace which results in the new trace Q (# concatenates the events of a trace). Nil is the base case of the model. *Fake* rule is used to model the power of the *Spy* to fake any possible messages based on his knowledge of a trace *t*. *Reception* rules, states that for any *Says* event we may have a *Gets* event (or we may not have this event which shows that the *Spy* intercepts the packets delivery). Other rules formalize how to extend a trace using the steps of the protocol.

$$\text{Nil: } \frac{}{[] \in \text{foo}}$$

$$\text{Fack: } \frac{t \in \text{foo};\ X \in \text{synth (analz (knows Spy } t))}{\text{Says Spy B X \# } t \in \text{foo}}$$

$$\text{Receptoin: } \frac{t \in \text{foo};\ \text{Says A B X} \in t}{\text{Gets B X \# } t \in \text{foo}}$$

$$\text{Receptoin}_{\text{BC}}\text{: } \frac{t \in \text{foo};\ \text{Broadcast A X} \in t}{\text{Gets B X \# } t \in \text{foo}}$$

$$\text{Foo1: } \frac{t \in \text{foo};\ V \neq A;\ \text{registration\_phase } t}{\text{Says V A } \{V, \sigma v(\chi(\xi(v, r), b))\}}$$

$$\text{Foo2: } \frac{\begin{array}{c} t \in \text{foo};\ V \neq A;\ \text{Gets A } \{V, \sigma v(X)\} \in t; \\ \text{registration\_phase } t; \text{Notes A } \{V\} \notin t; \\ V \in \text{legitimate\_voters} \end{array}}{\text{Notes A V \# Says A V } \sigma A(X) \# t \in \text{foo}}$$

$$\text{Foo3: } \frac{\begin{array}{c} t \in \text{foo};\ V \neq C;\ \text{casting\_phase } t; \\ \text{Gets V } \sigma A(\chi(X, b)) \in t \end{array}}{\text{Says V C } \sigma A(X) \# t \in \text{foo}}$$

$$\text{Foo4\_1: } \frac{t \in \text{foo};\ \text{Gets C } \sigma A(X) \in t;\ \text{casting\_phase } t}{\text{Notes C } \{|\text{ Number (CT } t), \sigma A(X) |\} \# t \in \text{foo}}$$

$$\text{Foo4\_2: } \frac{t \in \text{foo; talling\_phase } t;\ \text{Broadcast C X} \notin t}{\text{Broadcast C (setBoard } t) \# t \in \text{foo}}$$

$$\text{Foo5: } \frac{\begin{array}{c} t \in \text{foo};\ V \neq A;\ V \neq C; \text{talling\_phase } t; \\ \text{Says V C } \sigma A(\xi(v, r)) \in t; \\ \text{Broadcast C X}' \in t;\ \text{Gets V X}' \in t; \\ \{|\text{ Number l}, \sigma A(\xi(v, r))|\} \in (\text{analz } \{X'\}) \end{array}}{\text{Says V C } \{|\text{ Number l, r}|\} \# t \in \text{foo}}$$

References

1. Paulson, L.C.:The Inductive Approach to Verifying Cryptographic Protocols. J. Computer Security, vol. 6, 85–128 (1998)
2. Lowe, G.: Breaking and Fixing the Needham-Schroeder Public-Key Protocol using. FDR. In: Margaria, T., Steffen, B. (eds.) TACAS 1996. LNCS, vol. 1055, pp. 147–166 (1996)
3. Fujioka, A., Okamoto, T., Ohta, K.: A practical Secret Voting Scheme for Large Scale Elections. In Proc. of AUSCRYPT '92, vol. 718, pp. 244–251, (1992)
4. Bella, G., Paulson, L. C.: Mechanising BAN Kerberos by the Inductive Method. In Proc. of CAV'98 – 10th International Conference on Computer Aided Verification, pp. 416–427,(1998)
5. Bella, G., Paulson, L. C.:Kerberos Version 4: Inductive Analysis of the Secrecy Goals. In Proc. of 5th European Symposium on Research in Computer Security, pp. 361-375, (1998)
6. Kremer , S., Ryan , M. D.: Analysis of An Electronic Voting Protocol in the Applied pi-calculus. In Proc. of ESOP'05, LNCS, vol. 3444., pp. 186–200, Springer, (2005)
7. Delaune, S., Ryan, M., Smyth, B.: Automatic Verification of Privacy Properties in the Applied Pi Calculus. In Karabulut, Y., Mitchell, J. ,Herrmann, P. , Jensen, C. D. (eds.) IFIPTM'08. Proc. of the second Joint iTrust and PST Conferences on Privacy, Trust Management and Security, vol. 263, pp. 263–278, Springer, Norway (2008)
8. Mahrooghi, H. R., Haghighat, M., H., Jallil, R.: Formal verification of authentication-type properties of an electronic voting protocol using mcrl2. In: Proc. Fourth International Workshop on Verification and Evaluation of Computer and Communication Systems (VECoS 2010), (2010)
9. Bella, G., Massacci, F., Paulson L. C.: An overview of the verification of SET. J. International Journal of Information Security, 17–28 (2005)
10. Chothia, T., Orzan, S., Pang , J., Dashti, M. T.: A framework for automatically checking anonymity with μ-CRL. In Lecture Notes in Computer Science, Vol. 4661, pp. 301-318, (2006)
11. Baskar, A., Ramanujam, R., Suresh, S. P.: Knowledge-based Modeling of Voting Protocols. In Proc. 11th conference on Theoretical aspects of rationality and knowledge, pp. 25-27, Belgium (2007)