Verification of Resistance of Denial of Service Attacks in Extended Applied Pi Calculus with ProVerif

Bo Meng
School of Computer, South-Center University for Nationalities, Wuhan, China
Email: mengscuec@gmail.com

Wei Wang and Wei Chen
School of Computer, South-Center University for Nationalities, Wuhan, China
Email: {wangwei9852003@yahoo.com.cn, chenw2011@hotmail.com}

***Abstract*—Owning to the damage of denial of service attacks in security protocols, resistance of denial of service attacks plays an important role in remote voting protocols. Recently Meng et al. proposed a secure remote internet voting protocol that claims to satisfy formal definitions of key properties without physical constrains. In this study firstly the review of the formal model of resistance of denial of service attacks in security protocols is introduced, then extended applied pi calculus and Huang et al. formal model are reviewed, after that Meng et al. protocol is modeled in extended applied pi calculus, finally resistance of denial of service attacks is proved with ProVerif. The result we obtain is that Meng et al. protocol is not resistance of denial of service attacks because one denial of service attack is found. At the same time we also propose a method to prevent it from the denial of service attack. To our best knowledge, we are conducting the first mechanized proof of resistance of denial of service attacks in Meng et al. protocol for an unbounded number of honest and corrupted voters.**

***Index Terms*—protocol security, automatic verification, protocol state, symbolic model, availability**

I. Introduction

Owning to advantages of remote internet voting, it plays an important role in electronic government. In order to assess its securities and increase confidence of the voters in remote internet voting system and protocols, many researchers have paid attention to development and verification on secure remote internet voting systems and protocols[1, 2].

The practical secure remote internet voting protocol should include privacy, completeness, soundness, fairness and invariableness, universal verifiability, receipt-freeness, coercion-resistance and resistance of denial of service attacks. Previous works focus on implementation and formal analysis of receipt-freeness and coercion-resistance [1]. In the last twenty years many remote internet voting protocols [3-8], claimed on their security, have been proposed. To our best knowledge, until now resistance of denial of service attacks in these remote internet voting protocols has not been analyzed.

Denial of service attacks are attacks against availability, attempting to prevent legitimate users from accessing the network and distributed system. This kind of attacks aims at rendering a network an system incapable of providing normal service by targeting either the network, bandwidth or connectivity. Denial of service attacks is simple and effective. For example, the adversary can produce many bogus messages and send to target of attack. That make the target of attack can not provide normal service for legitimate user owning to process big bogus messages. At the same time it is not easy to find the adversary and adversary can mount another type attack based on denial of service attacks, for example, man-in-the-middle attack.

In order to prevent denial of service attacks, the first step is to analyze and prove resistance of denial of service attacks in protocol, network and distributed system with formal method, and then to increase the confidence of people in its security. There are two models that can be used: symbolic model in which cryptographic primitives are ideally abstracted as black boxes and computational model based on complexity and probability theory. Computational model is complicated and is difficult to get the support of mechanized proof tools. In contrast, symbolic model is simpler than the computational model, and can sometimes benefit from mechanized proof tools support. For example: ProVerif [9], SMV, NRL, Casper, Isabelle, Athena, Revere, SPIN, Brutus, Scyther.

In symbolic model there are mainly three formal frameworks in resistance of denial of service attacks. One is Yu-Gligor model [10] based on user agreement. The core of framework is based on access control policy. The second one is Meadows's cost-based model [11] built on the notion that a protocol is a sequence of operations with cause-effect relationships: an action by one principle usually causes a sequence of actions by another principle that incurs some cost. People pay much attention to it. The third one is Huang et al. model [12] which proposes

Corresponding author: Bo Meng, School of Computer, South-Center University for Nationalities, Wuhan, China, 430074

This study was supported in part by Natural Science Foundation of The state Ethnic Affairs Commission of PRC under the grants No: 10ZN09

the first automatic method of resistance of denial of service attacks with ProVerif. ProVerif is a mechanized proof of cryptographic protocol verifier based on a representation of the protocol by Horn clauses or applied pi calculus. It can deal with an unbounded number of sessions of the protocol and an unbounded message space. When ProVerif cannot prove a property, it can reconstruct an attack. ProVerif has been tested on many security protocols with very great results.

Recently Meng et al. [7] proposed a remote internet voting protocol that claims to satisfy formal definitions of key properties without physical constrains. Until now its resistance of denial of service attacks has not been analyzed. So here we use mechanized proof tool ProVerif to verify its resistance of denial of service attacks based on Huang et al. model [12].

The main contributions of this paper are summarized as follows:

- Review the formal model of resistance of denial of service attacks in security protocols. There are mainly three formal frameworks in resistance of denial of service attacks: Yu-Gligor model, Meadows's model and Huang et al. model which is the first automatic model of resistance of denial of service attacks with ProVerif. Until now resistance of denial of service attacks model based on computational model has not been proposed.
- Apply the mechanized formal model proposed by Huang et al. for mechanized proof of resistance of denial of service attacks. Therefore, Meng et al. protocol is modeled in extended applied pi calculus and resistance of denial of service attacks take into account. The proof itself is performed by mechanized proof tool ProVerif.
- The result we obtain is that Meng et al. protocol is not resistance of denial of service attacks because one denial of service attack is found by us. At the same time we also propose a method to prevent the denial of service attack. To our best knowledge, we are conducting the first mechanized proof of resistance of denial of service attacks in Meng et al. protocol for an unbounded number of honest and corrupted voters.

II. RELATED WORKS

In symbolic model there are mainly three formal frameworks in resistance of denial of service attacks: Yu-Gligor model [10] based on user agreement, Meadows's cost-based model [11], Huang et al. model [12] based on theorem proof. In computational model resistance of denial of service attacks analysis model has not been proposed. To our best knowledge until now resistance of denial of service attacks in remote voting protocol has not been analyzed.

May be one of the first attempts to formalize the notion of resistance of denial of service attacks was done by Gligor [13, 14] with maximum waiting time. He defines availability as the guaranty of a maximum specified waiting time for any operation, even in case of concurrent accesses. Then Yu and Gligor [10] propose a formal specification on resistance of denial of service attacks based on temporal logic by introduction of notion of user agreement. The core of framework is access control policy. It does not deal with denial of service attacks executed before authentication between sender and receiver in protocols, for example, SYN floods attacks. At the same time it does not support the automated tools. Bacic and Kuchta [15] argue that the core problem of resistance of denial of service attacks is resource allocation. They introduce the notion of a resource allocation monitor that has to have three reference monitor characteristics. Millen [16] extended Yu-Gligor model by representing the passage of time explicitly expressed as a finite-waiting-time policy.

Meadows [11] introduces a formal framework on resistance of denial of service attacks based on the costs spending on computation by the principles in security protocols. His model bases on fail-stop protocol. He analyzes the station to station protocol and point out that it is not resistance of denial of service attacks. But Meadows's model maybe not practical because the costs of generating a bogus message is smaller than costs of processing and verifying it, so all protocols are not resistance of denial of service attacks. Following this line, Ramachandran [17] analyzes JFK protocol and points that it is resistance of denial of service attacks with the conditions that bogus messages are handled in an appropriate way. Smith et al. [18] also analyze JFK protocol with Meadows's model. Lafrance and Mullins [19] present a method based on admissible interference for finding denial of service attacks in security protocols. Using SPPA and Meadows's framework, they introduce an information flow property called impassivity. Abadi et al. [20] use the observational-equivalence relation to formalize denial of service attacks and find JFK protocol is resistance of denial of services attacks. Tritilanunt et al. [21] and Tritilanunt [22] firstly point out that the cost analysis has only taken into account honest runs of the protocol in Meadows's model. They use the colored Petri nets to model the denial of service attacks based on cost-based and time-based model and analyze HIP protocol. They find that HIP protocol is not resistance of denial of service attacks in the conditions Type 3 adversary or Type 4 adversary. Zhou et al. [23] propose a model based on strand spaces and 4-way handshakes protocol is analyzed. They find that it is not resistance of denial of service attacks.

Huang et al. [12] present the first automatic method of resistance of denial of service attacks based on theorem proof with ProVerif. They extend the applied pi calculus from the attacker contexts and process expression, and then from the view of protocol state, then propose the first automatic method of resistance of denial of service attacks based on extended applied calculus. Resistance of denial of service attacks in JFK protocol and IEEE 802.11i four-way handshake protocol are analyzed. The results they obtained are that JFK protocol is resistance of denial of service attacks and IEEE 802.11i four-way handshake protocol is not. The methods to prevent resistance of denial of service attacks in IEEE 802.11i four-way handshake protocol are proposed.

Besides the three models, Amoroso [24] emphasizes the need for specifying a service model in terms of prevent (p, c) policies as predicates concerned subjects, resources and resource consumption operations. Based on modal logic and deontic logic, Cuppens and Saurel [25] propose a formal model to formalize availability policy by predicates expression of permissions, prohibitions and obligations of subjects. Cuppens et al. [26] use the formal security model called Nomad to specify availability requirements. They mainly concern the denial of service attacks in program. Agha et al. [27] use probabilistic extension of the Maude term rewriting system to model denial of service attacks and use a sublogic of Continuous Stochastic Logic to describe the rate of success of attack and use VESTA to analyze the TCP 3 3-way Handshaking protocol and find it is not resistance of denial of services attacks. Mahimkar and Shmatikov [28] use the alternating time temporal logic to model bandwidth consumption and resource exhaustion attacks and verify JFKr with MOCHA. They find that it is resistance of denial of service attacks.

III. Review of Huang et al. Formal Model

In this section we review the extended applied pi calculus, the definition of resistance of denial of service attacks, and the method of automated proof of resistance of denial of service attacks.

A. *Extended Applied Pi Calculus*

Here we only review adversary contexts and the process expression, the other content can be found in the reference [12]. The extended applied pi calculus is also supported by ProVerif.

❅ **Adversary contexts**

In extended applied pi calculus, according to abilities of adversary, the contexts of adversary are classified into two contexts: ideal context and real context. Real context is formalized as $\nu\tilde{n}.C\left[C\left[\bar{c}\langle u\rangle\right]\bar{u}\langle N\rangle.P\right], \nu\tilde{n}.C\left[C\left[c(u)\right]u(x).P\right]$, where $u\in\tilde{n}, c\notin\tilde{n}$. Real context is insecure environments. The adversary can overhear, intercept, and synthesize any message and is only limited by the constraints of the cryptographic methods used. Ideal context is formalized as $\nu\tilde{n}.C\left[\bar{u}\langle N\rangle.P\right], \nu\tilde{n}.C\left[u(x).P\right]$, where $u\in\tilde{n}$. Ideal context is secure environments. The adversary can not overhear, intercept, and synthesize any message.

❅ **Plain process**

In extended applied pi calculus, it has plain processes and extended processes. Plain processes in Figure 1 are built up in a similar way to processes in the pi calculus, except that messages can contain terms and that names need not be just channel names.

The process 0 is an empty process. The process $Q|P$ is the parallel composition of P and Q. The replication $!P$ produces an infinite number of copies of P which run in parallel. The process $\nu n.P$ firstly creates a new, private name then executes as P. The abbreviation $\nu\tilde{n}$ is a sequence of name restrictions $\nu n_1,\cdots,\nu n_l$. The process $\mathrm{in}\left(u,\widetilde{M}\right).P$ receives a message from channel u, and runs the process P by replacing formal parameter x by the actual message. We use $\mathrm{in}\left(u,\widetilde{M}\right).P$ for the input of terms $M_1\cdots,M_l$. The process $out\left(u,N\right).P$ is firstly ready to output the message N on the channel u, and then runs the process P. The process $out\left(u,\widetilde{N}\right).P$ is the abbreviation for the output of terms $N_1\cdots,N_l$. The conditional construct $if\ M=N\ then\ P\ else\ Q$ runs that if M and N are equal, executes P, otherwise executes Q in real context. The conditional construct $if\ M=N\ then\ P\ else\ C\left[\bar{c}\langle S\rangle\right].Q$ runs that if M and N are equal, executes P, otherwise executes $C\left[\bar{c}\langle S\rangle\right].Q$ in idea context.

$P,Q,R ::=$	plain processes
0	null process
$Q\|P$	parallel composition
$!P$	replication
$\nu n.P$	name restriction
$if\ M=N\ then\ P\ else\ Q$	conditional in real context
$if\ M=N\ then\ P\ else\ C\left[\bar{c}\langle S\rangle\right].Q$	conditional in idea context
$\mathrm{in}\left(u,x\right).P$	message input
$out\left(u,N\right).P$	message output

Figure 1. plain process

❅ **Process context**

$C ::=$	process context
$[\ ]$	null process context
$P\ \|\ C$	parallel composition
$C\ \|\ Q$	parallel composition
$!C$	replication
$\nu n.C$	name restriction
$if\ M=N\ then\ C\ else\ Q$	conditional
$if\ M=N\ then\ P\ else\ C$	conditional
$\mathrm{in}\left(u,x\right).C$	message input
$out\left(u,N\right).C$	message output

Figure 2. Process context

Process context in Figure 2 is a process with a hole $[\ ]$. The process 0 is an empty process context. The process $Q|P$ is the parallel composition of P and Q. The replication $!C$ produces an infinite number of copies of C which run in parallel. The process $\nu n.C$ firstly creates a new, private name then executes as C. The process $\mathrm{in}\left(u,x\right).C$ receives a message from channel u, and runs the process context C by replacing formal parameter x by the actual message. We use $\mathrm{in}\left(u,\widetilde{M}\right).C$ for the input of terms $M_1\cdots,M_l$. The process $out\left(u,N\right).C$ is firstly ready to output the message N on the channel u, and then runs the process context C. The process $out\left(u,\widetilde{N}\right).C$ is the abbreviation for the output of terms $N_1\cdots,N_l$. The conditional construct $if\ M=N\ then\ C\ else\ Q$ runs that if M and N are equal, executes process context C, then C

is a verified context. The conditional construct $if\ M = N\ then\ P\ else\ C$ runs that if M and N are not equal, executes C, then C is not a verified context.

B. Definition of Resistance of Denial of Service Attacks

P is an annotated Alice-and-bob specification in protocol, B is resistance of denial of service attacks if and only if set of association ω between any message M_i and M_j in set $Recv(B)$:

①. ω is null set $\varnothing$;

②. Any data items in ω are authenticated.

Where $Recv(B)$ is set where data items are in operations that are ordered in casually precedes in $act_j(B)[M_j, O_1^j, \cdots, O_k^j]$, where $i, j \in [1, n]$ and $i < j$.

C. Method of Automated Proof of Resistance of Denial of Service Attacks

Applying the extended applied pi calculus, the protocol can be modeled as an annotated Alice-and-Bob specification. Assume that the protocol exchanges $2n$ messages between principles Alice and Bob in a run. Principles Bob receives n messages M_i, where $i \in [1, n]$. Principles Bob sends n messages M_i', where $i \in [1, n]$. Protocol process $PP \equiv \nu \tilde{n}.(!Alice \mid !Bob)$ is a closed process and consists of parallel composition of any initiator processes $Alice$ and responder processes Bob.

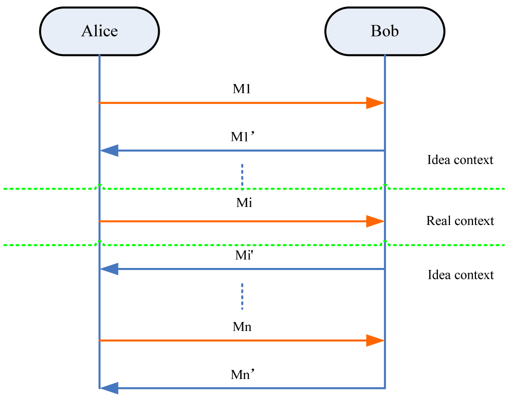


Figure 3. The formal model of messages

In order to use ProVerif to automatically prove resistance of denial of service attacks of Bob, the any messages M_i, where $i \in [1, n]$, is modeled with the extended applied pi calculus. If the adversary can get the secret $Secret$ on the public channel c, then the adversary can launch a denial of service attack by attack of message M_i. The method is used to model the messages $M_i\ i \in [1, n]$ in Figure 4. The message M_i is exchanged and processed in real context. The other messages are exchanged and processed in idea context. Protocol process PP is $PP \equiv \nu \tilde{n}.(!Alice_i \mid !Bob_i)$, c is public channel. c_j, where $j \in [2, n] \cap j \neq i$, are private channels used to receive messages M_j, where $j \in [1, n] \cap j \neq i$. If the adversary can get the secret message $Secret$ on the public channel c, then the adversary can launch a denial of service attacks by attacks of message M_i.

IV. Modeling Meng et al. Protocol in Extended Applied Pi Calculus

A. Meng et al. protocol

In Meng et al. protocol [7], when coerced by the adversary, the voter wants to lie about the decrypted message to a coercer and hence, escape coercion. In other word , the voter is able to decrypt the correct message from the registration authority, that is mean that all the information held by the voter when opened to a coercer, do not allow this coercer to verify the encrypted message ,or the coercer can not find the message is a fake message. Consequently, bribing or coercing the voter becomes useless from the very beginning. It includes four phases: preparation phase, registration phase, voting phase and tallying phase.

- **Preparation phase**

The registration authority A_R chooses a random element $\alpha \in \mathbb{Z}_{N^2}^*$, and sets $g = \alpha^2 \bmod N^2$, publishes publicly (N, g). Then the voter V_j gets (N, g) and chooses a random number $a \in [1, \mathrm{ord}(\mathrm{G})]$, computes $h = g^a \bmod N^2$ and publish publicly (g) .The public key of the registration authority A_R is given by the triplet ${}_{ElG}PU_R$, ${}_{BCP}PU_R$, while the corresponding secret key is private key ${}_{BCP}PR_R(p, q)$. At the same time the voter V_j can generates his public key ${}_{BCP}PU_j = (N, g, h)$ and private key ${}_{BCP}PR_j = a$ based on BCP cryptosystem. Finally he also creates his private key ${}_{ElG}PR_j = a$ and public key ${}_{ElG}PU_j = y = g^a \bmod p$ according to ElGamal cryptosystem. Because everyone can know the public key ${}_{BCP}PU_R = (N, g, h)$ of registration authority A_R, the voter V_j can get the registration authority A_R ' private key ${}_{BCP}PR_j = a$ through the knowledge of $h = g^a \bmod N^2$ and $N = p \times q$. Finally the registration authority A_R generates his public key and private key ${}_{ElG}PU_R$, ${}_{ElG}PR_R$ based on ElGamal cryptosystem registration authority A_R generates the ballot B^t and send B^t and its digital signature to bulletin board denoted by BB. Tallying authority A_T generates his public key ${}_{BCP}PU_T = (N, h)$ and private key ${}_{BCP}PR_T = (p, q)$ according to BCP commitment scheme.

- **Registration phase**

Voter V_j generates $Identif_j$ and ${}_{ElG}PR_j(Identif_j) \parallel {}_{ElG}PU_j \parallel Identif_j$, then sends it to registration authority A_R. Registration authority A_R

receives the message and uses its private key to verify the digital signature. Registration authority A_R checks $Identif_j$ that whether he has registered or not. If he has registered, registration authority A_R sends the error message to V_j. The protocol ends. If he has not registered, registration authority A_R produces $\left[{}_{ElG}PR_R\left\{(\partial,\wp)\,\|\,B=C(r,C_j)\right\}\|\,(\partial,\wp)\,\|\,B=C(r,C_j)\right]$ according to requirements of MW deniable encryption scheme, and then sends it to the voter by tappable channel. Finally, registration authority A_R sends ${}_{ElG}PR_R\left({}_{ElG}PU_R(C_j)\right)\|\;{}_{ElG}PU_R(C_j)$ to bulletin board.

☸ Voting phase

Voter V_j chooses his favor ballot. Using tallying authority A_T 'public key ${}_{BCP}PU_T=(N,h)$ voter V_j generates $B_C=C(r_1,C_j)\|\;B_B=C(r_1,B_t)$ with BCP commitment scheme and sends it to randomly in bulletin board by a tappable channel.

☸ Tallying phase

Firstly according to the rules the tallying authority eliminates the duplicate $B_C=C(r_1,C_j)\|\;B_B=C(r_1,B_t)$,then mixing authority mixes $B_C=C(r_1,C_j)\|\;B_B=C(r_1,B_t)$ and get the corresponding results are $\phi\left[B_C=C(r_1,C_j)\right]\|\,\phi\left[\;B_B=C(r_1,B_t)\right]$ .After that tallying authority A_T decrypts $\phi\left[B_C=C(r_1,C_j)\|\;B_B=C(r_1,B_t)\right]$ and gets C_j and B_t. At the same time tallying authority A_T verifies ${}_{ElG}PR_R\left({}_{ElG}PU_R(C_j)\right)\|\;{}_{ElG}PU_R(C_j)$ and let registration authority A_R decrypt ${}_{ElG}PU_R(C_j)$ and gets C_j. Finally tallying authority A_T tallies the ballot and publishes the results.

B. Function and Equational Theory

The function and equational theory is introduced in this section. We use extended applied pi calculus to model Meng et al. protocol. Figure 4 describes the functions and Figure 5 describes the equational theory in Meng et al. protocol.

```
Fun sign(x, PR).   Fun checksign(y, PK, x)     Fun PK(x, r)
Fun PR(x, r)    Fun mod(g, a, N)
Fun ELG_enc(x, PK, r)   Fun BCP_enc(x, PK)
Fun ELG_dec(x, PR)   Fun BCP_dec(x, PR)
Fun mod_inverse(x, g, p, q)    Fun multi(x, y)
```

Figure 4. Functions

```
equation mod_inverse(mod(a, g, multi(p, q)), g, p, q) = a
equation checksign(sign(x, PR(y, ELG)), PK(y, ELG), x) = true.
equation ELG_dec(ELG_enc(x, PK(y, ELG), r), PR(y, ELG)) = x.
equation BCP_dec(BCP_enc(x, PK(y, BCP)), PR(y, BCP)) = x..
```

Figure 5. Equational theory

ElGamal cryptosystem is modeled with decryption algorithm $ELG\_dec(x,\ PR)$ and encryption algorithm $ELG\_enc(x,\ PK,\ r)$. $ELG\_dec(x,\ PR)$ decrypts the ciphertext x with private key PR. $ELG\_enc(x,\ PK,\ r)$ encrypts the plaintext x with public key PK and random number r .BCP encryption scheme is expressed by decryption algorithm $BCP\_dec(x,\ PR)$ and encryption algorithm $BCP\_enc(x,\ PK)$. $BCP\_dec(x,\ PR)$ decrypts the ciphertext x with private key PR. $BCP\_enc(x,\ PK)$ encrypts the plaintext x with public key PK. The digital signature is modeled as being signature with message recovery. The digital signature algorithm includes the generation signature algorithm $sign(x,PR)$ which signs the message x with private key PR and the verification algorithm $checksign(y,\ PK,\ x)$ which verifies the digital signature y with public key PU and the message x. $PK(x,\ r)$ is the algorithm of generating the public key of cryptosystem r with the random number x. $PR(x,\ r)$ is the algorithm of generating the private key of cryptosystem r with the random number x. $mod(g,a,N)$ denotes modulus operator and $mod\_inverse(x,g,p,q)$ denotes inverse modulus operator. $multi(x,y)$ denotes multiplication operation.

C. Processes

The complete formal model of Meng et al. protocol in extended applied pi calculus is given in figures below. Figure 6 to 11 report the basic process including main process, voter process, corrupted voter process, registration authority process, identity issuer authority process and tallying authority process forming our of the model.

```
Meng et al.protocol ≜
    new voter; new reg;  new tal;
    let ELG_PK_voter = PK(voter, ELG) in
    let ELG_PR_voter = PR(voter, ELG) in
    let ELG_PK_Reg = PK(reg, ELG) in
    let ELG_PR_Reg = PR(reg, ELG) in
    let BCP_PK_Tal = PK(tal, BCP) in
    let BCP_PR_Tal = PR(tal, BCP) in
    out(pub, ELG_PK_voter); out(pub, ELG_PK_Reg);
    out(pub, BCP_PK_Tal);
    ((!voter) | (!Identity_Issue_Authority) | (!Tallying_Authority)
    | (!Registration_Authority) | (!corruptedvoter))
```

Figure 6. Main process

The main process in Figure 6 sets up private channels $chVR;\ chVII; chIIR; chTR$ and specifies how the processes are combined in parallel. $chVR$ is the private channel between voter and registration authority. $chVII$ is the private channel between voter and identity issuer authority. $chIIR$ is the private channel between registration authority and identity issuer authority. $chTR$ is the private channel between registration authority and tallying authority. At the same time the main process generates the key parameters $voter$ for voter, reg for registration authority, tal for tallying authority.

```
voter ≜
    new g; new p; new q; new nonceh;
    let N = multi(p, q) in
    out(chVR1, (n1, nonceh, N, g));
    in(chVR1, (= n2, = nonceh, h, e lg_pk_reg));
    let a = mod_inverse(h, g, p, q) in
    let ELG_BCP_PK_voter = PK(a, ELG) in
    let BCP_PK_voter = PK(a, BCP) in
    let ELG_BCP_PR_voter = PR(a, ELG) in
    let BCP_PR_voter = PK(a, BCP) in
    let ELG_BCP_PK_Reg = PK(a, ELG) in
    new nonce; new nonce1;
    out(chVII, (n1, nonce1)); in(chVII, (= n2, = nonce1, id));
    out(chVR2, (n1, nonce, id, ELG_PK_voter, sign(id, ELG_PR_voter)));
    in(chVR2, (= n2, = nonce, alphabet, Bc, signed));
    if checksign(signed, e lg_pk_reg, (alphabet, Bc)) = true then
    (
    let (r, cred) = BCP_dec(Bc, BCP_PR_voter) in
    in(chvote, vote);
    new r1; new r2;
    let Bc1 = BCP_enc((r1, cred), BCP_PK_Tal) in
    let Bb = BCP_enc((r2, vote), BCP_PK_Tal) in
    out(com, (Bc1, Bb))
    )
    else out(pub, secret).
```

Figure 7. Voter process

```
corruptedvoter ≜
    new g; new p; new q; new nonceh;
    let N = multi(p, q) in
    out(chVR1, (n1, nonceh, N, g));
    in(chVR1, (= n2, = nonceh, h, e lg_pk_reg));
    let a = mod_inverse(h, g, p, q) in
    let ELG_BCP_PK_voter = PK(a, ELG) in
    let BCP_PK_voter = PK(a, BCP) in
    let ELG_BCP_PR_voter = PR(a, ELG) in
    let BCP_PR_voter = PK(a, BCP) in
    let ELG_BCP_PK_Reg = PK(a, ELG) in
    new nonce; new nonce1;
    out(chVII, (n1, nonce1)); in(chVII, (= n2, = nonce1, id));
    out(chVR2, (n1, nonce, id, ELG_PK_voter, sign(id, ELG_PR_voter)));
    in(chVR2, (= n2, = nonce, alphabet, Bc, signed));
    if checksign(signed, e lg_pk_reg, (alphabet, Bc)) = true then
    let (r, cred) = BCP_dec(Bc, BCP_PR_voter) in
    out(c, cred).
```

Figure 8. Corrupted voter process

Voter process is modeled in extended applied pi calculus in Figure 7. Each voter gets the credential id from identity issuer authority, and then sends it to registration authority. After that registration authority generates the ciphertext of the genius credential Bc according to MW deniable encryption scheme and sends it to voter. Voter verifies Bc with $checksign\left(signed, ELG\_PK\_Reg, \left(alphabet, Bc\right)\right)$. If the result is true, then voter opens Bc with his private key BCP_PR_voter in BCP cryptosystem, otherwise it outputs $secret$ by public channel pub. Voter chooses his favorite ballot $vote$ and generates ciphertext $BCP\_enc\left(\left(r1, cred\right), BCP\_PK\_Tal\right)$ of credential $cred$ with public key BCP_PK_Tal of tallying authority. He also generates ciphertext $BCP\_enc\left(\left(r2, vote\right), BCP\_PK\_Tal\right)$ of ballot $vote$ with public key BCP_PK_Tal of tallying authority. Finally voter sends $BCP\_enc\left(\left(r1, cred\right), BCP\_PK\_Tal\right)$ and $BCP\_enc\left(\left(r2, vote\right), BCP\_PK\_Tal\right)$ through public channel com to bulletin board.

Corrupted voter process is modeled in Figure 8. The corrupted voter will register and get his secret credentials id from identity issuer authority, and then sends it to registration authority. After that registration authority generates the ciphertext of the genius credential Bc according to MW deniable encryption scheme and sends it to voter. Voter verifies Bc with $checksign\left(signed, ELG\_PK\_Reg, \left(alphabet, Bc\right)\right)$. If the result is true then voter open Bc with his private key BCP_PR_voter in BCP cryptosystem, sends credentials $cred$ on a public channel c, so that the attacker can impersonate them in order to mount any sort of attack.

```
Registration_Authority ≜
    in(chVR1, (= n1, nonceh, N, g));
    new a;
    let h = mod(a, g, N) in
    out(chVR1, (n2, nonceh, h, ELG_PK_Reg));
    let ELG_BCP_PK_voter = PK(a, ELG) in
    let BCP_PK_voter = PK(a, BCP) in
    (in(chTR, (= n1, nonceT));
    new nonce;
    out(chIIR, (n1, nonce));
    in(chIIR, (= n2, = nonce, id)); in(chVR2, (= n1, nonceV, = id, pk, signed));
    if checksign(signed, pk, id) = true then
    new r1; new r2; new cred; new r3; new r4;
    let alphabet = ELG_enc((r1, r2), ELG_BCP_PK_voter, r4) in
    let Bc = BCP_enc((r2, cred), BCP_PK_voter) in
    out(chVR2, (n2, nonceV, alphabet, Bc, sign((alphabet, Bc), ELG_PR_Reg)));
    let enccred = ELG_enc(cred, ELG_PK_Reg, r3) in
    out(chTR, (n2, nonceT, enccred, sign(enccred, ELG_PR_Reg)));
    out(pub, (enccred, sign(enccred, ELG_PR_Reg))))
    | in(chTR, (= n1, nonceT, enccred1));
    let cred1 = ELG_dec(enccred1, ELG_PR_Reg) in
    out(chTR, (n2, nonceT, cred1))..
```

Figure 9. Registration authority process

The registration authority process is modeled in Figure 9. The registration authority receives voters id, and then generates the secret credentials $cred$. After that the registration authority creates $ELG\_enc\left(\left(r1, r2\right), ELG\_PK\_voter, r4\right)$ and $BCP\_enc\left(\left(r2, cred\right), BCP\_PK\_voter\right)$ according to MW deniable encryption scheme. He produces the digital signature $sign\left(\left(alphabet, Bc\right), ELG\_PR\_Reg\right)$ of $ELG\_enc\left(\left(r1, r2\right), ELG\_PK\_voter, r4\right)$ and $BCP\_enc\left(\left(r2, cred\right), BCP\_PK\_voter\right)$ and sends it to voter by channel $chVR$ from voter to registration authority. The registration authority generates the ciphertext $ELG\_enc(cred, ELG\_PK\_\mathrm{Re}g, r3)$ of credential $cred$ and sends it to bulletin board through public channel pub. He also decrypts ciphertext $ELG\_enc(cred, ELG\_PK\_\mathrm{Re}g, r3)$ of credential with his private key $ELG\_PR\_\mathrm{Re}g$ and gets credential and sent it to tallying authority through channel $chTR$ from tallying authority to registration authority.

$$\begin{array}{l}
\text{Identity\_Issue\_Authority} \triangleq \\
\quad in(chVII,(=n1, nonceV)); in(chIIR,(=n1, nonceR)); new\ \ id; \\
\quad out(chVII,(n2, nonceV, id)); out(chIIR,(n2, nonceR, id)); out(pub, id).
\end{array}$$

Figure 10. Identity issue authority process

The identity issue authority is modeled in Figure10. The identity issuer authority generates id and sends it to voter and tallying authority. He also publishes id bulletin board in through the channel pub

$$\begin{array}{l}
\text{Tallying\_Authority} \triangleq \\
\quad new\ \ nonce; out(chTR,(n1, nonce)); \\
\quad in(chTR,(=n2, =nonce, enccred, signed)); in(com, result); \\
\quad let\ \ (Bc, Bb) = result\ \ in \\
\quad let\ \ (r1, cred) = BCP\_dec(Bc, BCP\_PR\_Tal)\ \ in \\
\quad let\ \ (r2, vote) = BCP\_dec(Bb, BCP\_PR\_Tal)\ \ in \\
\quad if\ \ checksign(signed, ELG\_PK\_\mathrm{Re}g, enccred) = true\ \ then \\
\quad new\ \ nonce; out(chTR,(n1, nonce, enccred)); \\
\quad in(chTR,(=n2, =nonce, cred1)); \\
\quad if\ \ cred = cred1\ \ then\ \ \ \ out(pub, vote);.
\end{array}$$

Figure 11. Tallying authority process

Tallying authority process is modeled in Figure11. After the voting time expires, the tallying authority gets the all ballots on bulletin board posted by allegedly eligible voters and then decrypt Bc and Bb with his private key BCP_PR_Tal to get $cred$ and $vote$. Tallying authority gets ciphertext $ELG\_enc(cred, ELG\_PK\_\mathrm{Re}g, r3)$ of the voter's credential and verifies $checksign(signed, ELG\_PK\_\mathrm{Re}g, enccred)$ the digital signature $enccred$. If the result is true, then he sends $enccred$ to registration authority. Registration authority decrypt $enccred$ with his private key to recover plaintext $cred1$ which is the credential. Finally tallying authority compares it with $cred$, if it is true, he publish the tallying result $vote$ on bulletin board.

V. Mechanized proof of Meng et al .Protocol with ProVerif

ProVerif can take two formats as input. The first one is in the form of Horn and applied pi calculus. The second one is in the form of a process in an extension of the pi calculus [12, 29]. In both cases, the output of the system is essentially the same.

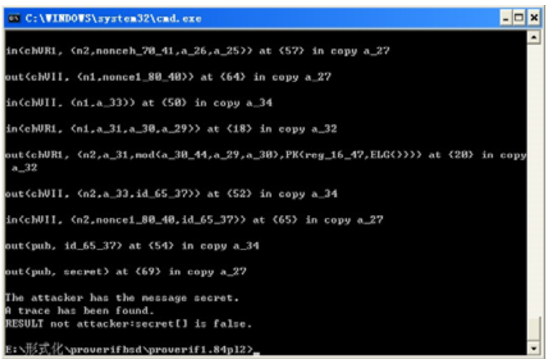
In this paper we use the extended pi calculus in Huang et al. model as the input of ProVerif. In order to prove resistance of denial of service attacks in Meng et al. protocol, the formal model based on the extended applied pi calculus is needed to be translated into the syntax of ProVerif and generated the ProVerif inputs in the extended pi calculus. The code in analysis of resistance of denial of service attacks in Meng et al. protocol is presented in Fig.12.

The result of resistance of denial of service attacks in Meng et al. protocol is presented in Fig. 13. The result we obtained is that Meng et al. protocol is not resistance of denial of service attacks because ProVerif outputs the message "$Secret$" through public channel c.

In Meng et al. protocol there is a denial of service attack found by us: in preparation phase registration authority publishes public keys ${}_{ElG}PU_R$, ${}_{BCP}PU_R$ on bulletin board without any security of these public keys by public channels. Thus the adversary can intercept public keys ${}_{ElG}PU_R$, ${}_{BCP}PU_R$ and modify it, then send it to bulletin board. In voting phrase voter v_j verifies Bc with $checksign(signed, ELG\_PK\_\mathrm{Re}g,(alphabet, Bc))$ and ${}_{ElG}PU_R$ according to MW deniable encryption scheme. Owning the adversary has modified the public keys ${}_{ElG}PU_R$, ${}_{BCP}PU_R$, hence the verification is not success, thus voter v_j can not vote. According to the specification, the voter v_j repeats several times, but he still can not vote. So adversary can make a denial of service attack. In order to protect Meng et al. protocol against the denial of service attack, we can use the digital certificate to distribute public keys ${}_{ElG}PU_R$, ${}_{BCP}PU_R$.

```
data true/0.
data ELG/0.
data BCP/0.

fun ELG_enc/3.
fun ELG_dec/2.

fun BCP_enc/2.
fun BCP_dec/2.

fun sign/2.
fun checksign/3.
fun PK/2.
fun PR/2.

fun mod/3.
fun mod_inverse/4.

fun multi/2.

equation mod_inverse(mod(a,g,multi(p,q)),g,p,q)=a.

equation checksign(sign(x,PR(y,ELG)),PK(y,ELG),x)=true.
equation ELG_dec(ELG_enc(x,PK(y,ELG),r),PR(y,ELG))=x.
equation BCP_dec(BCP_enc(x,PK(y,BCP)),PR(y,BCP))=x.

free pub,com.
private free chvote.
free chVR1.
private free chVR2.
free chVII.
private free chIIR.
private free chTR.

free va,vb.

free n1,n2.

private free secret.

query attacker:secret.
let votechooser =
        out(chvote,va) | out(chvote,vb).

let voter=
        new g;new p;new q;new nonceh;
        let N=multi(p,q) in
        out(chVR1,(n1,nonceh,N,g));
        in(chVR1,(=n2,=nonceh,h,elg_pk_reg));
        let a=mod_inverse(h,g,p,q) in
        let ELG_BCP_PK_voter=PK(a,ELG) in
        let BCP_PK_voter=PK(a,BCP) in
        let ELG_BCP_PR_voter=PR(a,ELG) in
        let BCP_PR_voter=PK(a,BCP) in
        let ELG_BCP_PK_Reg=PK(a,ELG) in
        new nonce;
        new nonce1;
        out(chVII,(n1,nonce1));
        in(chVII,(=n2,=nonce1,id));
        out(chVR2,(n1,nonce,id,ELG_PK_voter,sign(id,ELG_PR_voter)));
        in(chVR2,(=n2,=nonce,alphabet,Bc,signed));
        if checksign(signed,elg_pk_reg,(alphabet,Bc))=true then
        (
        let (r,cred)=BCP_dec(Bc,BCP_PR_voter) in
        in(chvote,vote);
        new r1;new r2;
        let Bc1=BCP_enc((r1,cred),BCP_PK_Tal) in
        let Bb=BCP_enc((r2,vote),BCP_PK_Tal) in
        out(com,(Bc1,Bb))
        )
            else out(pub,secret).

let corruptedvoter=
        new nonce;
        new nonce1;
        out(chVII,(n1,nonce1));
        in(chVII,(=n2,=nonce1,id));
        out(pub,id).

let Identity_Issue_Authority=
        in(chVII,(=n1,nonceV));
        in(chIIR,(=n1,nonceR));
        new id;
        out(chVII,(n2,nonceV,id));
        out(chIIR,(n2,nonceR,id));
        out(pub,id).

let Registration_Authority=
        in(chVR1,(=n1,nonceh,N,g));
        new a;
        let h=mod(a,g,N) in
        out(chVR1,(n2,nonceh,h,ELG_PK_Reg));
        let ELG_BCP_PK_voter=PK(a,ELG) in
        let BCP_PK_voter=PK(a,BCP) in
        (
        in(chTR,(=n1,nonceT));
        new nonce;
        out(chIIR,(n1,nonce));
        in(chIIR,(=n2,=nonce,id));
        in(chVR2,(=n1,nonceV,=id,pk,signed));
        if checksign(signed,pk,id)=true then
        new r1;new r2;new cred;new r3;new r4;
        let alphabet=ELG_enc((r1,r2),ELG_BCP_PK_voter,r4) in
        let Bc=BCP_enc((r2,cred),BCP_PK_voter) in
        out(chVR2,(n2,nonceV,alphabet,Bc,sign((alphabet,Bc),ELG_PR_Reg)));
        let enccred=ELG_enc(cred,ELG_PK_Reg,r3) in
        out(chTR,(n2,nonceT,enccred,sign(enccred,ELG_PR_Reg)));
        out(pub,(enccred,sign(enccred,ELG_PR_Reg))))
        |in(chTR,(=n1,nonceT,enccred1));
        let cred1=ELG_dec(enccred1,ELG_PR_Reg) in
        out(chTR,(n2,nonceT,cred1)).

let Tallying_Authority=
        new nonce;
        out(chTR,(n1,nonce));
        in(chTR,(=n2,=nonce,enccred,signed));
        in(com,result);
        let (Bc,Bb)=result in
        let (r1,cred)=BCP_dec(Bc,BCP_PR_Tal) in
        let (r2,vote)=BCP_dec(Bb,BCP_PR_Tal) in
        if checksign(signed,ELG_PK_Reg,enccred)=true then
        new nonceT;
        out(chTR,(n1,nonceT,enccred));
        in(chTR,(=n2,=nonceT,cred1));
        if cred=cred1 then
        out(pub,vote).

process new voter;new reg; new tal;
        let ELG_PK_voter=PK(voter,ELG) in
        let ELG_PR_voter=PR(voter,ELG) in
        let ELG_PK_Reg=PK(reg,ELG) in
        let ELG_PR_Reg=PR(reg,ELG) in
        let BCP_PK_Tal=PK(tal,BCP) in
        let BCP_PR_Tal=PR(tal,BCP) in
        out(pub,ELG_PK_voter);
        out(pub,ELG_PK_Reg);
        out(pub,BCP_PK_Tal);
        ((!voter)|(!Identity_Issue_Authority)|(!Tallying_Authority)|
(!Registration_Authority)|(!votechooser)|(!corruptedvoter)).
```

Figure 12. The code in analysis of resistance of denial of service attacks in voter in Meng et al. protocol



```
C:\WINDOWS\system32\cmd.exe

in(chVR1, (n2,nonceh_70_41,a_26,a_25)) at (57) in copy a_27

out(chVII, (n1,nonce1_80_40)) at (64) in copy a_27

in(chVII, (n1,a_33)) at (50) in copy a_34

in(chVR1, (n1,a_31,a_30,a_29)) at (18) in copy a_32

out(chVR1, (n2,a_31,mod(a_30_44,a_29,a_30),PK(reg_16_47,ELG()))) at (20) in copy
 a_32

out(chVII, (n2,a_33,id_65_37)) at (52) in copy a_34

in(chVII, (n2,nonce1_80_40,id_65_37)) at (65) in copy a_27

out(pub, id_65_37) at (54) in copy a_34

out(pub, secret) at (69) in copy a_27

The attacker has the message secret.
A trace has been found.
RESULT not attacker:secret[] is false.

E:\形式化\proverifbsd\proverif1.84pl2>_
```

Figure 13. The result of resistance of denial of service attacks in Meng et al. protocol

VI. ACKNOWLEDGEMENT

This study was supported in part by Natural Science Foundation of The state Ethnic Affairs Commission of PRC under the grants No: 10ZN09.

VII. CONCLUSION AND FUTURE WORK

Internet voting protocols play an important role in remote voting system. Meng et al. protocol is one of the most important remote internet voting protocols that claims to satisfy formal definitions of key properties without strong physical constrains. To our best knowledge until now its resistance of denial of service attacks has not been analyzed.

Recently owning to the contribution of Huang et al, Meng et al. protocol can be proved with mechanized proof tool ProVerif. In this paper the review of formal model of resistance of denial of service attacks in security protocols is presented. There are mainly three formal frameworks in resistance of denial of service attacks: Yu-Gligor model, Meadows's model and Huang et al. model. Until now resistance of denial of service attacks model based on computational model has not been proposed. Then apply the mechanized formal model proposed by Huang et al. model for mechanized proof of resistance of denial of service attacks. The result is that Meng et al. protocol is not resistance of denial of service attacks because one denial of service attack is found by us. At the same time we also propose the method to prevent it from the denial of service attack. To our best knowledge, we are conducting the first mechanized proof of resistance of denial of service attacks in Meng et al. protocol for an unbounded number of honest and corrupted voters.

As future work, it would be interesting to formalize the security properties of remote internet voting protocols in computational model with mechanized tool CryptoVerif.

REFERENCES

[1] B.Meng, "A critical review of receipt-freeness and coercion-resistance," *Information Technology Journal*, vol.8, pp.34-964, 2009.

[2] B.Meng, "A Survey on Analysis of Selected Cryptographic Primitives and Security Protocols in Symbolic Model and Computational Model," *Information Technology Journal*, vol.10, pp. 1068-1091, 2011.

[3] M.R.Clarkson, S. Chong and A.C. Myers, "Civitas: Toward a secure voting system," *In Proceeding of the 2008 IEEE Symposium on Security and Privacy*,18-21 May 2008, Oakland, California, USA 2008 pp. 354-368, 2008.

[4] B.Meng, "An Internet Voting Protocol with Receipt-free and Coercion- resistant," *In Proceeding of IEEE 7th International Conference on Computer and Information Technology*, October 16-19, 2007,University of Aizu, Fukushima, Japan,pp.721-726,2007.

[5] B.Meng, "A Secure Internet Voting Protocol Based on Non-interactive Deniable Authentication Protocol and Proof Protocol that Two Ciphertexts are Encryption of the Same Plaintext," *Journal of Networks*, vol.4,pp. 370-377,2009.

[6] B.Meng, "A Secure Non-Interactive Deniable Authentication Protocol with Strong Deniability Based on Discrete Logarithm Problem and its Application on Internet Voting Protocol," *Information Technology Journal*, vol.8,pp.302-309,2009.

[7] B.Meng, Z.M. Li, and J. Qin, "A receipt-free coercion-resistant remote internet voting protocol without physical assumptions through deniable encryption and trapdoor commitment scheme," Journal of Software, vol.5, pp. 942-949, 2010.

[8] B.Meng and J.Q. Wang, "An efficient receiver deniable encryption scheme and its applications," Journal of Networks, vol.5, pp.683-690, 2010.

[9] B.Blanchet, "An Efficient Cryptographic Protocol Verifier Based on Prolog Rules," *In 14th IEEE Computer Security Foundations Workshop*, Cape Breton, Nova Scotia, Canada, June 2001. pp. 82-96, 2001.

[10] C.Yu and V.Gligor, "A Formal Specification and Verification Method for the Prevention of Denial of Service," *IEEE Transactions on Software Engineering*, vol.16, pp.581-592, 1990.

[11] C.Meadows, "A cost-based framework for analysis of denial of service networks," *Journal of Computer Security*, vol.9, pp.143-164, 2001.

[12] W.Huang, B.Meng, D.J.Wang, "Automatic proof resistance of denial of service attacks in protocols," Journal on communication, unpublished

[13] V. D.Gligor, "A note on the denial-of-service problem," *In Proceeding of IEEE Symposium on Security and Privacy*, Oakland, Ca, 1983.

[14] V. D.Gligor, "A note on denial-of-service in operating systems," *IEEE Transactions on Software Engineering*, Vol.10, pp.320-324, 1984.

[15] E.Bacic and M.Kuchta, "Considerations in the Preparation of a Set of Availability Criteria," *In Proceeding of the Third Annual Canadian Computer Security Symposium*, Ottawa, Canada, May 1991.

[16] J.K.Millen, "A Resource Allocation Model for Denial of Service Protection," *Journal of Computer Security*, vol.2, pp.89-106, 1993.

[17] V.Ramachandran, "Analyzing DoS-resistance of protocols using a cost-based framework," *Technical Report DCS/TR-1239*, Yale University, 2002.

[18] J.Smith, J.M.Gonzalez-Nieto, and C.Boyd, "Modelling denial of service attacks on JFK with Meadows's cost-based framework," *In Proceedings of the 2006 Australasian workshops on Grid computing and e-research (ACSW Frontiers)*. Darlinghurst, Australia, 2006, pp.125-134, 2006.

[19] S.Lafrance and J.Mullins, "Using Admissible Interference to Detect Denial of Service Vulnerabilities," *In Proceedings of the Sixth International Workshop in*

Formal Methods (IWFM), Dublin City University, Ireland, 2003.

[20] M.Abadi, B.Blanchet, and C Fournet, "Just Fast Keying in the Pi Calculus," *ACM Transactions on Information and System Security*, vol.10,pp.1-59,2007.

[21] S.Tritilanunt, C. Boyd, E. Foo, and J. M.González Nieto, "Cost-based and time-based analysis of DoS-resistance in HIP," *In Proceedings of the thirtieth Australasian conference on Computer science (ACSC '07),* Darlinghurst, Australia, 2007,pp.191-200,2007.

[22] S.Tritilanunt, "Protocol engineering for protection against denial of service attacks," *Doctor Thesis,* Brisbane Australia .Queensland University of Technology, 2009.

[23] S.J.Zhou, R..Jing, and X. H.Yang, "DoS attacks on security protocols of the formal analysis," *Journal of Research Institute of China Electronics*, vol.3, pp.592-598, 2008.

[24] E.Amoroso, "A policy model for denial of service," *In Proceedings of Third IEEE Computer Security Foundations Workshop*, Franconia, NH, pp.110-114,1990.

[25] F.Cuppens, C. Saurel, "Towards a formalization of availability and denial of service," *In Proceedings of in Information Systems Technology Panel Symposium on Protecting NATO Information Systems in the 21st Century*. Washington, 1999

[26] F.Cuppens, N.Cuppens-Boulahia and T.Ramard, "Availability Enforcement by Obligations and Aspects Identification," *In Proceedings of the First International Conference on Availability, Reliability and Security (ARES '06)*, Vienna Austria, pp.229-239, 2006.

[27] G. Agha, C. Gunter, M.B. Greenwald, S. Khanna, J. Meseguer, K. Sen, and P. Thati, "Formal Modeling and Analysis of DoS Using Probabilistic Rewrite Theories," *In Proceedings of International Workshop on Foundations of Computer Security*, Chicago IL, pp. 91-102 ,2005.

[28] A.Mahimkar, V.Shmatikov, "Game-Based Analysis of Denial-of-Service Prevention Protocols,"*In Proceedings of the 18th IEEE workshop on Computer Security Foundations,* Aix-en-Provence, France, pp.287-301, 2005.

[29] M.Abadi and B. Blanchet, "Analyzing security protocols with secrecy types and logic programs," *Journal of the ACM,* vol.52, pp.102–146, 2005.

Bo Meng was born in 1974 in China. He received his M.S. degree in computer science and technology, Ph.D. degree in traffic information engineering and control from Wuhan University of Technology, at Wuhan, China, in 2000, 2003, respectively. From 2004 to 2006, he works in Wuhan University, China as Postdoctoral researcher in information security.

Currently he is an Associate Professor in school of computer, South-Center University for Nationalities, China. He has authored/coauthored over 50 papers in International/National journals and conferences. His current research interests include formal method, Internet voting, and protocol security, software verification.

Wei Wang was born in 1983 in China. He received his BA degree in computer science and technology in 2009 from South-Center University for Nationalities, China.

Currently he is a postgraduate student in college of computer science, South-Central University For Nationalities, at Wuhan, China. His current research interests include information security, formal method.

Wei Chen was born in 1989 in China. He received his BA degree in computer science and technology in 2011 from South-Center University for Nationalities, China.

Currently he is a postgraduate student in college of computer science, South-Central University For Nationalities, at Wuhan, China. His current research interests include formal method, and software verification.

.