http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 10 (4): 717-735, 2011
ISSN 1812-5638 / DOI: 10.3923/itj.2011.717.735

Automatic Verification of Deniable Authentication Protocol in a Probabilistic Polynomial Calculus with Cryptoverif

Bo Meng
School of Computer, South-Center University for Nationalities,
MinYuan Road #708, HongShan Section, Wuhan, Hubei 430074, China

Abstract: During the past few decades a lot of deniable authentication protocols have been proposed which claimed that have the security properties, for example, deniability. To our knowledge, these security properties and deniable authentication protocols are analyzed with informal method or with symbolic method by hand which is prone to make mistakes. So analysis of security properties and deniable authentication protocols with automatic tool in symbolic model or computational model play an important role in security protocol world. Especially analysis with automatic tool in computational model is a changeling issue. Owning to the contribution of Meng and Shao, Meng non-interactive deniable authentication protocol can be analyzed with automatic tool in computational model. In this study firstly the state-of-art of deniable authentication protocol and the proof included in symbolic model and in computational model are presented. Then the term, process and correspondence assertion in Blanchet calculus is used to model security properties included deniability and Meng non-interactive deniable authentication protocol with Meng and Shao mechanized framework of deniable authentication protocol in computational model with active adversary. Finally Meng non-interactive deniable authentication protocol is analyzed in the computational model and Meng and Shao framework with CryptoVerif. To our knowledge, we are conducting the first automatic analysis in computational model of Meng non-interactive deniable authentication protocol. The results show that Meng non-interactive deniable authentication protocol has weak deniability and strong deniability.

Key words: Strong deniability, weak deniability, mechanized proof, computational model, deniable authentication model

INTRODUCTION

With the development of Internet technology, many transactions are carried out through the Internet. Most of them can not be finished just by face-to-face approach. A key problem of how to authenticate the involved parties identities emerges subsequently. Therefore, many authentication protocols have been presented during the past few decades. Authentication protocol based on cryptographic technologies is used to confirm the identities of parties in the communication (Meng, 2009a).

However, some specified Internet applications such as electronic voting and electronic business, require deniable authentication protocols. Deniable authentication protocol allows a sender to authenticate a message for a receiver, in a way that the receiver can not convince a third party that such authentication (or any authentication) ever took place. Deniable authentication protocol has two characteristics that differ from traditional authentication protocol. One is that only the intended receiver can authenticate the true source of a given message. The other is that the sender can not provide the evidences to prove the source of the message to a third party in some condition and the receiver can provide the evidences to prove the source of the message to a third party (Raimondo and Gennaro, 2005).

The practical secure deniable authentication protocol should have the following properties: completeness or authentication; strong deniability (Raimondo and Gennaro, 2005); weak deniability (Raimondo and Gennaro, 2005); security of forgery attack (Shao, 2004); security of impersonate attack (Shao, 2004); security of compromising session secret attack (Lee *et al*., 2007); security of man-in-the-middle attack (Han *et al*., 2005). These secure properties play important roles in implementation of secure transactions over the public Internet.

During the past few decades deniable authentication protocol has been studied. A lot of deniable authentication protocols (Aumann and Rabin, 1998; Dwork *et al*., 1998; Deng *et al*., 2001; Fan *et al*., 2002; Raimondo and Gennaro, 2005; Han *et al*., 2005; Feng and Ma, 2007; Shao, 2004; Lu and Cao, 2005a, b; Qian *et al*., 2005; Shi and Li, 2005; Lee *et al*., 2007; Meng, 2009a) have been proposed.

In order to verify the security properties of security protocol include deniable authentication protocols and

increase the confidence of the people, two approaches have been developed for analyzing security protocols form the beginning of the 1980s. One approach relies on a symbolic model of protocol executions in which cryptographic primitives are treated as black boxes. Since, the seminal work of Dolev and Yao, it has been realized that this approach enables significantly simpler and many automatic tools have been developed. For example, SMV (McMillan, 1992), NRL (Meadows, 1996), Casper (Lowe, 1998), Isabelle (Paulson, 1998), Athena (Song, 1999), Revere (Kindred, 1999), SPIN (Maggi and Sisto, 2002), Brutus (Clarke *et al*., 2000), ProVerif (Blanchet, 2001), Scyther (Joseph and Cremers, 2006). In formal community, recently, Meng (2009b) proposed a framework of strong and weak deniability based on Kessler and Neumann logic. After that, the framework is applied to analyze the deniability of two typical deniable authentication protocols: Fan *et al*. (2002) interactive deniable authentication protocol (Fan *et al*., 2002) and Meng non-interactive deniable authentication protocol (Meng, 2009a). But the result of proof based on symbolic model is not quite clear.

The other approach relies on a computational model that base issues of complexity and probability. This approach use a strong notion of security, guaranteed against all probabilistic polynomial-time attacks. The computation approach can be more realistic, but it is difficult to automatic proof until the introduction of mechanized tool CryptoVerif (Blanchet, 2005, 2006, 2007a, b) which is the only automatic tool with computational model. Blanchet (2005, 2006, 2007a, b) proposed a probabilistic polynomial calculus based on computational model. In this calculus, messages are bitstrings and cryptographic primitives are functions operating on bitstrings. Blanchet calculus is adapted from the pi calculus and its semantics is purely probabilistic (no non-determinism). All processes run in polynomial time: polynomial number of copies of processes and length of messages on channels bounded by polynomials. Blanchet calculus has been carefully designed to make the automated proof security protocols. At the same time they develop a mechanized tool CryptoVerif (Blanchet, 2005, 2006, 2007a, b) which is the only automatic tool with computational model until now. CryptoVerif does not rely on soundness results for symbolic model but directly automate the proofs made in cryptography, based on sequences of games. It can directly prove security properties of cryptographic protocols in the computational model in which the cryptographic primitives are functions on bit-strings and the adversary is a polynomial-time Turing machine. It can proves secrecy properties and events that can be executed only with negligible probability, also it can handles various cryptographic primitives, for example, MACs, stream and block ciphers, public-key encryption, signatures, hash functions. CryptoVerif works for N sessions with an active adversary. It can also give a bound on the probability of an attack (exact security). CryptoVerif runs either automatically or interactively, in which case it receives guidance from the user for selecting transformations. CryptoVerif has been used to verify: FDH signature scheme (Blanchet and Pointcheval, 2006) PKINIT for Kerberos (Jaggard *et al*., 2007), Verification Protocol Implementations in ML (Bhargavan *et al*., 2007), a model of the Basic and Public-Key Kerberos protocol (Blanchet *et al*., 2008),Verification Protocol Implementations for TLS (Bhargavan *et al*., 2008), Diffie-hellman protocol (Blanchet, 2009). Recently, Meng and Shao (2010) proposed the first automatic framework of strong and weak deniability based on Blanchet calculus in computational model. Strong deniability and weak deniability of the deniable authentication protocols can be automatically verified with CryptoVerif.

To our best knowledge, this security properties and deniable authentication protocols are analyzed with informal method, or with symbolic method by hand (Meng, 2009b), which depends on experts' knowledge and skill and is prone to make mistakes. So analysis of security properties and deniable authentication protocols with automatic tool in symbolic model or computational model play an important role in security protocol world and is a significant works. Especially analysis of security properties and deniable authentication protocols with automatic tool in computational model is a changeling issue.

Inspired by the works of Blanchet and owning to the introduction of the powerful mechanized tool CryptoVerif which is the only automatic tool in computational model are have been successfully used to verify several cryptographic primitive and security protocols, at the same time owning to Meng non-interactive deniable authentication protocol is analyzed with informal method and its result is not clear, based on Meng and Shao (2010), in this study, we use Blanchet calculus in the computational model to mechanized analyze Meng non-interactive deniable authentication protocol (Meng, 2009a) with mechanized tool CryptoVerif.

CONTRIBUTION AND OVERVIEW

During the past few decades deniable authentication protocol has been studied. A lot of deniable authentication protocols has been proposed which claimed that have the security properties, for example, authentication; strong deniability; weak deniability;

security of forgery attack; security of impersonate attack; security of compromising session secret attack; security of man-in-the-middle attack and so on. In order to verify the security properties of security protocol include deniable authentication protocols and increase the confidence of the people, two approaches have been developed for analyzing security protocols form the beginning of the 1980s. One approach relies on a symbolic model of protocol executions in which cryptographic primitives are treated as black boxes. The other approach relies on a computational model that base issues of complexity and probability. This approach use a strong notion of security, guaranteed against all probabilistic polynomial-time attacks. The computation approach can be more realistic (Meng and Shao, 2010). To our best knowledge, this security properties and deniable authentication protocols are analyzed with informal method, or with symbolic method by hand (Meng, 2009b), which depends on experts' knowledge and skill and is prone to make mistakes.

So analysis of security properties and deniable authentication protocols with automatic tool in symbolic model or computational model play an important role in security protocol world and is a significant works. Especially analysis of security properties and deniable authentication protocols with automatic tool in computational model is a changeling issue.

Owning to the introduction of a probabilistic polynomial calculus proposed by Blanchet (2005, 2006, 2007a, b) and a powerful mechanized tool CryptoVerif (Blanchet, 2005, 2006, 2007a, b) which is the only automatic tool in computational model are have been successfully used to verify several cryptographic primitive and security protocols. Recently Meng and Shao (2010) proposed the first automatic framework of strong deniability and weak deniability based on Blanchet calculus in computational model. Inspired by the works of Blanchet and owning to the introduction of the powerful mechanized tool CryptoVerif, at the same time owning to Meng non-interactive deniable authentication protocol is analyzed with informal method and its result is not clear, based on Meng and Shao (2010), in this study, we use Blanchet calculus in the computational model to mechanized analyze Meng non-interactive deniable authentication protocol with mechanized tool CryptoVerif. The main contributions of this paper are summarized as follows in detail:

- Firstly the state-of-art of deniable authentication protocol and the proof included in symbolic model and in computational model are presented. We found that security properties and deniable authentication protocols are analyzed with informal method or with symbolic method by hand. There does not exist that analysis of security properties and deniable authentication protocols with automatic tool until Meng and Shao (2010) proposed the first automatic framework of strong and weak deniability based on Blanchet calculus in computational model.
- Applying Meng and Shao (2010) automatic model based on Blanchet calculus in computational model with active adversary for automatically analysis of Meng non-interactive deniable authentication protocol. So the term, process and correspondence assertion in Blanchet calculus is used to model security properties included deniability and Meng non-interactive deniable authentication protocol. The strong deniability and weak deniability are expressed by non-injective or injective correspondence. The analysis itself is performed by automatic tool CryptoVerif developed by Blanchet. Figure 1 describe the analysis model of deniable authentication protocols with Blanchet calculus
- Finally Meng non-interactive deniable authentication protocol is analyzed in the computational model and Meng and Shao 2010 automatic framework with mechanized tool CryptoVerif. Meng non-deniable authentication protocol (Meng, 2009a) is a secure non-interactive deniable authentication protocol based on discrete logarithm problem. It is secure and has properties: completeness, strong deniability, weak deniability, security of forgery attack, security of impersonate attack, security of compromising session secret attack and security of man-in-the-middle attack. The result shows that Meng non-interactive deniable authentication protocol has weak deniability and strong deniability, which is consist to

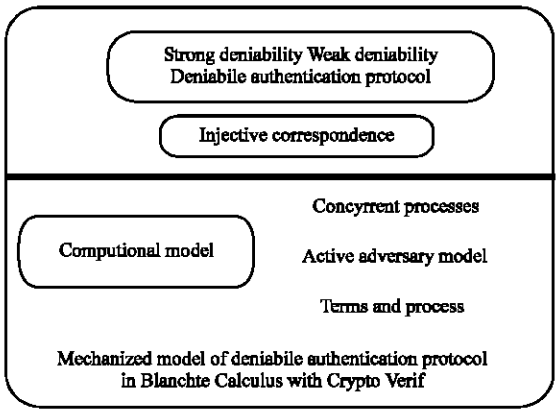

Fig. 1: Analysis model of deniable authentication protocols with Blanchet calculus

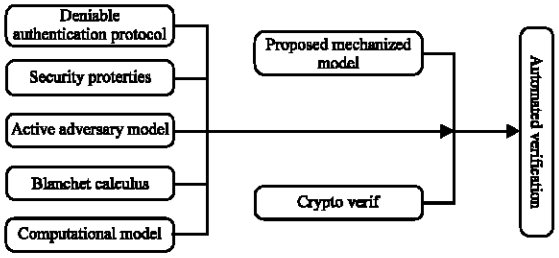

Fig. 2: Model of automatic verification of deniable authentication protocols

the claim in its paper. To our best knowledge, we are conducting the first automatic analysis in computational model of Meng non-interactive deniable authentication protocol in active adversary. Fig. 2 shows the model of automatic verification of Meng non-deniable authentication protocol.

Deniable authentication protocols allow a sender to authenticate a message for a receiver, in a way that the receiver can not convince a third party that such authentication ever took place. Deniable authentication has two characteristics that differ from traditional authentication: one is that only the intended receiver can authenticate the true source of a given message; the other is that the receiver can not provide the evidences to prove the source of the message to a third party.

A practical secure deniable authentication protocol should have the following properties: Completeness or authentication; Strong deniability (Raimondo and Gennaro, 2005), Weak deniability (Raimondo and Gennaro, 2005) Security of forgery attack (Shao, 2004); Security of impersonate attack (Shao, 2004); Security of compromising session secret attack (Lee *et al*., 2007) Security of man-in-the-middle attack (Han *et al*., 2005).

Dwork *et al*. (1998) proposed an interactive deniable authentication protocol based on the concurrent zero-knowledge proof. Aumann and Rabin (1998) proposed an interactive deniable authentication protocol based on factoring problem. Deng *et al*. (2001) proposed two interactive deniable authentication protocols based on factoring and the discrete logarithm problem, respectively. Fan *et al*. (2002) proposed another simple interactive deniable authentication protocol based on the Diffie-Hellman key distribution protocol. Han *et al*. (2005) proposed an interactive deniable authentication protocol resisting man-in-the-middle attack based on Diffie-Hellman key exchange protocol. Feng and Ma (2007) proposed a concurrent deniable authentication based on witness indistinguishable, which can support strong deniability.

The interactive deniable authentication protocols are inefficient. Hence, several non-interactive deniable authentication protocols are proposed. Shao (2004) pointed out there are three weakness in paper (Fan *et al*., 2002) and give an improved a generalized ElGamal signature scheme. Lu and Cao (2005a, b) proposed a non-interactive deniable authentication protocol based on bilinear pairings and factoring, respectively. Lee *et al*. (2007) pointed that protocols (Shao, 2004; Lu and Cao, 2005a, b) can not protect against compromising session secret attack and introduce a new deniable authentication protocol using generalized ElGamal signature scheme. But these non-interactive deniable authentication protocols have not strong deniability. Meng (2009a) proposed a secure non-interactive deniable authentication protocol based on discrete logarithm problem is developed. At the same time we prove that the proposed protocol has properties: completeness, strong deniability, weak deniability, security of forgery attack, security of impersonate attack, security of compromising session secret attack and security of man-in-the-middle attack.

In order to increase the confidence of the people, two approaches: symbolic model and computational model can be used to verify the security properties of deniable authentication protocols.

In formal community Meng (2009b) proposed a framework of strong and weak deniability based on Kessler and Neumann logic is proposed. After that, the framework is applied to analyze the deniability of two typical deniable authentication protocols: Fan *et al*. interactive deniable authentication protocol and Meng's non-interactive deniable authentication protocol.

In computational model, Meng and Shao (2010) proposed the first model of strong deniability and weak deniability with Blanchet calculus. Blanchet (2004, 2005, 2006, 2007a) proposed a probabilistic polynomial calculus based on computational model. In this calculus, messages are bitstrings and cryptographic primitives are functions operating on bitstrings. Blanchet calculus is adapted from the pi calculus and its semantics is purely probabilistic.All processes run in polynomial time: polynomial number of copies of processes and length of messages on channels bounded by polynomials. Blanchet calculus has been carefully designed to make the automated proof security protocols. Blanchet calculus consists of terms and processes. At the same time they develop a mechanized tool CryptoVerif (Blanchet, 2005, 2006, 2007a, b) with computational model. CryptoVerif does not rely on soundness results for symbolic model but directly automate the proofs made in cryptography, based on sequences of games. It can directly prove security properties of cryptographic protocols in the

computational model in which the cryptographic primitives are functions on bit-strings and the adversary is a polynomial-time Turing machine. It can proves secrecy properties and that events can be executed only with negligible probability, also it can handles various cryptographic primitives, for example, MACs, stream and block ciphers, public-key encryption, signatures, hash functions. CryptoVerif works for N sessions with an active adversary. It can also give a bound on the probability of an attack (exact security). CryptoVerif runs either automatically or interactively, in which case it receives guidance from the user for selecting transformations. In a recent case study, CryptoVerif is used to verify: FDH signature scheme (Blanchet and Pointcheval, 2006) PKINIT for Kerberos (Jaggard *et al*., 2007), Verification Protocol Implementations in ML (Bhargavan *et al*., 2007), a model of the Basic and Public-Key Kerberos protocol (Blanchet *et al*., 2008), Verification Protocol Implementations for TLS (Bhargavan *et al*., 2008), Diffie-hellman protocol (Blanchet, 2009).

Inspired by the works of Blanchet, to our best knowledge, deniable authentication protocols have not been analyzed in computational model, in this study, based on Meng and Shao (2010), we use Blanchet calculus in the computational model to mechanized analyze the typical deniable authentication protocols with and CryptoVerif.

MENG AND SHAO MODEL OF DENIABILITY

Meng and Shao (2010) described strong deniability and weak deniability. Meng and Shao model use Blanchet calculus to model the strong deniability and weak deniability.

Generally, deniable authentication protocol includes three roles, sender which is initiator, receiver which is responder and third party, represented by sender, receiver and thirdparty, respectively. We assume that sender plays only on the role of the initiator, receiver plays only the role of responder, thirdparty play only on the prover. The deniable authentication protocol consists of a sequence of messages exchanged between the sender and the receiver and the receiver and thirdparty and sender and thirdparty. In deniable authentication protocol sender can authenticate a message for receiver, in a way that the receiver can not convince a thirdparty that such authentication (or any authentication) ever took place. Deniable authentication protocol has two characteristics that differ from traditional authentication protocol. One is that only the intended receiver can authenticate the true source of a given message. The other is that the sender can not provide the evidences to prove the source of the message to a third party at some condition and the receiver can provide the evidences to prove the source of the message to a third party. The ability of adversary is defined in the previous section. It can control the channel channelSR between sender and receiver. It can not control the channels channelSR and channelRT. At the same time the adversary is a probabilistic polynomial-time attacker.

In Meng and Shao automatic model, it assume that shared-key encryption is modeled as encrypt-then-MAC, where the encryption is indistinguishability under chosen plaintext attacks and the MAC is enforceability under chosen message attacks; public-key encryption is assumed to be indistinguishability under adaptive chosen ciphertext attacks; signatures are assumed to be unforgeability under chosen message attacks. CryptoVerif has the decisional Diffie-Hellman assumption and Computational Diffie-Hellman assumption.

Definition DAP: A secure deniable authentication protocol with session functions sessionid and sessionid' process DAP for any probabilistic polynomial-time adversary:

$$\text{DAP} = \text{Initprosess};\ (!^{i\text{sender}\le n}\ \text{SenderProcess} \mid !^{i\text{receiver}\le n}\ \text{ReceiverProcess} \mid !^{i\text{thirdparty}\le n}\ \text{ThirdpartyProcess})$$

Such that:

- If the adversary just send receiver to sender process^{i} as the first message and relays faithfully between process sender process^{i} and process receiver process$^{i'}$, then process receiver process$^{i'}$ finishes with sender and process sender process^{i} finishes with receiver.
- With overwhelming probability, there exists an injective function that maps each index i of a process sender process^{i} that finished with receiver to the index i' of a process receiver process$^{i'}$ with intended principle sender such that sessioner

$$\text{sessioner}'\left(x_1[i], x_2[i], \cdots, x_1[i]\right) = \text{sessioner}'\left(z_1[i'], z_2[i'], \cdots, z_1[i']\right)$$

- With overwhelming probability, there exists an injective function that maps each index i of a process Receiver process$^{i'}$ that finished with sender to the index i' of a process sender process$^{I'}$ that finished with recevier such that sessioner

$$\text{sessioner}'\left(x_1[i], x_2[i], \cdots, x_1[i]\right) = \text{sessioner}'\left(z_1[i'], z_2[i'], \cdots, z_1[i']\right).$$

- If the adversary just send thirdparty to Receiver process^{i} as the first message and relays faithfully between thirdparty process^{i} and receiver process$^{'i}$; then thirdparty process finishes with receiver and receiver process$^{i''}$ finishes with thirdparty.
- With overwhelming probability, there exists an injective function that maps each index i of a process thirdparty process that finishes with receiver to the index i' of a process receiver process$^{i''}$ finishes with thirdparty such that sessioner sessioner' ($p^{i}[i]$ = sessioner' (q^{i} [i])

In the above definition of DAP the injective correspondence can be instead by non-injective correspondence.

In deniable authentication protocol DAP, the first several messages between sender and receiver are executed, then the last message are send to the thirdparty by the receiver.

It assume that DAP consists of odd number of rounds l, so that the t1h message and 1-1th message of DAP is from sender to receiver. The 1th message is from receiver to thirdparty. Here we only consider the only one message from receiver to thirdparty.

Initprosess process is an initialization process which responsible for generating the relative parameters for sender and receiver, for example, the random numbers, public and private keys and so on. sender process and receiver process do not contain replication. sender process process stores the messages of the deniable authentication protocol in variables $x_1, x_2, \ldots, x_i$. receiver process process stores the message from sender process process in variable $z_1, z_2, ., z_1$ and stores the message to thirdparty process in variable p_1. Thirdparty process process stores it in variable q_1. The process sender process is invoked by a receiving message or external request which contains the identity receiver of process receiver process with sender is supposed to run a session. After that the process receiver process is invoked by a receiving message or external request which contains the identity thirdparty of process thirdparty process with thirdparty is supposed to run a session. We designate by sender process^{i} the copy of sender process of index isender = i and by receiver process$^{i''}$ the copy of receiver process of ireceiver = i and by thirdparty process the copy of receiver process of ithirdparty = i. The 1-1th message is sent by sender process^{i} process. The receiver process$^{i''}$ process is assumed that it send the additional message one message include either $OK_{Receiver}$ (sender) or reject after receiver process^{i} process received and checked the 1-1th message of DAP protocol, where receiver is the identity of its intended principle. We say that receiver process^{i} finished with sender when it sends $Ok_{receiver}$ (sender) in the additional message two message.

In Meng and Shao automatic model the session function sessioner based on is used to match the conversion in the deniable authentication protocol. The session function is chosen to contain all messages of the protocol, except messages that are sent to or received from a trust third server. A sessioneris a function of messages in the protocol; sessioner ($x_1, x_2, \ldots, x_1$) is typically a subsequence of the whole sequence messages $x_1, x_2, \ldots, x_1$. A partial session function sessioner ($x_1, x_2, \ldots, x_1$) is also defined, useful since the 1-th message is not available to receiver when sender accepts. At the same time a partial session function sessioner' (θ) is also defined, useful since the 1th message is not available to thirdparty when receiver sends. It requires that sessioner ($x_1, x_2, \ldots, x_1$) = sessioner ($z_1, z_2, \ldots, z_1$) implies sessioner ($x_1, x_2, \ldots, x_{1-1}$) = sessioner ($z_1, z_2, \ldots, z_{1-1}$). sender process and receiver process$_i$ are intended principle when they have the same session function:

$$\text{sessioner}\left(x_1[i], x_2[i], \cdots, x_1[i]\right) = \text{sessioner}\left(z_1[i], z_2[i], \cdots, z_1[i]\right)$$

and receiver process and thirdparty process are intended principles when they have the same session function: sessioner ($z_1[i]$) = ($q_1[i]$).

The condition one describes the communications between sender and receiver without adversary. It deal with receiver authenticate sender. The condition two and three describe that sender has a distinct session with receiver and receiver has the same session with sender with overwhelming probability.

The condition four describes the communications between receiver and thirdparty without adversary. It deal with thirdparty authenticate receiver. The condition five describes that receiver has a distinct session with intended principle thirdparty and thirdparty has the same session with receiver with overwhelming probability.

Definition DAP with events: If DAP' satisfies the condition one and four in definition DAP and DAP' satisfies the correspondence:

$$\text{inj-event}\left(\text{whole}_{\text{Sender}}\left(\text{Receiver}, x\right)\right) \Rightarrow \text{inj-event}\left(\text{whole}_{\text{Receiverr}}\left(\text{Sender}, x\right)\right) \quad (1)$$

$$\text{inj-event}\left(\text{whole}_{\text{Thirdparty}}\left(\text{Receiver}, x\right)\right) \Rightarrow \text{inj-event}\left(\text{whole}_{\text{Thirdparty}}\left(\text{Sender}, x\right)\right) \quad (2)$$

Table 1: The table: event in DAP

Contents	Position (Before)
Event	
$portion_{Receiver}$ ($host_x$, sessioner' (z_1, z_2,..., z_{l-2}))	Receiver sends 1-2th message
$portion_{Sender}$ ($host_y$, sessioner' (x_1, x_2,..., x_{l-2}))	Sender sends 1--th message, OK_{Sender} ($host_y$)
$whole_{Sender}$ ($host_y$, sessioner' (x_1, x_2,..., x_{l-1}))	Receiver sends $OK_{Receiver}$ (Sender) in additional message one message
$whole_{Receiver}$ ($host_x$, sessioner' (z_1, z_2,..., z_{l-1}))	Receiver sends 1 th message, $OK_{Receiver}$ ($host_z$)
$whole_{Receiver}$ ($host_z$, sessioner (q_1))	Thirdparty sends $OK_{Thirdparty}$ (Sender) in additional message two message

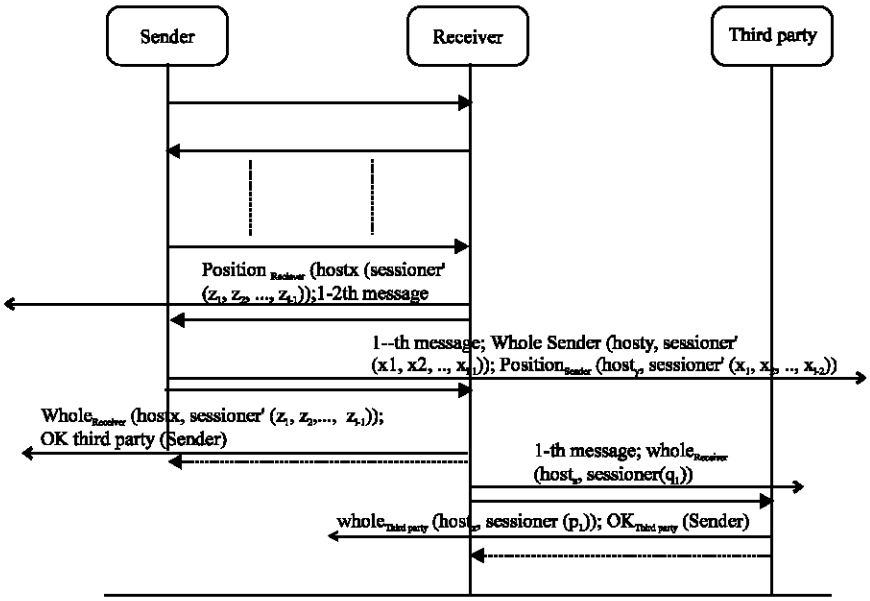

Fig. 3: The event in DAP "→" means the sequence of output in the message in DAP

with public variables V = ϕ, then DAP is a secure deniable authentication protocol with session functions (sessionid and sessionid').

DAP' process can be obtained from DAP by adding the following events in Table 1 and Fig. 3:

"→" means the sequence of output in the message in DAP.

- Deniability consists of strong deniability and weak deniability. The purpose of strong deniability is to protect the privacy of receiver. After execution of the deniable authentication protocol the sender can deny to have ever authenticated anything to receiver. If the prover (receiver or the any other party) wants to prove that the sender have authenticated messages to receiver, they must provide all the relevant evidence. The sender can provide his secret information to the thirdparty

A adversary model in strong deniability: When discussing the strong deniability, in addition the adversary has the ability in previous section, we always also suppose that the sender and the receiver cooperate with the judge or the prover or the any other party, which means that the sender and the receiver provide all the transcripts of the message in the deniable authentication protocol to them.

Definition of strong deniability: If DAP' satisfies the condition one and four in definition DAP and DAP' satisfies the correspondence inj-event (whole $_{sender}$ (receiver, x)) ⇒ inj-event (whole $_{receiver}$ (sender, x)) and (whole $_{third\ party}$ (receiver, x)) ⇒ inj-event (whole $_{third\ party}$ (sender, x)) with public variables V = θ, then DAP is a secure deniable authentication protocol with session functions (sessionid and sessionid') in a adversary model in strong deniability. In the above definition of DAP the injective correspondence can be instead by non-injective correspondence.

The purpose of weak deniability is to protect the privacy of sender. After execution of the deniable authentication protocol the receiver can prove to have spoken to sender but not the content of what the sender authenticated in a way that the receiver can not convince a third party that such authentication. If the receiver want

to prove that the sender have authenticated messages to receiver, he must provide the evidence related to the thing.

An adversary model in weak deniability: When discussing the weak deniability, in addition the adversary has the ability in previous section; we always suppose that only the receiver generates the evidence that the sender have authenticated messages to receiver. The receiver can not get the secret information of the sender, for example the private key of sender. The receiver can provide his secret information to thirdparty.

Definition of weak deniability: If DAP' satisfies the condition one in definition DAP and DAOP' satisfies the correspondence

$$\text{inj-event}\left(\text{whole}_{\text{Sender}}\left(\text{Receiver}, x\right)\right) \Rightarrow \text{inj-event}\left(\text{whole}_{\text{Receiverr}}\left(\text{Sender}, x\right)\right)$$

and

$$\text{inj-event}\left(\text{whole}_{\text{Thirdparty}}\left(\text{Receiver}, x\right)\right) \Rightarrow \text{inj-event}\left(\text{whole}_{\text{Thirdparty}}\left(\text{Sender}, x\right)\right)$$

with public variables $V = \theta$, then DAP is a secure deniable authentication protocol with session functions (sessionid and sessionid') in a adversary model in weak deniability. In the above definition of DAP the injective correspondence can be instead by non-injective correspondence.

MENG NON-INTERACTIVE DENIABLE AUTHENTICATION PROTOCOL

Meng non-interactive deniable authentication protocol (Meng, 2009a) is a secure non-interactive deniable authentication protocol based on discrete logarithm problem. It consists of authority, the sender and the receiver. Meng non-interactive deniable authentication protocol is secure and has properties: completeness, strong deniability, weak deniability, security of forgery attack, security of impersonate attack, security of compromising session secret attack and security of man-in-the-middle attack.

Meng non-interactive deniable authentication protocol in Fig. 4 is described as the following:

Initialized phase: The Authority firstly chooses a large prime numbers p; secondly, compute a random

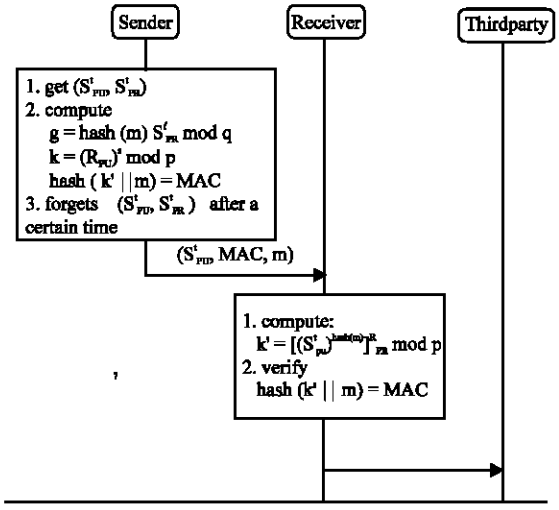

Fig. 4: Meng non-interactive deniable authentication protocol

multiplicative generator element g in finite field of p elements: GF (p); thirdly, send the g, p to the bullet board. The sender firstly, pick a serial random numbers $\Upsilon_i \epsilon_U Z_{P-1}$, $S^i_{PR} = \Upsilon$ i = 1…..*l*; secondly, compute his public key by $S^i_{PU} = G^{\upsilon i}$ (mode p) i = 1…..*l*; thirdly, send the S^i_{PU} to the bullet board. The receiver firstly, pick a random number $x \epsilon_U Z_{P-1}$ $R_{PU} = x$; secondly, compute his public key by: $R_{PU} = g^x$ (mode p) ; thirdly, send the R_{PU} to the bullet board.

When finishing the initialized phase the sender has serial public and private keys (S^i_{PU}, S^i_{PR}), at the same time receiver has his public and private keys (R_{PU}, R_{PR})

Execution of protocol phase: The sender firstly, chooses randomly public and private keys (S^i_{PU}, S^i_{PR}) with each run of the propose protocol are different, then computes: δ = hash (m) S^t_{PR} mod q and forget (S^i_{PU}, S^t_{PR}) after a certain time. $k = (R_{PU})^{\delta}$ mod p hash (km) = MAC. Finally he sends (S^t_{PU}, MAC, m) to the receiver.

The receiver firstly compute k' = $[(S^t_{PU})^{hash\,(m)}]^{Rm}$ then verifies hash (k' ||m) = MAC, if the result is true, the receiver accepts it. Otherwise the receiver rejects it.

MODELING MENG NON-INTERACTIVE DENIABLE AUTHENTICATION PROTOCOL IN BLANCHET CALCULUS

Adversary model: In modeling weak deniability, firstly the Meng protocol is described in Blanchet Calculus, we assume that the adversary is in an adversary model in weak deniability: when discussing the weak deniability, in addition the adversary has the ability in previous section;

we always suppose that only the receiver generates the evidence that the sender have authenticated messages to receiver. The third party process can not get the secret information of the sender, for example the private key of the sender. The receiver can provide his secret information to the third party.

In modeling strong deniability, firstly the Meng protocol is described in Blanchet Calculus, we assume that the adversary is in an adversary model in strong deniability: when discussing the strong deniability, in addition the adversary has the ability in previous section; we always suppose that the sender and the receiver cooperate with the third party, that is to say, the third party has capacity to access all transcripts of the sender and receiver.

Cryptographic assumptions: In our computational analysis model, we assume that shared-key encryption is modeled as encrypt-then-MAC, where the encryption is indistinguishability under chosen plaintext attacks and the MAC is enforceability under chosen message attacks; public-key encryption is assumed to be indistinguishability under adaptive chosen ciphertext attacks. We also assume that session key generator f (publickey, hash, private key) and (mb, hashfromS, enc (gabin receiver, pk TP, r4)) are indistinguishability under chosen plaintext attacks. Figure 5 gives cryptographic assumptions.

Events: In order to verify the weak deniability and strong deniability, according to the model proposed by Meng and Shao, the events are defined in Fig.6. event portion receiver (x) = = > event whole sender (x) and inj-event protion receiver (x) = = > inj-event whole sender (x) are used in authentication. Event whole third party (x) = => event whole receiver (x), inj-event whole third party (x) = = > inj-event whole receiver (x) and event portion third party = = > true. are used in strong deniability and weak deniability. In order to prove that the Meng protocol has the weak deniability and strong deniability, the non-injective or injective properties must satisfy.

Processes: The complete formal model of Meng protocol in Blanchet calculus is given in figures. Figure 7 to 11 report the basic process include DAP of fanprocess, sender process, receiver process, third party process in weak deniability and in strong deniability forming our of the model of Meng protocol.

The process DAP of mengprocess in Fig. 7 is assumed to run in interaction with an adversary, which

```
(*IND-CCA2 probabilistic public-key encryption *)
proba Penc.
proba Penccoll.
expand IND_CCA2_public_key_enc(keyseed, publickey, privatekey, key, encryptionofG,
                               randomseed, skgen, pkgen, enc, dec, injbot, Z, Penc, Penccoll).

(* one way hash function without key *)
fun HO(message):hash.
        forall x:message, y:message;
        HO(x)<>HO(y).

(* one way hash function  *)
fun H(message, key):hash.
        forall k:key, x:message, y:message;
        H(x,k)<>H(y,k).


(*session key generator*)
fun f(publickey, hash, privatekey):key .
fun g(publickey, hash, privatekey):key .
forall r1:keyseed, r2:keyseed,m:hash;
f(pkgen(r2),m,skgen(r1))=g(pkgen(r1),m,skgen(r2)).
```

Fig. 5: Cryptographic assumptions

```
(* events *)

event WholeSender(key).
event PortionReceiver(key).
event WholeReceiver(key).
event WholeThirdParty(key).
event PortionThirdParty.

(* Receiver  authenticate Sender *)
event PortionReceiver(x) ==> WholeSender(x).
event inj:PortionReceiver(x) ==> inj:WholeSender(x).

(* Third Party authenticate Receiver and deniable authentication *)
event WholeThirdParty(x) ==> WholeReceiver(x).
event inj:WholeThirdParty(x) ==> inj:WholeReceiver(x).
event PortionThirdParty ==> true.
```

Fig. 6: Events

```
DAP of Meng process
    start();
    new rs : keyseed;
    let skS = skgen(rs) in
    let pkS : publickey = pkgen(rs) in
    new rr : keyseed;
    let skR = skgen(rr) in
    let pkR = pkgen(rr) in
    new rtp : keyseed;
    let skTP = skgen(rtp) in
    let pkTP = pkgen(rtp) in
    start<pkTP>;
    ((!N SenderProcess) | (!N ReceiverProcess) | (!N ThirdpartyProcess) )
```

Fig. 7: DAP of Meng process

```
let SenderProcess =
    c1();
    new ma : message;
    let keyinS : key = f(pkR,HO(ma),skS) in
    event WholeSender(keyinS);
    let hashinS : hash = H(ma, keyinS) in
    c4<pkS, ma, hashinS>.
```

Fig. 8: Sender process

also models the network. DAP of mengprocess firstly receives an empty message on channel start, sent by the adversary. Then, it chooses randomly with uniform probability a bitstring rs in the type keyseed, by the construction new rs: keyseed. A type T, such as keyseed, aims at denoting a set of bitstrings. However, the considered set of bitstrings depends on the security parameter η, which determines the length of keys. Then, DAP of fanprocess generates the sender's private key skS corresponding to the coins rs, by calling the public-key generation algorithm skgen (). Similarly, it generates the sender's public key pkS corresponding to the coins rs, by calling the public-key generation algorithm pkgen (). After that it chooses randomly with uniform probability a bitstring rr in the type keyseed, by the construction new rr: keyseed. Then, DAP of fanprocess generates the receiver's private key skR corresponding to the coins rr,

```
let ReceiverProcess =
    c2(pkSfromS:publickey,mb:message,hashfromS:hash);
    find j<=N suchthat defined(hashinS[j])&&(hashfromS=hashinS[j]) then
    let keyinR : key = g(pkSfromS,HO(mb),skR) in
    let hashinR : hash = H(mb,keyinR) in
    if  hashinR = hashfromS then
    event PortionReceiver(keyinR);
    new r4 :randomseed;
    new r5 :randomseed;
    new b:bool;
    (* OK *)
    if b=true then
    event WholeReceiver(keyinR);
    c5<pkSfromS,mb, hashfromS, enc(keyinR, pkTP, r4)>
    else
    event WholeReceiver(keyinR);
    c5<pkS, mb, hashinR, enc(keyinR, pkTP, r5)>.
```

Fig. 9: Receiver process

```
let ThirdpartyProcess =
    c5(pkSfromR:publickey, mc:message, hashfromR : hash,encryptionofkey: encryption);
    let injbot(keyfromR : key) = dec( encryptionofkey,skTP ) in
    if  H(mc, keyfromR) = hashfromR  then
    find j <= N suchthat defined(keyinR[j]) && (keyfromR = keyinR[j])  then
    (event PortionThirdParty;event WholeThirdParty(keyfromR))
    c6<>.
```

Fig. 10: Third party process in weak deniability

```
let ThirdpartyProcess =
    c5(pkSfromR:publickey, mc:message, hashfromR : hash,encryptionofkey: encryption);
    let injbot(keyfromR : key) = dec( encryptionofkey,skTP ) in
    if  H(mc, keyfromR) = hashfromR  then
    find j <= N suchthat defined(keyinS[j]) && (keyfromR = keyinS[j])  then
    c6<> else (event PortionThirdParty;event WholeThirdParty(keyfromR)).
```

Fig. 11: Third party process in strong deniability

by calling the public-key generation algorithm skgen (). Similarly, it generates the sender's public key pkR corresponding to the coins rr, by calling the public-key generation algorithm pkgen (). After that it chooses randomly with uniform probability a bitstring rTP in the type keyseed, by the construction new rTP: keyseed. Then, DAP of fanprocess generates the third party's private key skTP corresponding to the coins rTP, by calling the public-key generation algorithm skgen (). Similarly, it generates the sender's public key pkTP corresponding to the coins rtp, by calling the public-key generation algorithm pkgen(). After outputting this message, the control passes to the receiving process, which is part of the adversary. Several processes are then made available, which represent the roles of sender, receiver and third party in the protocol: the process (! $^{i \le sender \le n}$ sender process), (! $^{i \le sender \le n}$ receiver process) is the parallel composition of (! $^{i \le sender \le n}$ sender process), (! $^{i \le sender \le n}$ receiver process) and it makes simultaneously available the processes defined in (! $^{I \le sender \le n}$ sender

process) , (! $^{i\ sender\ \leq n}$ receiver process) and(! $^{i\ sender\ \leq n}$ third party process). The replication (! $^{i\ sender\ \leq n}$ sender process) represents n copies of the process sender indexed by the replication index i_{sender}; The replication (! $^{i\ reser\ \leq n}$ recevier process) represents n copies of the process receiver process indexed by the replication index $i_{receiver}$; The replication (! $^{i\ third\ arty\ \leq n}$ third party process) represents n copies of the process thirdparty process indexed by the replication index $i_{third\ party}$.

The process sender process in Fig. 8 begins with an input on channel c1which is abbreviation of c1 [i_{sender}]; the channel is indexed with i_{sender} so that the adversary can choose which copy of the process sender process receives the message by sending it on channel c1 [i_{sender}] for the appropriate value of i_{sender}. The situation is similar for receiver process, which expects a message on channel c2 which is abbreviation of c2[i_{sender}]. At the same time for thirdparty process, which expects a message on channel c5 which is abbreviation of c14[$i_{third\ party}$].The adversary can then run each copy of sender process or receiver processor thirdparty process simply by sending a message on the appropriate channel c1 or c2 or c5. Process sender process chooses randomly with uniform probability a bitstring a in the type message, by the construct new ma: massage. After that it generates the key keyinS by the construct f(pkR, HO (ma), skS). Then it executes the event event whole sender (keyinS). This event does not change the state of the system. Events just record a certain program point that has been reached with certain values of the arguments of the event. After that it generates the message digest hashinS by the construct H(ma, keyinS). Finally process sender process outputs the message <pkS, ma, hashinS> on channel c4, so that the adversary has message (pkS, ma, hashinS).

The process sender process in Fig. 9 firstly expects on channel c2 a message (pkS fromS: publickey, mb: message, hasfromS: hash). It serves for starting a new session of the protocol, in which process receiver process interacts with the participant. After that, process receiver process checks hash fromS: hash from sender by the construct find j< = N such that defined (hasinS [j]) && (hashfromS = hashinS [j]). If it is true process receiver process computes the key keyin R: key by the construct g(pkS fromS, HO (mb), skR). It also computes the hush code hashinR: hash by the construct H (mb, keyinR. After that it checks hashinR = hash fromS. If it is true event portion receiver (keyinR) is executed. Process receiver process chooses randomly with uniform probability a nonce r4 in the type randomseed by the construct new r4: randomseed. Process receiver process also chooses randomly with uniform probability a nonce r5 in the type randomseed by the construct new r5: randomseed. Process receiver process chooses randomly with uniform probability a nonce b in the type bool by the construct new b: bool. If it is true and b = true then process receiver process executes event event whole receiver (keyinR) and outputs <pkS fromS, mb, hash fromS, enc (keyinR, pkTP, r4)> by the channel c5, after that the control then passes to the sender process, included in the adversary. But may proceed differently in order to mount an attack against the protocol, else reciver process executes event event whole receiver (keyinR) and outputs <pkS, mb, hashinR, enc (keyinR, pkTP, r5)> by the channel c5, after that the control then passes to the sender process, included the adversary, it also may proceed differently in order to mount an attack against the protocol.

Here, we first introduce the process thirdparty process in weak deniability in Fig. 10, then the process thirdparty process in strong deniability in Fig. 11. The control then passes to the process thirdparty process in weak deniability, which should forward this message (pkS fromR: publickey, mc: message, hash fromR: hash, encryption key: encryptionofG) on channel c5 if it wishes to run the protocol correctly. Process thirdparty process gets the session key key fromR by injbot (key fromR: key) and decryption algorithm dec(encryptionofkey, skTP). Then it checks H (mc, key from R) = hash fromR whether or not. If it is true then proces thirdparty process checks the matter that key fromR is from receiver weather or not by the construct find j< = N such that defind (keyinR [j]&& (key fromR = keyinR [j]). if the result is true, process thirdparty process will executes the event event portion third party and event event whole third party (key fromR).After that the control then passes to the sender process, included in the adversary, but may proceed differently in order to mount an attack against the protocol. Process thirdparty process gets the parameters of receiver related to the verification of the authenticator of message by the construct find j < = N such that defined (keyinR [j]) && (key fromR = keyinR [j]), not through the channel between the sender and the third party.

Here we introduce the process thirdparty process in strong deniability in Fig. 11. The control then passes to the process thirdparty process, which should forward this message (pkS fromR: publickey, mc: message, hash fromR): hash, encryption of keyL incrption on channel c5 between receiver and third party if it wishes to run the protocol correctly. The control then passes to the process thirdparty process in strong deniability, which should forward this message (pkS fromR: public key, mc: message, hash fromR: hash, encryption of key: incrption) on channel c5 if it wishes to run the protocol correctly.

Process thirdparty process gets the session key key fromR by inj boot (key fromR: key) and decryption algorithm dec (encryption key sk TP). Then it checks H (mc, key from R) whether or not. If it is true then proces thirdparty process checks the matter that key fromR is from sender weather or not by the construct find j < = N such that defind (key inS [j]) && (key fromR = keyinS [j]). if the result is true, process receiver process will executes $\bar{c}6$ <> else executes the event event portion third party and event event whole third party (key fromR).After that the control then passes to the sender process, included in the adversary, but may proceed differently in order to mount an attack against the protocol. Process thirdparty process gets the parameters of receiver related to the verification of the authenticator of message by the construct find j< = such that defind (keyinS [j]) && (key fromR = keyinS [j]), not through the channel between the sender and the third party.

AUTOMATIC VERIFICATION OF DENIABILITY IN MENG NON-INTERACTIVE DENIABLE AUTHENTICATION PROTOCOL WITH CRYPTOVERIF

CryptoVerif can take two formats as input. The first one is in the form of channels Front-end. The second one is in the form of oracles Front-end. In both cases, the output of the system is essentially the same. The main difference between the two front-ends is that the oracles front-end uses oracles while the channels front-end uses channels. In the channels front-end, channels must be declared by a channel declaration. There is no such declaration in the oracles front-end. Some constructs use a different syntax in the oracles front-end, to be closer to the syntax of cryptographic games.

In this study we use the form of channels Front-end as the input of CryptoVerif. In order to prove the weak deniability and strong deniability in Meng protocol, the Blanchet calculus model are needed to be translated into the syntax of CryptoVerif and generated the CryptoVerif inputs in the form of channels Front-end.

Verification of weak deniability: In order to prove the weak deniability, according to the model proposed by Meng and Shao (2010), without adversary, Meng protocol has the condition one and condition four. Also the following injective properties event potion receiver (x) = = > event whole sender (x), event whole third party (x) = = > event whole receiver (x) and event portion third party = => trueare satisfied; or non-injective properties inj-event portion receiver (x) = = > inj-event whole sender (x), inj-event whole third party (x) = = > inj-event whole receiver (x) and event portion third party = => true are satisfied. Fig. 12-19 gives the inputs of verification of weak deniability in CryptoVerif.

Table 2 gives the procedure and result of verification in weak deniability in CryptoVerif. The analysis was performed by CryptoVerif and succeeded.

The results in Table 2 are that:

- Proved event Portion Third Party ==> true
- Proved event inj:WholeThird Party(x) ==> inj: Whole Receiver (x)
- Proved event Whole Third Party(x) ==> Whole receiver(x)

Table 2: Procedure and result of verification of weak deniability

Source	Destination	Applying
Game 1	Game 2	simplify
Game 2	Game 3	move all binders
Game 3	Game 4	remove assignments of useless

Result proved event portion Thirdparty => true. Result proved event inj: Whole Thirdparty (x) => inj: Whole Receiver (x). Result proved event Whole Thirdparty (x) => Whole Receiver (x). Result proved event inj: Portion Receiver (x) => Whole Sender (x). Result proved event Portion Receiver (x) => Whole Sender (x)

```
param N.
type host.   (* the type of the principle identification in protocol*)
type keyseed [large,fixed].(* the type of the key seeds*)
type randomseed [fixed]. (* the type of the key seeds used in generation of random numbers*)
type publickey [bounded].(* the type of the public key *)
type privatekey [bounded].(* the type of the private key *)

type message [large,fixed].(* the type of the plaintext *)
type hash.(* the type of output of hush function *)
type encryption.(* the type of ciphertext *)
type Key [large,fixed]. (* the type of input in operation of enc *)
```

Fig. 12: Type definitions

```
(* IND-CCA2 probabilistic public-key encryption *)
proba Penc.
proba Penccoll.
expand IND_CCA2_public_key_enc(keyseed, publickey, privatekey, key, encryption,
                               randomseed, skgen, pkgen, enc, dec, injbot, Z, Penc, Penccoll).

(*  one way hash function without key *)
fun HO(message):hash.
        forall x:message, y:message;
        HO(x)<>HO(y).

(*  one way hash function  *)
fun H(message, key):hash.
        forall k:key, x:message, y:message;
        H(x,k)<>H(y,k).


(* ideal session key generator*)
fun f(publickey, hash, privatekey):key .
fun g(publickey, hash, privatekey):key .
forall r1:keyseed, r2:keyseed,m:hash;
f(pkgen(r2),m,skgen(r1))=g(pkgen(r1),m,skgen(r2)).
```

Fig. 13: Cryptographic assumptions

```
(*  events *)

event WholeSender(key).
event PortionReceiver(key).
event WholeReceiver(key).
event WholeThirdParty(key).
event PortionThirdParty.

(* Receiver  authenticate Sender *)
query x:key; event PortionReceiver(x) ==> WholeSender(x).
query x:key; event inj:PortionReceiver(x) ==> inj:WholeSender(x).

(* Third Party authenticate Receiver and deniable authentication *)
query x:key; event WholeThirdParty(x) ==> WholeReceiver(x).
query x:key; event inj:WholeThirdParty(x) ==> inj:WholeReceiver(x).
query event PortionThirdParty ==> true.
```

Fig. 14: Query events

```
channel start, c1, c2, c3, c4, c5, c6.
```

Fig. 15: Channels

```
let SenderProcess =
    in(c1, ());
    new ma : message;
    let keyinS : key = f(pkR,HO(ma),skS) in
    event WholeSender(keyinS);
    let hashinS : hash = H(ma, keyinS) in
    out(c4, (pkS, ma, hashinS)).
```

Fig. 16: Sender process

```
let ReceiverProcess =
    in(c2, (pkSfromS:publickey,mb:message,hashfromS:hash));
    find j<=N suchthat defined(hashinS[j])&&(hashfromS=hashinS[j]) then
    let keyinR : key = g(pkSfromS,HO(mb),skR) in
    let hashinR : hash = H(mb,keyinR) in
    if  hashinR = hashfromS then
    event PortionReceiver(keyinR);
    new r4 :randomseed;
    new r5 :randomseed;
    new b:bool;
    (* OK *)
    if b=true then
    event WholeReceiver(keyinR);
    out(c5, (pkSfromS,mb, hashfromS, enc(keyinR, pkTP, r4)))
    else
    event WholeReceiver(keyinR);
    out(c5, (pkS, mb, hashinR, enc(keyinR, pkTP, r5))).
```

Fig. 17: Receiver process

```
let ThirdpartyProcess =
    in(c5, (pkSfromR:publickey, mc:message, hashfromR : hash,encryptionofkey: encryption));
    let injbot(keyfromR : key) = dec( encryptionofkey,skTP ) in
    if  H(mc, keyfromR) = hashfromR  then
    find j <= N suchthat defined(keyinR[j]) && (keyfromR = keyinR[j])  then
    (event PortionThirdParty;event WholeThirdParty(keyfromR))
    else out(c6,()).
```

Fig. 18: Third party process in weak deniability

- Proved event inj: PortionReceiver(x) ==> inj: Whole Sender(x)
- Proved event PortionReceiver(x) ==> Whole Sender (x)

According to the definition of weak deniability proposed by us Meng protocol is proved to guarantee weak deniability in computation model.

Verification of strong deniability: In order to prove the weak deniability, according to the model proposed by Meng and Shao (2010), the following injective properties or non-injective properties event portion receiver (x) = = > event whole sender (x), event whole third party (x) = = > event whole receiver (x) and event portion third party = = > true. are satisfied; or inj-event portion receiver (x) = = > inj-event whole sender (x), inj-event

```
process
    in(start, ());
    new rs : keyseed;
    let skS = skgen(rs) in
    let pkS : publickey = pkgen(rs) in
    new rr : keyseed;
    let skR = skgen(rr) in
    let pkR = pkgen(rr) in
    new rtp : keyseed;
    let skTP = skgen(rtp) in
    let pkTP = pkgen(rtp) in
    out(start, (pkTP));
    ((!N SenderProcess) | (!N ReceiverProcess) | (!N ThirdpartyProcess) )
```

Fig. 19: Main process

```
let ThirdpartyProcess =
    in(c5, (pkSfromR:publickey, mc:message, hashfromR : hash,encryptionofkey: encryption));
    let injbot(keyfromR : key) = dec( encryptionofkey,skTP ) in
    if H(mc, keyfromR) = hashfromR then
    find j <= N suchthat defined(keyinS[j]) && (keyfromR = keyinS[j]) then
    out(c6,())
    else (event PortionThirdParty;event WholeThirdParty(keyfromR)).
```

Fig. 20: Third party process in strong deniability

whole third party (x) = = > inj-event whole receiver (x) and event porion third party = = > true. are satisfied. Figure 12-17, 19, 20 give the inputs of verification of strong deniability in CryptoVerif.

Table 3 gives the procedure and results of verification in strong deniability in CryptoVerif. The analysis was performed by CryptoVerif and succeeded. The results in Table 3 are that:

- Proved event Portion Third Party ==> true
- Proved event inj: Portion Receiver (x) ==> inj:WholeSender(x)
- Proved event Portion Receiver (x) ==> Whole Sender (x)
- Could not prove event inj: Whole Third Party (x) ==> inj: Whole Receiver (x), event Whole Third Party (x) ==> Whole Receiver(x)

From the result we can found the correspondences event inj: Whole Third Party (x) ==> inj:Whole Receiver (x), event Whole Third Party(x) ==> WholeReceiver(x) is not proved. Owning to the incomplete of CryptoVerif and the previous informal proof we can conclude that Meng protocol is proved to guarantee strong deniability in computation model.

Table 3: Procedure and results of verification of strong deniability

Source	Destination	Applying
Game 1	Game 2	simplify
Game 2	Game 3	move all binders
Game 3	Game 4	remove assignments of useless
Result Proved event PortionThirdParty ==> true		
Result Proved event inj:PortionReceiver(x) ==> inj:WholeSender(x)		
Result Proved event PortionReceiver(x) ==> WholeSender(x)		
Game 4	Game 5	Remove assignments of binder pkTP
Game 5	Game 6	Remove assignments of binder skTP
Game 6	Game 7	Equivalence
Game 7	Game 8	Simplify
Game 8	Game 9	Remove assignments of useless
Game 9	Game 10	Remove assignments of binder pkS
Game 10	Game 11	Equivalence
Game 11	Game 12	Simplify
RESULT time (context for game 6) = time (let injbot) * N+3. *Time(H) * N+time (= hash, length (H), m xlength (game 6: hashfromR[!_13])) *N+2. *time(HO) *N+2. *time(g, length(HO)) *N+time (= hash, maxlength (game 6: hashinR[!_12]), maxlength (game 6: hashfromS[!_12])) * N+N*N*time(= hash, maxlength (game 6: hashfromS[!_12]), maxlength (game 6: hashinS[!_11])) + time (pkgen) *N+time(skgen) *N+time (skgen)+time (pkgen). RESULT Could not prove event inj:WholeThirdParty(x) ==> inj:WholeReceiver(x), event WholeThirdParty(x) ==> WholeReceiver(x)		

CONCLUSION

During the past few decades deniable authentication protocol has been studied. A lot of deniable authentication protocols has been proposed which

claimed that have the security properties, for example, authentication; strong deniability; weak deniability; security of forgery attack; security of impersonate attack; security of compromising session secret attack; security of man-in-the-middle attack and so on. In order to verify the security properties of security protocol include deniable authentication protocols and increase the confidence of the people, two approaches have been developed for analyzing security protocols form the beginning of the 1980s. One approach relies on a symbolic model of protocol executions in which cryptographic primitives are treated as black boxes. The other approach relies on a computational model that base issues of complexity and probability. This approach use a strong notion of security, guaranteed against all probabilistic polynomial-time attacks. The computation approach can be more realistic. To our best knowledge, this security properties and deniable authentication protocols are analyzed with informal method or with symbolic method by hand which depends on experts' knowledge and skill and is prone to make mistakes.

To our best knowledge, this security properties and deniable authentication protocols are analyzed with informal method, or with symbolic method by hand (Meng, 2009b), which depends on experts' knowledge and skill and is prone to make mistakes. So analysis of security properties and deniable authentication protocols with automatic tool in symbolic model or computational model play an important role in security protocol world and is a significant works. Especially analysis of security properties and deniable authentication protocols with automatic tool in computational model is a changeling issue.

In this study, we use Blanchet calculus in the computational model to mechanized analyze Meng non-interactive deniable authentication protocol with mechanized tool CryptoVerif.

- The state-of-art of deniable authentication protocol and the proof included in symbolic model and in computational model are presented. We found that security properties and deniable authentication protocols are analyzed with informal method or with symbolic method by hand. There does not existing that analysis of security properties and deniable authentication protocols with automatic tool until Meng and Shao (2010) proposed the first automatic framework of strong and weak deniability based on Blanchet calculus in computational model. And the mechanized CryptoVerif is the only automatic tool with computational model
- Applying Meng and Shao automatic model based on Blanchet calculus in computational model with active adversary for automatically analysis of Meng non-interactive deniable authentication protocol. So the term, process and correspondence assertion in Blanchet calculus is used to model security properties included deniability and Meng protocol. The analysis itself is performed by automatic tool CryptoVerif developed by Blanchet
- The result shows that Meng non-interactive deniable authentication protocol has weak deniability and strong deniability. To our best knowledge we are conducting the first automatic analysis in computational model of Meng non-interactive deniable authentication protocol in active adversary
- Further studies concentrate on formal analysis of the security of forgery attack, security of impersonate attack, security of compromising session secret attack, security of man-in-the-middle attack

ACKNOWLEDGMENT

This study was supported in part by Natural Science Foundation of South-Center University for Nationalities under the grants No: YZZ06026, titled Research on the internet voting protocols with receipt-freeness, conducted in Wuhan, China from 06/11/2006 to 20/11/2009.

REFERENCES

Aumann, Y. and M. Rabin, 1998. Efficient deniable authentication of long messages. Proceedings of the International Conference on Theoretical Computer Science in Honor of Professor Manuel Blum's 60th Birthday, 1998. http://www.cs.cityu.edu.hk/dept/video.html.

Bhargavan, K., R. Corin and C. Fournet, 2007. Crypto-verifying protocol implementations in ML. Proceedings of the Workshop on Formal and Computational Cryptography-FCC 2007. http://www.msr-inria.inria.fr/projects/sec/fs2cv/draft.pdf.

Bhargavan, K., R. Corin, C. Fournet and E. Zalinescu, 2008. Cryptographically verified implementations for TLS. Proceedings of the 15th ACM Conference on Computer and Communications Security, Oct. 27-31, Alexandria, Virginia, USA., pp: 459-468.

Blanchet, B., 2001. An efficient cryptographic protocol verifier based on prolog rules. Proceedings of the 14th IEEE Workshop on Computer Security Foundations, June 11-13, IEEE Computer Society, Washington, DC., pp: 82-96.

Blanchet, B., 2004. Automatic proof of strong secrecy for security protocols. Proceedings of IEEE Symposium on Security and Privacy, May 9-12, CNRS, Ecole Normale Superieure, Paris, France, pp: 86-100.

Blanchet, B., 2005. A computationally sound mechanized prover for security protocols. Cryptology ePrintArchive, Report 2005/401, Nov. 2005. http://eprint.iacr.org/2005/401.

Blanchet, B., 2006. A computationally sound mechanized prover for security protocols. Proceedings of IEEE Symposium on Security and Privacy, May 21-24, Oakland, California, pp: 150-154.

Blanchet, B. and D. Pointcheval, 2006. Automated security proofs with sequences of games. Proceedings of the 27th IEEE Symposium on Security, Aug. 06, LNCS, Santa Barbara, CA, Springer Verlag, pp: 537-554.

Blanchet, B., 2007a. Computationally sound mechanized proofs of correspondence assertions. Proceedings of 20th IEEE Computer Security Foundations Symposium, July 6-8, Venice, Italy, pp: 97-111.

Blanchet, B., 2007b. A computationally sound mechanized prover for security protocols. IEEE Trans. Dependable Secure Comput., 5: 193-207.

Blanchet, B., A.D. Jaggard, A. Scedrov and J. Tsay, 2008. Computationally sound mechanized proofs for basic and public-key kerberos. Proceedings of the ACM Symposium on Information, Computer and Communications Security, March 2008, Tokyo, Japan, pp: 87-99.

Blanchet, B., 2009. Diffie-hellman in cryptoverif. http://www.di.ens.fr/~blanchet/talks/FormaCrypt09-DH.pdf.

Clarke, E.M., S. Jha and W. Marrero, 2000. Verifying security protocols with Brutus. ACM Trans. Softw. Eng. Methodol., 9: 443-487.

Deng, X., C.H. Lee and H. Zhu, 2001. Deniable authentication protocols. IEE Proc. Comput. Digital Techniques, 148: 101-104.

Dwork, C., M. Naor and A. Sahai, 1998. Concurrent zero-knowledge. Proceedings of the 30th Annual ACM Symposium on Theory of Computing, 1998, USA., pp: 409-418.

Fan, L., C.X. Xu and J.H. Li, 2002. Deniable authentication protocol based on Diffie–Hellman algorithm. Elect. Lett., 38: 705-706.

Feng, T. and J.F. Ma, 2007. Universally composable security concurrent deniable authentication based on witness indistinguishable. J. Software, 18: 2871-2881.

Han, S., W.Q. Liu and E. Chang, 2005. Deniable Authentication protocol resisting man-in-the-middle attack. Proc. World Acad. Sci. Eng. Technol., 3: 161-164.

Jaggard, A.D., A. Scedrov and J. Tsay, 2007. Computationally sound mechanized proof of PKINIT for kerberos. Proceedings of the Workshop on Formal and Computational Cryptography-FCC 2007. http://dimacs.rutgers.edu/~adj/Research/papers/jst07fcc.pdf.

Joseph, C. and F. Cremers, 2006. Scyther-semantics and verification of security protocols. http://alexandria.tue.nl/extra2/200612074.pdf.

Kindred, D., 1999. Theory generation for security protocols. Doctoral Thesis, Carnegie Mellon University.

Lee, W.B., C.C. Wu and W.J. Tsaur, 2007. A novel deniable authentication protocol using generalized ElGamal signature scheme. Inform. Sci., 177: 1376-1381.

Lowe, G., 1998. Casper: A complier for the analysis of security protocols. J. Comput. Sec., 6: 53-84.

Lu, R. and Z. Cao, 2005a. A new deniable authentication protocol from bilinear pairings. Applied Math. Comput., 168: 954-961.

Lu, R. and Z. Cao, 2005b. Non-interactive deniable authentication protocol based on factoring. Comput. Standards Interfaces, 27: 401-405.

Maggi, P. and R. Sisto, 2002. Using SPIN to verify security properties of cryptographic protocols. Proceedings of the 9th International SPIN Workshop on Model Checking of Software, April 11-13, Springer-Verlag, London, pp: 187-204.

McMillan, K.L., 1992. Symbolic model checking: An approach to the state explosion problem. Ph.D. Thesis, Carnegie Mellon University.

Meadows, C.A., 1996. The NRL protocol analyzer: An overview. J. Logic Program., 26: 113-131.

Meng, B., 2009a. A secure non-interactive deniable authentication protocol with strong deniability based on discrete logarithm problem and its application on Internet voting protocol. Inform. Technol. J., 8: 302-309.

Meng, B., 2009b. Formalizing deniability. Inform. Technol. J., 8: 625-642.

Meng, B. and F. Shao, 2010. Computationally sound mechanized proofs for deniable authentication protocols with a probabilistic polynomial calculus in computational model. Inform. Technol. J., 10: 611-625.

Paulson, L.C., 1998. The inductive approach to verifying cryptographic protocols. J. Comput. Sec., 6: 85-128.

Qian, H.F., Z.F. Cao, L.C. Wang and Q.S. Xue, 2005. Efficient non-interactive deniable authentication protocols. Proceedings of the 5th International Conference on Computer and Information Technology, Sept. 21-23, IEEE Computer Society Washington, DC. USA., pp: 673-679.

Raimondo, M.D. and R. Gennaro, 2005. New approaches for deniable authentication. Proceedings of the 12th ACM Conference on Computer and Communications Security, Nov. 7-11, ACM Press, New York, pp: 112-121.

Shao, Z., 2004. Efficient deniable authentication protocol based on generalized ElGamal signature scheme. Comput. Standards Interfaces, 26: 449-454.

Shi, Y. and J. Li, 2005. Identity-based deniable authentication protocol. Elect. Lett., 41: 241-242.

Song, D.X., 1999. Athena: A new efficient automatic checker for security protocol analysis. Proceedings of the 12th IEEE Workshop on Computer Security Foundations, June 28-30, IEEE Computer Society, Washington, DC., pp: 192-202.