On the Security of three Versions of the WAI Protocol in Chinese WLAN Implementation Plan

Qiang Tang*
Département d'Informatique, École Normale Supérieure
45 Rue d'Ulm, 75230 Paris Cedex 05, France
tang@di.ens.fr

April 3, 2007

Abstract

In this paper we investigate the security properties of three versions of the WAI protocol in Chinese WLAN implementation plan. We first revisit the security analysis that has been done to the version 1 and 2. we show that the security proof given by Li, Moon, and Ma is incorrect and the alternative protocol EWAP of Zhang and Ma is insecure. We further analyse the third version of the WAI protocol and prove its security in the Canetti-Krawczyk model. In addition, we also provide some practical security analysis of this version.

1 Introduction

Key establishment plays a fundamental role in enabling other security services, such as symmetric encryption and message authentication. The modern study of key establishment protocols can be traced back to the seminal work of Needham and Schroeder [28]. After the invention of public key cryptography by Diffie and Hellman [19], research into key establishment has grown rapidly, especially in the two-party setting, and a considerable number of protocols and security models have been proposed. Some of well known protocols include those in [6, 7, 8, 9, 11, 10, 14, 15, 16, 19]), however, we stress that it is impossible for us to enumerate all existing protocols. A number of key establishment protocols have been standardised by ISO

*The work was partially done when the author was a full-time Ph.D student at Royal Holloway, University of London.

[22, 23, 24] and IEEE [20, 21]. Summaries of key establishment protocols can be found in [12, 27].

In the literature, researchers have attempted to analyse key establishment protocols using a number of analysis methods such as heuristic analysis, formal method, and complexity-theoretic analysis. The first complexity-theoretic security model for key establishment was proposed by Bellare and Rogaway [6]. Later, this security model has been extended in a number of papers [5, 7, 11, 13, 17, 18, 25]. These security models use an indistinguishability-based approach to evaluate the session key security, i.e., a key establishment protocol is said to achieve session key security if it is infeasible for any attacker to distinguish between the session key and a randomly chosen string. In contrast to the indistinguishability-based approach, the simulatability-based approach is also widely used in the literature (e.g. [5, 29, 17]). When this approach is employed, an ideal functionality for key establishment is first defined, where the attacker's capabilities are highly restricted (compared with that in real-world), then a key establishment protocol is said to achieve session key security if it is infeasible to distinguish between an ideal-world execution of the protocol and a real-world execution, where the attacker's capabilities model the threats to key establishment protocols in practice.

The Canetti-Krawczyk model [17] is a well-known indistinguishability-based model for two-party key establishment protocols. In this model, a modular construction of key establishment protocols is also proposed and a simulatability-based approach is used to define the security.

1.1 Related work

Wireless Authentication and Privacy Infrastructure (WAPI) is the security mechanism in the Chinese Wireless LAN standard [1]. WAPI has two sub-modules: Wireless Authentication Infrastructure (WAI) and Wireless Privacy Infrastructure (WPI). The WAI protocol realise the functionality of authentication and key establishment between mobile Stations (STA) and Access Points (AP), while the WPI works on top of WAI to provide security guarantees for data communication. Until now, the WAI protocol has evolved through three different versions [1, 3, 4, 2].

Zhang and Ma [31] show that the first version of the WAI protocol [1] is not secure in the Canetti-Krawczyk model. They also proposed a protocol EWAP as an alternative, which is claimed to be secure in the Canetti-Krawczyk model. Li, Moon, and Ma [26] claimed that the second version of the WAI protocol is secure in the Canetti-Krawczyk model.

In this paper, we show that the EWAP protocol does not guarantee key authentication and entity authentication properties, therefore, it is not secure in the Canetti-Krawczyk model (and any other well-known security model for key establishment protocols). We shown that the second version of the WAI protocol is insecure in the Canetti-Krawxzyk model, which means that the security analysis of Li, Moon, and Ma is wrong.

Recently, the third version of the WAI protocol has been published [4, 2]. This new protocol adopts a Diffie-Hellman key exchange over a group based on elliptic curves. We prove that this protocol is secure in the Canetti-Krawczyk model.

1.2 Organisation

The rest of this paper is organised as follows. In Section 2 we review the second version of the WAI protocol and show that it is not secure in the Canetti-Krawczyk model. In Section 3 we show the EWAP protocol proposed by Zhang and Ma suffers from serious attacks against key authentication and entity authentication properties. In Section 4, we review the third version of the WAI protocol and prove that it is secure in the Canetti-Krawczyk model. In Section 5 we conclude the paper.

2 The Second Version of the WAI protocol

2.1 Description of the protocol

ASU is a trusted third party for STA and AP, and it generates a sign/verification key pair (pk'_{asu}, sk'_{asu}) for a signature scheme $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$. STA possesses a public/private key pair (pk_{sta}, sk_{sta}) for a public-key encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$. AP possesses a public/private key pair (pk_{ap}, sk_{ap}) of the same encryption scheme, and a sign/verification key pair (pk'_{ap}, sk'_{ap}) for the same signature scheme. In addition, the system has two hash functions $\mathsf{H}_1 : \{0,1\}^\ell \longrightarrow \{0,1\}^{2\ell}$ and $\mathsf{H}_2 : \{0,1\}^\ell \times \{0,1\}^\ell \longrightarrow \{0,1\}^\ell$.

When STA and AP wish to authenticate each other and establish a shared secret session key, they perform as follows:

1. AP sends an authentication request to STA, where the request only contains the type information "0" which means the message is a request.

2. After receiving the authentication request from AP, STA sends its certificate $Cert_{sta}$ and a timestamp t_{sta} to AP.

3. After receiving $Cert_{sta}$ and t_{sta}, AP sends $(Cert_{sta}, Cert_{ap}, t_{sta}, \sigma_{ap})$ to ASU, where $\sigma_{ap} = \mathsf{Sign}(Cert_{sta}||Cert_{ap}||t_{sta}; sk_{ap})$.

4. After receiving $(Cert_{sta}, Cert_{ap}, t_{sta}, \sigma_{ap})$, ASU first checks that

$$\mathsf{Verify}(Cert_{sta}||Cert_{ap}||t_{sta}, \sigma_{ap}; pk_{ap}) = 1$$

   If the check succeeds, ASU sends $(R_{sta}, R_{ap}, \sigma_{asu})$ to AP, where R_{sta} is the validation result of $Cert_{sta}$, R_{ap} is the validation result of $Cert_{ap}$, and $\sigma_{asu} = \mathsf{Sign}(R_{sta}||R_{ap}; sk_{ap})$. Otherwise, ASU aborts.

5. After receiving $(R_{sta}, R_{ap}, \sigma_{asu})$, AP checks the validation result. If the check succeeds, itforwards it to STA who will check the validation result. Otherwise, it aborts.

6. AP randomly selects $r_1 \in \{0,1\}^{\ell}$ and sends (SPI, c_1, σ_1) to STA, where SPI is the security parameter index, $c_1 = \mathsf{Enc}(r_1; pk_{sta})$, and $\sigma_1 = \mathsf{Sign}(SPI||c_1; sk_{ap})$.

7. After receiving (SPI, c_1, σ_1), STA sends (c_2, σ_2) to AP, where $r_1 = \mathsf{Dec}(c_1, sk_{sta})$, $r_2 \in \{0,1\}^{\ell}$ is randomly chosen, k_1 are the first half and k_2 is the second half of $\mathsf{H}_1(r_1 \oplus r_2)$, $c_2 = \mathsf{Enc}(r_2, pk_{ap})$, and $\sigma_2 = \mathsf{H}_2(k_1, SPI||c_2)$. In addition, STA sets k_1 as the session key.

8. After receiving (c_2, σ_2), AP computes $r_2 = \mathsf{Dec}(c_2, sk_{ap})$, computes k_1, k_2 in the same way as STA, and then checks σ_2. If the check succeeds, AP accepts k_1 as the session key; otherwise, AP aborts.

The first 5 steps is regarded as the authentication sub-protocol which is used for STA and AP to authenticate each other, and the rest is regarded as the key agreement protocol which enables STA and AP to establish a session key.

2.2 Security in the Canetti-Krawczyk model

Li, Moon, and Ma [26] claimed that the second version of the WAI protocol is secure in the Canetti-Krawczyk model. However, we observe that, in their security analysis, neither the Canetti-Krawczyk model is well interpreted nor the security analysis is carried out with a rigorous security reduction. We show below, the WAI protocol is insecure in the Canetti-Krawczyk model.

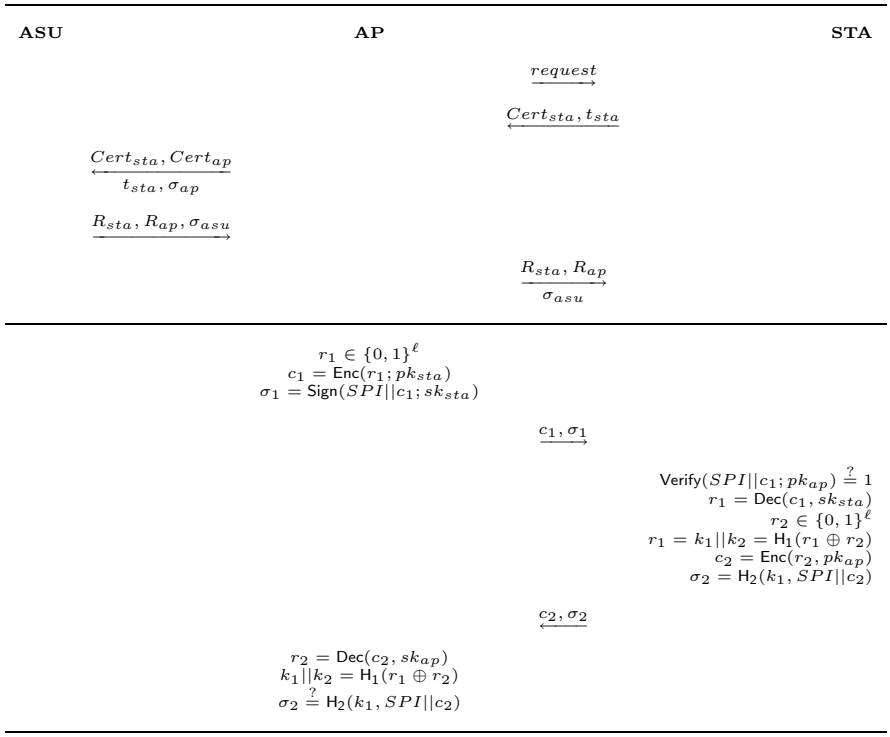

Figure 1: The Second Version of the WAI Protocol

2.2.1 Review of the Canetti-Krawczyk model

Let the user set be U_i $(1 \leq i \leq N)$. In the Canetti-Krawczyk model, an attacker may have access to the following types of oracle queries.

- activate, which on the input of $(ID_i, sid_i, ID_j, role)$, where sid_i a unique session identifier and $role$ is either "initiator" or "responder", creates an oracle $\Pi_{i,j}^{sid_i}$ to starts a session with U_j.
- send, which, on the input of an active oracle $\Pi_{i,j}^{sid_i}$ and a message m, delivers m to $\Pi_{i,j}^{sid_i}$.
- corrupt, which, on the input of any user identifier ID_i, returns U_i's long-term private key(s) and ephemeral internal states of U_i's active oracles.

- session-state-reveal, which, on the input of ID_i and a session identifier sid_i (belonging to an active oracle), returns $\Pi_{i,j}^{sid_i}$'s ephemeral internal state.

- session-output-reveal, which, on the input of an accepted oracle $\Pi_{i,j}^{sid_i}$, returns the session key possessed by this oracle.

- test-session, which, on the input of a fresh oracle $\Pi_{i,j}^{sid_i}$ (see the definition below), returns a string which is computed as follows: choose a random bit b from the set $\{0,1\}$, return the session key if $b = 1$, otherwise return a random string from the session key space.

In the Canetti-Krawczyk model, two oracles $\Pi_{i,j}^{sid_i}$ and $\Pi_{j,i}^{sid_j}$ are partnered if their roles are different and $sid_i = sid_j$. An oracle $\Pi_{i,j}^{sid_i}$ is defined to be complete if the protocol execution has successfully ended.

An oracle $\Pi_{i,j}^{sid_i}$ is defined to be unexposed if the following two requirements are satisfied:

1. Neither $\Pi_{i,j}^{sid_i}$ nor its partner oracle has been issued any session-output reveal or session-output reveal query;

2. Neither U_i nor U_j has been issued any corrupt query before $\Pi_{i,j}^{sid_i}$ and its partner oracle accept.

Note that the concept of "session (key) expiration" is also introduced in the original paper, however, it is easy to check that we can omit it here without affecting the security definition.

The attack game for *session key security* is carried out between a two-stage polynomial-time attacker $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ and a hypothetical challenger $\mathcal{C}$, as follows.

1. Setup: The challenger $\mathcal{C}$ generates both the public system parameters $param_1$ and private system parameters $param_2$.

2. Phase 1: The attacker runs $\mathcal{A}_1$ on the input of $param_1$. $\mathcal{A}_1$ can make the following types of queries: activate, send, session-state reveal, session-output reveal, and corrupt. $\mathcal{A}_1$ terminates by making a test query on the input of a complete, unexpired, and unexposed oracle $\Pi_{i,j}^{sid_i}$. In addition, $\mathcal{A}_1$ also outputs some state information *state*.

3. Challenge: The challenger $\mathcal{C}$ returns the output of test$(\Pi_{i,j}^{sid_i})$.

4. Phase 2: The attacker runs $\mathcal{A}_2$ on the input of *state* and the output of the challenger. $\mathcal{A}_2$ can make the same types of query as $\mathcal{A}_1$ in step 1, but no session-output-reveal query to $\Pi_{i,j}^{sid_i}$ or its partner oracle. $\mathcal{A}_2$ terminates by outputting a guess bit b'.

At the end of this game, the attacker wins if $b' = b$, i.e., the attacker's advantage is defined to be $|Pr[b = b'] - \frac{1}{2}|$.

If the attacker is not allowed to corrupt U_i and U_j in Phase 2, then (perfect) forward secrecy is not modelled.

Definition 1. *A key establishment protocol is defined to be* SK *secure if it achieves the following properties:*

1. *If two uncorrupted oracles have matching sessions, then they accept and compute the same session key.*

2. *Any polynomial-time attacker's advantage in the attack game for session key security is negligible.*

2.2.2 Insecurity in the Canetti-Krawczyk Model

We show that, in the Canetti-Krawczyk model (without modelling forward secrecy), the second version of the WAI protocol is insecure, i.e., an active attacker can obtain the session key in any target session by issuing session-state-reveal in other sessions.

Suppose that, in a session identified by SPI, in step 6, AP sends $c_1 = \mathsf{Enc}(r_1; pk_{sta})$, and $\sigma_1 = \mathsf{Sign}(SPI||c_1; sk_{sta})$ to STA, and in step 7, STA sends $c_2 = \mathsf{Enc}(r_2, pk_{ap})$, and $\sigma_2 = \mathsf{H}_2(k_1, SPI||c_2)$ to AP. The attacker mounts the attack as follows in two steps:

1. The attacker obtains the long-term private keys of the access point AP$'$ (different from AP) and the station STA$'$ (different from STA).

2. In a subsequent session identified by SPI' for AP$'$ (impersonated by the attacker) and STA, in step 6, the attacker sends $c_1' = c_1$ and $\sigma_1' = \mathsf{Sign}(SPI'||c_1'; sk_{ap'})$ to STA. It is straightward to verify that STA will succeed in verifying the attacker's message. The attacker then corrupts STA's session and obtains r_1.

3. In another subsequent session identified by SPI'' for AP and STA$'$ (impersonated by the attacker), after receiving $c_1'' = \mathsf{Enc}(r_1''; pk_{ap'})$ and $\sigma_1'' = \mathsf{Sign}(SPI''||c_1''; sk_{sta'})$ from AP in step 7, the attacker sends $c_2'' =$

c_2 and σ_2'' to AP, where σ_2'' is randomly chosen from the appropriate domain. The attacker then corrupts AP's session and obtains r_2.

4. With r_1 and r_2, the attacker can easily compute the session key belonging to the session identified by SPI.

It is clear that the attacker has played a valid game for session key security in the Canetti-Krawczyk model, so that it implies that the second version of the WAI protocol is insecure in this model.

2.3 Further Security Analysis

Besides the insecurity result in the previous section, we have the following additional comments on the WAI protocol.

- It is easy to see that the protocol does not achieve perfect forward secrecy, which means that, with the private keys of AP and STA, any attacker can recompute the session key using eavesdropped protocol messages. This has been noted by Li, Moon, and Ma [26]. In fact, this protocol cannot guarantee forward secrecy in some circumstances, where the attacker may have access to the ephemeral state of STA or AP.

  As an example, we show that, if the attacker has compromised the private keys of AP, then it able to recover AP's session keys in past sessions. Suppose that, in a session identified by SPI, in step 6, AP sent $c_1 = \mathsf{Enc}(r_1; pk_{sta})$, and $\sigma_1 = \mathsf{Sign}(SPI||c_1; sk_{sta})$ to STA, and in step 7, STA sent $c_2 = \mathsf{Enc}(r_2, pk_{ap})$, and $\sigma_2 = \mathsf{H}_2(k_1, SPI||c_2)$ to AP.

  In a subsequent session identified by SPI' for AP$'$ (impersonated by the attacker) and STA, in step 6, the attacker sends $c_1' = c_1$ and $\sigma_1' = \mathsf{Sign}(SPI'||c_1'; sk_{ap'})$ to STA. It is straightward to verify that STA will succeed in verifying the attacker's message. The attacker then corrupts STA's session and obtains r_1. Since the attacker can decrypt c_2 to obtain r_2, then it can compute the session key k_1.

- The protocol allows STA to fully control the session key, which means that STA can force the session key to be any pre-defined value k_1^*, where $k_1^*||k_2^* = \mathsf{H}_1(r^*)$ and r^* can be any value. Note that the session key is computed based on $r_1 \oplus r_2$, hence, STA can chooses r_2 to force $r_1 \oplus r_2$ to be a specific value. However, it is worth noting that, given H_1 is pre-image resiatant, it is hard to compute force the session key to be a randomly chosen value $k_1^\dagger$.

3 Analysis of an Alternative Protocol

Zhang and Ma [31] show that the first version of the WAI protocol [1] is not secure in the Canetti-Krawczyk model, and proposed an alternative, namely the EWAP protocol. They claimed that the EWAP protocol is secure in the Canetti-Krawczyk model. However, we show that their protocol is not secure in the Canetti-Krawczyk model (and any other well-known security model).

3.1 Description of the EWAP protocol

The system chooses two primes p, q satisfying $q|p-1$, and a generator g of a multiplicative group of order q in $\mathbb{Z}_p^*$, and a MAC function MAC. Let Alice and Bob be two users which possess identity ID_a and ID_b, respectively. Bob possesses a sign/verification key pair (pk_b, sk_b) for a signature scheme $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$.

If Alice and Bob wish to establish a shared secret key in a session identified by sid, then they perform as follows:

1. Alice randomly chooses $x \in \mathbb{Z}_q$, and sends (ID_a, sid, g^x) to Bob.
2. After receiving (ID_a, sid, g^x), Bob randomly chooses $y \in \mathbb{Z}_q$, and sends (ID_b, sid, g^y, σ) and $\mathsf{MAC}(K_{mk}; s, g^y)$ to Alice, where $K_{mk} = g^{xy}$ and $\sigma = \mathsf{Sign}(ID_b||sid||g^y||g^x||ID_a; sk_b)$.
3. After receiving (ID_b, s, g^y, σ), Alice verifies the signature and continues if the signature is valid; otherwise terminates. Alice then sends $\mathsf{MAC}(K_{mk}; sid||g^x)$ to Bob.
4. After receiving $\mathsf{MAC}(K_{mk}; sid||g^x)$, Bob checks the MAC value and accepts if the value is valid; otherwise terminates.

At the end of the protocol, Alice and Bob set the session key as K_{mk}.

3.2 Analysis of the EWAP Protocol

Zhang and Ma [31] claimed that the EWAP protocol is secure in the Canetti-Krawczyk model based on the DDH assumption, given that the signature scheme is secure against chosen message attacks and the MAC function is secure. However, as we show below, the protocol is not secure in the Canetti-Krawczyk model, and in fact, it achieves neither key authentication nor entity authentication. The attack is depicted in the Figure 2.

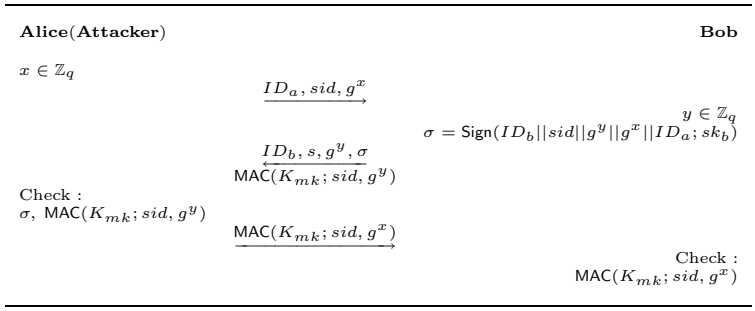

Figure 2: Attack against the EWAP Protocol

At the end of the attack, the attacker succeeds in impersonating Alice and obtains the session key possessed by Bob, therefore, the protocol does not achieve key authentication and entity authentication. In fact, this attack is due to the lack of authentication of messages from Alice and the improper use of the session key.

4 The third version of the WAI Protocol

4.1 Description of the protocol

For the simplicity of description, we assume that ASU is a trusted third party for STA and AP, and it generates a sign/verification key pair (pk_{asu}, sk_{asu}) for a signature scheme $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$. STA and AP possess the public/private key pair (pk_{sta}, sk_{sta}) and (pk_{ap}, sk_{ap}) for the same signature scheme, respectively. Let $\mathbb{G}$ be a additive group of prime order p based on an elliptic curve and P is a generator of $\mathbb{G}$. In addition, the system has two hash function $\mathsf{H}_1 : \{0,1\}^* \longrightarrow \{0,1\}^{256}$ and $\mathsf{H}_2 : \{0,1\}^* \longrightarrow \{0,1\}^{256}$.

When STA and AP wish to authenticate each other and establish a shared secret base key, they perform as follows:

1. AP sends an authentication request m_1 to STA, where

$$m_1 = (flag_1, A_{id}, ID_{asu}, Cert_{ap}, param),$$

   $param$ is a description of $(\mathbb{G}, p, P)$, $flag_1$ is a 8-bit string, and A_{id} is an authentication identifier. If the first bit of $flag_1$ is 0, then A_{id} is randomly chosen by AP, otherwise, A_{id} is set to be the pre-set value.

   If the first bit of $flag_1$ is 0, the protocol is run in a security association establishment mode. In this case, ASU is involved in the protocol

execution and responsible for checking the validaty of the certificates, as in step 3 and 4. If the first bit of $flag_1$ is 1, the WAI protocol is run in a base key update mode, where ASU is not required. In this case, the step 3 and 4 are omitted, and STA and AP locally check the validaty of the cetificates.

2. After receiving the request m_1 from AP, STA sends m_2 to AP, where $flag_2$ is some flag information, N_{sta} is a 256-bit nonce, x is randomly chosen from $\mathbb{Z}_p$, and

$$m_2 = (flag_2, A_{id}, N_{sta}, xP, ID_{ap}, Cert_{sta}, param, ID_{asu}, \sigma_{sta}),$$

$$\sigma_{sta} = \mathsf{Sign}(flag_2||A_{id}||N_{sta}||xP||ID_{ap}||Cert_{sta}||param||ID_{asu}; sk_{sta}).$$

3. After receiving m_2 from STA, AP sends m_3 to ASU, where $flag_3$ is the concatenation of the MAC adresses of AP and STA, and N_{ap} is a 256-bit nonce, and

$$m_3 = (flag_3, N_{ap}, N_{sta}, Cert_{sta}, Cert_{ap}, ID_{asu}).$$

4. After receiving m_3 from AP, ASU sends m_4 to AP, where $flag_4$ is some flag information, R_{sta} indicates whether or not $Cert_{sta}$ is valid, and R_{ap} indicates whether or not $Cert_{ap}$ is valid

$$m_4 = (flag_4, R_{sta,ap}, \sigma_{asu}),$$

$$R_{sta,ap} = \mathsf{Sign}(flag_4||N_{sta}||N_{ap}||R_{sta}||Cert_{sta}||R_{ap}||Cert_{ap}; sk_{asu}).$$

5. After receiving m_4 from ASU, AP checks the validation result and the siganture σ_{sta}. If both checks succeed, AP sends m_5 to STA, where R_{access} is the authentication result,

$$m_5 = (flag_5, N_{sta}, N_{ap}, R_{access}, xP, yP, ID_{ap}, ID_{sta}, R_{sta,ap}, \sigma_{ap}),$$

$$\sigma_{ap} = \mathsf{Sign}(flag_5||N_{sta}||N_{ap}||R_{access}||xP||yP||ID_{ap}||ID_{sta}||R_{sta,ap}; sk_{ap}).$$

Otherwise, AP aborts. AP sets the base key K and A_{id} as

$$K||A_{seed} = \mathsf{H_1}(xyP||N_{sta}||N_{ap}||str),\ A_{id} = \mathsf{H_2}(A_{seed}),$$

where $str =$ "base key expansion for key and additional nonce".

6. After receiving m_5 from AP, STA checks the certificate validation result and the signature σ_{ap}. If both checks pass, STA sets the base key K and A_{id} as

$$K||A_{seed} = \mathsf{H_1}(xyP||N_{sta}||N_{ap}||str),\ A_{id} = \mathsf{H_2}(A_{seed}).$$

Otherwise, STA aborts.

4.2 Security Analysis

In the third version of the WAI protocol, the authentication identifier A_{id} plays the role of session identifier, as required in the Canetti-Krawczyk model. We have the following theorem on the security of the protocol.

Theorem 1. *The third version of the WAI protocol is* SK *secure in the Canetti-Krawczyk model and Random Oracle model based on Computational Diffie-Hellman (CDH) assumption, given that* (KeyGen, Sign, Verify) *is existentially unforgeable.*

Proof. It is straightforward to verify that the protocol satisfies the first requirement of Definition 1. Next, we prove that it satisfies the second requirement, i.e., an attacker, which plays the attack game for session key security (defined in the Canetti-Krawczyk model), has only negligible advantage. Without loss of generality, we suppose that there are most n_1 oracles invoked for AP in the game. The security proof is done through a sequence of games.

Game_0 : In this game, the challenger $\mathcal{C}$ faithfully simulates the protocol execution (answers the oracle queries of the attacker). Let the attacker's advantage be Adv_0.

Game_1 : In this game, $\mathcal{C}$ performs in the same way as in Game_0, except that it aborts in case that one of the the follwoing events occurs:

- if the tested oracle is $\Pi_{sta,ap}^{A_{id}}$, there is no oracle $\Pi_{sta,ap}^{A_{id}}$ which possesses the same parameters xP and yP as $\Pi_{ap,sta}^{A_{id}}$.
- if the tested oracle is $\Pi_{ap,sta}^{A_{id}}$, there is no oracle $\Pi_{ap,sta}^{A_{id}}$ which possesses the same parameters xP and yP as $\Pi_{sta,ap}^{A_{id}}$.

Let this event be E_1 and attacker's advantage be Adv_1. It is easy to see that E_1 is the event that the attacker has forged a signature.

Game_2 : $\mathcal{C}$ obtains a CDH challenge $(\mathbb{G}, p, P, x^*P, y^*P)$ from a CDH challenger, and performs as follows. It randomly chooses an integer $1 \leq i \leq n_1$. Let the i-th oracle be $\Pi_{ap,sta}^{A_{id}^*}$. The DH parameter of $\Pi_{ap,sta}^{A_{id}^*}$ is set to be x^*P. If there is an oracle $\Pi_{sta,ap}^{A_{id}^*}$, then the DH parameter of this oracle is set to be y^*P. $\mathcal{C}$ aborts if the tested oracle is neither $\Pi_{ap,sta}^{A_{id}^*}$ or $\Pi_{sta,ap}^{A_{id}^*}$ (if exists). In other circumstances, $\mathcal{C}$ performs in the same way as in Game_1.

If $\mathcal{C}$ does not abort, let the attacker's advantage be Adv_2. It is easy to see that, if $\mathcal{C}$ does not abort during Game_2, then its simulation is identical

to that in Game_1 so that $\mathsf{Adv}_1 = \mathsf{Adv}_2$. In addition, the probability that $\mathcal{C}$ does not aborts is $\frac{1}{n}$.

Conclusion : In a summary, we have $|\mathsf{Adv}_0 - \mathsf{Adv}_2| \leq \epsilon$, where $\epsilon = Pr[E_1]$ is negligible based on our assumption that $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ is existentially unforgeable. Suppose the attacker issues n_2 queries to the hash function H_1 with the input of the form $x_1||x_2||x_3||x_4$, where x_i $(1 \leq i \leq 4)$ are properly defined. $\mathcal{C}$ randomly choose one of these inputs and outputs x_1 as the answer to the CDH challenge. Let the event that the attacker has queried the random oracle with the input $x^*y^*P||x_2||x_3||x_4$. then we have the following probability relationships based on the assumption that H_1 is a random oracle:

$$\begin{aligned} \mathsf{Adv}_2 &\leq |Pr[E_2] + \frac{1 - Pr[E_2]}{2} - \frac{1}{2}| \\ &= \frac{Pr[E_2]}{2} \end{aligned}$$

$$\mathsf{Adv}_0 \leq \frac{Pr[E_2]}{2} + \epsilon$$

It is clear that $\mathcal{C}$ can compute x^*y^*P at least with the probability $\frac{Pr[E_2]}{n_2}$, so that if Adv_0 is non-negligible then $\mathcal{C}$'s advantage is non-negligible. As a result, the theorem gets proved. □

Remark: We have shown that the protocol is secure based on CDH assumption in the random oracle model. Alternatively, we can also prove the security based on Decisional Diffie-Hellman (DDH) assumption without random oracle model. Because of space limit, we omit the detailed proof.

Note that the above protocol is indeed *insecure* in the Bellare-Rogaway model and its variants (e.g. those in [6, 5, 25, 30]). The reason is that, the partnership in these security models are defined based mathcing conversations, so that two oracles are partnered only if they possess the same protocol transcripts (or matching conversations). Because some bits in the flags, such as $flag_1$ and $flag_2$, are not used and can be set to be any value, therefore, these bit can be modified but the protocol execution will not be affected. Hence, it is obvious that the protocol is insecure in these model.

Nontheless, we regard the third version of the WAI protocol to be secure in practice, because the attacker cannot get any information in any target session without compromising the users.

We note that the authentication identifier A_{id} is stored as an local state, therefore, it is also secret information to an attacker in the base key update phase. When the protocol execution is run in the base key update mode,

STA will reject any message from AP if A_{id} is different from that in its local storage. In practice, A_{seed} may plays an important role in preventing DoS attacks.

5 Conclusions

In this paper we have shown that the security proof of Li, Moon, and Ma for the second version of the WAI protocol is incorrect and the alternative protocol EWAP of Zhang and Ma is insecure. We have also proved that the third version of the WAI protocol is in the Canetti-Krawczyk model.

References

[1] GB 15629.11-2003. Information technology–telecommunications and information exchange between systems–local and metropolitan area networks–specific requirements–part 11: Wireless lan medium access control (mac) and physical layer(phy) specifications, 2003.

[2] GB 15629.11-2003-XG1-2006. Information technology–telecommunications and information exchange between systems–local and metropolitan area networks–specific requirements–part 11:wireless lan medium access control(mac) and physical layer(phy) specifications amendment 1, 2006.

[3] GB 15629.1102-2003. Information technology–telecommunications and information exchange between systems–local and metropolitan area networks–specific requirements–part 11: Wireless lan medium access control (mac) and physical layer(phy) specifications: Higher-speed physical layer extension in the 2.4 ghz band, 2003.

[4] WAPI Alliance. WAPI implementation plan. http://www.wapia.org/files/Guide

[5] M. Bellare, D. Pointcheval, and P. Rogaway. Authenticated key exchange secure against dictionary attacks. In B. Preneel, editor, *Advances in Cryptology – EUROCRYPT '00*, volume 1807 of *Lecture Notes in Computer Science*, pages 139–155, 2000.

[6] M. Bellare and P. Rogaway. Entity authentication and key distribution. In D. R. Stinson, editor, *Advances in Cryptology – Crypto 1993*, volume

773 of *Lecture Notes in Computer Science*, pages 110–125. Springer-Verlag, 1993.

[7] M. Bellare and P. Rogaway. Provably secure session key distribution: the three party case. In *STOC '95: Proceedings of the Twenty-Seventh Annual ACM Symposium on Theory of Computing*, pages 57–66. ACM Press, 1995.

[8] S. M. Bellovin and M. Merritt. Encrypted key exchange: Password-based protocols secure against dictionary attacks. In *SP '92: Proceedings of the 1992 IEEE Symposium on Security and Privacy*, pages 72–84. IEEE Computer Society, 1992.

[9] S. M. Bellovin and M. Merritt. Augmented encrypted key exchange: A password-based protocol secure against dictionary attacks and password file compromise. In *Proceedings of the First ACM Conference on Computer and Communications Security*, pages 244–250, 1993.

[10] S. Blake-Wilson, D. Johnson, and A. Menezes. Key agreement protocols and their security analysis. In M. Darnell, editor, *Proceedings of Cryptography and Coding, 6th IMA International Conference*, volume 1355 of *Lecture Notes in Computer Science*, pages 30–45. Springer-Verlag, 1997.

[11] S. Blake-Wilson and A. Menezes. Entity authentication and authenticated key transport protocols employing asymmetric techniques. In B. Christianson, B. Crispo, T. Lomas, and M. Roe, editors, *Proceedings of Security Protocols, 5th International Workshop*, volume 1361 of *Lecture Notes in Computer Science*, pages 137–158. Springer-Verlag, 1997.

[12] C. Boyd and A. Mathuria. *Protocols for Authentication and Key Establishment*. Springer, 2004.

[13] E. Bresson, O. Chevassut, D. Pointcheval, and J.-J. Quisquater. Provably authenticated group Diffie-Hellman key exchange. In *Proceedings of the 8th ACM Conference on Computer and Communications Security*, pages 255–264. ACM Press, 2001.

[14] M. Burmester and Y. Desmedt. A secure and efficient conference key distribution system. In A. D. Santis, editor, *Advances in Cryptology—EUROCRYPT '94*, volume 950 of *Lecture Notes in Computer Science*, pages 275–286. Springer-Verlag, 1994.

[15] M. Burmester and Y. Desmedt. A secure and efficient conference key distribution system. In A. D. Santis, editor, *Pre–Proceedings of Eurocrypt '94*, pages 279–290, 1994.

[16] M. Burmester and Y. Desmedt. A secure and scalable group key exchange system. *Inf. Process. Lett.*, 94(3):137–143, 2005.

[17] R. Canetti and H. Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In B. Pfitzmann, editor, *Advances in Cryptology — Eurocrypt 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 453–474. Springer-Verlag, 2001.

[18] L. Chen and C. Kudla. Identity based authenticated key agreement protocols from pairings. In *Proc. of the 16th IEEE Computer Security Foundations Workshop — CSFW 2003*, pages 219–233. IEEE Computer Society Press, 2003.

[19] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, 1976.

[20] Institute of Electrical and Electronics Engineers, Inc. *IEEE P1363 Standard Specifications for Public-Key Cryptography*, 2000.

[21] Institute of Electrical and Electronics Engineers, Inc. *IEEE P1363.2 draft D20, Standard Specifications for Password-Based Public-Key Cryptographic Techniques*, March 2005.

[22] International Organization for Standardization. *ISO/IEC FCD 11770–2, Information technology — Security techniques — Key management — Part 2: Mechanisms Using Symmetric Techniques*, 1996.

[23] International Organization for Standardization. *ISO/IEC FCD 11770–3, Information technology — Security techniques — Key management — Part 2: Mechanisms Using Asymmetric Techniques*, 1999.

[24] International Organization for Standardization. *ISO/IEC FCD 11770–4, Information technology — Security techniques — Key management — Part 4: Mechanisms based on weak secrets*, December 2004.

[25] C. Kudla and K. G. Paterson. Modular security proofs for key agreement protocols. In B. K. Roy, editor, *Advances in Cryptology — ASIACRYPT 2005*, volume 3788 of *Lecture Notes in Computer Science*, pages 549–565. Springer-Verlag, 2005.

[26] X. Li, S. Moon, and J. Ma. On the security of the authentication module of chinese wlan standard implementation plan. In J. Zhou, M. Yung, and F. Bao, editors, *Proceedings of the 4th International Conference on Applied Cryptography and Network Security*, volume 3989 of *Lecture Notes in Computer Science*, pages 340–348, 2006.

[27] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1997.

[28] R. Needham and M. Schroeder. Using encryption for authentication in large networks of computers. *Commun. ACM*, 21(12):993–999, 1978.

[29] V. Shoup. On formal models for secure key exchange. Technical report, IBM Research Report RZ 3120, 1998.

[30] M. Strangio. On the resilience of key agreement protocols to key compromise impersonation. In A. Atzeni and A. Lioy, editors, *Public Key Infrastructure, Third European PKI Workshop: Theory and Practice, EuroPKI 2006, Turin, Italy, June 19-20, 2006, Proceedings*, volume 4043 of *Lecture Notes in Computer Science*, pages 233–247. Springer, 2006.

[31] F. Zhang and J. Ma. Security analysis on chinese wireless lan standard and its solution. In *34th International Conference on Parallel Processing Workshops*, pages 436–443. IEEE Computer Society, 2005.