Security Analysis and Improvements on WLANs

Li Wang
Clayton School of IT, Melbourne, Australia
Email: Li.Wang@monash.edu

Bala Srinivasan and Nandita Bhattacharjee
Clayton School of IT, Melbourne, Australia
Email: {Sri; Nandita.Bhattacharjee}@monash.edu

Abstract—IEEE 802.11i standard defines the security specifications of IEEE 802.11 series Wireless Local Area Network (WLAN). It is the replacement of the old security standard named Wired Equivalent Privacy (WEP), and it aims to eliminate all known attacks against WEP. It certainly provides solutions to the confidentiality, mutual authentication and integrity aspects of the WLAN security but not the availability aspect. Many researchers have shown that IEEE 802.11i standard cannot prevent various Denial of Service (DoS) attacks including de-authentication, disassociation and memory/CPU DoS attacks. Besides, IEEE 802.11i has reserved the pre-shared key (PSK) mode of WEP for flexibility and backward compatibility. However, the PSK mode in IEEE 802.11i standard fails to provide sufficient security to prevent offline dictionary attacks and internal attacks. In this paper, we present our solutions that can effectively improve the security of IEEE 802.11i. For memory/CPU DoS attack against 4-way Handshake protocol, we propose an alternative Enhanced 3-way Handshake protocol which can effectively prevent this attack and can save computation cost compared to the original one. For the vulnerability in PSK mode, we proposed a novel Elliptic curve Diffie–Hellman (ECDH) protocol to prevent the offline dictionary attacks and internal attacks. The formal proofs of above two proposed protocols are also provided using Protocol Composition Logic (PCL).

Index Terms—IEEE 802.11i, DoS attack, offline dictionary attack, inside attack, PSK, WLAN, 4-way Handshake

I. INTRODUCTION

Wireless Local Area Network (WLAN) is one kind of wireless networks and which is a wireless network in limited area where laptop and mobile devices can connect to it freely [1]. WLAN is popular due to its flexibility, mobility and portability and are widely deployed in schools, commercial organizations or in home uses. However, a WLAN suffers from all the constraints of wireless networks including low transmit speed and bandwidth, memory and processing power which limits the implementation of security approaches as can be implemented in wired LANs which includes public-key ciphers. Moreover, due to the feature that signals are transmitted in the air, an adversary can easily monitor and intercept all signals. Thus, robust, efficient and effective security measurements are essential for all WLANs. IEEE 802.11i is the security standard for 802.11 series WLAN, and it provides enhanced security in Medium Access Control (MAC) layer. It well defines the solutions to the confidentiality, mutual authentication and integrity aspects of the WLAN security. Nevertheless, IEEE 802.11i has reserved the pre-shared key (PSK) mode of WEP for flexibility and backward compatibility. However, the PSK mode in IEEE 802.11i standard fails to provide sufficient security to prevent offline dictionary attacks and internal attacks. In this paper, we present our solutions that can effectively improve the security of IEEE 802.11i. For memory/CPU DoS attack against 4-way Handshake protocol, we propose an alternative Enhanced 3-way Handshake protocol which can effectively prevent this attack and can save computation cost compared to the original one. For the vulnerability in PSK mode, we proposed a novel Elliptic curve Diffie–Hellman (ECDH) protocol to prevent the offline dictionary attacks and internal attacks. The formal proofs of above two proposed protocols are also provided using Protocol Composition Logic (PCL).

II. SECURITY ANALYSIS OF IEEE 802.11i

In this section, we shall introduce the IEEE 802.11i standard; next we shall identify 3 major vulnerabilities existing in IEEE 802.11i.

A. Overview of IEEE 802.11i standard

IEEE 802.11i defines two kinds of authentication approaches. One approach is based on IEEE 802.1x authentication and the other is the pre-shared key (PSK) mode. IEEE 802.11x is port-based authentication without specifying any particular encryption or cipher. For example, IEEE 802.11 may work within Extensible Authentication Protocol (EAP) using one of its various encryption methods for authentication such as EAP-TLS and EAP-MD5. PSK mode was firstly deployed in WEP and is also reserved in IEEE 802.11i for backward compatibility and flexibility. It is widely used in small and ad hoc WLANs for its simple configurations such as...
in small companies and home WLANs. Fig.1 shows the establishment procedure of IEEE 802.11i.

**Stage 1: Network discovery**

In this stage, a supplicant finds the network by monitoring the beacon frame broadcast by Access Point (AP) or sending a probe request frame to get the response from the existing AP. Then, the supplicant will find a list of available APs which can be chosen for later communication.

**Stage 2: IEEE 802.11 authentication and association**

In this stage, the supplicant will choose an AP from the list to authenticate and associate. Supplicant and AP will negotiate the security parameters during the association. Open System Authentication [1] used here is only for backward compatibility without any security. Though supplicant and the AP is authenticated and associated during this stage, the security achieved here is not adequate and more secure mutual authentication will be held in the next stage.

**Stage 3: IEEE 802.1x authentication**

In this stage, supplicant will authenticate to authenticator using upper layer authentication approaches like EAP-TLS and EAP-MD5. The Master Session Key (MSK) is generated during this stage which is used for generating the Pairwise Master Key (PMK). This stage provides strong mutual authentication using EAP based authentication. Note if PSK mode or cached PMK is used this stage will be skipped.

**Stage 4: 4-Way Handshake protocol**

In this stage, 4-Way Handshake will be performed to generate the fresh Pairwise Transient Key (PTK). 4-Way Handshake is an essential part in IEEE 802.11i as it confirms the supplicant and authenticator can generate a fresh PTK. During this stage, nonce plays an important role to prevent various attacks. Fig.2 shows the process of 4-way Handshake where AA and SPA is the MAC address of authenticator and supplicant respectively, ANonce and SNonce are two nonce from the authenticator and supplicant respectively, SN represents the sequence number and MIC is Message Integrity Code. Besides, PTK is generated as:

\[
PTK = PRF-n(PMK, "Pairwise key expansion", Min(AA, SPA) || Max(AA, SPA) || Min(ANonce, SNonce) || Max(ANonce, SNonce))
\]

Where PRF is Pseudo Random Function and \( n \) means the number of bits required.

**Stage 5: Group key handshake**

This stage is optional and will be performed after the 4-Way Handshake if it is necessary. Group key handshake provides fresh Group Transient Key (GTK) which is used to secure multicast applications which are requested by supplicant.

**Stage 6: Secure Data communication**

After all necessary stages above, a secure channel of data communication is established. The supplicant can connect to the WLAN enjoying the secure communication protected by fresh PTK or GTK. Though 802.11i provide enhanced security over old version of WLAN security standard, there are still many threats. We will analyse these possible threats at each stage.

For stage 1 and 2, many attacks can be mounted including passive and active eavesdropping, interception, deletion, modification, DoS attacks. The reason for it is because there is no protection of data frames as the supplicant and authenticator have not shared a secret yet. The management and control frames are transmitted without authentication, so the adversary can easily forge the de-authentication and disassociation frames and send forged frames to the supplicants to disconnect them. In other words, there is no improvement on 802.11i over WEP in the first two stages. The vulnerability that easy to suffer DoS attacks are still needed to be solved.

In the stage 3, the data frames will be protected by the upper layer authentication protocols. It should be considered to be secure though the strength of security.
depends on the upper layer authentications [12-14]. The other situation is that stage 3 is skipped as cached PMK or PSK is used. In this situation, stage 4 will be performed to derive the fresh PTK from the cached PMK or PSK.

In the stage 4, the data frames used are in the clear but there are several mechanisms to protect them from being forged such as the use of nonce, MIC and Sequence Number (SN). These mechanisms can mitigate or prevent the attacks which are modification, deletion, interception and replay attacks. However, 4-Way Handshake has a deadly vulnerability which enables DoS attacks to be mounted easily [8]. Moreover, another off-line dictionary attack on the PSK can easily be mounted by collecting the information of 4-Way Handshake as no data is encrypted. However, this attack is not the vulnerability of 4-Way Handshake but the vulnerability in the stage 2 and the reserved PSK mode.

After the stage 3 and 4, supplicant and authenticator have been mutually authenticated with 802.1x protocol, so eavesdropping, interception, modification will be no longer be threats as the data frames transmitted will be protected by the fresh PTK generated in Stage 4. Above all, when it comes to stage 4, passive and active eavesdropping, modification, interception, deletion cannot cause much damage as the data frames will be protected by fresh PTK or GTK.

B. Identified vulnerabilities

From above analysis, we identified 3 major vulnerabilities existing in IEEE 802.11i, which are

1) No protection on the management and control frames in stage 1 and 2.

Many attacks are derived from this vulnerability including DoS attacks by broadcasting de-authentication and disassociation frames, modification, passive and active eavesdropping and further attacks like man-in-the-middle (MitM) attacks and session hijacking. Fortunately, IEEE 802.11 group has developed a new standard named IEEE 802.11w [15] to solve this vulnerability. Thus, we will not address this vulnerability in this paper.

2) DoS attacks against 4-way Handshake.

This attack is caused when the first message in 4-way Handshake is sent without protection [16]. Therefore, an adversary can forge the first message and overload the supplicant with messages which will induce the de-authentication of supplicant. We will analyse this vulnerability and present our proposed approach to handle this problem in Part III.

3) Offline dictionary and internal attack against PSK mode.

In PSK mode, an adversary can successfully mount an offline dictionary attack by passive sniffing and then brute force attack on the passphrase if it is not long enough. What is worse, if the adversary has cracked the PSK or a legal user who shares the PSK, he/she can crack all the PTKs used by other users just by passive monitoring. That means all network traffics will be revealed just by passive monitoring. We will introduce this attack in Part 4 in detail and we also propose a novel protocol to prevent this attack.

III. DoS ATTACKS TO 4-WAY HANDSHAKES

A. Detailed of this attack

DoS attack can be launched in the 4-way Handshake as there exists no protection of Message 1 as shown in Fig.2. Supplicant cannot identify if Message 1 is from a legitimate authenticator until it receives Message 3 and verifies it, and this can cause an attack to block 4-way handshake. This problem occurs when a forged Message 1 is sent to supplicant after the supplicant has received Message 1, sent Message 2 and before it receives Message 3. Since Message 1 is transmitted in clear, the adversary can easily capture all the information including AA, SN and ANonce. The adversary only needs to modify the ANonce and send forged Message 1’ to supplicant before it receives the legitimate Message 3. Then supplicant will generate a new PTK’ corresponding to the received ANonce’ in Message 1’, and then the legitimate PTK will be updated to the wrong PTK’. Because of inconsistency in the PTKs between the supplicant and authenticator, authenticator will de-authenticate the supplicant. Thus, 4-way Handshake is blocked by just sending one forged Message 1.

One possible modification to prevent this attack is that supplicant stores all generated PTKs and uses them to verify Message 3. However, if flooding of forged Message 1s are sent to supplicant, then it has to store all the PTKs and ANonce in order to make the 4-way handshake non-blocking [8]. Then another problem arises as supplicant may not have sufficient memory to store all received PTKs and ANonce and this will cause the supplicant memory exhaustion. So it is the inherent vulnerability of 4-way Handshake. Thus for a malign intruder, he/she can always exploit this vulnerability of 4-way Handshake and mount attacks. This DoS attack is an alternative choice for adversaries to stop the authentication between supplicant and authenticator besides other DoS attacks such as deauthentication and disassociation DoS attacks. Therefore, this potential vulnerability of 4-way Handshake should be solved before practical attacks have been implemented.

B. Related Works to Improve on 4-way handshake Protocol

There are many reported solutions to mitigate DoS attacks against 4-way Handshakes. He and Mitchell [8] propose two possible solutions to solve this problem: one is to authenticate Message 1 with PMK and the other is to reuse the SNonce. The first approach uses the shared secret between supplicant and authenticator which is PMK to generate a MIC of Message 1. The other approach is to reuse the SNonce which means that when supplicant receives Message 1, it generates PTK, constructs Message 2 and sends it without storing the PTK and ANonce but storing the SNonce. This SNonce will not be updated if it receives new Message 1 before Message 3 is verified and PTK is installed. By this
approach, a supplicant only needs to store one SNonce and re-compute PTK when it receives Message 3. This approach solves the problem of memory exhaustion, but another potential problem occurs which is CPU exhaustion as if flooding of forged Message 3 are sent to supplicant then it has to keep on computing PTKs and verify them. Rango, Lentini and Marano [17] analyze that approach in detail which is reusing SNonce proposed by He and Mitchell [8], and they expand it with resource-aware variant approach to get a tradeoff between memory and CPU exhaustion. However, it does not eliminate the inherent flaw of 4-way handshake.

C. Proposed Enhanced 3-way Handshake

![Proposed Enhanced 3-way Handshake Diagram](image)

Figure 3: Proposed Enhance 3-way Handshake

DoS attacks on 4-way Handshake are caused by no protection of the Message 1 of 4-way Handshake. We propose the Enhanced 3-way Handshake as shown in Fig. 3 and the message flows are as follows:

- **Message 1:** A → S: AA, ANonce, SN, msg1;
- **Message 2:** S → A: SPA, ANonce, SNonce, SN, msg2, MIC\_PTK\{SPA, ANonce, SNonce, SN, msg2\};
- **Message 3:** A → S: AA, SNonce, SN+1, msg3, MIC\_PTK(\{AA, SNonce, SN+1, msg3\})

Where used PTK generation formula is:

PTK = PRF-n(PMK, “Pairwise key expansion”, Min(AA, SPA)||Max(AA, SPA)||SNonce) \( (2) \)

As shown in Fig. 3, supplicant computes a SNonce and and PTK when it receives Message 1, and if authenticator receives the Message 2, it will generate the PTK using (2) and then verifies MIC. If MIC is valid, then authenticator will install the PTK, construct Message 3 and send it to supplicant. Message 3 is used as a notification to supplicant that authenticator has installed PTK, so that supplicant will also install the PTK for later communication.

The main difference between the 4-way handshake and the proposed Enhanced 3-way handshake is that the later one only uses AA, SPA and SNonce for PTK generation compared to (1). Thus, for each session, a PTK is only generated upon SNonce instead of ANonce and SNonce. Besides, ANonce only plays a role to prevent replay attacks against authenticator. Furthermore, SNonce is reused in each session so that supplicant can actually pre-generate the PTK before the 3-way Handshake. Thus, it will save computational cost for supplicant in proposed protocol. However, the freshness of the PTK is also ensured since a new SNonce will be used for each session.

From above analysis, our proposed Enhanced 3-way Handshake fulfills the basic purposes of original 4-way Handshake. Besides, it can also prevent memory DoS attacks since both supplicant and authenticator only need to store one nonce which is generated by themselves. Moreover, the CPU DoS attacks cannot be mounted against both sides since supplicant can generate the PTK in advance while authenticator is protected by the ANonce. In addition, the last message in original 4-way Handshake is abandoned as it is redundant since authenticator has verified Message 2 and installed PTK. The details of this protocol can refer to the work of Wang and Srinivasan [18], and we provide formal modelling and proof of this approach in part V.

IV. Offline Dictionary and Inside Attack Against PSK Mode of WLAN

IEEE 802.11i has reserved PSK mode for backward compatibility and flexibility. However, many literatures [9-10] have shown that PSK mode is not robust and can be easily cracked by some tools (e.g. coWPAtty [9]) by offline dictionary attacks. However, PSK mode is widely used in home WLANs and ad hoc WLANs for its flexibility and easy configuration. One cause of this offline dictionary attack is that the chosen passphrase for WLAN is weak. Only those passphrases with a length more than 20 characters can effectively defend this attack [9]. But most of WLAN users will not choose such a passphrase as it is hard to memorize and inconvenient to use.

The procedure to mount this attack can be described in the following steps:

1) Firstly, collect all necessary information including Service Set Identifier (SSID) of AP, nonce used in 4-way handshake, MAC addresses of supplicant and authenticator by eavesdropping authentication messages.
2) Secondly, perform the offline dictionary attack to generate the PSK by the formula from [1]: $PSK = PBKDF2(passphrase, SSID, SSIDLength, 4096, 256)$ where PBKDF2 is the Passphrase-Based cryptography standard and it means the passphrase, SSID and length of SSID will be hashed 4096 times to generate a 256 bits PMK.
3) Thirdly, use the generated PSK to compute the related PTK by (1), once the PTK is generated use it to check the MIC code to see whether it is the correct PTK. If it is correct, then record the passphrase, otherwise redo the above three stages.
4) At last, use the derived passphrase to connect the AP.
PSK is shared by all users in the same WLAN, so an insider adversary can directly use the PSK to monitor other users’ network traffics. However, it is impossible to prevent insider attack if using symmetric ciphers as the passphrase is unique and an insider adversary can always decrypt the cipher by that passphrase.

Several works have been done to address this problem. Moskowitz suggests that passphrase should be set longer than 20. However, this suggestion can only prevent external offline dictionary attack. Miao and Striegel [19] developed a lightweight enhanced 4-way handshake by implementing the Diffie-Hellman protocol. However, the computation cost is quite high if large bits of keys are required. Xing et al [20] developed a new mechanism using Certificate Authentication (CA). However, in their approach, a server is necessary for each WLAN, which is inconvenient for small and ad hoc WLANs.

Thus, we proposed a novel Elliptical Curve Diffie-Hellman (ECDH) protocol which uses an asymmetric cipher named Elliptical Curve Cryptography (ECC) to handle above attacks. In the following section, we shall firstly introduce the ECC and ECDH; next we shall present the proposed novel ECDH protocol.

A. ECC Algorithm and ECDH protocol

ECC is a public key cryptography based on the algebraic structure of elliptic curve over finite field, and it was proposed independently by Koblitz [21] and Miller [22]. It can provide same security and functions like other public key cryptography such as RSA but with smaller keys. The strength of ECC relies on the difficulty of the Elliptic Curve Discrete Logarithm Problem (ECDLP). This problem can be explained as follows: Given two points G and Q where dG = Q (it uses scalar point multiplication in a finite group) in an elliptic curve, then it is intractable to generate d known G and Q. The sizes of elliptic curves determine the difficulty of this problem, and it is believed that a smaller group can achieve the same security level as those public key cryptographies based on the intractable of integer factorization problem such as RSA [23]. More details about ECC algorithm can refer to [21-22]. Table I shows the comparison of key length to achieve the same level of security between different ciphers, where Digital Signature Algorithm/ Diffie-Hellman (DSA/DH), RSA and ECC are public key cryptography while triple Data Encryption Standard (3DES) and Advanced Encryption Standard (AES) are block ciphers.

We can find that the length of ECC is linear increasing as the same as block ciphers while key length of other public key cryptography based on the integer factorization problem increases exponential. In addition, ECC provides the highest per bit ratio of security among all public key ciphers. Therefore, the future developments of those public key cryptographies based on integer factorization difficulty and discrete logarithm meet the bottleneck due to the increase of necessary key length to achieve the same level of the security as other block ciphers. So the advantage of ECC is more efficient than other public key ciphers for the smaller key needed, and especially, it is much suitable to be deployed in those handheld and wireless devices as it can save computation power, memory and bandwidth [23].

B. ECDH Key Exchange protocol


Before executing the ECDH key exchange, the involved two principals should share some public domain parameters which are (p, a, b, G, n, h) in the prime case or (m, f(x), a, b, G, n, h) in the binary case of which prime case is better for software implementation while binary case is better for hardware implementation. These public domain parameters are used to define the elliptic curve. The field is defined by the p and m, f(x) in the prime and binary cases, respectively. Parameter G denotes the base point of the curve. And parameters a and b denotes the coefficients of the curve while h and n denotes the cofactor and the order of G, respectively. Then for each ECDH key exchange, each principal should choose a secret key which are dA for principal A and dB for principal B. Let us assume principal A is the initiator of this protocol, then A firstly send a public key QA = dAG to B. When B receives the public key QA, it will send its public key QB = dBG to A. Finally, both A and B can generate a shared key K = dAdBG.

The ECDH has not defined an approach to authenticate the messages as the same in Diffie-Hellman protocol. In our proposed protocol, the passphrase is used to protect the messages from tampering which will be introduced in the next section.

C. A Novel Protocol Proposed to Prevent Offline Dictionary Attacks against PSK Mode of 802.11i

We proposed a PMK deriving protocol based on ECDH which can provide secure PMK exchange between supplicant and authenticator in PSK mode. Fig.4 shows our proposed protocol briefly. The parameters dA and dB represent the secret key of authenticator and supplicant, respectively. G is the base point of chosen curve, and dAG and dBG represent the public keys of ECC algorithm. Note these public keys are not known to public as they are XOR with PSK. Therefore, an adversary cannot crack ECC algorithm by cryptanalysis if he/she does not know the PSK. Besides, the adversary will never know the PSK by passive eavesdropping since there is no pattern of daG and dsG to check whether the PSK tried is correct. PSK
can be generated the same as the defined in IEEE 802.11i. After this protocol, a long term PMK is generated for both supplicant and authenticator, and this PMK will be used to derive PTK for each session. The following is the messages flows of our proposed protocol.

1) Message 1: \(A \rightarrow S: \text{PSK} \oplus d_sG, \text{msg1}\)
2) Message 2: \(S \rightarrow A: \text{PSK} \oplus d_A G, \text{PSK} \oplus d_S G, \text{msg2}, H(d_sG, \text{PSK} \oplus d_A G, \text{PSK} \oplus d_S G, \text{msg2})\)
3) Message 3: \(A \rightarrow S: \text{PSK} \oplus d_A G, \text{msg3}, H(d_A G, \text{PSK} \oplus d_A G, \text{msg3})\)
4) Message 4: \(S \rightarrow A: \text{PSK} \oplus d_S G, \text{msg4}, H(d_sG, \text{PSK} \oplus d_A G, \text{msg4})\)

![Diagram](image)

**Figure 4 Proposed PMK deriving protocol based on ECDH**

D. Security analysis

In proposed protocol, it is obviously that offline dictionary can be prevented as an adversary cannot break an ECC cipher. Besides, for an internal adversary, even if he/she knows the PSK, she cannot calculate the shared PMK as he/she does not have the secret key. Thus, he/she has to solve the ECC problem to get the PMK which is considered infeasible. Besides, external MitM attacks can be avoided as the public keys of supplicant and authenticator have been both XOR with the shared PSK. Thus, if the adversary cannot crack the PSK, then the MitM attack cannot go further beyond Message 2. However, for an internal adversary, he/she may mount a MitM attack as he/she knows the PSK. Nevertheless, MitM attacks are unsolved issue for those Diffie-Hellman based approach [19]. It is similar to another attack existing in all kinds of wireless networks which is forged APs attacks. One possible solution is binding a CA to each AP which is certificated by a public organization or vendors, and then each supplicant can verify the AP before each connection. However, our approach solves both introduced offline dictionary and insider attacks.

V. PROOF OF PROPOSED 3-WAY HANDSHAKE AND ECDH PROTOCOL

A. Introduction to Protocol Composition Logic (PCL)

We use PCL to model and prove the security properties of our proposed protocols. PCL [24-26] is a formal logic which can prove security properties of network protocols. The results of PCL include two parts which are a set of computation soundness theorems and composition theorems. Computation soundness theorem guarantees the security properties of protocol. It is used to prove the correctness of a protocol. Especially for authentication and key distribution protocols, the correctness includes two security properties which are session authentication and key secrecy. Composition theorems can prove complex protocols as the complex protocol can be built up of components proofs. It is an advantage of PCL as most of nowadays security protocols consist of various component protocols. Take an example, 802.11i security standard consist of 802.1x authentication, 4-way handshake and group handshake Protocol. By using PCL, the correctness of 802.11i can be proved by proving the correctness of its component protocols. Thus, it will reduce the complexity when proving those complex protocols. The details of PCL refer to [26].

B. Formal Proof of Proposed Enhanced 3-way Handshake

1) Modelling Enhanced 3-way Handshake

In this model, we use 3WAY: AUTH and 3WAY: SUPP respectively to represent the sequences of actions of authenticator and supplicant in the proposed Enhanced 3-way Handshake as shown in Table II. This model can be explained as follows: The \(X\) and \(Y\) represent the role of authenticator and supplicant, respectively. \(\text{Supp}\) and \(\text{Auth}\) represent the name of thread \(X\) and \(Y\), respectively. Before the handshake, the supplicant and authenticator should share a PMK and MAC address of each one which are the initial conditions of the handshake. The authenticator \(X\) firstly generates a fresh nonce \(x\) and then sends the Message 1 to supplicant \(Y\). Message 1 contains the nonce \(x\) and \(msg1\) where the term \(msg1\) represents additional information in Message1. When supplicant \(Y\) receives Message 1 which is represented as \(z\), it will do the match action to match the patterns of nonce \(x\) and \(msg1\). Then supplicant \(Y\) will generate a fresh nonce \(y\) and corresponding PTK and then send Message 2 which contains the nonce \(y\), \(msg2\), and \(\text{HASH}_{ptk}(y, \text{msg2})\). Note term \(ptk\) and \(\text{HASH}_{ptk}(y, \text{msg2})\) represents the PTK and MIC of Message 2, respectively. When authenticator \(X\) receives Message2, it will firstly match the patterns of received Message 2 and then generate the PTK and verifies the MIC. Then authenticator \(X\) will send Message 3 which contains the received nonce \(y\), \(msg3\) and \(\text{HASH}_{ptk}(y, \text{msg3})\). Note authenticator \(X\) will install the PTK after sending Message 3. Once supplicant \(Y\) received Message 3, it will verify the nonce \(y\) and \(\text{mic}\) and will install the PTK if they are valid.
two security properties as following 2 definitions:

**Definition 1**: (Key secrecy of Enhanced 3-way Handshake protocol)

The Enhanced 3-way Handshake is said to provide key secrecy of $\phi_{3\text{way,sec}}$ holds, where:

$$\phi_{3\text{way,sec}} := \text{Honest}(\bigwedge) \land \text{Honest}(\bigwedge) \supset ((\text{Has}(\bigwedge, \text{ptk}) \supset \bigwedge) = \bigwedge \land \text{Has}(\bigwedge, \text{ptk}) \land \text{Has}(\bigwedge, \text{ptk}))$$

**Definition 2**: (Session authentication of Enhanced 3-way Handshake)

The Enhanced 3-way Handshake is said to provide session authentication for the supplicant if $\phi_{3\text{way,auth}}$ holds, where

$$\phi_{3\text{way,auth}} := \text{Honest}(\bigwedge) \land \text{Honest}(\bigwedge) \supset \exists X.\text{ActionInOrder}($$

Send($X, \{ \bigwedge \bigwedge \bigwedge, \text{Message } 1\}$);
Receive($Y, \{ \bigwedge \bigwedge \bigwedge, \text{Message } 1\}$);

Send($Y, \{ \bigwedge \bigwedge \bigwedge, \text{Message } 2\}$);
Receive($X, \{ \bigwedge \bigwedge \bigwedge, \text{Message } 2\}$);

Send($X, \{ \bigwedge \bigwedge \bigwedge, \text{Message } 3\}$);
Receive($Y, \{ \bigwedge \bigwedge \bigwedge, \text{Message } 3\}$)

Definition 1 defines that if any principal holds PTK, then it must be the supplicant or the authenticator, thus it confirms the *key secrecy* property of our proposed protocol. Definition 2 defines that if the supplicant and authenticator are honest, then they should do the actions in order as shown in $\phi_{3\text{way,auth}}$ which confirms that authenticator is authenticated. Because supplicant receives Message 3 in our proposed protocol, if authenticator is authenticated by the supplicant, then it means authenticator and supplicant are mutual authenticated. From these two definitions, we can get the Theorem 1 as follows:

**THEOREM 1** (Enhanced 3-way Handshake authentication guarantee) On the execution of the supplicant role, key secrecy and session authentication are guaranteed if the formulas in Fig.5 hold. Formally, $\Gamma_{3\text{way,1}} \land \Gamma_{3\text{way,2}} \vdash \phi_{3\text{way,3-WAY:SUPP}} \land \phi_{3\text{way,auth}}$

$$\phi_{3\text{way}} := \text{Has}(\bigwedge, \text{ptk}) \land \text{Has}(\bigwedge, \text{ptk})$$

$\Gamma_{3\text{way,1}} := \text{Compu}$$

$\Gamma_{3\text{way,2}} := \text{Compu}$

$\phi_{3\text{way}}$ states that the pre-condition of proposed Enhanced 3-way Handshake protocol, which means that the supplicant and authenticator should share the PMK before execute this protocol. $\Gamma_{3\text{way,1}}$ states that any principal should not send its generated PTK to any other principals. $\Gamma_{3\text{way,2}}$ states that any honest principal in Enhanced 3-way Handshake should not act both the roles of supplicant and authenticator. The main result of Theorem 1 states that the security guarantees both supplicant and authenticator. We will show the proof of Theorem 1 in Appendix A.

**C. Formal Proof of Proposed ECDH Protocol**

1) **Modelling ECDH protocol**

The proposed ECDH PMK exchange protocol has 4 messages, and its model can be seen in Table III. This model can be explained as follows: **ECDH: AUTH** and **ECDH: SUPP** represent the sequences of actions of authenticator and supplicant in the proposed PMK exchange protocol, respectively. The initial conditions of authenticator $X$ and supplicant $Y$ are that they shares the $psk$ and other constants and variables of selected elliptic curve such as the base point $G$. Message 1 is from authenticator, and it contains encrypted authenticator’s public key “$PSK\oplus d\times G$” and additional information which are represented as $ENC_{psk}(xG)$ and msg1, respectively. In this model, we use $x$ and $y$ as the authenticator’s and supplicant’s secret key, respectively. When the supplicant receives the Message 1, it will decrypt the term $ENC_{psk}(xG)$ with the shared $psk$ as $ENC_{psk}^{-1}(xG)$ to get the authenticator’s public key $xG$. Then the supplicant can generate the PMK $pmk$ as $xyG$. After that, the supplicant will send both principals’ encrypted public keys together with a hash value to
authenticator in Message 2. The reason of sending both two public keys is that these encrypted public keys can play a role of preventing replay attacks since their values seem random to an adversary. And that hash value is generated by hashing the total Message 2 and derived pmk, so it can provide both authentication and integrity confirmation. Thus, we use \( \text{HASH}_\text{pmk}(\text{ENC}_\text{psk}(xG), \text{ENC}_\text{psk}(yG), \text{msg}2) \) to represent this hash value since pmk will be verified on authenticator’s side. On receiving Message 2, the authenticator \( X \) will verify its encrypted public key firstly, then decrypting the supplicant’s public key as \( yG \). So authenticator \( X \) can generated the pmk as \( xyG \), and it will be used to verify the hash value. If Message 2 is valid, the authenticator \( X \) will send Message 3 to supplicant \( Y \). The supplicant \( Y \) will install the pmk and send Message 4 to authenticator \( X \) if Message 3 is valid. When authenticator \( X \) has received and verified Message 4, it will install the pmk. Thus for authenticator \( X \), it finishes the ECDH PMK exchange protocol.

**TABLE III.**

<table>
<thead>
<tr>
<th>EC DH: AUTH=</th>
<th>((x, \text{psk}, G))</th>
</tr>
</thead>
<tbody>
<tr>
<td>new ( x ), ( \text{send} ( \text{new} ( x ), \text{ENC}_\text{psk}(xG), \text{msg}1) )</td>
<td></td>
</tr>
<tr>
<td>receive ( y ), ( z ), match ( z/\text{xyG}, \text{ENC}_\text{psk}(xG) )</td>
<td></td>
</tr>
<tr>
<td>( \text{ENC}_\text{pmk}(yG)), ( \text{msg}2), ( \text{hash}1) )</td>
<td></td>
</tr>
<tr>
<td>match ( \text{ENC}_\text{psk}(yG)/\text{xyG}); match ( \text{xyG}/\text{pmk})</td>
<td></td>
</tr>
<tr>
<td>match ( \text{hash}1/\text{HASH}<em>\text{pmk}(\text{ENC}</em>\text{psk}(xG), \text{ENC}_\text{psk}(yG), \text{msg}2))</td>
<td></td>
</tr>
<tr>
<td>send ( xG, \text{ENC}<em>\text{psk}(xG)), ( \text{msg}3), ( \text{HASH}</em>\text{pmk}(\text{ENC}_\text{psk}(xG), \text{msg}3))</td>
<td></td>
</tr>
<tr>
<td>receive ( v ), ( \text{match} ( v/{\text{ENC}<em>\text{psk}(yG)}, \text{msg}4, \text{hash}3); ( \text{match} \text{hash}3/\text{HASH}(\text{pmk}, \text{ENC}</em>\text{psk}(yG), \text{msg}4))</td>
<td></td>
</tr>
</tbody>
</table>

\[
\text{EC DH: SUPP=} \quad (y, \text{psk}, G) \\
\text{receive} \( \text{z} \), \( \text{match} \( z/\text{ENC}_\text{psk}(xG), \text{msg}1\) \) |
| new \( y \), \( \text{match} \text{ENC}_\text{psk}(xG)/xG\), \( \text{match} \text{xyG}/\text{pmk}\) |
| send \( xG, \text{ENC}_\text{psk}(xG)\), \( \text{ENC}_\text{psk}(yG), \text{msg}2\), \( \text{HASH}_\text{pmk}(\text{ENC}_\text{psk}(xG), \text{ENC}_\text{psk}(yG), \text{msg}2)\) |
| receive \( w \), \( \text{match} \( w/\{\text{ENC}_\text{psk}(yG)\}, \text{msg}3, \text{hash}2\) \) |
| match \( \text{hash}2/\text{HASH}_\text{pmk}(\text{ENC}_\text{psk}(xG), \text{msg}3)\) |
| send \( \text{ENC}_\text{psk}(yG)\), \( \text{msg}4\), \( \text{HASH}_\text{pmk}(\text{ENC}_\text{psk}(yG), \text{msg}4)\) |

2) **Security Properties of Proposed Protocol**

Similar with our proposed Enhanced 3-way Handshake, the security properties of this proposed ECDH PMK exchange protocol include session authentication and key secrecy. **Session authentication** confirms that authenticator and supplicant executes actions in the model in Table III exactly sequentially. **Key secrecy** means that the derived PMK is known to only authenticator and supplicant. Thus we define 2 definitions to as follows:

**DEFINITION 3:** (Key secrecy of DHEC PMK exchange protocol)

The proposed ECDH PMK exchange protocol is said to provide key secrecy of \( \phi_{\text{ECDH,sec}} \), where,
\[
\phi_{\text{ECDH,sec}} \overset{\triangleq}{=} \text{Honest}(\mathbb{A}_X) \land \text{Honest}(\mathbb{A}_Y) \supset ((\text{Has}(x, \text{pmk}) \land \text{Has}(y, \text{pmk})) \lor \text{msg}1, \text{msg}2, \text{hash}1) \land \text{msg}3, \text{hash}2) \land \text{msg}4, \text{hash}3)
\]

**DEFINITION 4:** (Session authentication of ECDH PMK exchange protocol)

The proposed ECDH PMK exchange protocol is said to provide session authentication for the authenticator if \( \phi_{\text{ECDH,auth}} \) holds, where
\[
\phi_{\text{ECDH,auth}} \overset{\triangleq}{=} \text{Honest}(\mathbb{A}_X) \land \text{Honest}(\mathbb{A}_Y) \supset \exists Y. \text{ActionInOrder}(Y, \{ \text{msg}1, \text{msg}2, \text{msg}3 \}, \text{msg}4, \text{msg}4)
\]

Definition 3 defines that if any principal holds PMK, then it must be the supplicant or the authenticator, thus it confirms the key secrecy property of our proposed protocol. Definition 4 defines that if the supplicant and authenticator are honest, then they should do the actions in order as shown in \( \phi_{\text{ECDH,auth}} \) which confirms that supplicant is authenticated by authenticator. We can find that only both two definitions are satisfied then the correctness of proposed protocol can be proved. From these two definitions, we can get the Theorem4 as follows:

**THEOREM 2** (ECDH PMK exchange protocol session authentication guarantee)

On the execution of the supplicant role, key secrecy and session authentication are guaranteed if the formulas in Figure 6.5 hold. Formally,
\[
\Gamma_{\text{ECDH1}} \land \Gamma_{\text{ECDH2}} \vdash \theta_{\text{ECDH}}[\text{ECDH-AUTH}] \land \phi_{\text{ECDH,sec}} \land \phi_{\text{ECDH,auth}}
\]

**TABLE IV.**

| \( \theta_{\text{way}} := \text{Has}(x, \text{psk}) \land \text{Has}(y, \text{psk}) \) |
| \( \Gamma_{\text{ECDH1}} := \text{Computes}(x, xG) \supset \neg (\text{Send}(x, \text{msg}1) \land \text{Contains}(\text{msg}1, xG)) \) |
| \( \Gamma_{\text{ECDH2}} := \text{Computes}(x, yG) \supset \neg (\text{Send}(x, \text{msg}2) \land \text{Contains}(\text{msg}2, xG)) \) |
| \( \Gamma_{\text{ECDH3}} := (\text{Honest}(\mathbb{A}_X) \land \text{Receive}(X, \text{msg}1) \supset \neg \text{Send}(X, \text{msg}1)) \land \text{Has}(x, \text{pmk}) \land \text{Has}(y, \text{pmk}) \) |
| \( \neg (\text{Send}(X, \text{msg}2)) \land \text{Send}(X, \text{msg}4) \) |
The main result of Theorem 2 states that the security guarantees for both supplicant and authenticator. We will show the proof of Theorem 2 in the Appendix B.

VI. CONCLUSION

In this paper, we identified 3 major threats against WLANs, which are DoS attacks against management/control frames, Memory/CPU DoS attacks against 4-way Handshake and offline line dictionary attacks. Since the new IEEE 802.11 iw [15] has addressed the first kind threats, we present two novel protocols which can successfully solve the rest two threats. Furthermore, formal modelling and proofs by PCL are provided in order to show the correctness of proposed protocols.

APPENDIX A  FORMAL PROOF OF PROPOSED 3-WAY HANDSHAKE

HASH1 Computes(\(X, Hash_{k}(x)\)) \(\Rightarrow\) Has(\(X, x\)) \& Has(\(X, K\))

HASH2 Computes(\(X, Hash_{k}(x)\)) \(\Rightarrow\) Has(\(X, Hash_{k}(x)\))

HASH3 Receive(\(X, Hash_{k}(x)\)) \(\Rightarrow\) \(\exists Y.\) Computes(\(Y, Hash_{k}(x)\)) \& Send(\(Y, Hash_{k}(x)\))

HASH4 Has(\(X, Hash_{k}(x)\)) \(\Rightarrow\) Computes(\(X, Hash_{k}(x)\)) \& Computes(\(X, Hash_{k}(x)\)) \& Send(\(Y, m\)) \& Contains(\(m, Hash_{k}(x)\))

SPMK Honest(\(x, Z\)) \& Honest(\(y\)) \(\Rightarrow\) Has(\(Z, pmk\)) \(\Rightarrow\) \(Z = \gamma\) \& \(Z = \phi\)

Define Computes(\(X, Hash_{k}(x)\)) \(\Rightarrow\) Has(\(X, K\)) \& Has(\(X, x\))

The above axioms define the extended axioms based on PCL which will be used in the following proof of the proposed Enhanced 3-way Handshake, and these extended axioms are adopted from works of He et al. [27]. The explanations of these axioms are as follows:

Hash1 and Hash2 state the possession rules of keyed hash function. If the principal X computes a keyed hash value, it must know the key x and the message z as well as the hash value.

Hash3 states that if the principal received a hash value, then there must be someone has computed and sent it to it.

Hash4 states that if a principal X has a hash value, then it either computes this value or receives from other principal.

SPMK states that for honest supplicant and authenticator, they should have shared the PMK.

The following part is the proof of security guarantee for supplicant’s roles in proposed Enhanced 3-way Handshake, note all axioms refer to [26]. The explanations of this formal proof are as follows:

- For an honest supplicant, it must know its own sequential actions in this protocol as presented in line (1).
- If the supplicant have received and verified Message 3, then there must be some principal Z who shared the ptk and sent it out as presented in line (2) and (3).

- If the principal Z shares the same pmk as the supplicant, it must share the same ptk as presented in line (4) – (6).
- Since the principal Z has the pmk, it must be the supplicant authenticator. For honest supplicant, it will not send itself Message 3, thus Message 3 is from authenticator as presented in line (7) – (9).
- Since the authenticator received and verified the Message 3, there must be some principal Z has sent the Message 2 and shares the ptk to it as presented in line (10) – (12).
- Since the principal Z generated the ptk, it must has the pmk. For the same reason, it must be the supplicant as presented in line (13) – (16).
- Because the ANonce from authenticator is fresh, the supplicant can only send the Message 2 after the authenticator has sent the Message 1 to supplicant as presented in line (17) – (19).
- Line (21) shows that the Definition 2 is proved which the session authentication is provided to supplicant.
- Line (20) shows that the Definition 1 is proved which only the valid supplicant and authenticator shares the ptk.
- Line (22) shows that the correctness of proposed Enhanced 3-way Handshake is proved.
APPENDIX B: FORMAL PROOF OF PROPOSED ECDH PROTOCOL

SPSK
\[ \text{Honest}(Z) \land \text{Honest}(y) \Rightarrow \text{Has}(Z, psk) \Rightarrow Z = Z \lor Z = Y \]

ECDH1
\[ \text{Comptes}(X, xyG) \Rightarrow \text{Has}(X, xyG) \]

ECDH2
\[ \text{Has}(X, xyG) \Rightarrow \text{Comptes}(X, xyG) \lor \exists m. (\text{Receive}(X, m) \land \text{Contains}(m, xyG)) \]

ECDH3
\[ \text{Receive}(X, m) \land \text{Contains}(m, xyG) \Rightarrow \exists Y, m'. (\text{Computes}(Y, m') \land \text{Send}(Y, m') \land \text{Contains}(m', xyG)) \]

ECDH4
\[ \text{Fresh}(X, x) \Rightarrow \text{Fresh}(X, xG) \]

Define \[ \text{Computes}(X, xyG) \equiv (\text{Has}(X, x) \land \text{Has}(X, yG)) \lor (\text{Has}(X, y) \land \text{Has}(X, xG)) \]

The above axioms define the extended axioms based on PCL which will be used in the following proof of the proposed novel ECDH protocol. The explanation of these axioms is as follows:

- **SPSK** states that only legitimate supplicant and authenticator share the psk.
- **ECDH1** states that if some thread has all necessary information to compute the secret key of ECDH, then it also has this secret key.
- **ECDH2** states that the only way to get the ECDH secret key is to compute it or receive from other thread.
- **ECDH3** states that if someone receives a message containing the ECDH secret key, then there must be someone compute and send it out by itself.
- **ECDH4** states that the public key of ECDH is fresh if the corresponding private key of ECDH is fresh.

The following part is the proof of the proposed ECDH PMK exchange protocol, and the explanation of this proof is as follows:

- For an honest authenticator, it must know its sequential actions in this protocol as presented in line (1).
- If the supplicant have received and verified Message 4, then there must be some principal Z who shared the pmk and sent it out as presented in line (2) and (3).
- If the principal Z shares the same pmk as the supplicant, it must share the same psk as presented in line (4) – (8).
- Since the principal Z has the psk, it must be the supplicant authenticator. For honest supplicant, it will not send itself Message 4, thus Message 4 is from supplicant as presented in line (9) – (11).
- Since the supplicant received and verified the Message 3, there must be some principal Z has sent the Message 3 and shares the pmk to it as presented in line (10) – (14).
- Since the principal Z generated the pmk, it must has the psk. For the same reason, it must be the authenticator as presented in line (15) – (17).
- Because the secret keys from authenticator and supplicant are fresh, then the public keys of them are also fresh. So supplicant can only send out the Message 2 after the authenticator has sent the Message 1 to supplicant, and authenticator can only send out Message 3 after the supplicant sent out the Message 2 as presented in line (18) – (22).
- Line (23) shows that the Definition 4 is proved which the session authentication is provided to authenticator.
- Line (24) shows that the Definition 3 is proved which the only valid supplicant and authenticator share the pmk.
● Line (25) shows that the correctness of proposed ECDH PMK exchange protocol is proved.

AA1, AA4, θ_{ECDH-AUTH}:
ARP
Send(X, {Y, X, ENC_{psk}(Y), msg1}) <
Receive(X, {Z, Y, ENC_{psk}(Y), ENC_{psk}(Y), msg2},
HASH_{psk}(ENC_{psk}(Y), ENC_{psk}(Y), msg2)) <
Send(Y, {Z, X, ENC_{psk}(Y), ENC2, ENC_{psk}(Y), msg3},
HASH_{psk}(ENC_{psk}(Y), msg3)) <
Receive(X, {Z, Y, ENC_{psk}(Y), ENC_{psk}(Y), msg4},
HASH_{psk}(ENC_{psk}(Y), msg4)) )

HASH3, ARP
θ_{cond}receive Y, Z, v; match v = (ENC_{psk}(Y), msg4), hash3; match hash3/ HASHpmk,
ENC_{psk}(Y), msg4), Y) <
Receive(X, {Z, Y, ENC_{psk}(Y), ENC_{psk}(Y), msg4}) <
3Z(ComputeZ(X, HASH_{psk}(ENC_{psk}(Y), msg4)) &
Send(Z, {Z, Y, ENC_{psk}(Y), ENC_{psk}(Y), msg4}, HASH_{psk}(ENC_{psk}(Y), msg4)) <
Receive(X, {Z, Y, ENC_{psk}(Y), ENC_{psk}(Y), msg4},
HASH_{psk}(ENC_{psk}(Y), msg4)) )

HASH1 Computes(Z, HASH_{psk}(ENC_{psk}(Y), msg4)) ⊃
Has(Z, pmk) ∧ Has(Z, ENC_{psk}(Y), msg3)) ⊃
(4)

SPSK
Honest[Z] ∧ Honest[Y] ⇒ Compute(Z,
HASH_{psk}(ENC_{psk}(Y), msg3)) ⇒
Has(Z, pmk) =
Has(Z, Y, xG) ⇒ Has(Z, pmk) = Z = Y = Z = Y

AA1, ARP, θ_{ECDH-SEND}:
Send(Y, {Z, Y, ENC_{psk}(Y), msg3}) <
Receive(X, {Z, Y, ENC_{psk}(Y), ENC_{psk}(Y), msg4}) <
Send(Y, {Z, Y, ENC_{psk}(Y), ENC_{psk}(Y), msg4},
HASH_{psk}(ENC_{psk}(Y), msg4)) )

FA1, AN3, θ_{cond}new x, v;
Send(x, ENC_{psk}(Y), msg1) )<
Receive(x, {Z, Y, ENC_{psk}(Y), msg1})<
(21)

FA2, θ_{cond}new x, v;
Send(X, {Z, Y, ENC_{psk}(Y), msg1}) <
Receive(Y, {Z, Y, ENC_{psk}(Y), msg1})<
(22)

© 2011 ACADEMY PUBLISHER
Li Wang received the BEng degree in software engineering from Zhe Jiang University and MCS degree in computer science from Monash University in 2006 and 2009, respectively. He is currently a research student in the Clayton school of IT at Monash University. His research interests include network security, information security and biometrics.

Bala Srinivasan is a Professor in Monash University, Melbourne, Australia. He has authored and jointly edited 6 technical books and authored and co-authored more than 200 international refereed publications in journals and conferences in the areas of Databases, Multimedia Retrieval Systems, Distributed and Mobile Computing and Data Mining.

Nandita Bhattacharjee is a senior lecturer in Monash University, Melbourne, Australia. She received his Ph.D. degree from Monash University in 2006. Her research includes information security, image processing and audio visual communications, ultrasonic imaging, digital systems hardware and architecture.