IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 16, 2019, accepted July 7, 2019, date of publication July 10, 2019, date of current version August 6, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2927772

Design and Formal Analysis of an Authentication Protocol, eWMDP on Wearable Devices

BO LU[1,2], RUOHAN CAO[1,3], YUEMING LU[1,3], AND XUETING LUO[1,2]

[1]Key Laboratory of Trustworthy Distributed Computing and Service, Ministry of Education, Beijing University of Posts and Telecommunications (BUPT), Beijing 100876, China
[2]College of Cyberspace Security, Beijing University of Posts and Telecommunications (BUPT), Beijing 100876, China
[3]School of Information and Communication Engineering, Beijing University of Posts and Telecommunications (BUPT), Beijing 100876, China

Corresponding author: Yueming Lu (ymlu@bupt.edu.cn)

This work was supported in part by the National Key R&D Program of China under Grant 2016YFF0204001 and Grant 2016YFB0800302, and in part by the Beijing University of Posts and Telecommunications (BUPT) Excellent Ph.D. Students Foundation under Grant CX2019230.

ABSTRACT For wearable devices, this paper proposes an authentication protocol eWMDP. We formally model and analyze it. In the execution model of the protocol, a Dolev-Yao adversary is constituted. With the above, various security properties can be defined for measuring security performance. Our protocol can achieve the injective-auth property, in which case it surpasses another existing wearable authentication protocol—WMDP. In addition, our analysis work is confirmed by Scyther and tamain-prover, and the two protocols are implemented on PC and on Ti CC3200 LAUNCHPAD widely used in wearable devices in the meanwhile. The results of experiments highlight the higher transmission efficiency of eWMDP under various configurations, including different platforms, sorts of combinations of encryption algorithms and hash algorithms, and sending packets in different sizes.

INDEX TERMS Security protocol, formal analysis, wearable device protocol, scyther, tamarin-prover.

I. INTRODUCTION

With the miniaturization of electron devices, the intelligentization of function, and the popularization of phone, wearable devices are widely used in the fields of medical treatment, households, sports and services. Meanwhile, the threats which are imposed on wearable devices increase rapidly [1], [2]. These threats not only come from the privacy leaks of consumers' data (medical records, insurance policies, etc.) by service providers, but the analysis in network transmission by hackers. Some of these threats have brought great risks. In particular, hackers can launch multiple attacks, such as DDoS attack [3], man-in-the-middle attack [4], and reflection attack [5], which may cause huge destruction to wearable devices.

A. RELATED WORK

It is cleared that security analysis of protocol has a great importance to defend these attacks. Especially for authentication protocol whose security analysis have been developing rapidly in recent years. Formal semantics analysis of authentication protocol is a hot topic in current research. Therefore, we can use some specific logic systems to analyze the behavior of attack protocol, such as Burrows Abadi Needham logic [6], Gong Needham Yahalom logic [7], [8], omniscience-free temporal logic [9], embedded security protocol logic [10], etc. In a logic system, the security property can be defined as a predicate to analyze protocol's security. It can be analyzed by logical deduction whose procedure can be accomplished by programs [11]–[16]. Most of the programs used in the general field work interactively. But automation tools can be designed for certain field. Using these automation tools (ProVerif [17], Scyther [18], tamarin-prover [19], AVISPA [20] etc.) increases analysis efficiency. With the help of these tools, many security protocol vulnerabilities have been found. In [18], Cas Cremers et al. provide the first systematic analysis of the ISO/IEC 11770 standard for key management techniques and analyze the claimed security properties by Scyther tool. In [19], Martin Dehnel-Wild et al. analyze the security of the Secure Authentication v5 (SAv5) protocol by tamarin-prover. In [17], Lucca Hirschi et al. analyze the security of the e-voting protocols by ProVerif. The method is also used to analyze wearable device protocols [20]. There are various methods of wearable device certification protocol design and security analysis. The authentication protocols are designed for different application

The associate editor coordinating the review of this manuscript and approving it for publication was Masood Ur-Rehman.

scenarios. The methods for designing these protocols can be divided into two kinds. One is based on biological characteristics [21]–[26], and the other is mathematical difficulty [27]–[35]. In terms of security analysis, [21], [22], [24], [33], [34] establish capabilities of adversary to analyze the security of protocols. [28], [32], [33] verify the security of protocols against a limited number of attacks. And [20] analyzes security of protocol by using formal analysis.

It is easy to generate identity and random information by using biological characteristics. The information helps protocols implement authentication. The performance metric of these schemes is hard to measure. This type of protocol security analysis also takes adversary capabilities into account. To be more precise, Chen *et al.* [21] propose lightweight and real-time key sharing scheme, in which the security is based on random numbers generated by shaking arm, and the adversary ability consists of imitation attacks, passive eavesdrop, active attacks and knowledge of all the procedures and methods. Zhang *et al.* [22] propose an implicit authentication by novel 3D magnetic finger motioning pattern. The adversary can obtain users' secret information such as a PIN, password. Samangouei *et al.* [23] present a method using facial characteristics for continuous authentication. Peris-Lopez *et al.* [24] propose a continuous authentication which is based on electrocardiogram (ECG). The adversary consists of an unknown adversary, known adversary, blind-model adversary. Shen *et al.* [25] propose a continuous and implicit authentication which is based on motion sensor (accelerometer and gyroscope). Zhang *et al.* [26] propose an authentication based on ECG for smart health-care systems. There is no adversary working in the system.

In contrast, protocols, which use one-way function based on difficult problems such as encryption and hashing, can avoid this problem to some extent. Liu *et al.* [27] proposes a quick response (QR) authentication protocol whose structure is challenge-response. They use QR code to complete the authentication, which means that they have to face to face each other. He and Zeadally [28] present a new authentication model which is suitable for Ambient Assisted Living (AAL) system. This system is used to provide electronic monitoring and telemedicine service for users. The authors only analyze limited known attacks, and cannot predict unknown attacks and security properties. Long and Lin [29] present a new authentication protocol for wearable medical devices. It satisfies the secrecy property, only partially implements the authentication property, and does not satisfy the injective authentication property. Shen *et al.* [30] propose an anonymous, lightweight authentication protocol. The adversary is a dishonest user who works on a random oracle model. Van hammy *et al.* [31] propose a multi-modal active authentication scheme whose trust model combines policy and reputation. Vijayakumar *et al.* [32] propose an alert system for sending the private and confidential SMS messages from heart patients. The adversary can launch DoS attack, masquerading, and man-in-the-middle attack. Alhothaily *et al.* [33] propose a threshold-based authentication scheme in which users can register upon n devices without creating or remembering any credentials and provide access control features. They take three situations into account which are adversary attacks device with stealing equipment, malware attacks and single point of failure. Jiang *et al.* [34] propose an end-to-end mutual authentication protocol in wearable health monitoring systems (WHMSs) to protect health data from unauthorized access attack. The adversary that steals secret parameter by side channels to attack can capture all messages sent or received in session and obtain stolen or lost devices of a legal user. Wang *et al.* [35] propose an authentication protocol using key agreement schemes. The adversary is based on Dolev-Yao model. The security analysis is based on the random oracle model.

The mentioned above adversary capabilities are not closely tied to the protocol steps designed, and only limited forms of attack are considered. Li *et al.* [20] presents a lightweight authenticated key establishment protocol which is analyzed automatically by AVISPA tool and works for wearable sensors. The formal proof is based on BAN logic. However, the adversary based on BAN and GNY logic are weaker than Dolev-Yao adversary based on embedded security protocol logic (ESPL).

B. CONTRIBUTIONS

In summary, we need to base the security of the protocol on the one-way function. In the security analysis, we should consider the adversary attack as an integral part of the system treated as formal, logical system, of which the properties are adversary behaviors. Due to the complexity of the computation process, the derivation process is automatized as much as possible.

Compared with the above work, formal semantic analysis on protocol security has the advantage of resisting unknown attacks, and the security is based on the mathematical difficulties. In the research, ESPL is used to analyze protocol security. The capabilities of the Dolev-Yao adversary are defined as controlling the network and deleting, injecting, modifying, and intercepting messages on the network.

In this paper, our contributions include the following:

(1) We model and formalize the wearable medical device protocol (WMDP) and eWMDP, and build the operation semantics of protocols.

(2) Correspondingly, the security properties of protocol eWMDP are inferred according to the rule chain, which shows that eWMDP satisfies secrecy, non-injective synchronization authentication and injective authentication, and is also verified by Scyther and tamarin-prover.

II. WEARABLE DEVICE PROTOCOL

The standard IEEE 802.15.6 which is published for the body area networks (BAN) by IEEE in 2012 has shown that the standard is applied to short-range, wireless communication in the vicinity of, or inside a human body (but not limited to humans) [36]. In some scenarios, the wearable devices of the body such as watches, bracelets, belts can be connected

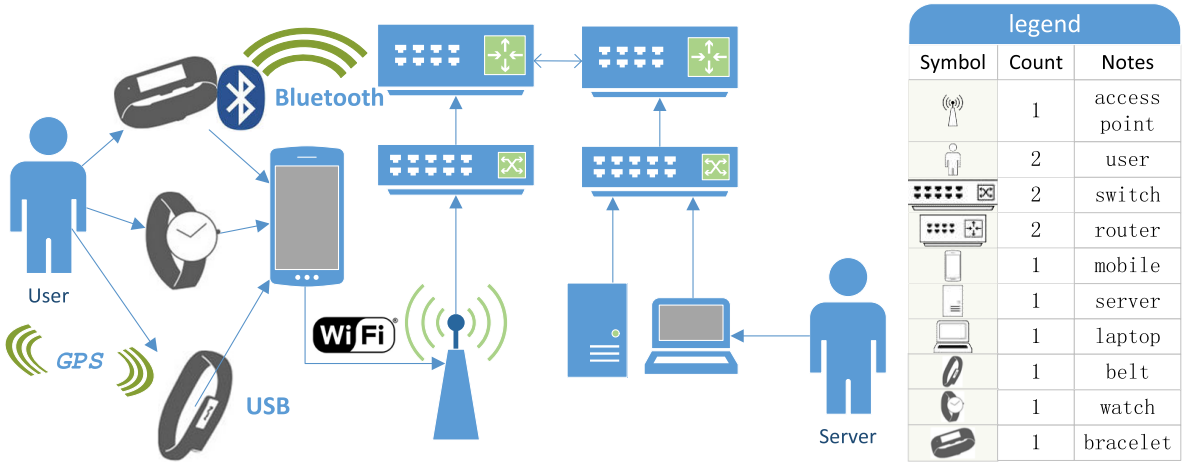

FIGURE 1. Wearable devices interconnected through network.

to Bluetooth, USB, Wi-Fi and mobile phones or mid-service equipment. On mobile phones or computers, users can browse data onto devices in detail, and can also do more complex network operations, such as data uploading, downloading, recording and sharing, etc. At the same time, the mobile phone is connected to the access point through Wi-Fi. In addition, wearable devices can access the internet indirectly, and be able to interact with remote service providers through the traditional network equipment such as switches and routers. The network structure is shown with FIGURE 1 below.

Ti CC3200 LAUNCHPAD is an excellent implementation scheme for wearable devices [37]–[40], which has been widely used in ECG monitoring, temperature sensing, environmental monitoring and other fields. In the implementation, the protocol is a lightweight protocol with only symmetric encryption and hash operation in the case of shared key.

As illustrated above, the existing partner of the protocol can be classified as being user (I) and server (R). Both share secret messages such as symmetric keys k, secret information s and public messages such as opad (32-byte long sequence of 0×36) and ipad (32-byte long sequence of $0 \times 5C$). They communicate data security to each other by sending and receiving messages. The protocol WMDP which is working on the network and defined in [29] is represented as several core steps:

Step 1 User I generates a random number c which is a fresh value and sends it to server R. Fresh value as a random number provides an 'aliveness' verification mechanism, by which a user tries to prove himself to another one that he is online as a partner.

Step 2 Server R receives the value c, generates a random number r and computes $h_{crs} = h(c, r, s)$, and next, sends message $\langle r, h_{crs} \rangle$ to user I. He uses cryptology hash functions $h(\_)$ to provide message integrity against adversary tampering with the message c, r, s.

Step 3 User I receives the message, and compares $h(c, r, s)$ with h_{crs}. If the expression is false, then he terminates the protocol. Otherwise, he computes the $MAC = h(k \oplus opad, h(k \oplus ipad), m)$, and encrypts the cipher text $C_m = \{m, MAC\}_k$. Next, sends C_m to server R. After the previous interaction, the user starts sending messages to another partner. He uses cryptology encryption $\{\}_k$ to provide the secrecy for the message sent by I.

Step 4 Server R receives the message, decrypts and computes the expression $h(k \oplus opad, h(k \oplus ipad), m) == MAC$. If the expression is false, then an exception is executed. Otherwise, he completes the communication.

Where $\oplus$ is xor operation symbol. The symmetric key between user I and server R is noted as k and the data is m. Next, we introduce the proposed protocol. There are some shortcomings in the above scheme. In communication process, there is no security protection for random number c. The first two steps as separate authentication processes and the third step as subsequent messaging processes (there is no cohesion between shared messages such as random numbers) are disconnected. In cryptology encryption, the hash algorithm with two nested sets is redundant.

III. PROPOSED PROTOCOL

We improved the above protocol. Security analysis and safety verification of the newly designed protocols are conducted. The initial knowledge of user I, server R contains only symmetric key k, thus simplifying usage conditions compared with that of WMDP. Other optimizations are covered later. The protocol eWMDP is represented as several core steps:

Step 1 User I generates a random number c, computes $MAC = h(c, m)$, and encrypts the cipher text $C_m = \{c, m, MAC\}_k$. Next, he sends the message to server R.

Step 2 Server R receives the value C_m, decrypts and compares $h(c, m)$ with MAC. If the expression is false, then terminates the protocol. Otherwise, he generates random number r, computes $h_{crk} = h(c, r, k)$, next, sends message $\langle r, h_{crk} \rangle$ to user I.

Step 3 User I receives the message, and compares $h(c, r, k)$ with h_{crk}. If the expression is false, then he terminates the protocol. Otherwise, he completes communication.

The number of handshakes is only twice in eWMDP less than third in WMDP. In addition, the hash operation $h(\_)$ used for calculation and verification in the protocol eWMDP is only 4 times, less than 6 times of WMDP protocol. Compared with WMDP, the proposed protocol eliminates xor $\oplus$, and we uses no opad or ipad on it. The proposed protocol complies with the principle of ''fresh value and hash operation binding'' in the field of design, which embodies freshness and data integrity by depending on each other mutually. According to the text description of WMDP protocol sequence given above, we use natural language to give a trace of adversary attacking protocol. The following example clearly violates the security goal originally set in the protocol, in which the adversary cheats the communicator. With this example in mind, we begin to identify what security goal (that is, security properties) is required for the authentication protocol. With respect to security, there is an attack of authentication aiming at protocol WMDP. The capabilities of the attacker are based on the Dolev-Yao attack model. For example,

Step 1 User I generates a random number c and sends it to server R.

Step 2 Server R receives the value c, generates a random number r and computes $h_{crs} = h(c, r, s)$. Next, he sends message $\langle r, h_{crs}\rangle$ to user I.

Step 3 User I receives the message, and compares $h(c, r, s)$ with h_{crs}. Next, he computes the $MAC = h(k \oplus opad, h(k \oplus ipad), m)$, and encrypts the cipher text $C_m = \{m, MAC\}_k$. Finally, he sends C_m to server R.

Step 4 Adversary I_E pretending as user I generates a random number c_E and sends it to server R.

Step 5 Server R follows protocol steps to send messages $\langle r_E, h_{crs_E}\rangle$ to user I_E.

Step 6 Adversary I_E sends the message intercepted in step 3 to the server R.

Step 7 Server R receives the message, decrypts and compares $h(k \oplus opad, h(k \oplus ipad), m)$ with MAC. And then he completes communication.

The steps 1-3 represent a normal session between I and R. The steps 4-6 indicates that the adversary uses the message of previous normal session to make the other party think he himself is having a normal session, which is not found. In the second session built by adversary, server R never knows that he is not communicating with the intended user I. In addition, he also cannot determine which thread that executes user I he himself is communicating with. We compare three security properties of the protocol which are secrecy, non-injective authentication and injective authentication. The secrecy property denoted as φ_{sec} (23) means that the message msg sent by roles is unknown to the adversary. The non-injective authentication property denoted as $\varphi_{\text{non-injective-auth}}$ (24) means that the partner is able to confirm that he is communicating with the intended communication partner, but cannot determine whether the thread executing the other partner is the intended thread when the communication is completed. As can be seen from the above example, the WMDP protocol does not satisfy the non-injective authentication. The injective authentication property denoted as $\varphi_{\text{injective - auth}}$ (25) means that, according to the thread i which executes a role, the partner can determine the thread j which executes the other role. Obviously, WMDP also does not satisfy the injective authentication. By contrast, protocol eWMDP satisfies all three security properties whose proof may be described in detail later. To prove that there is no adversary trace of breaking security properties, we cannot use an exhaustive search. The formal semantic model needs to be built first. Next, we begin the security analysis by establishing the corresponding protocol model.

IV. SECURITY ANALYSIS OF PROTOCOL WMDP AND EWMDP

To analyze the security properties of the above two protocols, we need to establish protocol specification models, protocol execution models, and protocol security properties like secrecy and authentication.

A. PROTOCOL SPECIFICATION MODEL

Our model of security protocol is a set of roles such as I and R that are given by a script specified by the initial knowledge of the roles and role steps. A sequence $RoleStep$ is constructed by $Send_l(Pat)$ or $Recv_l(Pat)$, where l is a label, and Pat is a patterns of message. The initial knowledge of a role is a set of Msg such as key and public information. A protocol can be executed by any amount of agents who are the executor of the protocol roles. Each role instance is a thread which has a thread id. The message pattern Pat is constructed by:

$$Pat ::= Constant|Fresh|Var|h(Pat)|(Pat, Pat) \\ |Msg_{Pat}|Pat^{-1}|k(Pat, Pat), \quad (1)$$

The message Msg is constructed by:

$$Msg ::= Constant|Fresh|\#TID|h(Msg)|(Msg, Msg) \\ |Msg_{Msg}|k(Agent, Agent), \quad (2)$$

where $\#TID$ is a thread id. A protocol $WMDP$ is written as $WMDP ::= I, R$, where $WMDP(I)$ and $WMDP(R)$ are defined as follows:

$$\begin{aligned} &WMDP(I) \\ &= (\{I, R, s, k_{(I,R)}, \text{opad}, \text{ipad}\}, \\ &\quad [Send_1(\tilde{c}),\ Recv_2(V, h(\tilde{c}, V, s)), \\ &\quad Send_3(\{m, h(k(I, R)\oplus\text{opad}, h(k(I, R) \oplus \text{ipad}), m))\}_{k(I,R)}) \\ &\quad]), \\ &WMDP(R) \\ &= (\{I, R, s, k_{(I,R)}, \text{opad}, \text{ipad}\}, \\ &\quad [Recv_1(W),\ Send_2(\tilde{r}, h(\tilde{c}, V, s)), \\ &\quad Recv_3(\{U, h(k(I,R)\oplus\text{opad}, h(k(I,R)\oplus\text{ipad}), U))\}_{k(I,R)})]), \end{aligned} \quad (3)$$

where the initial knowledge s as secret information is shared by devices, and $k_{(I,R)}$ is long-term key between two partners. Both of them belong to *Fresh*, and *Constant* consists of I, R, opad, ipad. The uppercase letters such as V, W, U, represent the message variables that may be received. The *Var* consists of *AVar* (agent variable) and *MVar* (message variable).

Correspondingly, the protocol specification of eWMDP is defined in the same way as follows:

$$\begin{aligned} eWMDP(I) = &(\{I, R, k_{(I,R)}\}, \\ &[Send_1(\{\tilde{c}, \tilde{m}, h(\tilde{c}, \tilde{m})\}_{k(I,R)}) \\ &Recv_2(V, h(\tilde{c}, V, k_{(I,R)})) \\ &]), \\ eWMDP(R) = &(\{I, R, k_{(I,R)}\}, \\ &[Recv_1(\{V, W, h(V, W)\}_{k(I,R)}), \\ &Send_2(\tilde{r}, h(V, \tilde{r}, k_{(I,R)})) \\ &]), \end{aligned} \tag{4}$$

where in eWMDP protocol, *Constant* consists of I,R, and here, $k_{(I,R)}$ is a *term* which will be substituted by a fresh value. Formulas (3) and (4) both provide the preconditions for the formal analysis of protocol security. As the conclusion of the formal reasoning, security properties need to be proved by inference rules. In addition to the formal model of the protocol, the corresponding semantics need to be established afterwards. We use operation semantics to establish the corresponding initial state of machine (see formulas 5-7), state transition function (see formulas 10-15) for the formalized system.

B. PROTOCOL EXECUTION MODEL

The trace records the history of executed role steps and the messages learned by adversary. The *TraceEvent* is defined as follows:

$$TraceEvent ::= St(TID, RoleStep) | \mathrm{Ln}(\mathcal{P}(Msg)), \tag{5}$$

where $\mathcal{P}(Msg)$ means the power set of Msg. The step trace event $St(i, s)$ means that the thread whose *TID* is i executes the role steps s. The learn trace event $Ln(M)$ means that the message is known by adversary. The trace tr is a sequence which is constructed by trace events. To record the history of the protocol execution and the messages which are gained by the adversary's reasoning. The system state of operation semantics is defined by a triple (tr, th, σ). The thread pool th is a partial function:

$$th : TID \rightharpoonup (Role \times RoleStep^*), \tag{6}$$

where $RoleStep^*$ denotes the reflexive transitive closure of $RoleStep$. The σ is a variable store function which is defined as follows:

$$\sigma : Var \times TID \to Msg, \tag{7}$$

where σ stores the assignment that shows the content of thread variable. The initial knowledge of adversary AK_0 is defined as follows:

$$Ak_0 = Constant \cup Agent \cup \bigcup_{a \in Agent, c \in Agent_C} \{k(a, c), k(c, a)\}, \tag{8}$$

where in WMDP *Constant* consists of opad and ipad; Agent consists of a, b, e; $a, b \in Agent_H$ (honest agent), $e \in Agent_C$ (compromise agent). We use the inference rules $l \to r$ to denote that l entails r. This rule is also denoted by:

$$\frac{l}{r}\text{RULENAME}, \tag{9}$$

where l is a premise, and r is a conclusion. As for an execution of protocol WMDP, it can be modeled in the transition system which is defined as follows:

$$\frac{th(i) = (R, [\mathrm{Send}_l(pt) \cdot todo])\ inst_{\sigma,i}(pt) \neq \bot}{\begin{array}{l}(tr, th, \sigma) \to (tr \cdot [\mathrm{St}(i, \mathrm{Send}_l(pt)), \\ \mathrm{Ln}(learns_{tr}(inst_{\sigma,i}(pt)))], th[i \mapsto (R, todo)], \sigma)\end{array}}\text{SEND}, \tag{10}$$

$$\frac{\begin{array}{l}th(i) = (R, [\mathrm{Recv}_l(pt) \cdot todo])\ inst_{\sigma,i}(pt) \\ \in Know(tr)\ inst_{\sigma,i}(pt) \neq \bot\end{array}}{\begin{array}{l}(tr, th, \sigma) \to (tr \cdot [\mathrm{St}(i, \mathrm{Recv}_l(pt))], \\ th[i \mapsto (R, todo)], \sigma)\end{array}}\text{RECV}, \tag{11}$$

$$\frac{x, y \in Know(tr)\ \langle x, y\rangle \notin Know(tr)}{(tr, th, \sigma) \to (tr \cdot [\mathrm{Ln}(\{\langle x, y\rangle\})], th, \sigma)}\text{PAIR}, \tag{12}$$

$$\frac{m \in Know(tr)\ h(m) \notin Know(tr)}{(tr, th, \sigma) \to (tr \cdot [\mathrm{Ln}(\{h(m)\})], th, \sigma)}\text{HASH}, \tag{13}$$

$$\frac{m, k \in Know(tr)\ \{m\}_k \notin Know(tr)}{(tr, th, \sigma) \to (tr \cdot [\mathrm{Ln}(\{\{m\}_k\})], th, \sigma)}\text{ENCR}, \tag{14}$$

$$\frac{\{m\}_k \in Know(tr)\ k^{-1} \in Know(tr)}{(tr, th, \sigma) \to (tr \cdot [\mathrm{Ln}(learns_{tr}(m))], th, \sigma)}\text{DECR}, \tag{15}$$

where $Know(tr)$ is defined by $\bigcup_{\mathrm{Ln}(M) \in tr} M$. The $inst_{\sigma,i}(pt)$ is defined as follows:

$$inst_{\sigma,i}(pt) := \begin{cases} pt & \text{if } pt \in Constant \\ pt\#i & \text{if } pt \in Fresh \\ \sigma(pt, i) & \text{if } pt = Var \\ h(inst_{\sigma,i}(x)) & \text{if } pt = h(x) \\ (inst_{\sigma,i}(x), inst_{\sigma,i}(y)) & \text{if } pt = (x, y) \\ \{inst_{\sigma,i}(x)\}_{(inst_{\sigma,i}(k))} & \text{if } pt = \{x\}_k \\ (inst_{\sigma,i}(x))^{-1} & \text{if } pt = x^{-1} \\ k(inst_{\sigma,i}(I), inst_{\sigma,i}(R)) & \text{if } pt = k(I, R) \\ \bot & \text{otherwise} \end{cases} \tag{16}$$

In combination with the previously defined formal semantic model, we define the attack trace of WMDP using formal symbols. This helps us clarify the fact that the conclusion of the adversary's attack contradicts that of the security property, leading to the proof failure. This just proves that WMDP doesn't satisfy the security property. The example

attacking WMDP can be shown by the execution model as follows:

$$\begin{aligned} th := &\{1 \mapsto (I, []),\\ &2 \mapsto (R, []),\\ &3 \mapsto (I, []),\\ &4 \mapsto (R, [R_2])\}, \end{aligned} \tag{17}$$

$$\begin{aligned} \sigma := &[(\tilde{c}, 1) \mapsto \tilde{c}^{\#1}][(I, 1) \mapsto a][(R, 1) \mapsto b]\\ &[(I, 2) \mapsto a][(R, 2) \mapsto b][(W, 2) \mapsto \tilde{c}^{\#1}]\\ &[(\tilde{r}, 2) \mapsto \tilde{r}^{\#2}][(V, 1) \mapsto \tilde{c}^{\#1}][(\tilde{m}, 1) \mapsto \tilde{m}^{\#1}]\\ &[(U, 2) \mapsto \tilde{m}^{\#3}][(I, 3) \mapsto a][(R, 3) \mapsto b]\\ &[(I, 4) \mapsto a][(R, 4) \mapsto e][(\tilde{c}, 3) \mapsto \tilde{c}^{\#3}]\\ &[(W, 4) \mapsto \tilde{c}^{\#3}][(\tilde{r}, 4) \mapsto \tilde{r}^{\#2}][(V, 3) \mapsto \tilde{c}^{\#3}]\\ &[(\tilde{m}, 3) \mapsto \tilde{m}^{\#3}], \end{aligned} \tag{18}$$

$$\begin{aligned} tr := &[\mathrm{Ln}(Ak_0), \mathrm{St}(1, I_1), \mathrm{Ln}(\tilde{c}^{\#1}), \mathrm{St}(2, R_1), \mathrm{St}(2, R_2),\\ &\mathrm{Ln}(\tilde{r}^{\#2}), \mathrm{Ln}(h(\tilde{c}^{\#1}, \tilde{r}^{\#2}, k(a, b))),\\ &\mathrm{Ln}(\langle \tilde{r}^{\#2}, h(\tilde{c}^{\#1}, \tilde{r}^{\#2}, k(a, b))\rangle), \mathrm{St}(1, I_2), \mathrm{St}(3, I_1),\\ &\mathrm{Ln}(\tilde{c}^{\#3}), \mathrm{St}(4, R_1), \mathrm{St}(4, R_2), \mathrm{Ln}(\tilde{r}^{\#2}),\\ &\mathrm{Ln}(h(\tilde{c}^{\#3}, \tilde{r}^{\#4}, k(a, b))), \mathrm{Ln}(\langle \tilde{r}^{\#4}, h(\tilde{c}^{\#3}, \tilde{r}^{\#4}, k(a, b))\rangle),\\ &\mathrm{St}(3, I_2), \mathrm{St}(3, I_3),\\ &\mathrm{Ln}(\{\tilde{m}^{\#3}, h(k(a, b) \oplus \mathrm{opad}, h(k(a, b) \oplus \mathrm{opad}, \tilde{m}^{\#3}))\}_{k(a,b)}),\\ &\mathrm{St}(1, I_3)\mathrm{St}(2, R_4)], \end{aligned} \tag{19}$$

where the second session between agent a and b is interrupted by adversary, and that the compromised agent e disguises agent b to communicate with agent a while he uses another Msg $\{\tilde{m}^{\#3}, h(k(a, b) \oplus \mathrm{opad}, h(k(a, b) \oplus \mathrm{opad}, \tilde{m}^{\#3}))\}_{k(a,b)}$ sent by agent b, which means that users a cannot determine whether he is communicating with the expectant partner in WMDP. The operation symbol $\oplus$ denotes xor operation.

C. PROTOCOL SECURITY PROPERTIES MODEL

In this formal semantic model, protocol security properties are defined precisely by using symbolic semantics. Before that, we first introduce the concepts of predicates $Q_0(P)$, $reachable(P)$ and a relationship $\prec_{tr}$, which will help us express security properties succinctly. As for the transition system, the initial state $Q_0(P)$ of protocol P is defined as follows:

$$\begin{aligned} Q_0(P) := \{([\mathrm{Ln}(Ak_0)], th, \sigma) | (&\forall v \in AVar, i \in TID\\ &. \sigma(v, i) \in Agent) \wedge (\forall i \in dom(th)\\ &. \exists R \in P.th(i) = (R, [\pi_2(P(R))]))\}, \end{aligned} \tag{20}$$

where for each thread i of role R, they has no executed step yet. The function π_2 returns the second of the pairs. Corresponding to the transition system, the reachable state can be defined as follows:

$$reachable(P) := \{q | \exists q_0 \in Q_0(P).q_0 \rightarrow^* q\}. \tag{21}$$

Next, the aforementioned state of protocol WMDP in formulas (17)(18)(19) is a reachable one. We use event order relation $\prec_{tr}$ to represent the possible execution space of protocols:

$$\begin{aligned} x \prec_{tr} y := &\exists tr_1, tr_2.tr = tr_1.tr_2\\ &\wedge (x \in Know(tr_1) \vee x \in steps(tr_1))\\ &\wedge (y \in Know(tr_2) \vee y \in steps(tr_2)), \end{aligned} \tag{22}$$

where step events $steps(tr)$ can be written as: $steps(tr) := \{(i, s) | \mathrm{St}(i, s) \in tr\}$. Our purpose is to prove that the execution of the security protocol satisfies some security properties, i.e., the validity of some formulas in this logical system. Furthermore, the trace of the space of the reachable state in this system is closely associated with a property. For protocol P, the secrecy property is defined in (23). Now let's formally define these security properties by:

$$\begin{aligned} \varphi_{\mathrm{sec}} := &\forall q \in reachable(P).\forall i \in TID.\forall m \in Msg.\\ &claim_q(i, m) \Rightarrow m \notin Know(tr). \end{aligned} \tag{23}$$

where $claim_q(\_)$ is a predicate formalized for every state q when the parameter claims something. The non-injective authentication property is defined in (24):

$$\begin{aligned} \varphi_{\text{non-injective - auth}} := &\forall i_1, i_2, j \in TID.partner_q(i_1, j)\\ &\wedge partner_q(i_2, j) \Rightarrow i_1 = i_2. \end{aligned} \tag{24}$$

where $partner_q(i, j)$ is a predicate formalized for every state q where a thread j is a partner of a thread i. The injective authentication property is defined in (25):

$$\begin{aligned} \varphi_{\text{injective-auth}} := &\forall q \in reachable(P).\forall i \in TID.claim_q(i)\\ &\Rightarrow \exists j \in TID.partner_q(i, j). \end{aligned} \tag{25}$$

Note that the aforementioned state of protocol WMDP in formulas (17)(18)(19) is a reachable state but doesn't satisfy the non-injective authentication property.

D. CHAIN RULE AND INSTANCE

The following rule derivations have been proved based on high order logic(HOL) via the Isabelle tool. In formulas 26-28, 30, their correctness can be easily verified according to the definitions of Pair, Relation $\prec_{tr}$, and $Know(tr)$. Formula 29 can be concluded from the definition of the semantic model (formulas 10-15), which reveals that the event a of the adversary learning message from the event b must occur after b itself.

$$\frac{\langle m_1, m_2\rangle = \langle m_3, m_4\rangle}{m_1 = m_3 \wedge m_2 = m_4}\mathrm{EQUALS}, \tag{26}$$

where EQUALS rule states that the equation $\langle m_1, m_2\rangle = \langle m_3, m_4\rangle$ holds if and only if the corresponding elements in pair are equal.

$$\frac{\langle m_1, m_2\rangle \in Know(tr)}{m_1 \in Know(tr)}\mathrm{KN}_1, \quad \frac{\langle m_1, m_2\rangle \in Know(tr)}{m_2 \in Know(tr)}\mathrm{KN}_2, \tag{27}$$

where KN_1, KN_2 rules state that if adversary knows a pair $\langle m_1, m_2\rangle$, he also knows m_1 and m_2.

$$\frac{\langle m_1, m_2\rangle \prec_{tr} e}{m_1 \prec_{tr} e}\mathrm{ORD}_1, \quad \frac{\langle m_1, m_2\rangle \prec_{tr} e}{m_2 \prec_{tr} e}\mathrm{ORD}_2, \tag{28}$$

where D_1, D_2 rules state that the message $\langle m_1, m_2\rangle$ is learned by adversary before the event e happens; similarly, both m_1 and m_2 are also learned before the event e.

$$\frac{m \prec_{tr} e}{m \in Know(tr)}\text{KNOWN}, \quad \frac{(i,s) \prec_{tr} e}{(i,s) \in Know(tr)}\text{EXEC}, \tag{29}$$

where rules KNOWN and EXEC state that the adversary knows the message or step, which implies that the event that adversary learns message m has happened or the event e has.

$$\frac{e \prec_{tr} e}{false}\text{IRR}, \quad \frac{e_1 \prec_{tr} e_2 \; e_2 \prec_{tr} e_3}{e_1 \prec_{tr} e_3}\text{TRANS}, \tag{30}$$

where rules Irr and Trans show that the relation $\prec_{tr}$ is a strict partial order. In other words, it's not reflexive closure, but transitive closure. Formulas 31-32 are a further expansion on the basis of formula 30. To be specific, the correlation of the two orders is illustrated by substituting the events in formula 30 into the role events in the protocol.

$$\frac{role_{th}(i) = R \; s' <_R s \; (i,s) \in steps(tr)}{(i,s') \prec_{tr} (i,s)}\text{ROLE}, \tag{31}$$

where rule ROLE states that if thread i that is an instance of role R has executed role step s, then all role steps s' ($s' <_R s$) have been executed before s via thread i.

$$\frac{(i, \text{Recv}_l(pt)) \in steps(tr)}{inst_{\sigma,i}(pt) \prec_{tr} (i, \text{Recv}_l(pt))}\text{INPUT}, \tag{32}$$

where rule INPUT states that before the step $\text{Recv}_l(pt))$ happens, the adversary has learned the instance of pattern pt.

$$\frac{m \in Know(tr)}{\begin{array}{c}(m \in AK_0) \vee \\ (\exists x.m = h(x) \wedge x \prec_{tr} h(x)) \vee \\ (\exists x,k.m = \{x\}_k \wedge x \prec_{tr} \{x\}_k \wedge k \prec_{tr} \{x\}_k) \vee \\ (\exists x,y.m = \langle x,y\rangle \wedge x \prec_{tr} \langle x,y\rangle \wedge y \prec_{tr} \langle x,y\rangle) \vee \\ (\exists R \in P. \exists \text{Send}_l(pt) \in R. \exists i. role_{th}(i) = R \wedge \\ chain_{tr}(\{i, \text{Send}_l(pt)\}, inst_{\sigma,i}(pt), m))\end{array}}\text{CHAIN}, \tag{33}$$

where rule CHAIN shows the ways adversary can learn message m. The predicate $chain_{tr}(E, m', m)$ is denoted by:

$$\begin{aligned} &chain_{tr}(E, m', m) \\ &\quad := (m' = m \wedge (\forall e \in E. e \prec_{tr} m)) \\ &\qquad \vee (\exists x,k. m' = \{x\}_k \wedge (\forall e \in E. e \prec_{tr} \{x\}_k) \\ &\qquad \wedge chain_{tr}(\{\{x\}_k, k^{-1}\}, x, m)) \\ &\qquad \vee (\exists x,y. m' = \langle x,y\rangle \\ &\qquad \wedge (chain_{tr}(E,x,m) \vee chain_{tr}(E,y,m))). \end{aligned} \tag{34}$$

where E denotes a trace event. Formula 34 is the ability of the adversary's derivation of the semantic model, and the induction of formulas 10-15 proves all the ways the adversary can obtain m. According to the type of messages, it is easy to see the way of obtaining specific types of messages. Symmetric keys can only come from the adversary's initialized knowledge. Fresh values can only be obtained by decrypting the corresponding session key or the message sent directly by the sender. Hash values and encrypted messages can only be generated by the adversary or sent directly by the sender. Combing eWMDP specification (see formula 4), we have the following simplified chain rules:

$$\frac{k(I,R) \in Know(tr)}{k(I,R) \in AK_0}\text{KCHAIN}_{IR}, \tag{35}$$

where rule N_{IR} states that the session key can be learned by the adversary only through AK_0.

$$\begin{array}{c}\dfrac{n\#i \in Know(tr)}{\begin{array}{l}(role_{th}(i) = I \wedge role_{th}(j) = R \wedge ((i,I_1) \\ \prec_{tr} \{\tilde{c}, \tilde{m}, h(\tilde{c}, \tilde{m})\}_{k(\sigma(I,i),\sigma(R,i))} \prec_{tr} \tilde{m}\#i \\ \wedge k(\sigma(s,i), \sigma(s,j)) \prec_{tr} \tilde{c}\#i \wedge \\ (\tilde{c} = n \vee \tilde{m} = n)) \vee ((j,R_2) \prec_{tr} \\ (\tilde{r}\#j, h(\sigma(V,j), \tilde{r}\#j, k(\sigma(I,i), \\ \sigma(R,j)))) \prec_{tr} \tilde{r}\#j \wedge \tilde{r} = n))\end{array}}\text{NCHAIN}_{IR}, \\ \end{array} \tag{36}$$

where rule NCHAIN_{IR} states that the fresh value $n\#i$ can be learned by the adversary only through the decryption from session key $k(\sigma(I,i), \sigma(R,i))$ or the message $r\#j$ sent by role R.

$$\begin{array}{c}\dfrac{h(x) \in Know(tr)}{\begin{array}{l}(x \prec_{tr} h(x)) \vee (\exists i. role_{th}(i) = I \wedge \exists j. role_{th}(j) \\ = R \wedge ((j,R_2) \prec_{tr} (\tilde{r}\#j, h(\sigma(V,j), \tilde{r}\#j, \\ k(\sigma(I,j), \sigma(R,j)))) \prec_{tr} \tilde{r}\#j \wedge (\sigma(V,j), \tilde{r}\#j, \\ k(\sigma(I,j), \sigma(R,j))) = x))))\end{array}}\text{HCHAIN}_{IR}, \\ \end{array} \tag{37}$$

where rule HCHAIN_{IR} states that the adversary learns the message $h(x)$ by constructing it himself or the message sent by role R.

$$\begin{array}{c}\dfrac{\{txt\}_x \in Know(tr)}{\begin{array}{l}(txt \prec_{tr} \{txt\}_x \wedge x \prec_{tr} \{txt\}_x) \vee \\ (\exists i. role_{th}(i) = I \wedge \exists j. role_{th}(j) = R \wedge (((i,I_1) \\ \prec_{tr} (\{\tilde{c}\#i, \tilde{m}\#i, h(\tilde{c}\#i, \tilde{m}\#i)\}_{k(\sigma(I,i),\sigma(R,i))} \\ = \{txt\}_x)))\end{array}}\text{ECHAIN}_{IR}, \\ \end{array} \tag{38}$$

where rule ECHAIN_{IR} states that adversary learns the message $\{txt\}_x$ by constructing it himself or the message sent by role I.

E. INSTANCE OF SECURITY PROPERTIES OF EWMDP AND PROOF

In combinition with eWMDP specification (see formula 4), we define its secrecy property as:

$$\begin{aligned} \varphi_{\text{sec}_{eWMDP}}(tr, th, \sigma) &:= \forall i \in TID. role_{th}(i) \\ &= I \wedge \sigma(R,i) \notin Compromise \\ &\Rightarrow (\tilde{m}\#i \notin Know(tr) \wedge \tilde{c}\#i \notin Know(tr)). \end{aligned} \tag{39}$$

Proposition 1: eWMDP satisfies $\varphi_{\text{sec}_{eWMDP}}(tr, th, \sigma)$ *which the proof is Appendix VIII.*

In combinition with eWMDP specification, we define its injective authentication property as:

$$\begin{aligned}&\varphi_{\text{non-injective-auth}_{eWMDP}}(tr, th, \sigma)\\ &:= \forall i \in TID.role_{th}(i)\\ &= I \wedge \sigma(R, i) \notin Compromise \wedge (i, R_2) \in steps(tr)\\ &\Rightarrow (\exists j \in TID.role_{th}(j) = R \wedge \sigma(R, i) = \sigma(R, j)\\ &\quad \wedge \tilde{c}\#i = \sigma(W, j) \wedge (i, I_1) \prec_{tr} (j, R_1) \wedge (j, R_2) \prec_{tr} (i, I_2))\end{aligned} \tag{40}$$

Proposition 2: eWMDP satisfies $\varphi_{\textit{non-injective-auth}_{eWMDP}}$ *of which the proof is Appendix VIII.*

Theorem 4.1 (Non-Injective to Injective Authentication): Suppose the protocol satisfies non-injective two-party authentication property

$$\begin{aligned}&\forall q \in reachable(P). \forall i \in TID.claim_q(i)\\ &\qquad \Rightarrow \exists j \in TID.partner_q(i, j),\end{aligned} \tag{41}$$

for the definition of the predicate claim and partner which satisfies the injectivity property, the injective authentication property

$$\begin{aligned}&\forall q \in reachable(P). \exists f.inj_{\{i \in TID.claim_q(i)\}}(f)\\ &\qquad \wedge \forall i \in TID.claim_q(i) \Rightarrow partner_q(i, f(j))\end{aligned} \tag{42}$$

also holds, where $inj_A(f)$ *means that* f *is injective on set* A*. It is defined as:* $A \subseteq dom(f)$*.* f *is* $inj_A(f)$ *if and only if* $\forall x, y \in A. f(x) = f(y) \Rightarrow x = y$*.*

Theorem 4.1 has been proved by Isabelle/HOL tool in [41]. Next, we prove that the protocol eWMDP also satisfies the injective-auth property.

Proposition 3: eWMDP satisfies $\varphi_{\textit{injective-auth}_{eWMDP}}$ *whose proof is Appendix VIII.*

The idea of proof: the proposition is proved to be correct by assuming that the unsafe role step event has occurred and by tracing back through the way (chain rule) the adversary gets the message, so as to obtain the contradiction with the premise.

V. ANALYSIS SECURITY PROTOCOLS BY TAMARIN-PROVER

To further understand the security of the protocol, we use Scyther [42] and tamarin-prover [41], [43] to analyze WMDP and eWMDP. They all have their own syntax. We need to define the protocol specification to tell the software the protocol content and security properties for analysis. To simplify the writing of the text, the Dolev-Yao model has been predefined in the tool and can be seen through the graphic interface (see formulas 49-53). The syntax involves terms, facts and rules. The terms are defined as follows:

$$\begin{aligned}term ::= &\ 'true'|'false'\\ &|ID\\ &|Function(termlist)\\ &|\{term\}_{term},\end{aligned} \tag{43}$$

where $'true'$ and $'false'$ are constant symbols, and ID is either variable or constant. $Function(termlist)$ is function symbols which is a kind of signature occurring in equation theory. Let f be a function, $dom(f)$ and $ran(f)$ denote the domain and the range of f respectively. $\{term\}_{term}$ is a function symbol which means using the key $_{term}$ to encrypt/decrypt $\{term\}$. It is a well-formed expression which is produced by applying operation. A fact is a first-order formula which is a combination of logical symbols (e.g ¬(not), ∧(and), ∨(or), ∀(for all), ∃(exist), ≈(equal)) and terms. In other words, terms can be split into four parts (function, predicate, variable, constant) respectively. In our model, the set of functions whose elements occur free is $\{first, second, xor, Fresh, AgSt, RecvCS, Out, In, h, KU, KD\}$. The set of the functions which only occur in traces is $\{Equals, AnswerRequest, SessKeyC, Know\}$. The set of variables is $\{k, c, m, r, hcm, S, B, x, y\}$. The set of constants is $\{A, B, X\}$.

A. EQUATION THEORY

Equation theory is a set of first-order atomic formulas which consist of equation symbol $\simeq$. See following:

$$\begin{aligned}first(x, y) &\simeq x,\\ second(x, y) &\simeq y,\\ \{\{x\}_k\}_k &\simeq x,\\ xor(x, y) &\simeq xor(y, x),\\ xor(x, xor(x, y)) &\simeq y,\\ xor(xor(x, x), y) &\simeq y.\end{aligned} \tag{44}$$

where these equations provide a relationship between terms that help the reasoning system deduce the state of the system. The operations which occur frequently in protocol elements include pair(x,y) operation, encryption/decryption, xor operation.

B. PROTOCOL SPECIFICATION RULES

This part is used to simulate the process of sending/receiving messages to each other in tamarin-prover. Combining the following adversary attack behavior system, we analyze the security properties of the protocol. They are the factual elements of execution system of the protocol. See formulas 45-48.

$$Registekey\ \frac{Fresh(\tilde{k})}{AgSt(A, \tilde{k}), AgSt(B, \tilde{k})} \tag{45}$$

$$SendCS\ \frac{Fresh(\tilde{c}), Fresh(\tilde{m}), AgSt(A, B, k)}{RecvCS(B, \tilde{c}, \tilde{m}, k), Out(\{\tilde{c}, \tilde{m}, h(\tilde{c}, \tilde{m})\}_k)} \tag{46}$$

$$\begin{aligned}&Fresh(\tilde{r}),\\ &AgSt(\mathrm{A}, \mathrm{B}, k),\\ &RecvCS\\ &SendSC\end{aligned}\ \frac{In(\{c, m, hcm\}_k)}{Out(\tilde{r}, h(c, \tilde{r}, k))} \begin{bmatrix} Equals(h(c, m), hcm),\\ AnswerRequest(\mathrm{B}, c, m)\end{bmatrix} \tag{47}$$

$$RecvSC\ \frac{\begin{matrix}RecvCS(B, c, m, k),\\ In(r, h(c, r, k))\end{matrix}}{} \begin{bmatrix} SessKeyC(S, k),\\ SessKeyC(S, m),\\ SessKeyC(S, c),\\ Equals((c, r, k),\\ h(c, r, k))\end{bmatrix} \tag{48}$$

Out() is a public channel which can be listened by the adversary, see formula (50). The value of the terms $Fresh(\tilde{k})$, $Fresh(\tilde{c})$, $Fresh(\tilde{r})$ is generated by a thread freshly. The value is the element belonging to $dom(Fresh)$, in which *Fresh* is a function symbol. The symbol $\#i$ ($i \in \mathbb{N}$, and $\mathbb{N}$ is natural number) means that i is the ID of a thread. $Fresh(a)\#i$ denotes an instance of the term a at thread i. Therefor, in $(a \approx b) \wedge Fresh(a)\#i \wedge Fresh(b)\#j \Rightarrow (i = j)$, a, b are infact the same term.

C. ADVERSARY MODEL

The Dolev-Yao adversary which is simulated by the set of rules can control the communication network. The definition of the rules of the adversary model in the following shows that the adversary can generate fresh values by the function $Fresh(\tilde{r})$, and increase the adversary's knowledge via the function $Know(x)$.

$$AdvSend\frac{KU(x)}{In(x)}[Know(x)] \tag{49}$$

$$AdvRecv\frac{Out(x)}{KD(x)} \tag{50}$$

$$coerce\frac{KD(x)}{KU(x)}[KU(x)] \tag{51}$$

$$pub\frac{}{KU(\mathrm{X})}[KU(\mathrm{X})] \tag{52}$$

$$fresh\frac{Fresh(\tilde{x})}{KU(\tilde{x})}[KU(\tilde{x})] \tag{53}$$

The term $KU(x)$ denotes the messages that the adversary can send. The messages can be generated either by a public constant such as $KU(A)$ or by the user's output that is denoted by $Out(x)$. The reason is that $KD(x)$ which the adversary can receive is generated by $Out(x)$. In addition, as a fresh value, it can also be generated by $Fresh(\tilde{x})$. In these rules, we can empower the adversary to get information denoted by $Know(x)$ from multiple sources. Compared with $Know(x)$, $KU(x)$ also holds information generated or combined by the adversary itself. On the other hand, the message the adversary can send is the message that all receivers may receive such as $In(x)$.

D. RESTRICTION OF THE SET OF TRACES

The constraint section is used to simulate the matching process in which the user is receiving the message, in order to detect the adversary's behavior in time. A trace atom is the term *false*, a term like $t_1 \approx t_2$, a time point of a thread ordering ($i < j$ or $i \doteq j$), or action $f@j$ for a fact f and a time point i.

$$\forall x, y, i\,.\,(Equals(x, y)@i) \Rightarrow (x \approx y), \tag{54}$$

where the role of constraint restriction is to ensure that the receiver performs a matching check after the message is received, e.g., $Equals((c, r, k), h(c, r, k))$. See formula (48).

VI. VERIFICATION

We use tamarin-prover to verify the security properties of the protocol WMDP. The procedure is as follows:

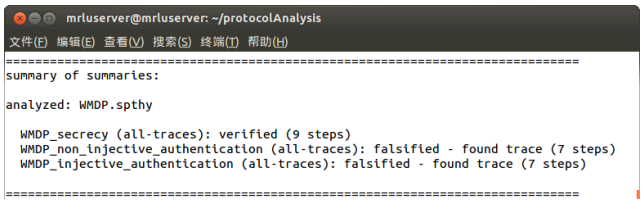

FIGURE 2. The analysis of WMDP by tamarin-prover [44].

The analysis of WMDP in FIGURE 2 is the result of operation under tamarin-prover. Three security properties of the protocol WMDP are analyzed, in which the tool finds the attack trace of the second and third through seven steps. It shows that the protocol WMDP doesn't satisfy the non-injective authentication property and the injective authentication property. The topological relation of security properties can be referred to [42].

A. PROPERTIES OF EWMDP

Now we prove that the eWMDP satisfies the following properties by way of tamarin-prover tool. These properties are expressed in the form of the syntax of tamarin-prover, which corresponds to the security properties defined earlier. The *reachable*(P) below represents all executions of protocol P.

Proposition 4: The secrecy property of session key is denoted as follows:

$$\begin{aligned}&\forall t \in reachable(P)\\ &\quad \neg(\exists S, k \in t, i, j.\\ &\qquad ((SessKeyC(S, k)@i) \wedge\\ &\qquad\quad (Know(k)@j)\\ &\qquad)\\ &\quad).\end{aligned} \tag{55}$$

where proposition 4 corresponds to the security property $\varphi_{sec_{eWMDP}}$.

Proposition 5: The non-injective authentication property of eWMDP between two partners is denoted as follows:

$$\begin{aligned}&\forall t \in reachable(P)\\ &\quad \forall S, k \in t, \exists i.(SessKeyC(S, k)@i)\\ &\Rightarrow ((\exists a.AnswerRequest(S, k)@a)).\end{aligned} \tag{56}$$

where proposition 5 corresponds to the non-injective authentication property $\varphi_{\mathrm{non-injectiv-auth}_{eWMDP}}$.

Proposition 6: The injective authentication property is denoted as follows:

$$\begin{aligned}&\forall t \in reachable(P)\\ &\quad \forall S, k \in t, \exists i.(SessKeyC(S, k)@i) \Rightarrow\\ &\qquad ((\exists a.AnswerRequest(S, k)@a) \wedge\\ &\qquad\quad (\forall j.(SessKeyC(S, k)@j) \Rightarrow\\ &\qquad\quad (i = j)\\ &\qquad)\\ &\quad).\end{aligned} \tag{57}$$

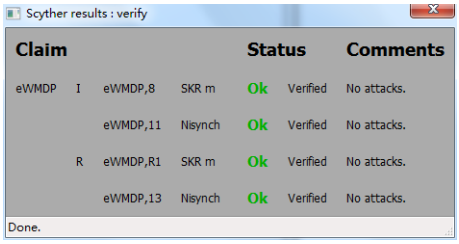

FIGURE 3. The analysis of eWMDP via Scyther.

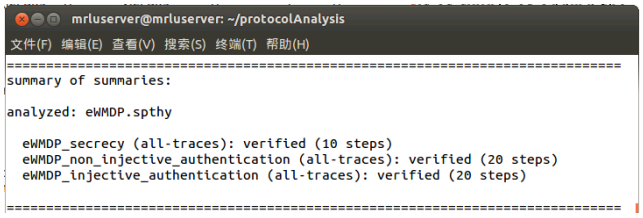

FIGURE 4. The analysis of eWMDP via tamarin-prover.

TABLE 1. Performance and security comparison for eWMDP and WMDP.

	Int	Enc	Dec	Ran	h	SKR	non-A	Injec
WMDP	3	1	1	2	6	√	×	×
eWMDP	2	1	1	2	4	√	√	√

where proposition 6 corresponds to the injective authentication property $\varphi_{injective-auth_{eWMDP}}$.

Scyther can only analyze the secrecy, liveness and non-injective authentication properties [45]. The tool cannot complete the proof of injective authentication and does not support self-definition of the operator. We only use it to analyze the security properties of eWMDP. The analysis in FIGURE 3 shows that eWMDP satisfies the secrecy property. The roles of I and R share the non-injective authentication property.

The analysis in FIGURE 4 shows that eWMDP satisfies the secrecy property, non-injective authentication property, and injective authentication property. Next, we compare the analysis results of the two protocols in the table.

In TABLE 1 below, we compare WMDP with eWMDP in several aspects, including Int (interaction wheel number), Enc (number of encryption), Dec (number of decryption), Ran (number of random), h (number of hash), SKR (secrecy property), non-A (non-injective authentication), and Injec (injective authentication), which shows that our protocol (eWMDP) is obviously more efficient than WMDP, and WMDP doesn't satisfy the last two security properties.

VII. IMPLEMENTATION

In the experiment, we first compare the handshake time of the two protocols communicating on TI CC3200 LAUCHPAD and PC in FIGURE 5. Secondly, the handshake time of eWMDP under different encryption and hash algorithms is tested in FIGURE 6. Finally we test the handshake time of eWMDP under different encryption and hash algorithms between PCs in FIGURE 7. The two communication roles I (client) and R (server) in the protocol run on

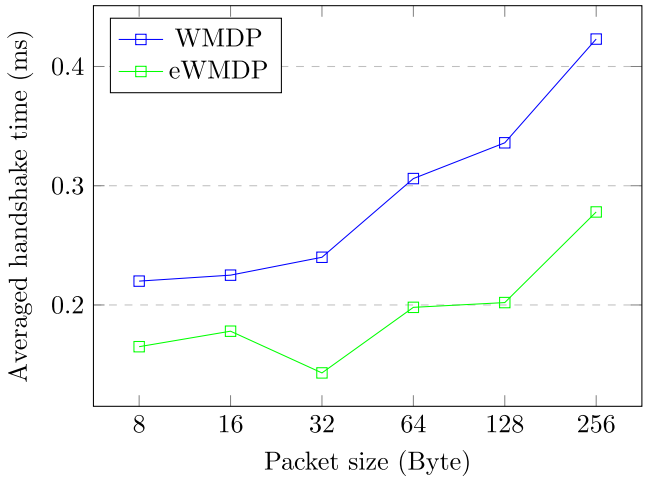

FIGURE 5. Handshake time with packet size between PC and Ti CC3200, where encryption algorithm = AES, hash algorithm = SHA256.

Ti CC3200 LAUNCHPAD and PC respectively. The server program which runs in Ubuntu operation system (Intel(R) Xeon(R) CPU E3-1220 V2 @ 3.10GHz) is implemented in C language and be compiled by GCC in CCSv7 (Code Composer Studio) platform. The security algorithm section deployed on server invokes some of the APIs such as DES, 3DES, AES, Blowfish for the OpenSSL library. The client deployed on Ti CC3200 LAUNCHPAD invokes for CC3200 SDK 1.3.0 library.

The grouping length of our DES algorithm is 64 bits, that of the 3DES 192bits, that of the AES 128 bits, and that of the Blowfish 128 bits. The lengths of hash text outputed by hash function including MD5, SHA160, SHA224, SHA256, SHA384, and SHA512 are 128 bits, 160 bits, 224 bits, 256 bits, 384 bits and 512 bits respectively. All below are used in CBC mode and implemented based on UDP protocol.

In the first part, we implement two protocols in the same environment platform in which they communicate between LAUNCHPAD and PC.

The two protocols are implemented by combining AES encryption algorithm with hash algorithm SHA256. In FIGURE 5, data is transmitted in size of 256, 128, 64, 32, 16 and 8 respectively. The time of the handshake of the two protocols is compared. As can be seen from the FIGURE 5 above, our protocol is significantly better than WMDP. The reason for using handshake twice rather than three times is that it is enough to satisfy the security properties.

As is shown in FIGURE 6 below, we extend the implementation of eWMDP to different scenarios, including the combination of DES, AES, and 3DES encryption algorithm with MD5, SHA160, SHA224, and SHA256.

By comparison, it can be seen that with the increasing complexity of algorithm, the time of running the protocol will be much longer. The choice of protocol algorithm depends on the balance between security requirements and performance requirements. The next experiment is to test the protocol between PCs. Based on the above implementation, Blowfish

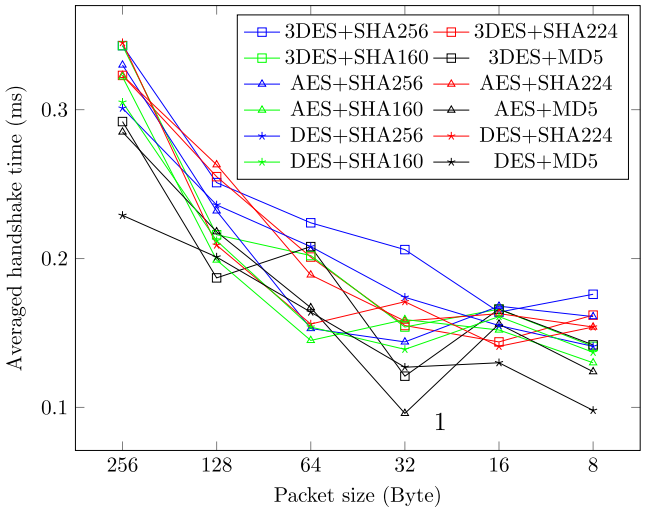

FIGURE 6. Handshake time with packet size between PC and Ti CC3200 LAUCHPAD under eWMDP.

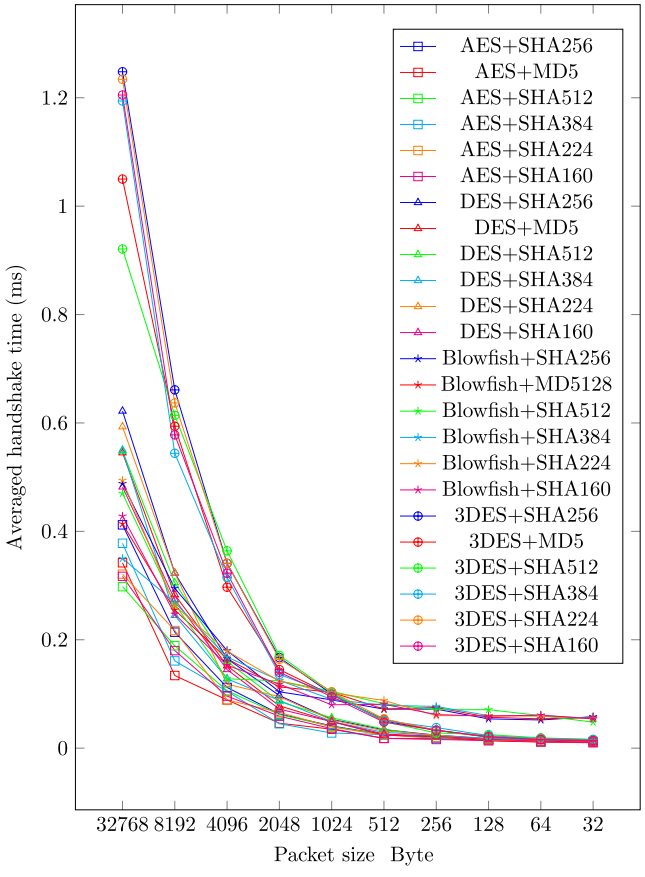

FIGURE 7. Handshake time with packet size between PCs under eWMDP.

encryption algorithm, SHA384 and SHA512 hash algorithms are newly added.

The average time of completing handshake of eWMDP between PC and LAUNCHPAD platform is 0.194181 milliseconds. As a reference, we have done more experiments between PCs. The communication latency between PC and LaunchPad is much longer than that between PCs. The time of completing handshake of eWMDP between PCs is showed in FIGURE 7.

As can be seen from the data above, 3DES algorithm performs poorly in eWMDP protocol. Moreover, in order to compare the performance differences between the completing time between PC and LAUCHPAD and that of between PCs, the sizes 256, 128, 64, 32 are selected. The combinations (twelve kinds in total) between DES, AES, 3DES, and MD5, SHA160, SHA224 SHA256 are selected. The average time between PCs is 0.084729167 milliseconds. By contrast, the average time between PC and LAUCHPAD is 0.1996625 milliseconds. The speed of the protocol eWMDP between PCs is 2.356478977 times that of between PC and LAUCHPAD.

VIII. CONCLUSION

In this paper, we have shown that WMDP doesn't satisfy injective-authentication property, and we provide eWMDP which holds in secrecy, non-injective authentication and injective-authentication properties, which shows that there is no trace that the adversary can know the secrecy between two communication partners or can communicate with one partner pretending to be the other. We have tried several schemes to implement the protocol and test their performances. The result can show that the protocol has a good performance under the various options.

APPENDIX A

Proposition 1: eWMDP satisfies $\varphi_{\text{sec}_{eWMDP}}(tr, th, \sigma)$

Proof: Suppose the secrecy property φ_{sec} does not hold for a state $(tr, th, \sigma) \in reachable(eWMDP)$. By the definition of the secrecy property, it can be explained as follows: $role_{th}(i) = I$, $\sigma(R, i) \notin Compromise$, and $\tilde{m}\#i \in Know(tr)$, which means that the adversary has learned $\tilde{m}$. And by means of the chain rules, we can figure out that there's only one case, i.e., the NCHAIN_{IR}. From the conclusion, we can figure out that the adversary can learn $\tilde{m}\#i$ only by decrypting the message $\{\tilde{c}, \tilde{m}, h(\tilde{c}, \tilde{m})\}_{k(\sigma(s,i),\sigma(s,j))}$, which implies $k(\sigma(s, i), \sigma(s, j)) \prec_{tr} \tilde{m}\#i$. Using KNOWN rule, we have $k(\sigma(s, i), \sigma(s, j)) \in Know(tr)$, and from KCHAIN_{IR}, we get $k(\sigma(s, i), \sigma(s, j)) \in AK_0$. According to the AK_0 definition, we have $\sigma(R, i) \in Compromise$, which yields a contradiction of our assumption.

Similarly, to prove the secrecy property of variable $\tilde{c}$, just replace $\tilde{m}$ above with $\tilde{c}$. Thus, we conclude that φ_{sec} holds for all reachable states q of the eWMDP protocol. □

APPENDIX B

Proposition 2: eWMDP satisfies $\varphi_{non\text{-}injective\text{-}auth_{eWMDP}}$

Proof: From $(i, R_2) \in steps(tr)$, we get $h(\tilde{c}\#i, \tilde{r}\#j, k(a, b)) \prec_{tr} (i, R_2)$ by using INPUT rule and get $h(\tilde{c}\#i, \tilde{r}\#j, k(a, b)) \in Know(tr)$ by using KNOWN rule. Applying HCHAIN_{IR} rule yields the following conclusions:

$$(1)(\tilde{c}\#i, \tilde{r}\#j, k(a, b)) \prec_{tr} h(\tilde{c}\#i, \tilde{r}\#j, k(a, b))$$

$$(2) \vee (\exists i.role_{th}(i) = I \wedge \exists j.\exists j \in TID.role_{th}(j) = R \wedge (j, R_2) \prec_{tr} h(\sigma(V, j), \tilde{r}\#j, k(\sigma(I, i), \sigma(R, j))) \wedge h(\sigma(V, j), \tilde{r}\#j, k(\sigma(I, i), \sigma(R, j))) = h(\tilde{c}\#i, \tilde{r}\#j, k(\sigma(I, i), \sigma(R, j))). \quad (58)$$

Case (1) means that the adversary builds the message $h(\tilde{c}\#i, \tilde{r}\#j, k(a, b))$ using $\tilde{c}\#i \in Know(tr)$ by himself. This contradicts the secrecy property we have proved in φ_{sec}.

Case (2) implies that there is a server thread j, $role_{th}(j) = S$, and j sends the message received by the client thread i. From $h(\sigma(V, j), \tilde{r}\#j, k(\sigma(I, j), \sigma(R, j))) = h(\tilde{c}\#i, \tilde{r}\#j, k(\sigma(I, i), \sigma(R, i)))$ and injectivity of hash function $h(\_)$, we can derive $\sigma(V, j) = \tilde{c}\#i$ and $\sigma(I, j) = \sigma(I, i) \wedge \sigma(R, j) = \sigma(R, i)$ by using EQUALS rule. From

$$(j, R_2) \prec_{tr} h(\sigma(V, j), \tilde{r}\#j, k(\sigma(I, i), \sigma(R, j))), h(\sigma(V, j), \tilde{r}\#j, k(\sigma(I, j), \sigma(R, j))) = h(\tilde{c}\#i, \tilde{r}\#j, k(\sigma(I, i), \sigma(R, i))),$$

and

$$h(\tilde{c}\#i, \tilde{r}\#j, k(\sigma(I, i), \sigma(R, i))) \prec_{tr} (i, I_2),$$

we can get $(j, R_2) \prec_{tr} (i, I_2)$. From $(j, R_2) \prec_{tr} (i, I_2)$, we have $(j, R_1) \prec_{tr} (j, R_2)$ by using rules EXEC and ROLE. Hence, we can get $\{\tilde{c}\#i, \tilde{m}\#i, h(\tilde{c}\#i, \tilde{m}\#i)\}_{k(\sigma(I,i),\sigma(R,i))} \prec_{tr} (j, R_1)$ by using the rules EXEC and INPUT and $\sigma(V, j) = \tilde{c}\#i$ and $\sigma(I, j) = \sigma(I, i) \wedge \sigma(R, j) = \sigma(R, i)$. From KNOWN rule, we have $\{\tilde{c}\#i, \tilde{m}\#i, h(\tilde{c}\#i, \tilde{m}\#i)\}_{k(\sigma(I,i),\sigma(R,i))} \in Know(tr)$. Applying ECHAIN_{IR} rule and EQUALS rule yields following results:

$$(2.1)(\tilde{c}\#i \prec_{tr} \{\tilde{c}\#i, \tilde{m}\#i, h(\tilde{c}\#i, \tilde{m}\#i)\}_{k(\sigma(I,i),\sigma(R,i))} \wedge k(\sigma(I, i), \sigma(R, i)) \prec_{tr} \{\tilde{c}\#i, \tilde{m}\#i, h(\tilde{c}\#i, \tilde{m}\#i)\}_{k(\sigma(I,i),\sigma(R,i))})$$
$$(2.2) \vee (\exists i'.role_{th}(i') = I \wedge \exists j'.role_{th}(j') = R \wedge ((i', I_1) \prec_{tr} (\{\tilde{c}\#i', \tilde{m}\#i', h(\tilde{c}\#i', \tilde{m}\#i')\}_{k(\sigma(I,i'),\sigma(R,i'))} = \{\tilde{c}\#i, \tilde{m}\#i, h(\tilde{c}\#i, \tilde{m}\#i)\}_{k(\sigma(I,i),\sigma(R,i))})) \quad (59)$$

Case (2.1) states that the adversary fakes the message, which again contradicts the secrecy property proven in φ_{sec} due to $\tilde{c}\#i \prec_{tr} \{\tilde{c}\#i, \tilde{m}\#i, h(\tilde{c}\#i, \tilde{m}\#i)\}_{k(\sigma(I,i),\sigma(R,i))}$ and the rule KNOWN.

Case (2.2) implies $i' = i$ since $k\#i' = k\#i$. Hence, we have

$$(i, I_1) \prec_{tr} \{\tilde{c}\#i, \tilde{m}\#i, h(\tilde{c}\#i, \tilde{m}\#i)\}_{k(\sigma(I,i),\sigma(R,i))} = \{\tilde{c}\#i, \tilde{m}\#i, h(\tilde{c}\#i, \tilde{m}\#i)\}_{k(\sigma(I,j),\sigma(R,j))} \prec_{tr} (j, R_1). \quad (60)$$

This implies that $\sigma(R, j) = \sigma(R, i)$ and that $(i, I_1) \prec_{tr} (j, R_1)$, which completes the proof. □

APPENDIX C

Proposition 3: eWMDP satisfies injective-auth property.

Proof: From the $\varphi_{\text{non-injective-auth}_{eWMDP}}$ we have:

$$claim_{(tr,th,\sigma)}(i) := role_{th}(i) = I \wedge \sigma(R, i) \notin Compromise \wedge (i, R_2) \in steps(tr)$$
$$partner_{(tr,th,\sigma)}(i, j) := role_{th}(j) = R \wedge \sigma(R, i) = \sigma(R, j) \wedge \tilde{m}\#i = \sigma(W, j) \wedge (i, I_1) \prec_{tr} (j, R_1) \wedge (j, R_2) \prec_{tr} (i, I_2). \quad (61)$$

From the condition that $\tilde{m}\#i = \sigma(W, j)$, we get that both threads i, j agree on the nonce $\tilde{m}\#i$ which is generated by thread i. For all threads i_1, i_2, j, from the condition that $\tilde{m}\#i_1 = \sigma(W, j)$ and the condition that $\tilde{m}\#i_2 = \sigma(W, j)$, we get $\tilde{m}\#i_1 = \sigma(W, j) = \tilde{m}\#i_2$. According to the fresh value property, we have $i_1 = i_2$, which is consistent with the injective definition (25). Using Theorem 4.1, we conclude that the role I of eWMDP is injective with the role R. □

REFERENCES

[1] A. J. Mills, R. T. Watson, L. Pitt, and J. Kietzmann, "Wearing safe: Physical and informational security in the age of the wearable device," *Bus. Horizons*, vol. 59, no. 6, pp. 615–622, Nov./Dec. 2016.
[2] S. Banerjee, T. Hemphill, and P. Longstreet, "Is IOT a threat to consumer consent? The perils of wearable devices' health data exposure," *SSRN Electron. J.*, vol. 2017, pp. 1–42, Sep. 2017.
[3] B. Wang, Y. Zheng, W. Lou, and Y. T. Hou, "DDoS attack protection in the era of cloud computing and software-defined networking," *Comput. Netw.*, vol. 81, pp. 308–319, Mar. 2015.
[4] M. Conti, N. Dragoni, and V. Lesyk, "A survey of man in the middle attacks," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 3, pp. 2027–2051, 3rd Quart., 2016.
[5] S. Almousa and M. Barbeau, "Delay and reflection attacks in authenticated semi-quantum direct communications," in *Proc. IEEE Globecom Workshops (GCWkshps)*, Dec. 2016, pp. 1–7.
[6] M. S. Farash, M. Turkanović, S. Kumari, and M. Hölbl, "An efficient user authentication and key agreement scheme for heterogeneous wireless sensor network tailored for the Internet of Things environment," *Ad Hoc Netw.*, vol. 36, pp. 152–176, Jan. 2016.
[7] K. Diab, L. Fei, S. Giombi, I. R. Klebanov, and G. Tarnopolsky, "On C_J and C_T in the gross-Neveu and O(N) models," *J. Phys. A, Math. Gen.*, vol. 49, no. 40, Oct. 2016, Art. no. 405402.
[8] X. Zhang, Q. Gao, and M. K. Saad, "Looking at a class of RFID APs through GNY logic," *Int. J. Secur. Netw.*, vol. 5, nos. 2–3, pp. 135–146, Mar. 2010.
[9] S. Ahmadi and M. S. Fallah, "An omniscience-free temporal logic of knowledge for verifying authentication protocols," *Bull. Iranian Math. Soc.*, vol. 44, no. 5, pp. 1243–1265, Oct. 2018.
[10] S. Meier, C. Cremers, and D. Basin, "Efficient construction of machine-checked symbolic protocol security proofs," *J. Comput. Secur.*, vol. 21, no. 1, pp. 41–87, Feb. 2013.
[11] S. Shiraz and O. Hasan, "A library for combinational circuit verification using the HOL theorem prover," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 37, no. 2, pp. 512–516, Feb. 2018.
[12] Q. Carbonneaux, J. Hoffmann, T. Reps, and Z. Shao, *Automated Resource Analysis With Coq Proof Objects*. Heidelberg, Germany: Springer, 2017, pp. 64–85.
[13] A. Spector-Zabusky, J. Breitner, C. Rizkallah, and S. Weirich, "Total Haskell is reasonable Coq," in *Proc. 7th ACM SIGPLAN Int. Conf. Certified Programs Proofs*, Jan. 2018, pp. 14–27.
[14] M. M. Wenzel, "Isabelle/Isar—A versatile environment for human-readable formal proof documents," Ph.D. dissertation, Lehrstuhl Softw. Syst. Eng., Technische Universität München, Munich, Germany, Oct. 2018.
[15] P. Masci, P. Curzon, D. Furniss, and A. Blandford, "Using PVS to support the analysis of distributed cognition systems," *Innov. Syst. Softw. Eng.*, vol. 11, no. 2, pp. 113–130, Jun. 2015.
[16] C. Kaliszyk, K. Pąk, and J. Urban, "Towards a Mizar environment for Isabelle: Foundations and language," in *Proc. 5th Conf. Certified Programs Proofs*, Petersburg, FL, USA, 2016, pp. 58–65.
[17] C. Cremers and L. Hirschi, "Improving automated symbolic analysis for E-voting protocols: A method based on sufficient conditions for ballot secrecy," 2017, *arXiv:1709.00194*. [Online]. Available: https://arxiv.org/abs/1709.00194
[18] C. Cremers and M. Horvat, "Improving the ISO/IEC 11770 standard for key management techniques," in *Security Standardisation Research*, vol. 8893, L. Chen and C. Mitchell, Eds. Cham, Switzerland: Springer, 2014, pp. 215–235.

[19] C. Cremers, M. Dehnel-Wild, and K. Milner, "Secure authentication in the grid: A formal analysis of DNP3: SAv5," in *Computer Security-ESORICS*, vol. 10492, S. Foley, D. Gollmann, and E. Snekkenes, Eds, Cham, Switzerland: Springer, 2017, pp. 389–407.
[20] X. Li, M. H. Ibrahim, S. Kumari, A. K. Sangaiah, V. Gupta, and K.-K. R. Choo, "Anonymous mutual authentication and key agreement scheme for wearable sensors in wireless body area networks," *Comput. Netw.*, vol. 129, pp. 429–443, Dec. 2017.
[21] Z. Chen, W. Ren, Y. Ren, and K.-K. R. Choo, "LiReK: A lightweight and real-time key establishment scheme for wearable embedded devices by gestures or motions," *Future Gener. Comput. Syst.*, vol. 84, pp. 126–138, Jul. 2018.
[22] Y. Zhang, M. Yang, Z. Ling, Y. Liu, and W. Wu, "FingerAuth: 3D magnetic finger motion pattern based implicit authentication for mobile devices," *Future Gener. Comput. Syst.*, to be published.
[23] P. Samangouei, V. M. Patel, and R. Chellappa, "Facial attributes for active authentication on mobile devices," *Image Vis. Comput.*, vol. 58, pp. 181–192, Feb. 2017.
[24] P. Peris-Lopez, L. González-Manzano, C. Camara, and J. María de Fuentes, "Effect of attacker characterization in ECG-based continuous authentication mechanisms for Internet of Things," *Future Gener. Comput. Syst.*, vol. 81, pp. 67–77, Apr. 2018.
[25] C. Shen, Y. Chen, and X. Guan, "Performance evaluation of implicit smartphones authentication via sensor-behavior analysis," *Inf. Sci.*, vols. 430–431, pp. 538–553, Mar. 2018.
[26] Y. Zhang, R. Gravina, H. Lu, M. Villari, and G. Fortino, "Pea: Parallel electrocardiogram-based authentication for smart healthcare systems," *J. Netw. Comput. Appl.*, vol. 117, pp. 10–16, Sep. 2018.
[27] S. Liu, S. Hu, J. Weng, S. Zhu, and Z. Chen, "A novel asymmetric three-party based authentication scheme in wearable devices environment," *J. Netw. Comput. Appl.*, vol. 60, pp. 144–154, Jan. 2016.
[28] D. He and S. Zeadally, "Authentication protocol for an ambient assisted living system," *IEEE Commun. Mag.*, vol. 53, no. 1, pp. 71–77, Jan. 2015.
[29] W. J. Long and W. Lin, "An authentication protocol for wearable medical devices," in *Proc. 13th Int. Conf. Expo Emerg. Technol. Smarter World (CEWIT)*, Nov. 2017, pp. 1–5.
[30] J. Shen, Z. Gui, S. Ji, J. Shen, H. Tan, and T. Yi, "Cloud-aided lightweight certificateless authentication protocol with anonymity for wireless body area networks," *J. Netw. Comput. Appl.*, vol. 106, pp. 117–123, Mar. 2018.
[31] T. Van hamme, D. Preuveneers, and W. Joosen, "Managing distributed trust relationships for multi-modal authentication," *J. Inf. Secur. Appl.*, vol. 40, pp. 258–270, Jun. 2018.
[32] P. Vijayakumar, P. Pandiaraja, M. Karuppiah, and L. J. Deborah, "An efficient secure communication for healthcare system using wearable devices," *Comput. Electr. Eng.*, vol. 63, pp. 232–245, Oct. 2017.
[33] A. Alhothaily, A. Alrawais, C. Hu, and W. Li, "One-time-username: A threshold-based authentication system," *Procedia Comput. Sci.*, vol. 129, pp. 426–432, Jan. 2018.
[34] Q. Jiang, J. Ma, C. Yang, X. Ma, J. Shen, and A. C. Shehzad, "Efficient end-to-end authentication protocol for wearable health monitoring systems," *Comput. Elect. Eng.*, vol. 63, pp. 182–195, Oct. 2017.
[35] K.-H. Wang, C.-M. Chen, W. Fang, and T.-Y. Wu, "A secure authentication scheme for Internet of Things," *Pervasive Mobile Comput.*, vol. 42, pp. 15–26, Dec. 2017.
[36] *IEEE Standard for Local and Metropolitan Area Networks—Part 15.6: Wireless Body Area Networks*, IEEE Standard 802.15.6-2012, Feb. 2012, pp. 1–271.
[37] D. Piccinini, N. B. Andino, S. D. Ponce, M. A. Roberti, and N. López, "Wearable system for acquisition and monitoring of biological signals," *J. Phys., Conf. Ser.*, vol. 705, no. 1, Apr. 2016, Art. no. 012009.
[38] R. K. Kodali, V. Jain, S. Bose, and L. Boppana, "IoT based smart security and home automation system," in *Proc. Int. Conf. Comput., Commun. Automat. (ICCCA)*, Apr. 2016, pp. 1286–1289.
[39] R. K. Kodali, "An implementation of MQTT using CC3200," in *Proc. Int. Conf. Control, Instrum., Commun. Comput. Technol. (ICCICCT)*, Dec. 2016, pp. 582–587.
[40] F. Bamarouf, C. Crandell, S. Tsuyuki, J. Sanchez, and Y. Lu, "Cloud-based real-time heart monitoring and ECG signal processing," in *Proc. IEEE SENSORS*, Nov. 2016, pp. 1–3.
[41] S. Meier, "Advancing automated security protocol verification," Ph.D dissertation, ETH Zürich, Zürich, Switzerland, 2013.
[42] C. Cremers, *Scyther User Manual*. Springer, 2014.
[43] The Tamarin Team, *Tamarin-Prover Manual Security Protocol Analysis in the Symbolic Model*. Berlin, Germany: Springer, 2018.
[44] D. Basin, C. Cremers, J. Dreier, and R. Sasse, "Symbolically analyzing security protocols using tamarin," *ACM SIGLOG*, vol. 4, no. 4, pp. 19–30, Oct. 2017.
[45] C. Cremers and S. Mauw, "Operational semantics and verification of security protocols," in *Information Security and Cryptography*. Berlin, Germany: Springer, 2014.

BO LU received the B.Eng. degree from the Business College of Shanxi University, Taiyuan, China, in 2010, and the master's degree from Shanxi University, Taiyuan, China, in 2014. He is currently pursuing the Ph.D. degree with the College of Cyberspace Security, Beijing University of Posts and Telecommunications (BUPT), Beijing, China. His research interests include protocol security analysis and safety critical systems.

RUOHAN CAO received the B.Eng. degree from the Shandong University of Science and Technology (SDUST), Qingdao, China, in 2009, and the Ph.D. degree from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2014. From November 2012 to August 2014, she also served as a Research Assistant for the Department of Electrical and Computer Engineering, University of Florida, supported by the China Scholarship Council. She is currently with the Institute of Information Photonics and Optical Communications, BUPT, as a Postdoctoral. Her research interests include physical-layer network coding, multiuser multiple-input-multiple-output systems, and physical-layer security.

YUEMING LU received the B.S. and M.S. degrees in computer science from the Xi'an University of Architecture and Technology, in 1994 and 1997, respectively, and the Ph.D. degree in computer architecture from Xi'an Jiaotong University, in 2000. He is currently a Professor with the Beijing University of Posts and Telecommunications (BUPT). His research interests include security control, evaluation, and data protection.

XUETING LUO received the B.Eng. degree from the Civil Aviation University of China (CAUC), Tianjin, China, in 2013. She is currently pursuing the master's degree with the Beijing University of Post and Telecommunications (BUPT), Beijing, China. Her research interest includes network security.