Modeling and Verification of IEEE 802.11i Security Protocol for Internet of Things

Yuteng Lu and Meng Sun

LMAM & Department of Informatics, School of Mathematical Sciences, Peking University, Beijing, China {luyuteng,sunm}@pku.edu.cn

Abstract—IEEE 802.11i is the IEEE standard that provides enhanced MAC security and has been widely used in wireless networks and Internet of Things. It improves IEEE 802.11(1999) by providing a Robust Security Network (RSN) with two new protocols: the 4-way handshake and the Group-key handshake. These protocols utilize the authentication services and port access control described in IEEE 802.1X to establish and change the appropriate cryptographic keys. In this paper, we carry out a formal modeling and verification approach based on timed automata for IEEE 802.11i protocol, using the UPPAAL model checker, to check correctness of the changes in IEEE 802.11i protocol and provide better security.

Keywords: IEEE802.11i Protocol, Model Checking, 4-Way Handshake, Group-Way Handshake, UPPAAL

I. INTRODUCTION

The Internet-of-Things (IoT) [5], [17] is an evolving paradigm that offers a family of sophisticated computing services and physical instruments which cooperate with each other over the Internet. It has gained increased attention in the past decade due to rapid development in computing and storage technologies and easy access to the Internet, and brings promising opportunities and challenges. IoT has played a key role in the next generation of information, network, and communication systems. With the rapid development of IoT, security of communication, being of paramount importance, has attracted more and more attention. Since the intruder can attack a network remotely, the industry has been trying to introduce various security protocols to improve the security of the wireless network and IoT.

IEEE 802.11 is a set of standards defined by IEEE for wireless network communication, such as 802.11e for QoS enhancement of 802.11, 802.11k for radio resource management, and 802.11n for high throughput enhancement, and so on. In the actual usage, IEEE802.11 exposed a lot of security issues, so IEEE also developed a set of IEEE802.11 amendment to make up for its fragile security encryption, which is the IEEE802.11i protocol being studied in this paper.

Industrial practice has shown that in the design of a complex protocol, more time and effort are usually spent on verification of the correctness of the protocol, rather than in the formulation of the protocol itself. Formal verification techniques, especially model checking [3], aim to establish correctness with mathematical rigor and offer a large potential to obtain an early integration of verification in the design process, to make verification activities more effective, and to reduce the verification time. For these reasons, model checking has been recognized as an important method to guarantee the correctness of protocols formally and avoid further odious problems caused by errors in early stage.

A large body of literature for analysis and verification of IEEE 802.11i protocol already exist. For example, the Highlevel Petri Net (HPN) model was adopted in [10] to specify the protocol framework of the 4-way handshake protocol, and model checking techniques are used to carry out the security verification on the HPN models. In [11] a logic based approach is taken to verify several properties of some typical methods in extensible authentication protocol, which are major solutions in IEEE 802.11i implementation. The behavior tree models for IEEE 802.11i RSN were developed and verified by using the SAL model checker in [15]. The IEEE 802.11i amendment is analyzed and a number of potential threats are identified in [18]. The 4-way handshake authentication WPA-PSK protocol in IEEE802.11i was verified in [13] using the CasperFDR model checker. On the other hand, the 4-way handshake phase in IEEE 802.11i Standard has been analyzed in [1] using theorem prover Isabelle to identify a new Denial-of-Service (DoS) attack. A key refreshing technique to reduce 4-way handshake latency in 802.11i based networks was proposed in [14] which provides per frame key freshness and generates a new refreshed secret key for encryption of each frame. However, most of these works only focus on part of IEEE 802.11i protocol and ignore the other parts.

In this paper, we investigate the usage of UPPAAL [16] to analyze and verify different protocols in the IEEE 802.11i standard. UPPAAL is a toolbox for verification of real-time systems that can be modeled as networks of timed automata (TA) [2] extended with integer variables, structured data types, and channel synchronization. UPPAAL can be used to automatically check whether a given property is satisfied by a system. The query language of UPPAAL used to specify the properties to be checked is a subset of CTL. It uses a client-server architecture, splitting the tool into a graphical user interface and a model checking engine. The UPPAAL model checker is based on the theory of timed automata and its modeling language offers additional features such as bounded integer variables and urgency. In the past decades, UPPAAL has been applied successfully in various industrial case studies

DOI reference number: 10.18293/SEKE2018-060.

ranging from communication protocols to multimedia applications, such as Bounded Retransmission Protocol [4], Bang & Olufsen audio / video protocol [6] and Philips audio protocol [9].

In an actual network communication, after the user login the network, whether its landing time is more than the scheduled time and in each time period whether it is still online are the problems that the authentication side must concern about. So timing parameters play an essential role in the IEEE 802.11i protocol, especially for communications in wireless networks and IoT. In addition, our work also examines whether the protocol will be deadlocked, which is a vital security attribute in communication. For wireless network and IoT, it is also an important issue that the interactive port can eventually be opened normally.

The rest of this paper is organized as follows: The IEEE 802.11i standard is briefly described in Section II. Section III presents the analysis of the IEEE 802.1x protocol which is used in the authentication process of IEEE 802.11i. Then the verification of 4-Way Handshake for key management and distribution and the Group Key Handshake in UPPAAL is provided in Section IV. Section V concludes the paper and discusses possible future work.

II. THE IEEE 802.111 PROTOCOL

IEEE802.11i is an IEEE standard designed to provide enhanced MAC security in wireless networks, which enhances the IEEE 802.11 standard in terms of security by providing a Robust Security Network (RSN) [7]. The protocol consists of several parts, including a 802.1X authentication phase using TLS over EAP, the 4-way handshake protocol to establish a fresh session key, and an optional Group Key Handshake protocol for group communications. This series of protocols together define a RSN.

The 4-way handshake protocol and the Group Key Handshake protocol utilize the authentication services and port access control described in IEEE 802.1X [8]. Authentication services involve three parts: the applicant, the certifier, and the authentication server. The applicant is a client device that needs to be connected to a LAN / WAN, and can also refer to software that runs on the client and provides credentials to the certifier. The verifier behaves like a guard of a protected network. Applicants (such as client devices) do not allow authenticated access to the protected side of the network until the identity of the applicant is verified and authorized. And the authentication server is typically a host running RADIUS and EAP-enabled protocols. The EAP data is first encapsulated in an EAPOL frame and transmitted between the supplicant and the authenticator, and then encapsulated in RADIUS or Diameter, transmitted between the verifier and the authentication server.

The initial authentication process is carried out either using a pre-shared key (PSK), or following an EAP exchange through 802.1X. If a 802.1X EAP exchange was carried out, the PMK is derived from the EAP parameters provided by the authentication server. The supplicant and the authenticator use this handshake to confirm the existence of the PMK, verify the selection of the cipher suite, and derive a fresh Pairwise Transient Key(PTK), based on the shared PMK, the nonces and MAC addresses. In case of multicast application, the authenticator will generate a fresh Group Temporary Key(GTK) and may distribute a GTK to supplicants. During the Group Key Handshake, the same PMK may be used repeatedly for multiple times.

III. VERIFYING THE IEEE802.1X PROTOCOL

In this section we first introduce the working principle of the IEEE 802.1x protocol and verify some important properties. The details of the message forwarding in the protocol and its specific implementation process are described in Figure 1.



Figure 1. Message Forwarding in the IEEE 802.1X Protocol

When a user requests to access the network, the supplicant receives the user's request login information and sends the start message "EAPoL-Start" to the authenticator to trigger the authentication process. Once receiving the "EAPoL-Start" packet from the supplicant, the authenticator sends a request to the supplicant asking for the user identity. After receiving the "EAPoL-Request [Identity]" packet from the authenticator, the supplicant sends the user identity to the authenticator. Then, the authenticator passes the user identity through the RADIUS Access-Request "EAPoL-Response [Identity]" to the server. After receiving the packet, the server sends an Access-Challenge packet "EAPoL-Request [MD5 Challenge]" to the authenticator to request a password, and then the authenticator sends the message "EAPoL-Request [MD5 Challenge]" to the supplicant. After the supplicant receives the packet, it sends the password to the authenticator through the message "EAPoL-Response[MD5 Challenge]".



Figure 2. UPPAAL Model for Supplicant in IEEE 802.1x Protocol



Figure 3. UPPAAL Model for Authenticator in IEEE 802.1x Protocol

After receiving the packet, the authenticator sends the message to the server for verification. If the authentication succeeds, the server sends the authenticator success message "EAP-Success" to the device. The authenticator sends the message "EAPoL-Success" to the supplicant to notify the user that the authentication is successful. Otherwise, the server sends an authenticator failure packet, and the authenticator informs the supplicant of the failure of the authentication message so that the user knows that the authentication fails. The user can communicate normally once the authentication successes. However, although the authentication is successful, the legitimate users may be disconnected due to abnormal circumstances, there may be the situation that illegal users replace legitimate users, so the re-certification mechanism was introduced. That is, after successful authentication for the first time, the server authenticates the user to determine whether the user is online and whether it is legitimate over a period of time.



Figure 4. UPPAAL Model for Server in IEEE 802.1x Protocol

The UPPAAL templates of the automata for the supplicant, the authenticator and the server in the IEEE 802.1x model are provided in Figure 2, 3 and 4. They run exactly as the protocol orders them. The main work of the IEEE802.1x protocol is to carry out the exchange of information, for which we have defined a number of channels. For example, the supplicant model moves from state S_0 to state S_1 , and sends the authentication start message "EAPOLStart", synchronizing with the authenticator by output action EAPOLStart!, for which we define the channel EAPOLStart. As a receiver terminal apparatus, the authenticator moves from state A_0 to state A_1 , and synchronize with the supplicant by the corresponding co-action EAPOLStart?. The supplicant, the authenticator and the server interact with each other and their concurrent composition leads to the protocol.

After the supplicant receives the "EAPoL-Request [MD5 Challenge]" packet, it sends the password to the authenticator through the corresponding "EAPoL-Response [MD5] Challenge]" and the authenticator sends the message to the server for verification. Once the authentication is successful, we need to consider re-certification issues. Our model introduces the time variable x to record the elapsed time after the first authentication succeeds and the ReCertificationTIME records re-authentication interval. After the authentication is successful, the port is opened and the recording time starts, correspondingly, x is initialized to 0. The state transition of the authenticator sending the re-certification request is guarded by "x == ReCertificationTIME", so once the property "x will eventually be equal to ReCertificationTIME" is satisfied, it is guaranteed that the re-certification process can occur normally.

Since the IEEE 802.1x protocol is used to authenticate the identity of the user, the first property being required is that once the authentication is successful, the network port will open normally. Correspondingly, once the authentication fails, the network port will not open. In addition, the protocol should also ensure that the port can always be closed or opened, so as to avoid the situation that the port is always closed or

open. Since we also consider the issue of re-authentication, we should also ensure that the port can be opened or closed according to the situation after each re-authentication.



Figure 5. UPPAAL Model for DosAttack in IEEE 802.1x Protocol

Authentication phase of 802.11i suffers against DoS attacks mainly due to lack of authentication of packets [12]. In Figure 5 we define the DosAttack model to simulate a denial of service and define a new channel, FakeEAPoStart. After a denial of service attack, if there is no manual recovery, a deadlock will occur and the authenticator model will stay in the state A_{14} . Once the Repair channel is added to the model, the deadlock behavior will disappear.

A family of properties can be reformulated as CTL formula. For example:

- 1) E<>port==1, on behalf of the port can be opened.
- 2) E<>port==0, on behalf of the port can be closed.
- 3) A[] not deadlock, on behalf of the IEEE 802.1x protocol will not be deadlocked.
- authenticator.A10->port==0, representing that the port will be closed once the re- authentication process begins.
- 5) authenticator.A8->port==0, representing that once the authentication failed the port will not be opened.
- 6) E<>x==ReCertificationTIME, representing that the re-certification process will occur.
- authenticator.A0->port==0, representing that the port was closed initially.

The verification results given by UPPAAL show that all these properties have been proved to be satisfied. Thus, the port can be normally opened or closed, and there is no deadlock in authenticating the identity of the user. After the 802.1X authentication, a shared secret key is generated, called the Pairwise Master Key (PMK). The PSK is derived from a password that is put through the cryptographic hash function. In a pre-shared-key network, the PSK is actually the PMK. If a 802.1X EAP exchange is carried out and no deadlock can be guaranteed, the PMK is derived from the EAP parameters provided by the authentication server.

IV. 4-WAY HANDSHAKE AND GROUP KEY HANDSHAKE PROTOCOLS

In this section, we first consider security properties of the 4-way handshake protocol. During the handshake, the authenticator and the supplicant generate fresh nonces, then derive a fresh PTK based on the shared PMK, the nonces, and their MAC addresses, so that the authenticator and the supplicant can independently prove to each other that they know the PMK, without ever disclosing the key. They authenticate the key material generated using keyed hashes. After this stage, the IEEE 802.1x ports are unblocked for data packets.



Figure 6. The Authentication Process in the 4-Way Handshake Protocol

Figure 6 describes the authentication process in the 4way handshake protocol. In the form of first message the authenticator (AP) sends the random number "ANonce" and MAC address of itself. In response, the supplicant generates another random number "SNonce", and sends it to the AP with the MAC address and message integrity code (MIC) using PTK. Then, a third message is sent by the AP after generating GTK and verifying MIC based on the PTK derived. The supplicant verifies MIC of this message and sends a MIC and install PTK at supplicant. After receiving the MIC message, AP also installs PTK and the 4-way handshake communication is completed. This is the normal way of handshake behavior. Once the supplicant does not receive the first message "ANonce+MAC" within the expected time interval, it will try the authentication again. On the other hand, the authenticator will timeout and retry the message if it does not receive the expected message "SNonce+MAC+MIC" within the configured time intervals.



Figure 7. UPPAAL Model for Authenticator in the 4-Way Handshake Protocol



Figure 8. UPPAAL Model for Supplicant in the 4-Way Handshake Protocol



Figure 9. UPPAAL Model for Hacker in the 4-Way Handshake Protocol

The UPPAAL templates of authenticator and supplicant in the 4-way handshake protocol are described in Figure 7 and 8. The authenticator and the supplicant synchronize through the channel. For example, Msg1! represents authenticator sends the first message "ANonce+MAC" to supplicant, synchronizing with Msg1?, representing supplicant has received the message.

In Uppaal, we build a Hacker model and consider the number of attacks. Hacker multiple attacks will exhaust the memory and complete a denial of service attack. Here we have a FakeMsg1 channel. Every time when the hacker sends FakeMsg1 message, the Supplicant will be attacked. We use Attack to record the number of hacker attacks. Combined with the experience of the industry, we can notice that the agreement will collapse when the number of attacks exceeds a certain number limitation. In our model, once we set the number of attacks more than 100 times, Supplicant will not be able to reject the false information normally, thus it cannot be restored to the initial state.

A family of properties related to the 4-way handshake protocol have been verified in UPPAAL. For example, we have checked whether the handshake behavior will be deadlocked, and can eventually generate PTK. The corresponding properties are reformulated in CTL as follows:

1) A[] not deadlock, representing that the 4-way handshake behavior will not be deadlocked.

 E<>MIC==1, meaning that MIC eventually can be verified so that the last message can be sent successfully and PTK will be installed by the supplicant.

The verification results in UPPAAL show that the above properties are satisfied.

We have not consider the attacker in this model. In the transition from state S_0 to state S_1 , ANonce is updated to 1, followed by the state transition as a guard so that once ANonce has not been updated normally, the model blocks in state S_1 . Attacker is in accordance with this principle on the handshake attack. This Dos attack arises from the vulnerability of the message Msg1. Actually, the 4-way handshake protocol is vulnerable to Dos attack during handshake. So we use the guard $y \leq ExpectedTIME1&ANonce=1$ to make sure that we get the correct ANonce.

The authenticator may distribute a Group Temporary Key (GTK) to supplicants in multicast applications. PTKs are used to encrypt unicast data between a supplication station and an authenticator, and GTKs are used to encrypt multicast data between a supplicant station and an authenticator.

Group-key handshake contains 2-way handshake. The authenticator sends message GrpMsg1 containing the new GTK to each supplicant in the network. The GTK is encrypted and assigned to the supplicant, and protects the data from tampering, by using a MIC. Then, the supplicant confirms receipt of the new GTK and sends reply to the authenticator. MICs are used to provide authentication and message integrity. This 2-way handshake lends simplicity and much less overhead to multicast key generation and distribution to supplicants.



Figure 10. UPPAAL Model for Group Key Handshake Protocol

Figure 10 shows the models in the group-key handshake protocol. Since group-key handshake is 2-way handshake, we only need to define two states respectively. The authenticator sends GrpMsg1 to the supplicant. Once the supplicant synchronizes with the authenticator successfully (represented by GrpMsg1? in Figure 10), the GTK used to protect the data from tampering is updated (represented by the assignment GTK:= 1 in Figure 10). Once the GTK has been updated to 1,

it means that the GTK has been successfully encrypted. Thus, what we should ensure in our model now is that the supplicant sends reply to the authenticator after confirming receipt of the new GTK. The edge transforming from state A_1 to state A_0 is guarded by the condition GTK==1 so that only when GTK is successfully received, the supplicant can send a reply.

Through the verification of our model, we can prove that the property A[] not deadlock is satisfied, which means that the model is not deadlock. It is further illustrated that once the GTK has been successfully encrypted and sent to the supplicant, group-key handshake will be able to keep proceeding.

V. CONCLUSION AND FUTURE WORK

This paper analyzes the IEEE802.11i protocol for mutual authentication, group communications and key establishment. And we have considered the case of re-certification in the protocol, which means we need adding timing issues to the extended protocol. Using UPPAAL allows us to simulate, debug and verify the IEEE802.11i protocol in a real time setting. UPPAAL can be used to generate the protocol's simulation path and the study of time allows us to ensure that the re-authentication process will occur. In addition, we introduced a validation analysis of deadlock. Ensuring that the deadlock does not occur in the protocol is the primary requirement for regulatory security.

Designing secure and efficient key management protocol in 802.11i standard is a significant issue. In this paper, we model the 4-way handshake protocol with four synchronization channels, simulating the interaction of the protocol and consider the timeout issues. Based on the formal model, we perform an integrated formal verification of the protocol using UPPAAL. The verification results show that the 4-way handshake protocol will not be deadlock if the Dos attack can be prevented. Thus, in our further study we will consider an enhanced 4-way handshake to repair vulnerability to attack.

Basic analysis for the 4-way handshake protocol in this paper is based on the idealized handshake protocol shown in Figure 6. In our model we have not considered subsequent verification, such as verifying Message Integrity Code based on the PTK. The PTK is generated by concatenating the attributes PMK, ANonce, STA SNonce, MAC address of authenticator and supplicant. In fact, the attacker precisely makes the handshake deadlock by providing the wrong ANonce to derive a wrong PTK. In the future, we will also consider more situations, such as not successfully verifying MIC, in our model.

Furthermore, we have just considered whether the protocol is safe and proceeds normally and simplified the key passing process in this work. Assuming that once the key has been generated, the sub-protocol can run normally. This means that we have not considered emergencies that can cause transformation of the key failed. We will add the impact of the environment into the formal model in the future work as well.

ACKNOWLEDGEMENT

The work was partially supported by the National Natural Science Foundation of China under grant no. 61772038, 61532019, 61202069 and 61272160.

REFERENCES

- A. Alabdulatif, X. Ma, and L. Nolle. Analysing and attacking the 4-way handshake of ieee 802.11i standard. In *Proceedings of 8th International Conference for Internet Technology and Secured Transactions*, pages 382–387. IEEE, 2013.
- [2] R. Alur and D. L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.
- [3] C. Baier and J.-P. Katoen. Principles of Model Checking. The MIT Press, 2008.
- [4] P. R. D'Argenio, J. Katoen, T. C. Ruys, and J. Tretmans. The bounded retransmission protocol must be on time! In *Proceedings of TACAS '97*, volume 1217 of *LNCS*, pages 416–431. Springer, 1997.
- [5] S. Dustdar, S. Nastic, and O. Scekic. Smart Cities The Internet of Things, People and Systems. Springer, 2017.
- [6] K. Havelund, A. Skou, K. G. Larsen, and K. Lund. Formal modelling and analysis of an audio / video protocol: An industrial case study using uppaal. In *Proceedings of the 18th IEEE Real-Time Systems Symposium*, pages 2–13. IEEE Computer Society, 1997.
- [7] IEEE Std 802-2014. Ieee standard for local and metropolitan area networks: Overview and architecture, 2014.
- [8] IEEE Std 802.1X-2010. Ieee standard for local and metropolitan area networks-port-based network access control, 2010.
- [9] K. G. Larsen, P. Pettersson, and W. Yi. Diagnostic model-checking for real-time systems. In *Proceedings of the 4th DIMACS Workshop* on Verification and Control of Hybrid Systems, volume 1066 of LNCS, pages 575–586. Springer, 1996.
- [10] J. Liu, X. Ye, J. Zhang, and J. Li. Security verification of 802.11i 4way handshake protocol. In *Proceedings of ICC 2008*, pages 1642–1647. IEEE, 2008.
- [11] X. Liu and A. O. Fapojuwo. Formal evaluation of major authentication methods for IEEE 802.11i WLAN standard. In *Proceedings of VTC Fall* 2006, pages 1–5. IEEE, 2006.
- [12] S. pyo Hong and J. Lee. Supporting secure authentication and privacy in wireless computing. In 2006 International Conference on Hybrid Information Technology(ICHIT), pages 594–599. IEEE Computer Society, 2006.
- [13] K. V. K. Raju, V. Vallikumari, and K. Raju. Modeling and analysis of ieee802.11i wpa-psk authentication protocol. In *Proceedings of 3rd International Conference on Electronics Computer Technology*, pages 72–76. IEEE, 2011.
- [14] R. Singh and T. P. Sharma. A key refreshing technique to reduce 4way handshake latency in 802.11i based networks. In *Proceedings of 4th International Conference on Computer and Communication Technology*, pages 157–162. IEEE, 2013.
- [15] E. Sithirasenan, S. Zafar, and V. Muthukkumarasamy. Formal verification of the IEEE 802.11i WLAN security protocol. In *Proceedings of* (ASWEC 2006), pages 181–190. IEEE Computer Society, 2006.
- [16] UPPAAL. http://www.uppaal.org/.
- [17] R. H. Weber and R. Weber. *Internet of Things Legal Perspectives*. Springer, 2010.
- [18] X. Xing, E. M. Shakshuki, D. G. Benoit, and T. R. Sheltami. Security analysis and authentication improvement for IEEE 802.11i specification. In *Proceedings of GLOBECOM 2008*, pages 1887–1891. IEEE, 2008.