A Security Protocol Model of Internet of Things for Resisting Known Plaintext Attack

Yong Shen\textsuperscript{1,2,*}, Yan Li\textsuperscript{1}, Hongwei Kang\textsuperscript{1,2}, Xingping Sun\textsuperscript{1}, Qingyi Chen\textsuperscript{1,2} and Chuanyu Zang\textsuperscript{1}

\textsuperscript{1}School of Software, Yunnan University, Kunming, Yunnan, 650091, China
\textsuperscript{2}Key Laboratory for Software Engineering of Yunnan Province Republic of China, Yunnan University, Kunming, Yunnan, 650091, China

*Corresponding author e-mail: sheny@ynu.edu.cn

Abstract. Based on the current research status of security problems in the Internet of things and the communication characteristics of the internet of things, an Internet of things security protocol model that can resist known plaintext attacks was proposed. It realizes identity authentication, key distribution and data encryption in the process of information interaction. By using Casper and FDR tool to model and formalize the security protocol proposed, the security of this protocol was proved. The performance of the protocol was analyzed by comparing it with other IoT security protocols and calculating the expected network communication overhead.

1. Introduction

Nowadays, with the rapid development of the Internet of things technology, people are gradually deepening their research on the Internet of Things\cite{1}. With the passage of time, the Internet of Things technology will get more and more practical applications\cite{2}. Along with the development and research of hardware and software related to the Internet of Things, many problems have been gradually highlighted\cite{3}. It is particularly important to ensure the information security between the communication subjects of the Internet of Things. However, realizing security communication of the Internet of Things through the security protocol has become a hot topic\cite{4}.

Based on the analysis of the traditional IoT security protocol, an IoT communication security protocol model is proposed to resist known plaintext attacks due to factors such as weak node computing ability, limited storage space, and weak attack resistance. It transfers the resource consumption of key distribution in the network to the key distribution center as far as possible, and by introducing "super-large random number", "data authentication" to ensure that the communication process is sufficient to resist external attacks.

2. The Formal Analysis Method

2.1. Introduction to CSP and FDR

CSP(Communicating Sequential Processes), presented by the famous computer scientist C.A.R.Hoare\cite{5}, is an algebraic theory for solving concurrent Processes. It is an abstract language that can describe the interactive process for different processes in a system to deliver messages.
FDR (false discovery rate) is a common tool for model detection to detect whether various process-based communication systems meet the security specification standards[6]. In this paper, the FDR tool is used to detect the CSP description corresponding to the security protocol.

2.2. Introduction to the Casper Tool

Casper simplifies the process of generating CSP descriptions[7]. It is a scripting program developed by Gavin Lowe to make it easier for model detection tool FDR to analyze security protocols. The tool proposes a set of modeling syntax rules for protocols. When using Casper, we first need to model the security protocol according to the syntax standard provided by Casper, then compile the modeled file through the Casper tool, and finally generate the CSP description that can be used for FDR tool detection.

3. Protocol Design, Formal Validation and Improvement

3.1. Protocol Body Design

- 1. person1 to S: { person1, person2, na1, na2, {ta} {passwd(person1)} } {pks}
- 2. S to person2: person1, person2
- 3. person2 to S: { person2, person1, nb1, nb2, {tb} {passwd(person2)} } {pks}
- 4. S to person1: {na1, key (+) na2} {passwd(person1)}
- 5. S to person2: {nb1, key (+) nb2} {passwd(person2)}
- 6. person2 to person1: {rb} {key}
- 7. person1 to person2: {frb, ra} {key}
- 8. person2 to person1: {fr(a)ra} {key}

The first 5 steps of the protocol describe the process of session key negotiation, which involves: ① Timestamp encryption, which adopts symmetric encryption algorithm, is mainly aimed at completing the identity authentication of the communication entity by key distribution center S. ② Encrypting the session message initiated by the communication entity and key distribution center S, using the public key encryption algorithm ③ Encrypting the transmission process of the session key. Symmetric encryption technology is adopted here to enhance the speed of message encryption and reduce the computing burden of communication subjects ④ Transfer the result of XOR of session key and a new random number. This increases the anti-attack capability of the message.

The last three steps of the protocol describe the process of information interaction between communication entities, which involves: ① The message is encrypted using the session key. Symmetric encryption algorithm is adopted to improve the speed of message encryption. And the session key can be updated periodically according to actual requirements, that is, the first 5 steps of the protocol are executed. ② The hash function is introduced to complete the authentication of communication entities to each other and the integrity verification of messages.

The negotiation process of session key in the security protocol model of the Internet of Things does not need to be carried out every time of communication. As long as the session key is generated before, it can be used again. Of course, you need to ensure the freshness of the session key, so you can set the update period of the session key according to the actual needs. If this period is exceeded, the first 5 steps of the protocol need to be executed when communicating.

3.2. Protocol Formal Validation

The formal verification technology of security protocol used in this paper mainly involves Casper tool and FDR model detection tool, both of which are running in Ubuntu system.

Detection results from the FDR tool can be found. The description of the attack path is:

The intruder verifies the guess of passwd (Alice) using verifier{Alice, Mallory, na1, na2, {0} {passwd(Alice)}} {PKs}.

The information described above means that an intruder can guess or decipher the password by analyzing the first or third message in the protocol, that is, messages in the format: { person1, person2, na1, na2, {ta} {passwd(person1)} } {PKs}.
According to the analysis results given by FDR, the intrusion method of intruder belongs to password analysis. Obviously, the intruder chose a method in password analysis called known plaintext attack. In addition to mastering the ciphertext, the attackers also mastered the correspondence between some plaintext and ciphertext. For example, if the conversation is in compliance with the IoT security protocol, since the protocol USES a fixed communication format, such as: the message contains two identity identifiers and two random Numbers, and the timestamp format encrypted with passwd is the same, and with fixed keywords, such as passwd, the corresponding ciphertext of this keyword can be determined through analysis. If the amount of data transmitted is large enough, since most messages have a fixed format and some agreed text, under the condition of more intercepted messages, combined with statistical analysis, some sensitive content such as word phrases can be inferred.

Therefore, we can conclude that this protocol cannot resist the "known plaintext attack" in password analysis, and we need to improve the security of the protocol.

3.3. Protocol Improvement

3.3.1. Improved protocol body

We add the confounder in step 1 and step 3 of the protocol interaction process. Then, the modified protocol body information is shown as follows (ca and cb represent the confounder):

1. person1 to S : {person1, person2, na1, na2, ca, {ta}{passwd(person1)}} {pks}
2. S to person2 : person1, person2
3. person2 to S : {person2, person1, nb1, nb2, cb, {tb}{passwd(person2)}} {pks}
4. S to person1 : {na1, key (+) na2} {passwd(person1)}
5. S to person2 : {nb1, key (+) nb2} {passwd(person2)}
6. person2 to person1 : {rb} {key}
7. person1 to person2 : {f(rb), ra} {key}
8. person2 to person1 : {f(ra)} {key}

In the first step of the protocol, we add the confounder ca into the message sent by person1 to S, which is packaged together with the identity identification of both person1 and person2, the new random number na1 and na2, and the encrypted timestamp, and then use the public key of S to encrypt the transmission. In step 3 of the protocol, the modification is the same.

3.3.2. Formal verification of the improved protocol

The formal analysis process of the improved Internet of Things security protocol model is as follows:

1) initialize the Casper tool and FDR model detection tool;
2) Load the protocol-revision.spl file that completed the modeling into the Casper tool;
3) Casper compiles the protocal-revision.spl file and generates the protocal-revision.csp file automatically;
4) start the FDR model monitoring tools;
5) FDR detects previously generated protocal-revision.csp files;
6) after several rounds of model checking, found no vulnerability.

Therefore, we can draw a conclusion that the improved protocol model of the Internet of Things is safe.

4. Protocol Performance Analysis

4.1. Key Storage

The management control platform is responsible for the distribution and update of public and private keys in each communication area key distribution center. It needs to store M public and private key pairs, with a storage complexity of o (M), and is responsible for the transregional information transmission of different communication entities. The key distribution center needs to store the identity information and password key information of each communication entity. The storage
complexity is $o(2N)$. The communication entity needs to store its own password key information and the public key information of the key distribution center, and the storage complexity is $o(2)$.

4.2. Complexity Comparison

In order to further analyze the characteristics of communication entities in this physical network communication model, this scheme compares the computational complexity, communication complexity and storage complexity with the previously proposed IoT key distribution and identity authentication scheme. The results are shown in Table 1:

| Key Management Model | Computation Complexity | Communication Complexity | Storage Complexity |
|----------------------|------------------------|--------------------------|--------------------|
| Q-Composite          | $o(m)$                 | $o(2)$                   | $o(m)$             |
| E-G                  | $o(k)$                 | $o(2)$                   | $o(k)$             |
| PIKE                 | $o(2)$                 | $o(\sqrt{n})$            | $o(\sqrt{n})$      |
| LEAP                 | $o(d^2/N)$             | $o(\log N)$              | $o(d+L)$           |
| This protocol        | $o(2)$                 | $o(2 \text{ or } 3)$     | $o(2)$             |

The E-G scheme consists of three stages: key pre-allocation, shared key discovery and key path establishment. The user node needs to store its own key ring composed of $K$ keys for sharing key discovery. Therefore, both computational complexity and storage complexity are $0(k)$. The q-composite[8] scheme is a random key authentication scheme. In this scheme, $m$ different keys are randomly selected from the key pool. After deployment, the two adjacent nodes need to share at least $q$ keys to establish the matching key directly. In this scheme, the nodes select $m$ keys from the random key to form their own key rings. Similar to the E-G scheme, the computational complexity and storage complexity are $o(m)$. Through similar analysis methods, the computational complexity of PIKE[9], protocol LEAP protocol can be obtained, etc., as shown in the table above. In this scheme, the session key can be obtained by two secret computations per communication entity, so its computational complexity is 2.

4.3. Communication Overhead

In order to compare the communication costs of communication entities in the same or different network regions, we assume that the probability of communication entity A and communication entity B in the same network region is $P$, the cost of each message transmission is $\log_2 t$.

When the session key negotiation is complete, the communication entities can communicate with each other. In the security protocol model of Internet of Things, three sessions are required for each communication. As can be seen from the structure diagram, the communication process of nodes in the same network area needs 3 times of message transmission. For nodes in different network areas, the communication process requires 12 message transfers. Therefore, when the nodes communicate with each other, the expected cost is:

$$E(\log_2 t) = 3P\log_2 t + 12(1-P)\log_2 t$$

(1)

Where, we set $t$ as 1-10, unit of kb, and simulate the communication cost of communication entities by adjusting the size of probability $P$. The results are shown in figure 1. From the figure 1, we can find that with the increase of probability $P$, the expected result of communication overhead between communication entities is also increasing. Therefore, the communication overhead between communication nodes will change with the change of communication area. When two communication entities are in different network areas, the communication cost becomes significantly larger. This protocol is relatively suitable for applications in relatively few network areas.
5. Conclusion

"large random interference number" is introduced in the process of key distribution to ensure that the message cannot be deciphered. "Hash function" and "timestamp" are introduced in the process of message integrity verification and identity authentication to improve the communication efficiency. After modeling the security protocol according to the Casper syntax specification, the security of the protocol is verified by using the model monitoring tool FDR. The security of the protocol was verified by the model monitoring tool FDR, and the final conclusion was drawn that the improved protocol model of the Internet of Things was safe. Compared with other scheme the complexity of this security protocol model is better.

Acknowledgments

This work is funded by the Open Foundation of Key Laboratory of Software Engineering of Yunnan Province under Grant No. 2015SE204 and Open Foundation of Key Laboratory of Software Engineering of Yunnan Province under Grant No. 2012SE306.

References

[1] Tervonen J, Isoherranen V, Heikkilä M. A review of the cognitive capabilities and data analysis issues of the future industrial Internet-of-Things[C]. IEEE International Conference on Cognitive Infocommunications: IEEE, 2016.127-132.

[2] Welbourne E, Battle L, Cole G, et al. Building the Internet of Things Using RFID: The RFID Ecosystem Experience[J]. IEEE Internet Computing, 2016, 13(3):48-55.

[3] Zhang Y. Technology framework of the Internet of Things and its application[J]. IEEE, 2011:4109-4112.

[4] Bertino E, Islam N. Botnets and Internet of Things Security[J]. Computer, 2017, 50(2):76-79.

[5] Goldsmith M. FDR2 user manual[J]. Formal Systems Ltd, 2000.

[6] Roscoe A W. Model-checking CSP[J]. A Classical Mind Essays in Honour, 1994:353-378.

[7] Lowe G. A compiler for the analysis of security protocols[J]. Journal of Computer Security, 2009, 6(1-2):18-30.

[8] Kumar M, Tuli A, Tuli R, et al. Secure Communication Using Needham-Schroeder Protocol[J]. Chaos, 2012, 24(3):775-783.

[9] Chan H, Perrig A. PIKE: peer intermediaries for key establishment in sensor networks[A]. INFOCOM 2005. Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE[C]. IEEE, 2005:524-535 vol. 1.