A Novel XTEA based Authentication Protocol for RFID Systems

Gul N. Khan, X. Yu, and Fei Yuan

Department of Electrical and Computer Engineering, Ryerson University,
Toronto, Ontario M5B 2K3 Canada
gnkhan@ee.ryerson.ca, x7yu@ee.ryerson.ca, fyuan@ee.ryerson.ca

Abstract

RFID technology has been widely used in logistic, automation and authentication applications. It still has many potential issues of privacy and security. We present a novel XTEA encryption based authentication protocol. Analysis of its security and privacy is performed using FPGA based prototyping. Different attack models are implemented, and the results show that the protocol is robust and safe against major attacks. The protocol is analyzed and compared with other two similar protocols, Analysis of its performance as compared to related works show its advantages in code size, clock cycle, communication cost and scalability.

1. Introduction

RFID (Radio Frequency Identification) technology has a wide range of applications due to which finding suitable security and authentication protocol is challenging. Any practical design of an authentication protocol has to strike a balance between security, cost and performance. In this paper, we investigate and propose a novel authentication protocol based on eXtended Tiny Encryption Algorithm (XTEA) [1]. The security protocol is prototyped and analyzed using FPGA platform. We present the experimental results for different types of major attacks for our scheme to evaluate its security level. We also evaluate the performance of our protocol by comparing with similar works that also use symmetrical key encryption.

An authentication protocol consists of a cipher used to encrypt the message and the processes of how the messages are exchanged. Since a back-end server has more computational resource than a tag, the efficient implementation of an RFID protocol relies on lightweight ciphers that have a small size, consume less power and provide satisfactory level of security. Feldhofer et al. introduced an efficient implementation of AES (Advanced Encryption Standard) technique in CMOS [2]. Tiny Encryption Algorithm (TEA) has become of interest for RFID cipher solutions [3]. However, it suffers from key attacks and its actual key size is reduced. In order to overcome these weaknesses, proposed XTEA cipher was introduced [1]. Jens-Peter Kaps has investigated low power implementation of XTEA indicating that it needs low power and resources than AES [4]. However, very few RFID authentication protocols are based on XTEA cipher.

2. Proposed Authentication Protocol

To solve RFID security and privacy issues, we propose a novel secure authentication protocol based on XTEA encryption. It is assumed that the RF-tag has a 64-bit ID and a 128-bit key set that can be dynamically updated. The tag has the encryption and decryption capability using XTEA cipher. One key set consists of four sub-keys of 32 bits each. Initially each tag and back-end server shares a random secret key set. Our RFID authentication protocol has various steps that are depicted in Figure 1 and its message exchange process is described as following:

(1) The reader sends a query message to read the tag.

(2) Once being queried, the tag transmits its ID encrypted by XTEA cipher with the current key set k0.

(3) The reader forwards this message to the back-end server for authentication.

(4) The back-end server checks the database of expected response of known tags to find the corresponding tag ID. Then it generates a key update message. The key update message format is shown in Figure 2(a). It consists of 30-bit tag ID field, a 2-bit field for the sub-key number and 32-bit sub-key value field. The message must be encrypted using the current key set k0. The back-end server computes the tag response for the next query based on the new key set k1. Meanwhile, it backups the current response in case the key update failed for some reasons. The back-end server also generates a key update acknowledgement in a format shown in Figure 2(b), which is 64 bit wide and encrypted with next key set k1. It consists of the tag ID and the previous sub-key that is to be replaced. The server sends the tag ID, key update message and acknowledgement to the reader.

(5) The reader forwards this key update message to the tag, which decrypts it using the current key set k0 and verifies the tag ID. If verification succeeds, it continues to extract the sub-key value and number to update the key set to k1. If this process fails, it continues to wait for another key update message within a timeout.

(6) After receiving the key update and verification passes, the tag generates the key update acknowledgement, encrypts it with the previous key set and sends it back to the reader in the format of Figure 2(b).

(7) The reader verifies the correct key update acknowledgement using the current key set. Successfully receiving the acknowledgement message terminates the authentication process.

(8) If the reader fails to receive the acknowledgement, it resends the key update message. The tag will respond with acknowledgement again even if it has already sent out the acknowledgement. The reader can re-send this key update message several times before warning the back-end server that the tag is faked. If the key-update process fails because the tag is a faked one, the tag record in the database can be reverted back.
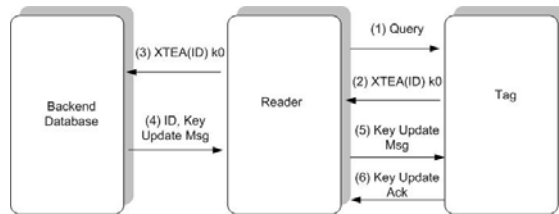

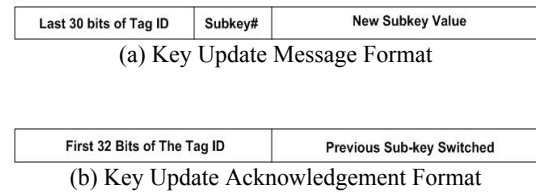Figure 1. Novel Protocol Based on XTEA Encryption

Last 30 bits of Tag ID	Subkey#	New Subkey Value

(a) Key Update Message Format

First 32 Bits of The Tag ID	Previous Sub-key Switched

(b) Key Update Acknowledgement Format

Figure 2. Message and Acknowledgement Formats

3. Protocol Analysis for Attacks

In this section, we demonstrate the implementation of our protocol for various attack models using Altera FPGA board [5]. The hardware system is generated using SOPC builder and configured the FPGA device using Quartus-II tool. In the replay attack, reader and the tag communicate via an insecure channel and attacker can intercept the message exchange between them. Figure 3 shows the Nios-II CPU system of this model. First, the reader queries the tag and the attacker records the tag response during the authentication process. The reader then queries the attacker to see if the attacker can be authenticated by replaying the message it records. The initial tag ID is {0x22332212, 0x21314783} and the key set is {0x247DC618, 0x70BC93E4, 0x902C729A, 0xAB47856C}.The sample of result in the first 3 sessions is summarized in Table 1 indicating that the tag not only replies with the encrypted message in every session but also update one sub-key of its key set. The replay attacker tries to replay the previous tag response but cannot be authenticated, verifying that our protocol is safe against replay attack.

3.1 Active Scanning and Eavesdropping Attack

For active scanning attack, the tag is isolated from the legitimate readers and the attacker scans the tag without authorization. The purpose of this attack is to find if it can keep sending useful messages. The message exchanges that happen in the first 3 sessions are shown in Table 2. As one can observe from tag response, the reply message is encrypted and it is not meaningful even if it is exposed to the attacker. In the case of eavesdropping, the attacker can intercept the message coming from the tag and reader. In our implementation, the reader first queries the tag in a normal operation. Then, the reader checks whether the message recorded by the attacker matches the secrets it has or not. As observed from the result shown in Table 3, the tag replied with the encrypted message in each new session. The eavesdropping attacker cannot obtain the secret of the tag because it does not know the key set.

3.2 Denial-of- Service and Man-in-the-Middle Attack Models

In denial-of-service attack, the communication channel between the reader and tag is blocked by the attacker. The attacker blocks the key update message and then allows it when the reader resends it. The results for the first three sessions are shown in Table 4 indicating a successful key update operation of the tag that verifies our protocol as safe. In the case of a man-in-the-middle attack, the communication is intercepted and modified by the attacker. The reader first queries the tag and receives the tag reply. When the key update message arrives, the attacker tries to randomly modify it and sends it to the tag. The results for sessions 28 to 30 are shown in Table 5. The tag decrypts the key update message, but the field with the last 30 bits of its tag ID does not match. In the timeout period, the reader does not receive the correct key update acknowledgement message. Therefore it resends the key update message. The key update message acknowledgement from the tag indicates the reader and the tag are synchronized.
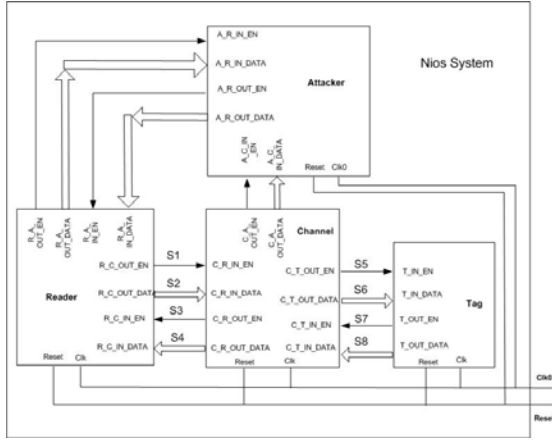
Figure 3. Replay Attack Model

Table 1. Replay Attack and Response

Session	Tag Response Message	Tag Key Set	Attacker Replay
1	B5718393, 0FE7A64B	AB47856C, 902C729A, 70BC93E4, 247DC618	B5718393, 0FE7A64B
2	AF4D7FD9, DDF60A22	AB47856C, 902C729A, 70BC93E4, DA6B4E51	AF4D7FD9, DDF60A22
3	97E6D6FC, 8072E812	7825DFCB, 902C729A, 70BC93E4, DA6B4E51	97E6D6FC, 8072E812

Table 2. Active Scanning Attack & Response

Session	Message Sent by Attacker	Tag Response Message
1	Tag Query	B5718393,0FE7A64B
2	Tag Query	B5718393,0FE7A64B
3	Tag Query	B5718393,0FE7A64B

Table 3. Eavesdropping Attack & Response

Session	Tag Response Message To Query	Tag ID
1	B57183930FE7A64B	2233221221314783
2	AF4D7FD9DDF60A22	2233221221314783
3	97E6D6FC8072E812	2233221221314783

Table 4. Denial-of-Service Attack & Response

Session	Tag Response Message	Tag Key Update Message	Key Update Reader Resend	Key Update ACK
1	B5718393, 0FE7A64B	8EA3D37C, 3F5198CA	8EA3D37C, 3F5198CA	2B6EDC58, 1B49A4D8
2	AF4D7FD9, DDF60A22	BE0B6D17, 27D8F698	BE0B6D17, 27D8F698	A7C5352E, B059EBD
3	97E6D6FC, 8072E812	70F67232, FAA7DB1A	70F67232, FAA7DB1A	110D71D7, 99059299

Table 5. Man-in-the-Middle Attack & Response

Session	Tag Response To Query	Tag Key Update Message	Key Update Reader Resend	Key Update ACK
28	BE0F3DF6, DD11E327	7C6F3423, DA1EE993	7C6F3423, DA1EE993	51B7093F, 4E4B5E76
29	2962986F, 2B17C3FA	F6D59CA9, F92AF59B	F6D59CA9, F92AF59B	685BF097, DBECF678
30	D7F46FAE, F0FE0ADF	63C4CFEF, C12C399C	63C4CFEF, C12C399C	A8A9C52, 4F35367B

4. Performance Comparison

Various researchers are working on RFID authentication protocols. It will be meaningful to compare different protocols that use similar ciphers. Therefore, our comparison focuses on those protocols using symmetric key encryption techniques, due to the same security level of the cipher. In this section, we analyze and compare our protocol with the protocols by Toiruul and Lee [6] and Kim et al [7].

4.1 Memory Storage/Code Size and Communication Cost

We have implemented the protocol by Toiruul and Lee [6], Kim et al protocol [7] and our authentication protocol using the same Nios-II multiprocessor platform consisting of only reader and tag modules. These modules are implemented as 32-bit RISC Nios II processors and the communication between them are achieved by Parallel I/O type ports. It is determined that the reader code is 3524 bytes and tag code is 1944 bytes. The same method is used to evaluate the code size of other two protocols. For Toiruul and Lee protocol the tag code is 5700 bytes, while the reader code is 7836 bytes. For Kim et al's [7], the tag code is 6440 bytes and reader code size is 9544 bytes.

In order to evaluate the communication cost of each protocol, consider the situation where a reader needs to authenticate *n* number of tags. In our protocol, when a reader is to query the tags, it needs to send out only one query message for all the tags to respond. The communication cost of step (1) is only one. If *n* tags are responding, their response consists of 64**n** bits and the communication cost in the second step is 64**n** bits. Similarly, the communication cost for step (3) and (4) are 64**n** bits. In this way, the total communication cost for our protocol is (64+ 3 x **n**64) bits. For the Toiruul and Lee protocol, when a reader queries **n** tags; it has to initiate *n* messages in

step (1). Due to usage of AES encryption that is 128 bits in length, the communication cost of first step is 128**n** bits. In the second step, **n** tags send back 128**n** bits to the reader and the total communication cost will be (2 x **n**128) bits. Kim et al.'s protocol consists of four steps for over-the-air communication channel. When a reader queries **n** tags, each step has 128**n** bits in communication cost due to the use of AES encryption. In this way, the total communication cost for Kim et al protocol is (4 x **n**128) bits. It indicates that our scheme is more efficient than the other two schemes in terms of communication cost specifically for multiple tags.

4.2 Execution Time and Clock Cycles

The clock cycles needed to execute a protocol is an important parameter because it is proportional to power consumption. Fewer clock cycles also means higher throughput and less likelihood of being attacked. The clock cycle is measured by implementing a timestamp timer within the Nios-II system for the reader module. We start the timer and obtain the first timestamp at the beginning of the execution. After the execution is completed in each session, we obtain the second timestamp. Then, by calculating the timestamp difference, the execution time of each session is calculated. To ensure better accuracy, timestamp difference is measured 20 times and the average value is obtained. We determined that our protocol takes 1.060 ms to execute, which is equivalent to about 53000 cycles when the communication time is zero. We apply the same method to analyze the clock cycle of Kim et al [7] as well as Toiruul and Lee [6] protocols. The timestamp difference for Toiruul and Lee and Kim et al protocols converges to 11.382ms and 12.507ms respectively. It indicates that our protocol takes far less clock cycles to execute.

4.3. Scalability

We also compared our protocol with Toiruul and Lee as well as Kim et al.'s protocols in terms of scalability. In our protocol, when a reader initiates a reading operation, it just needs to issue a query message. The backend server need to compare the tag reply with the list of expected replies from tags. The search time does not increase significantly with more tags added into the system. In Toiruul and Lee protocol [6], the reader relies on other anti-collision protocol to singularize a tag out of many. Therefore, it is considered to be scalable too. For Kim et al.'s protocol [7], in order to initiate a reading operation the reader needs to know the tag M secret in advance. If a reader does not know the tag's M, it has to brute force search for known M one by one to read a tag. As the population of tag grows larger, it becomes harder to identify a tag. Obviously, this scheme is not scalable as compare to ours.

5. Conclusions

A novel authentication protocol based on XTEA encryption is proposed as a robust solution for RFID security. In the protocol, one sub-key of the tag is dynamically updated and which sub-key is to be updated is random in nature. The reader and the tag are capable of authenticating each other mutually. The protocol and different attack models are prototyped. It is proved that the novel protocol is safe against major attacks. Performance is also evaluated and compared with similar protocols and the result shows that our scheme is lightweight and efficient.

6. References

[1] R. M. Needham and D. J. Wheeler. "TEA Extensions" Technical report, Computer Lab, Cambridge Univ, October 1997

[2] M. Feldhofer and J. Wolkerstorfer "Strong Crypto for RFID Tags – A Comparison of Low-Power Hardware Implementations" Institute for Applied Information Processing and Communications, Graz University of Technology Inffeldgasse 16a, 8010 Graz, 2007.

[3] R. M. Needham and D. J. Wheeler, "TEA, a Tiny Encryption Algorithm" Lecture Notes in Computer Science (Leuven, Belgium: Fast Software Encryption: 2nd Int. Workshop), Dec 16, 1994.

[4] Jens-Peter Kaps "Chai-Tea, Cryptographic Hardware Implementations of XTEA" Volgenau School of IT&E, George Mason University, Fairfax, VA, USA.

[5] Altera Development and Education Board URL: http://www.altera.com/education/univ/materials/boards/unv-de2-board.html

[6] B. Toiruul and K. O. Lee "An Advanced Mutual-Authentication Algorithm Using AES for RFID Systems" International Journal of Computer Science and Network Security, Vol. 6, No. 9B, September 2006.

[7] K. Kim, K. Chung, J. Shin, H. Kang, S. Oh, C. Han and K. Ahn "A Lightweight RFID Authentication Protocol using Step by Step Symmetric Key Change" 8th IEEE International Conf. Dependable, Autonomic and Secure Computing, 2009.