香港城市大學
City University
of Hong Kong

Department of Electronic Engineering

FINAL YEAR PROJECT REPORT

LMC-10-BEIE-2007/08

RFID Security

Student Name: So Chun Wa

Student ID:

Supervisor: Dr. Cheng, L M

Assessor: Dr . Tsang, Peter W M

Bachelor of Engineering (Honours) in

Information Engineering

Student Final Year Project Declaration

I have read the student handbook and I understand the meaning of academic dishonesty, in particular plagiarism and collusion. I declare that the work submitted for the final year project does not involve academic dishonesty. I give permission for my final year project work to be electronically scanned and if found to involve academic dishonesty, I am aware of the consequences as stated in the Student Handbook.

Project Title: RFID Security

Student Name: So Chun Wa

Student ID:

Signature

Date: 29-11-2007

ABSTRACT

Low cost Radio Frequency Identification (RFID) aims at replacing barcode counterparts and it is expected to be applied massively in our daily lives application soon. However the serious security problems may obstruct the deployment of RFID. In this project, I based on the scheme proposed in the paper titled "Enhancing security of EPCglobal Gen-2 RFID tag against Traceability and Cloning" published by Duc et al. to evaluate the possible attacks and vulnerabilities of the scheme by simulations. The project proposed a new enhanced protocol with improved Forward Secrecy, as well as prevented key de-synchronization and man-in-the-middle attacks.

ACKNOWLEDGEMENT

I acknowledge with gratitude to my supervisor Dr. Cheng, L M (Associate Professor, Department of Electronic Engineering, City University of Hong Kong) for his continuous guidance and encouragement throughout the whole study period.

I would like to extend my gratitude to my assessor Dr. Tsang, Peter W M (Associate Professor, Department of Electronic Engineering, City University of Hong Kong) for his genuine support and encouragement.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1 INTRODUCTION

Radio Frequency Identification (RFID) technology is a radio frequency system that has been applied to identify object and is able to gather data automatically as well as massively in different application. Since the use of RFID in Second World War until today's electronic payment system, it has been successfully used in various aspects. In the near future, it is planned to deploy massively in the product pallet level, which aims at replacing barcode counterpart. The convenient features of RFID technology will make it become the most pervasive microchips in history.

The typical RFID system consists of radio frequency (RF) tags and RF tag reader. Reader is usually connected with backend database server to store and retrieve the tag's information.

1.1 Advantages of RFID System

Nowadays, RFID system has been applied in many aspects, for instance, in access control, electronic payment system, logistic and supply chain management. The small-sized tag and contactless reader can communicate with more than one tag at the same time, it increases the efficiency significantly. It can also collaborate with many new applications, for example in the artificial intelligence electric appliances.

1.2 Hardware Limitation

Due to the restricted computation power and the memory size of the EPCglobal Gen-2 RFID tag[1], the implementation of well-known cryptographic algorithms on the tags are still very computational intensive and is not possible at this moment. By Moore's law[2], it is optimistically estimated that cryptographic function will be finally available in the low cost tag.

1.3 Privacy and Security Concerns

Since standard cryptographic are not feasible at this generation of tag, only simple authentication schemes are using in the current system. The current system is vulnerable to many security risks and they are obsolete to the deployment of RFID. Tracking and trace may expose company's confidential logistic data, Denial of Service Attack (DoS)[3] may decrease the efficiency in the use of supermarket payment gateway and unauthorized read may expose customer privacy information.

CHAPTER 2 BACKGROUND STUDIES

2.1 Background of EPCglobal

Several organizations like EPCglobal[4] and ISO[5] are currently actively working on RFID standardization. Particularly, EPC is a joint venture between EAN International (Europe)[6] and Uniform Code Council (USA)[7] aim at standardizing the electronic product code (EPC) technology and achieving world-wide adoption of the RFID standard. Since EPCglobal unifies the two main organizations who are responsible for the barcode technology and its board of governors include representatives from The Fillette Company, Procter & Gamble, Wal-Mart, Auto-ID Labs[8] and other, the standard highly potential to influence the RFID technology at the global scale.

2.2 EPCglobal Class-1 Gen-2 RFID Specification[1]

EPCglobal Class-1 Gen-2 RFID is one of the most important standards proposed by EPCglobal. The paper "EPC Radio-Frequency Identity Protocol Class-1 Generation-2 UHF RFID Protocol for Communications at 860 MHZ – 960 MHZ Version 1.0.9"[1] has clearly defined the functions and operations of a RFID tag. In table 2.1, I have briefly summarized the properties of Gen-2 RFID tags:

Table 2.1 Summary of EPCglobal Class-1 Gen-2 RFID Tag Properties

Type	Passive – power is supplied from the reader
Operating Frequency and	UHF band 800 – 960 MHz
Operating Range	2 – 10 m
Pseudo-Random Number Generator (PRNG)	On chips 16 bit PRNG is available
Cyclic Redundancy Code (CRC)	On chips CRC – 16 is available
Kill password	A 32-bit va lue to kill the tag and render it silent there-after
Access password	A 32-bit value to trans it the tag to th e secured state which can perform the read and write operation on the tag

2.3 Pseudo-Random Number Generator

PRNG is an algo rithm which uses a seed as input to generate a sequence of num bers that the properties are like ly random. However , the se quences are not truly random, since the number will repeat after certain trial.

Linear cong ruential gen erator is on e of the common PRNG algorithm . The for mula are given as : $X_{n+1} = (aX_n + c) \bmod m$ where X_{n+1} is the random number ,Xn is the seed , a is a constant , c and m are relatively prime constant. After each generation, X_{n+1} will be the input as X_n to generate a new number and finally to form a sequence.

2.4 Cyclic Redundancy Code

CRC is a form of error detecting checksum code, a simply way to ensure the original data integrity. It is usually used in telecommunication or storage to detect error in data transmission. In fact, CRC shares some properties of a hash function, it takes a data stream of an arbitrarily length of input and produces a fixed length of output. Inside a CRC function, it actually performs a long division with a polynomial which the quotient is discarded and the remainder becomes the result. There exist numbers of different kinds of CRC algorithms, including CRC-16-CCITT and CRC -32.

2.5 Electronic Product Code

EPC is a family of coding schemes similar to bar code we are using today . EPC was created by MIT Auto-ID Center[9] and is currently managed by EPCglobal aiming at an eventual successor to the bar code. EPC accommodates existing coding schemes and defines new schemes where necessary.

2.6 Security Requirement

In the following paragraphs, I have summarized the security requirements that we should concern in RFID security.

Tag Anonymity: Adversary can not track and trace the tag.

Data confidentiality: Tag's data can be only retrieved by authenticated parties only.

Data Integrity: Tag's data can be only modified by authenticated parties only.

Mutual authentication between reader and tag: Both tag and reader are able to identify and respond to the authorized.

Forward scenery: Even the tag is compromised, the past communication can not be traced.

Anti-cloning: Adversary can not clone the tag without tampering with tag.

2.7 Related Works

Juels[10] proposed minimalist cryptography which used the pseudonym-throttling scheme to prevent tracking and trace. This mutual authentication scheme stores a list of pseudonyms and keys on tag and backend database server. To resist cloning and eavesdropping, the scheme updates tag's pseudonym list using one-time pad. However, the scheme required extra memory for Gen-2 tag, which required extra cost for the tags. The scheme communication cost is relatively high because of the "refresh" on each successful authentication.

Dimitriou[11] proposed a communication scheme that aims at performing efficient identification of multiple tags and taking the concerns of privacy issues. The scheme avoid tracing by avoiding the transmission of static message from tag. It makes use of a PRNG and a pseudo random function (PRF) for symmetric key encryption. However, there exists a serious problem is that since all the tag share the same secret keys, if any tag is compromised, the entire system's security collapses.

CHAPTER 3 PROJECT OBJECTIVES

Since the standard cryptography algorithms are not available on the current generation tag, on the other hand current authentication scheme has only pay little attention to the privacy and security issue I have mentioned earlier. Therefore it is necessary to develop a secure mutual authentication scheme without using the standard cryptography algorithm for the current generation tag.

My project will aim at developing a scheme which can secure the communication in the RFID system, and is able to resist various attacks. I will evaluate the possible attacks and vulnerabilities of the proposed scheme in the paper titled "Enhancing Security of EPCglobal Gen-2 RFID Tag Against Traceability and Cloning" published by Duc at el.. I will base on Duc et al.'s scheme to develop a new protocol which will enhance the security performance.

CHAPTER 4 DUC ET AL.'S SCHEME REVIEW

Duc et al. proposed a communication scheme[13] to protect user privacy for RFID system. The scheme based on a synchronous session key between tags and back-end database server to authenticate each other. This mutual authenticate scheme takes the advantages of the hash properties of CRC function and a PRNG that are supported by EPCglobal Class-1 Gen-2 tags. The underlying idea is by using the same PRNG with the same seed at both tag and back-end data base to generate the same session key on both side. To prevent tag send static message before update of the session key, a random number is added in the authentication process. Data will be encrypted by performing logic operation Xor with the session key before transmission. Session key will be updated after each successful authentication. The following paragraphs will briefly explain the protocol flow.

4.1 Symbol Notations

T - RFID Tag

R - RFID Reader

S - Backend Database Server

r - Pseudo-Random Number Generated by Tag's PRNG

CRC(:) - CRC Function

PRNG(:) - PRNG Function

K_i - Session Key for i^{th} Session

A - Adversary

4.2 Initialization of Tags and Back-End Database Server

Initially during the manufacturing time, the tag has assembled with its EPC and the necessary parameters for the PRNG. A random seed number for PRNG and PIN is chosen and then stored into both *T*'s memory and *S* entry corresponding to the matching EPC. This is very important that each EPC must exactly match with its PRNG seed number and PIN, otherwise the tag can not be authenticated by the back-end server.

4.3 Communication Channel between *R* and *S*

The scheme assumes that *R* is communicating with *S* in a secure channel, both *R* and *S* are able to perform standard cryptography authentication. *S* can send the EPC and data to *R* in an encrypted form. *S* can even depend on the privilege of *R*, to determent what kind of information can send to the reader.

4.4 Protocol Flow

R: First of all, *R* sends a query request to *T*

T: *T* generates a nonce *r* and form the message $M_{1T} = CRC(EPC||r)$ Xor K_i and C = $CRC(M_{1T}$ Xor $r)$. CRC in M_{1T} actually is acting like a hash function while in C functioned as error detection purpose. M_{1T}, C and *r* then will send to reader.

R: R forwards M_{1T} , C and r to S.

S: For each tuples in *S*, it generates a message M_1 in the same way as the generation of M_{1T} in *T* until a match where $M_{1T} = M_1$ is found. If a matched tuple is found, *T* is successfully identified and authenticated. *S* forwards *T*'s information to *R*. However, if no matched tuple is found, *S* will send a tag reject message to *T* via *R*.

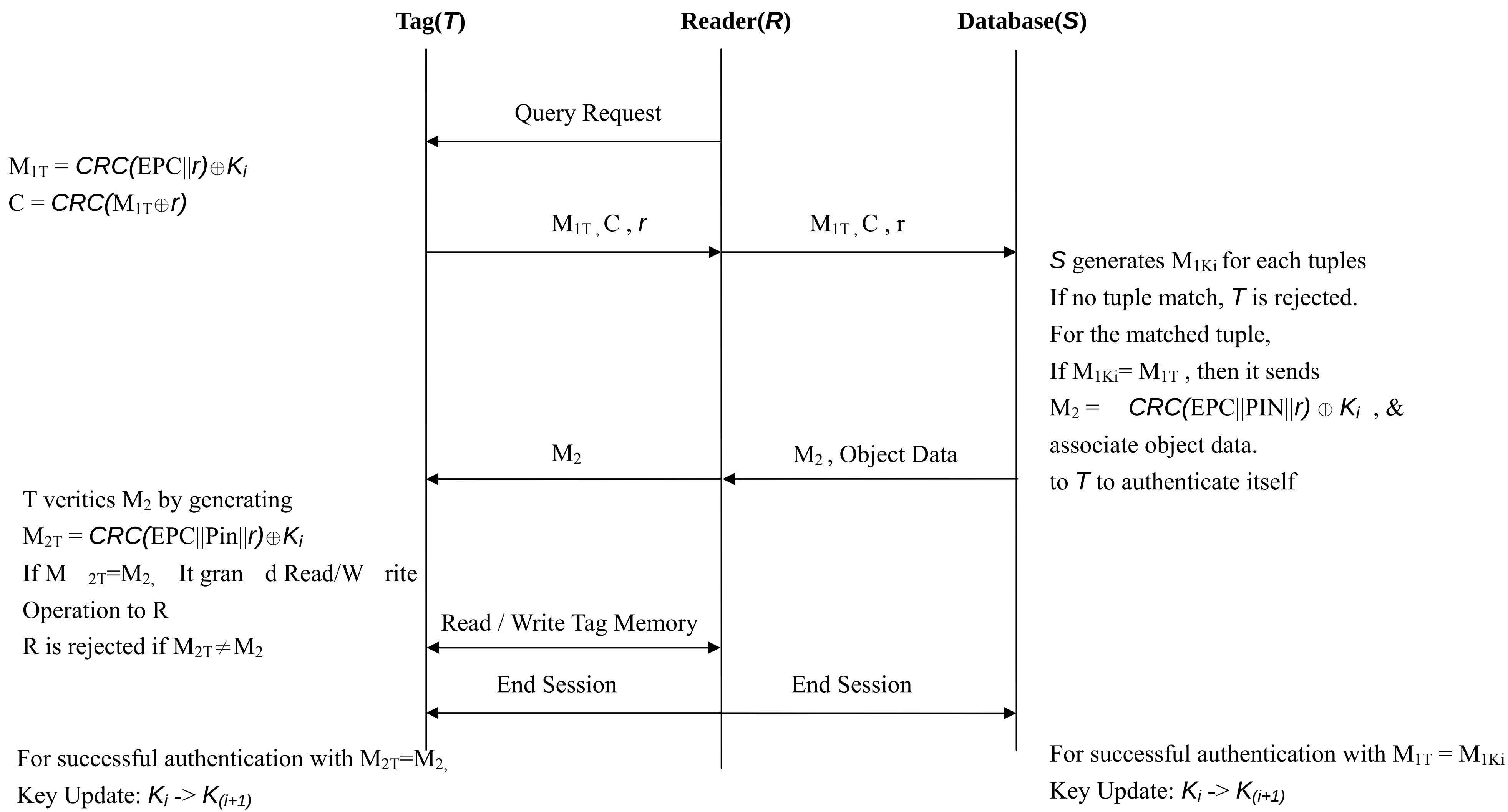
To update the information on *T*, *R* requires to authenticate itself to *T* with the generation of M_2.

S uses the matched tuple's EPC, PIN and K_i to generate the message M_2. where $M_2 = CRC(EPC||PIN||r)$ Xor K_i Finally *S* sends the corresponding object data and M_2 to *T* via R.

T: *T* generates a message M_{2T} to verify M_2 from *R*. *T* uses its EPC, PIN, *r* and K_i to generate the message M_{2T} in the same ways for M_2. If M_{2T} is equal to M_2 then the authentication procedure completed. Data exchange is Xor with the session key K_i to encrypt or decrypt. However, if M_{2T} is not matched to M_2 reader is rejected and the session end immediately.

When data exchange is completed, R signals an "end session" message to both *S* and *T*. Both *S* and *T* updates the session key where $K_{i+1} = PRNG(K_i)$. Figure 4.1 has given the protocol flow diagram.

Duc et al.'s Protocol Flow Diagram



Data on T
EPC , PIN , PRNG's Seed for K_i

Fig. 4.1 Duc et al.'s Protocol Flow Diagram

Data in S
EPC , PIN , PRNG's Seed for K_i

CHAPTER 5 POSSIBLE ATTACKS AND VULNERABILITIES ON DUC ET AL.'S SCHEME

Duc et al.'s protocol is not able to resist the DoS attack, and it can not provide forward secrecy to the RFID system . Since this au thentication repl y on the synchronized session key between *T* and *S*, an adversary can initiate replay attack, man-in-the-middle attack and brute force attack, as a r esult of DoS in the RFI D system. If any one of the "end session" comm and was intercepted, the shared session key between *T* and *S* will be out of sync hronization. As a result, *T* can not be authenticated anymore. The above DoS att acks actually are based on this vulnerable, aiming at intercepting the delivery of the "end session" command sent from *R* to *T*.

5.1 Replay Attack

An adversary can use a spoofed *R* to send a query request to tags, then record the replay m essages M_{1T} and nonce *r* fro m *T*. Recorded m essage will r eplay with a session started with an authorized *R*, finally *S* will update its sess ion key while *T*'s session key will rem ain unchanged. As th e session key is out of synchronization between *T* and *S*, therefore *T* can n ot be authenticated any more. This is one of the high level threats to the RFID system, as the replay attack can perform on a large numbers of *T* at a time.

5.2 Man-In-The-Middle Attack.

Man-in-the-middle attack is very similar to replay attack, an adversary acts like a hubs to store and forward messages between *R* and *T*. However, an adversary will intercept the command "end session" from *R* to *T*, to make the session key out of synchronization. Man-in-the-middle attack is a high level threat to the RFID system too, as it can also perform on a large numbers of tag at a time.

5.3 Brute Force Attack

Since tag-to-reader authentication relies on the correspondence between nonce *r*, EPC and session key. It is important to take note that, before the update of the session K_i in a successful authentication, the session key will remain unchanged, while the EPC is always a constant. Therefore the variance of M_{1T} is basically determined by *r*. An adversary can take this property to initiate a brute force attack on the message M_{1T}. A random message M is chosen, then an adversary can send along with different *r* in each session until a reply of M_2 from reader. As the length of *r* is 16 bits only, the maximum trial times for *r* in particular M is only 65536. The probabilities that the random message finds a match in *S* is mainly depends on the number of tuples exist in *S*. This is a very dangerous attack to the whole system, as the message length of M_{1T} is 16 bits only, an adversary can send all the combination of M_{1T} and *r* to *R*, it only cost 2^{32} trial times to match all the tuples exists in database.

5.4 Forward Secrecy

If the tag is compromised, an adversary can obtain the EPC, PIN, K_i. From the eavesdropped communication data, we can trace the past communication record between *T* and *R* by computing the respective M_{1T} and M_2 with the obtained parameters. For instance, an adversary can take M_{1T} Xor M_2 from the past communication, that can eliminate the session key and remain only the *CRC (*EPC Xor *r)* Xor *CRC (*EPC||PIN||*r)*. Then we may use the obtained parameter from *T* and generate with *r* to trace the past communication of *T* from the eavesdropped past communication data.

CHAPTER 6 MY NEW PROPOSED PROTOCOL

With respect to the possible attacks and vulnerabilities in Duc et al.'s security scheme, I have developed a new security scheme to improve the security performance for RFID system.

The major differences between my scheme and Duc et al.'s scheme are the additional random number challenge from the reader, database will keep the old session key for each tag, update of access PIN after each successful authentication and acknowledgement of M_2 from *T*. The following paragraphs will briefly explain the flow of my proposed protocol.

6.1 Symbol Notations

T - RFID Tag

R - RFID Reader

S - Backend Database Server

r - Pseudo-Random Number Generated by on Tag's PRNG

n - Pseudo-Random Number Generated by Reader RNG

$CRC(:)$ - CRC Function

PRNG(:) - PRNG Function

K_i - Session Key for i^{th} Session

K_{iT} - Session Key in the Tag's memory for i^{th} Session

PIN_i - Access PIN for i^{th} Session

A - Adversary

6.2 Initialization of Tags and Back-End Database Server

Initially during the manufacturing time, the tag has assembled with its corresponding EPC, and necessary parameter for the PRNG. A random seed number for PRNG and PIN is chosen and is stored into both T's memory and S entry corresponding to the matching EPC. The database will store the session key K_{i-1} and the PIN_{i-1} after the first authentication as well.

6.3 Communication Channel

Communication between *R* and *S* is assumed in a secure channel, which cryptographic algorithm can be used in authentication and the object data exchange. The protocol below can secure communications between *R* and *T* in an insecure wireless channel.

6.4 Protocol Flow

R: *R* generates a 16-bit random number *n* by its Random Number Generator(RNG) and send it together with query message to *T*.

T: *T* generates a 16-bit random number *r* by on *T*'s PRNG, then it generates the message M_{1T} = *CRC (*EPC||*n*||*r)* () Xor K_i and the error checksum code C = *CRC (*M_{1T} Xor *n* || *r)*. Finally, *T* will send M_{1T} , C and *r* to *R*.

R: *R* check C = *CRC (*M_{1T} Xor *n*||*r)*to detect error in transmission, *R* forwards M_{1T}, C , *r* and *n* to *S*, otherwise, tag is rejected.

S: *S* generates M_{1Ki} = *CRC (*EPC||*n*||*r)* Xor K_i and

$M_{1K(i-1)}$ = *CRC (*EPC||*n*||*r)* Xor K_{i-1} for each tuples in *S*.

If no tuple matched for $M_{1Ki} = M_{1T}$ or $M_{1T} = M_{1K(i-1)}$, the tag is rejected.

If $M_{1T} = M_{1K(i-1)}$, it reveal that the session key is out of synchronization.

Therefore, *S* generates $M_2 = M_{2K(i-1)}$ = *CRC (*EPC||PIN_{i-1}||*n*||*r)* Xor K_{i-1} and send it to *T* via *R*. *S* then informs *R* to send the "end session" command to *T*, in order to update its K_i and PIN_i while *S* keeps both K_i and PIN_i and K_{i-1} and PIN_{i-1} unchange. In this session, *R* did not perform any read and write operation to *T*, as it was regarded as an unsuccess ful authentication due to session key was out of synchronization. Finally *R* will initiate a new session with *T* with an updated session key.

If the tuple is matched where $M_{1T} = M_{1Ki}$, it generates $M_2 = M_{2Ki}$ = *CRC (*EPC|| PIN_i||*n*||*r)* Xor K_i . *S* sends M_2 and the associated object data to *R*. *R* then forward M_2 to *T*.

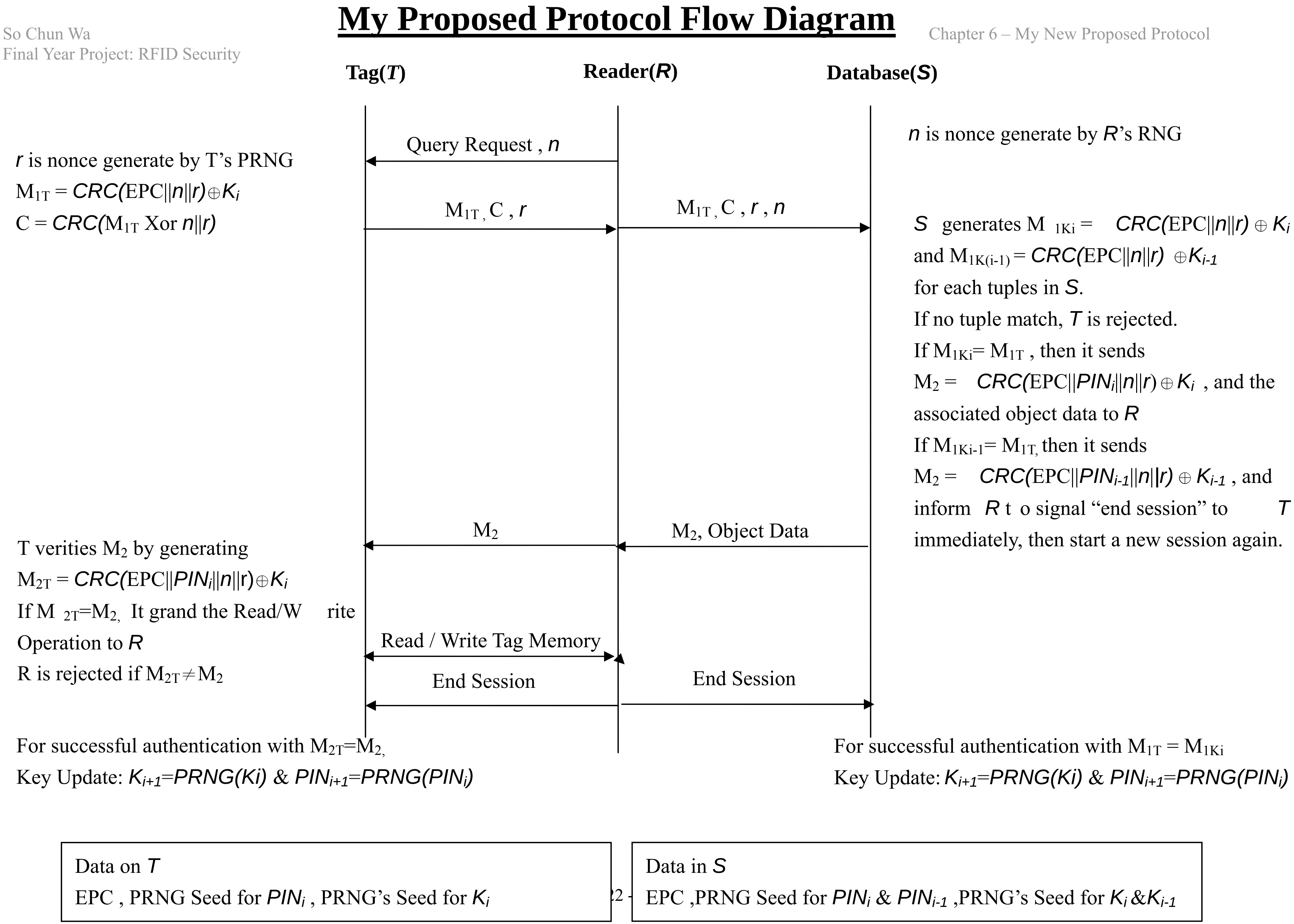
T: *T* verifies M_2 by computing M_{2T} = *CRC (*EPC|| PIN_i||*n*||*r)* Xor K_i, if $M_{2T} = M_2$, *R* is authenticated, reading and writing *T*'s memory is grand to *R*.

Otherwise, *R* is rejected.

R: Data exchange between *T* and *R* is encrypted and decrypted by Xor with the session key K_i.

When *R* has finished the reading and writing operation to *T*, *R* sends an "end session" command to both *R* and *S* to trigger the key update process. Both *T* and *S* will update K_i and PIN_i by $K_i = PRNG(K_{i-1})$ and $PIN_i = PRNG(PIN_{i-1})$. Figure 5.1 has given the protocol flow diagram.

Fig. 6.1 My Proposed Protocol Flow Diagram



6.5 Tag's M_2 Acknowledgement

During the stage of tag authentication in back-end database server, an exceptional case may happen. Back-end database server may find more than one tuples match with M_{1T} as the CRC is 16-bit only. If the server continues the authentication process with any one of the matched tuple, but ignoring other matched tuples, the reader may be rejected due to failure in verification of M_2. The reader has to initiate a new session to the tag again until it is verified, which will increase the communication cost. In a high population tag RFID system, it has high probability that more than one tuple is matched in most session, as a result decreases the efficiency of the system.

To deal with this problem, I propose an extra acknowledgement steps for tag. Back-end database generates and send M_2 for all matched tuples. The tag acknowledges to server which M_2 is successfully verified via the reader. Although the addition of M_2 acknowledgement will increase the communication cost, it can prevent M_2 collision in database which lead to failure in tag-to-reader authentication, which may even induce higher costs.

The collision problem may significantly decrease if longer length of CRC value is used. However only 16-bit CRC is available in the current generation of tag, it is suggested to add this extra acknowledgement step in the protocol in a high RFID tag population system.

CHAPTER 7 PROGRAMME SIMULATION

In order to find out the average appearing time of M_{IT} for a given tag, a simulation programme is built for simulating both Duc et al.'s protocol and my protocol. The programme simulates M_{IT} out of 65535 trials before a successful session key update for a given tag.

7.1 Tag's Parameter Selection

There are four essential parameters for each tag which includes a 96-bit EPC, 32-bit PIN, 16-bit K_i and PRNG's parameters. EPC, PIN_i and seed for K_i are randomly chosen from a RNG. Another random number testing programme(Figure 7.1) is built for choosing the PRNG's parameters to satisfy the requirement in Gen-2 tag specification.

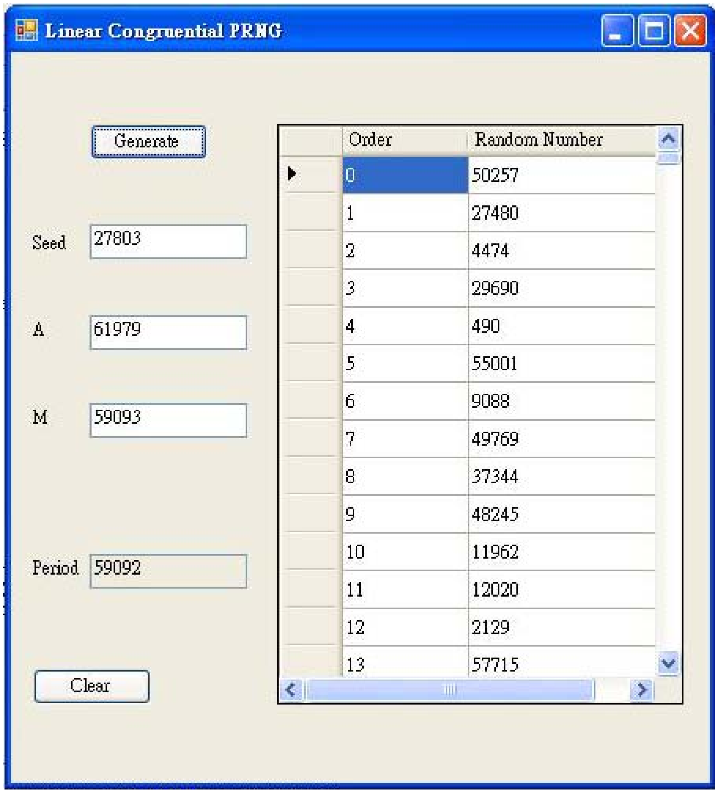


Fig. 7.1 Linear Congruential PRNG Period Test Simulation Programme

7.2 Linear Congruential Generator

A popular class of PRNG, linear congruential generator has been chosen. It gives the form of $X_{i+1} = (aX_i+c) \bmod m$ where a, c and m are PRNG parmeters, X_i is the seed. Value a = 61979 , c = 0 and m = 59093 have been chosen as the PRNG parameters for the tag in the simulation. The repeating period is 59092 which satisfies the requirement of Gen-2 tag.

7.3 Protocol Simulation Programme

Figure 7.2 shows the layout of my simulation programme. The main screen in the centre simulates the protocol flow. The gird view on the left simulates the population of tags, while the right hand side gird view simulatse tag's information tuples maintained in the database. On the right hand corner, the programme performs simulation for M_{1T} out of a given trial times before a successful session key update for a selected tag in the tag's grid view. In order to find out how the CRC function affect the average appearing times out of a given trial, the programme can simulate the generation of M_{1T} for both CRC-16-CCITT used in current Gen-2 tag and CRC-32.

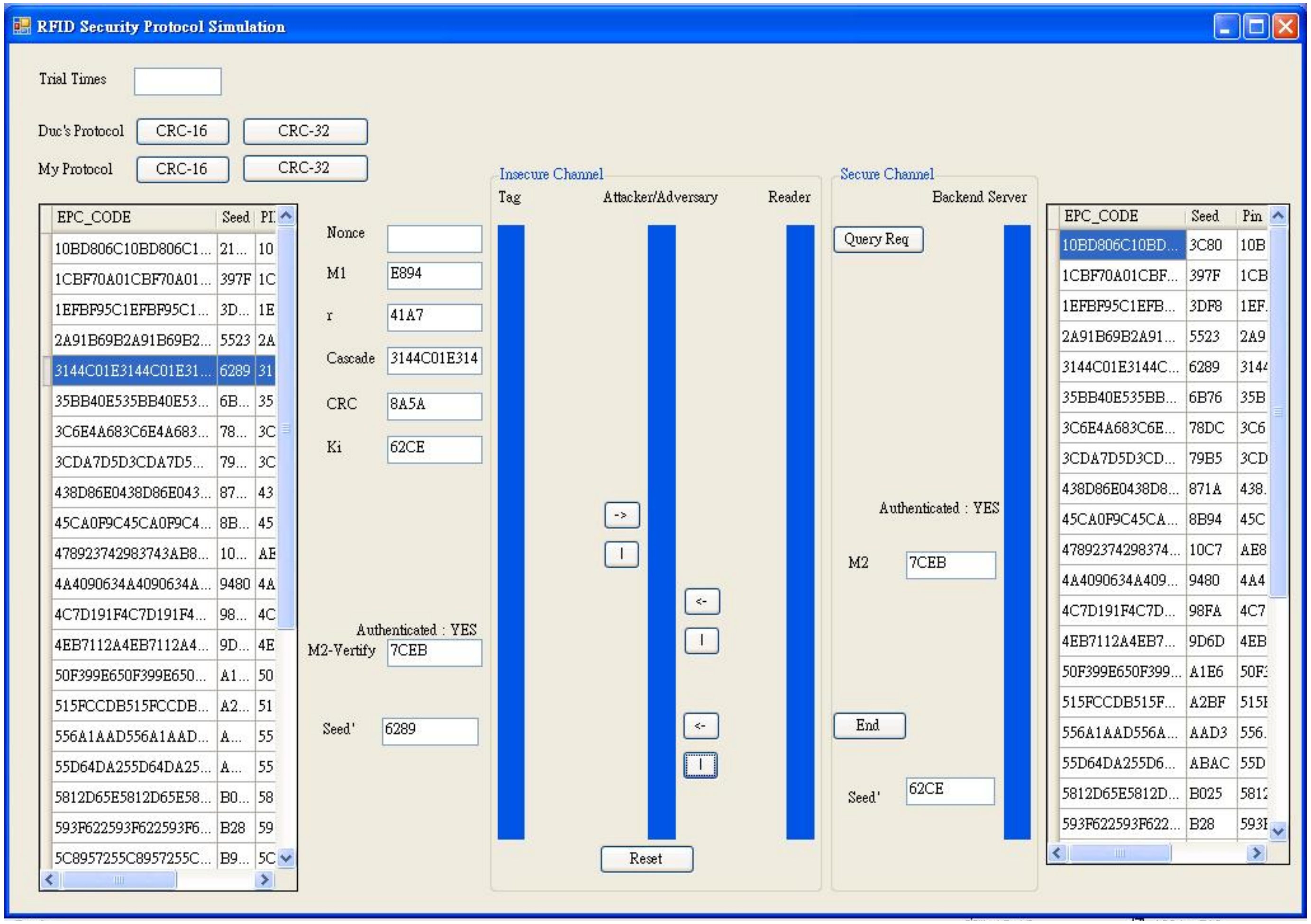


Fig. 7.2 Protocol Simulation Programme Main Screen

7.4 Duc et al.'s Protocol under Man-In-The-Middle Attack

An adversary appears in between the tag and the reader, acting like a store and forward hubs. It forwards query request from the reader and then send M_{1T}, C, *r* to the reader received from tag, and forward M_2 to tag like an ordinary authentication process. However after the mutual authentication, it blocks the "end session" command send from reader. As a result, the tag can not be authenticated anymore. Since the tag remains its session key and PIN unchanged while back-end database server updates them with PRNG,. The simulation programme shows the tag is rejected in next authentication in Figure 7.3. It can perform massively to make whole RFID system collapse.

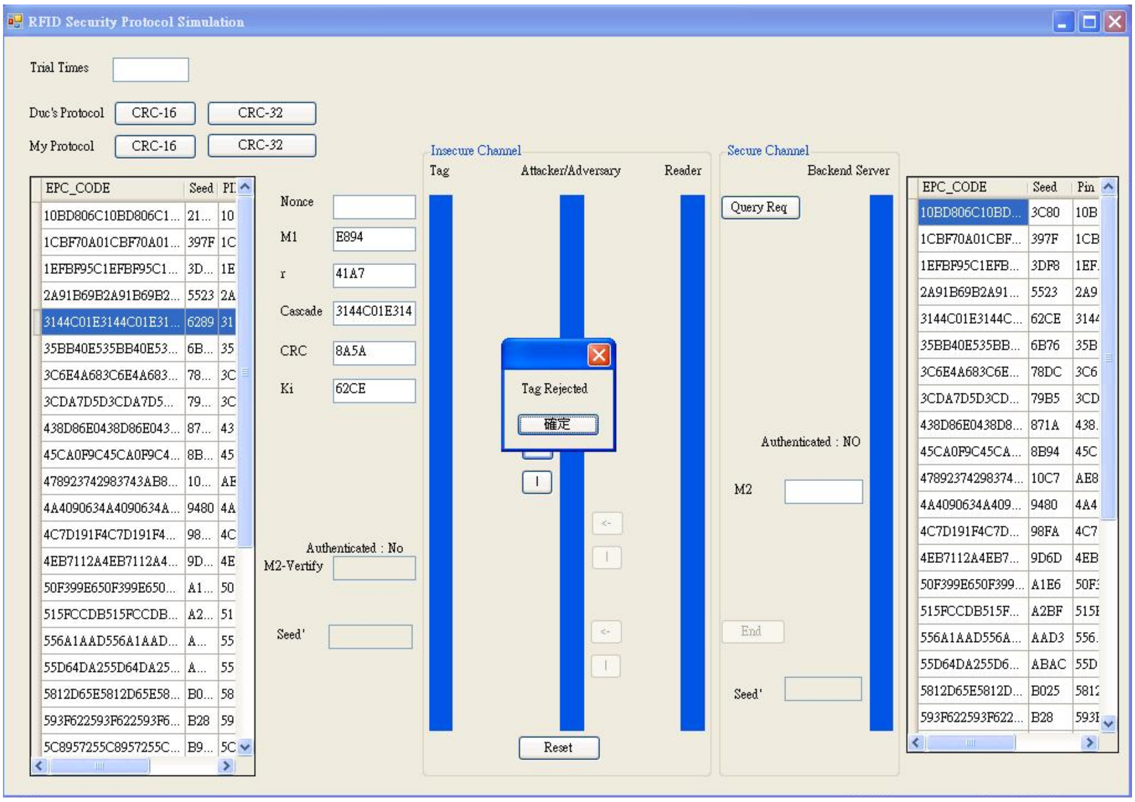


Fig. 7.3 Result of DoS Attack in Duc et al.'s Protocol Simulation Screen

7.5 Simulation Procedure for Average Appearing Time of M_{1T}

A tag was randomly chosen to loop recursively to generate 65535 trials for M_{1T} in the simulation programme, the result appears in the data grid view is shown in Figure 7.4. Finally, the result in the data grid view wa s exported to either excel or txt file for further analysis in excel. The simulation tags parameters are shown on T

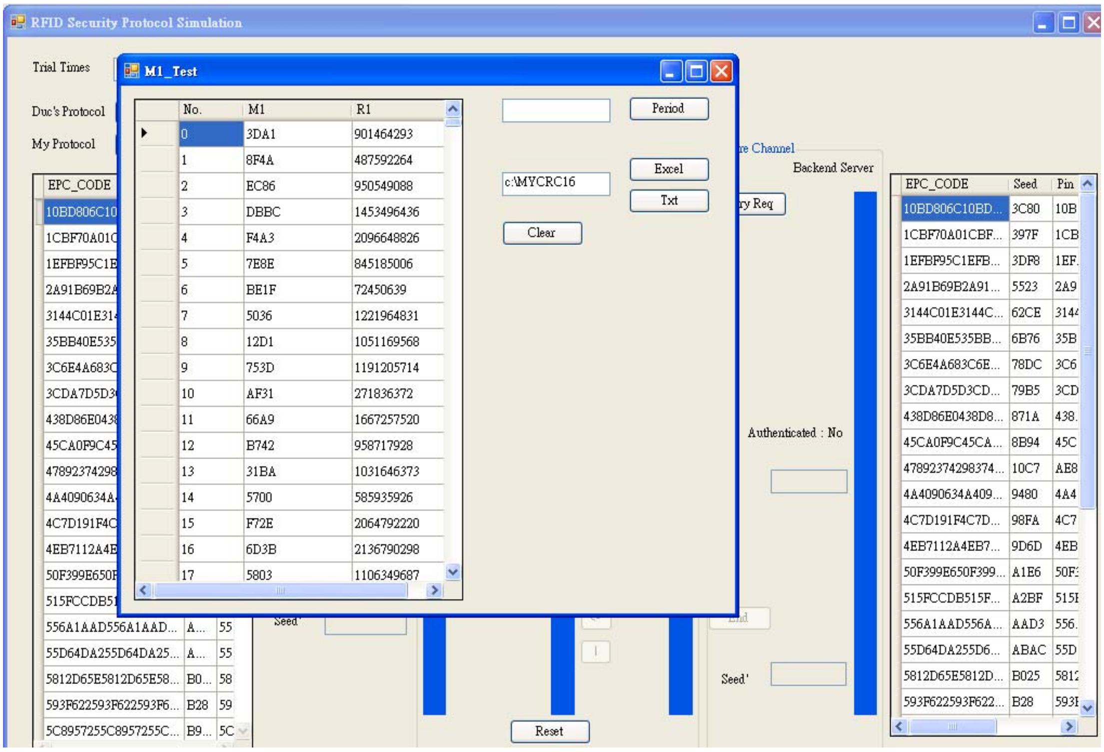


Fig. 7.4 Screen of M_{1T} Appearing Times out of 65535 Trial in Simulation

7.6 Programme Simulation Result for using CRC-16-CCITT

The Figure 7.5 and Table 7.1 show the result of average appearing times of M_{1T} out of 65535 trials for my protocol and Duc et al.'s protocol. The main difference of M_{1T} between the two protocols is the addition of random number challenge n from reader in my protocol while all other parameters remain the same.

Table 7.1 Average Appearing Time of M_{1T} with 16-bit n

	Average Appearing Time of M_{1T} out of 65535 Trial
Duc et al.'s Protocol	1.514805
My Protocol	1.531227

Table 7.2 Average Appearing Time of M_{1T} with 32-bit n

	Average Appearing Time of M_{1T} out of 65535 Trial
Duc et al.'s Protocol	1.514805
My Protocol(r =32-bit)	1.523425

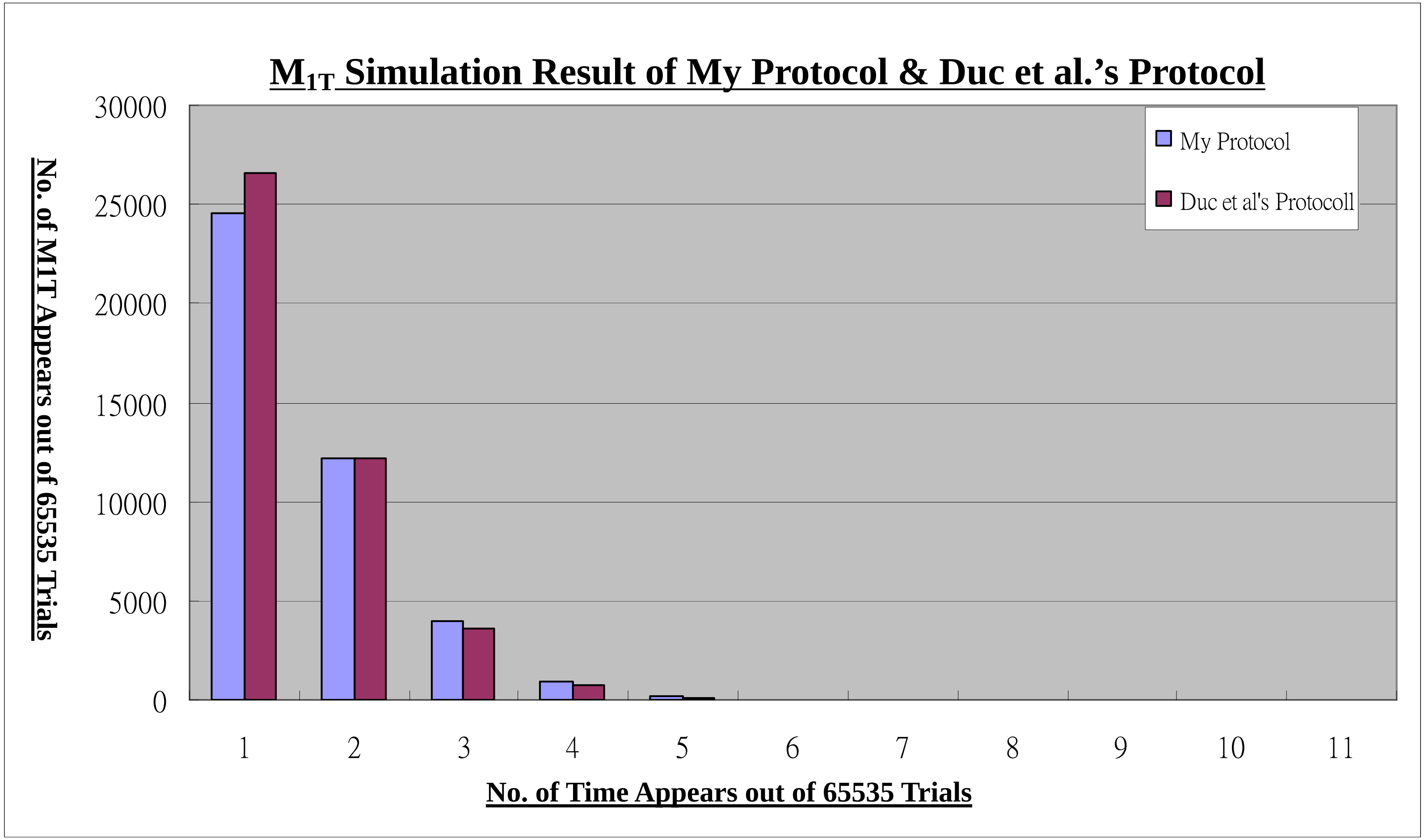


Fig. 7.5 Graph of M_{1T} Simulation Result of My Protocol & Duc et al.'s Protocol

7.6.1 Symbol Notation

T_D - Average Appearing Time out of 65535 Trial for Duc et al.'s Protocol

T_M - Average Appearing Time out of 65535 Trial for My Protocol

7.6.2 Simulation Result Analysis

The simulation result reveals that the period of CRC output in M_{1T} does not follow the period of *r*. Since *r* is the only changing parameters in the M_{1T} throughout the trial, it is expected that T_D should approximate equal to *r*'s period. The period of *r* is found to be 59092 time, therefore the average appearing times out of 65535 trial should be around 1.1. However T_D is found to be around 1.5 which has a significant difference from *r*'s period.

The simulation result also shows that T_M and T_D remain similar. An addition of random number challenge did not reduce T_M, although the combined number from *n* and *r* is found to have no repetition out of the 65535 trials. It contradicts my expectation that increasing the CRC input length in M_{1T} from 112-bit to 128-bit which has increased the randomness of the CRC input, as a result may reduce the appearing times of M_{1T} before session key update. In order to further test whether the length of CRC input affect T_M and T_D, another simulation was performed with increased length of random number challenge from reader to 32-bit. However the result remains the same as the pervious simulation. The results are shown in Table 7.2.

It is possible that the bottleneck is on the CRC function but not the randomness or the length of the CRC input. I have conducted another simulation for using CRC-32 in M_{IT} generation for both protocols to investigate the new average appearing time of M_{IT} out of 65535 trials.

7.7 Simulation Result for M_{1T} using CRC-32

Table 6.3 Average Appearing Time of M_{1T} with 16-bit *n* Using CRC-32

	Average Appearing Time of M_{1T} out of 65535 Trial
Duc et al.'s Protocol (CRC-32)	1.000168
My Protocol (CRC-32)	1.000015

7.7.1 Simulation Result Analysis

This simulation result(Table 6.3 & Figure 6.6) has a significant difference from all the pervious simulation. Both T_M and T_D have dropped from around 1.5 to around 1. Although it is better to draw the conclusion after conducting more simulations with different type of CRC function, time is limited in my project. However, with this significant reduction in T_M and T_D, we may still conclude that the bottleneck is in the CRC function.

In conclusion, the average appearing time of M_{1T} in my protocol and Duc et al.'s protocol relies on the CRC function, but not the length or randomness of the input. With the simulation result, it is concluded that the CRC is the bottleneck. To reduce the average appearing time of M_{1T}, we should use CRC-32 instead. However it is not available in the current generation of tag, hopefully it will appear soon in the near future. In my protocol, although the random number challenge does not help in reducing the average appearing time, it prevents replay attack effectively to enhance the protocol's security performance, it is considered necessary in the protocol.

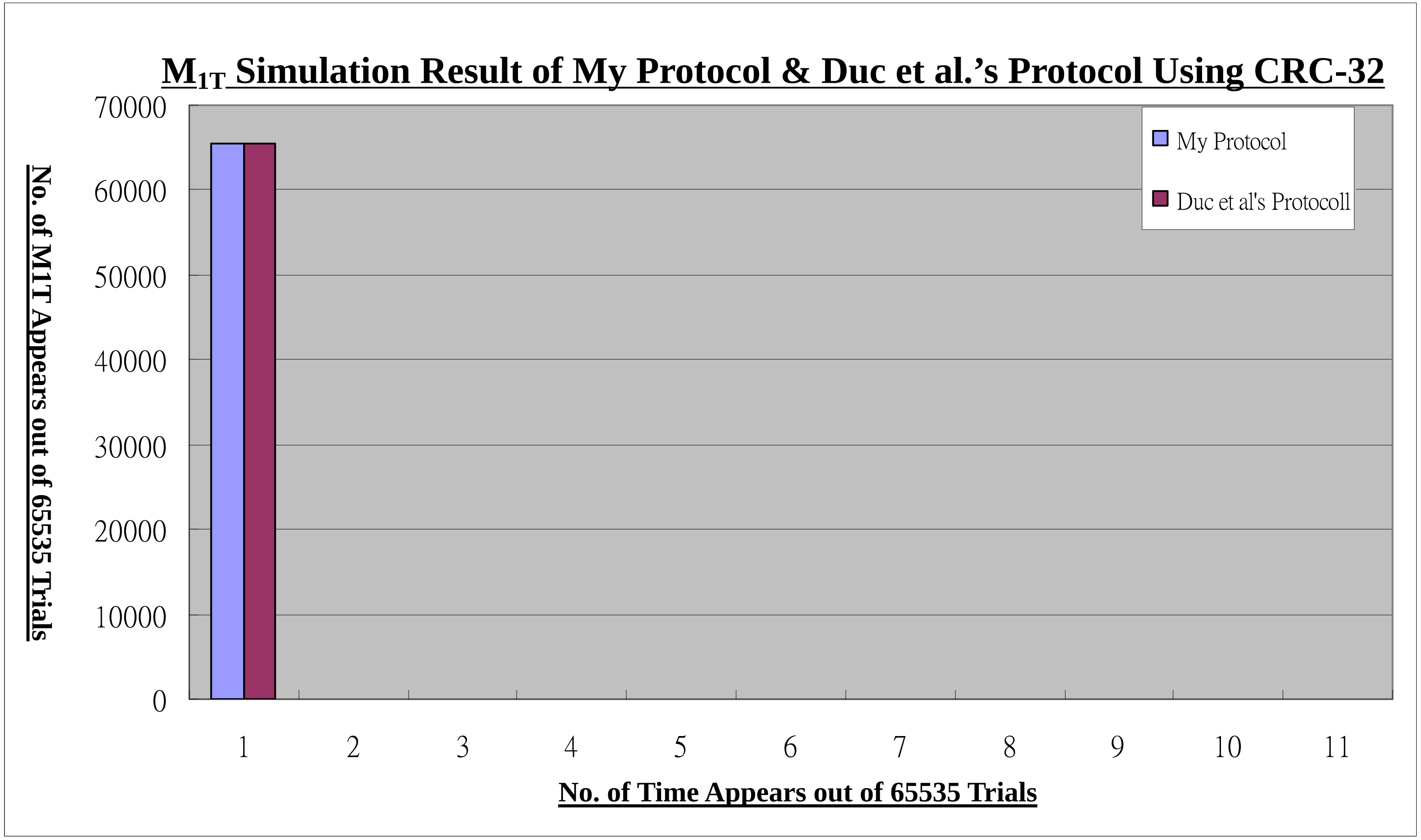


Fig. 7.6 M_{1T} Simulation Result of My Protocol & Duc et al.'s Protocol Using CRC-32

CHAPTER 8 SECURITY AND COMPLEXITY ANALYSIS

My proposed security scheme has solved the security loopholes in Duc et al.'s scheme I have raised in chapter 5 and guaranteed a secure mutual authentication between reader and tag in an insecure wireless channel .

8.1 Security Analysis

I analyzed the proposed scheme's security performance in tag anonymity, data privacy, mutual authentication, forward secrecy, key attack, DoS attack and replay attack in this section

8.1.1 Tag Anonymity

Tag will never emit static ID, a new random number is chosen from Reader and Tag in each session to ensure tag anonymity.

8.1.2 Data Privacy

Tag never sends plain text data through insecure channel, data is always encrypted by a session key with nonce. Reader can use cryptography algorithm to exchange data between back-end database server. Therefore data privacy is strongly protected.

8.1.3 Mutual Authentication

My protocol performs both tag-to-reader and reader-to-tag authentication. Database authenticates the tag by verifying the message M_{1T}. Tag verifies M_2 generated by database. This mutual authentication scheme ensures data exchange will only grand to authenticated parties only.

8.1.4 Forward Secrecy

Even the tag is compromised at some time later, as the PIN and session key is updated after each successful authentication, an adversary can not trace and track the compromised tag from the past eavesdropped communication data. Therefore the forward secrecy is protected.

8.1.5 Key Attack

The shared secret session keys are chosen randomly for each tag and they are different from each other. Exposure for a single key will therefore not expose other's tags secret information.

8.1.6 DoS Attack

The database will maintain six values including the old session key and old PIN for each tag. Even though the tag is out of synchronization with the database, it can still communicate with the database, by performing a session key and PIN update process to synchronize with database. Although it may increase the communication cost, it can ensure that M_{1T} was not a replay attack.

8.1.7 Replay Attack

The random number challenge from the reader can effectively prevent replay attack from the spoofed tag. The generation of M_{1T} has involved the random number from reader, therefore an adversary can not replay M_{1T} from a eavesdropped communication between spoofed reader and tag.

8.2 Complexity Analysis

I analyzed the proposed security scheme complexity in terms of computation, storage and authentication phrase.

8.2.1 Computation Complexity

-Reader

To communicate with tag, reader requires only a RNG and cryptography algorithm to authenticate and transfer data between reader and back-end database server. The requirements are feasible in the current generation reader. In authentication process, reader actually acts like a store and forward hubs between back-end database server and tag, as the computation complexity are distributed to the back-end database server.

-Tag

Tag-to-Reader authentication process requires a 2CRC, a PRNG to generate the message M_{1T} while Reader authentication process requires a CRC and Xor operation to verify M_2. The key and PIN update process requires 2 PRNG. There are total of 3 CRC, 3 PRNG involved in the whole authentication protocol.

-Database

The database generates both M_{1Ki} and M_{1Ki-1} for each tuple, so the computation complexity is 2N (2CRC + PRNG), where N is number of tuples in database.

8.2.2 Storage

-Tag: Tag is required to store 3 parameter only, i.e.: EPC , PIN_i and K_i.

-Database: Database is required to store five values for each tag including tag's EPC, PIN_i, PIN_{i-1}, K_i, K_{i-1}.

8.2.3 Authentication Phrase

My proposed security scheme is a three-phrase mutual authentication protocol. Phrase one: Random number challenge from reader. Phrase Two: Tag generates M_{1T} to authenticate itself to reader. Phrase Three: Back-end database server generates M_2 which included tag's access PIN to authenticate itself to tag, in order to grand the read and write right to reader.

Table 8.1 Security and Complexity Comparison

	Duc et al.'s Protocol	My Protocol
Backend Server's Complexity	N O(CRC)	2 N O(CRC)
Tag's Complexity	2CRC + 2PRNG	3CRC + 3PRNG
Reader's Complexity	Send ,receive and forward	Send ,receive and forward + 1RNG
Reader Authentication	Yes	Yes
Tag Authentication	Two Phrase	Three Phrase
Resist to Dos Attack	No	Yes
Resist to Replay Attack	No	Yes
M_{IT} Collision in Database	No	Yes
Forward Secrecy	No	Yes

N – Number of tuples in Back-End Database Server

O(CRC) - Computational complexity of CRC algorithm

Table 8.1 has summarized the security features and com plexity of m y protocol and Duc et al.'s protocol. Compare with Duc et al.'s protocol, my prot ocol do not require any extra memory for tag. In order to preven t replay attack, an addition authentication phrase with a random number challenge for tag has been included in my protocol. Database stores both old and current session key to prevent key de-synchronization DoS attacks on tag in my protocol. With an acknowledgement of M_2 from the tag, reader is still able to authenticate itself to tag in the condition with more than one tuples matched for M_{1T} in database. As forward secrecy is protected by updating the access PIN in each session, so an extra PRNG is required in my protocol. It is worth highlighting that, my proposed protocol not only improves the security performance of the RFID system significantly, but also conforms to EPCglobal Gen-2 RFID tag specification.

CHAPTER 9 CONCLUSION

In this project, I have evaluated Duc et al.'s security schemes under different attacks and pointed out its vulnerabilities. It is subjected to different kinds of DoS attacks, exist a weakness in forward secrecy and reader-to-tag authentication collision in database. With respect to the above weakness, I have proposed a new protocol that has improvement in all above weakness in which ensure mutual authentication in RFID system. My scheme has distributed the authentication computation complexity to the back-end database server and reader from tag and my scheme is still conformed to EPCglobal Gen-2 specification.

The simulation results conclude that out the average appearing time of M_{1T} is determined by the CRC function but not only the random number input. This bottleneck can be solved by using a CRC-32 function instead, so hopefully it will be available in next generation of RFID tag in near future.

References

[1] EPCglboal Inc., "EPC radio-Frequency Identify Protocols Class-1 Generation-2 UHF RFID Protocol for Communications at 860 MHz – 960 Mhz Version 1.09", Available at http://www.epcglobalinc.org/standards_technology/specifications.html.

[2] Gordon E. Moore, Moore's Law, Available at http://www.intel.com/technology/mooreslaw/index.htm

[3] Wikipedia, "Denial of Service Attack", Available at http://en.wikipedia.org/wiki/Denial-of-service_attack

[4] EPCglobal Inc., http://www.epcglobalinc.org/.

[5] ISO, http://www.iso.org/

[6] EAN International (Europe), http://www.ean.be/

[7] Uniform Code Council, http://www.uc-council.org/

[8] Auto-ID Labs., http://www.autoidlabs.org/

[9] MIT Auto-ID Center, http://autoid.mit.edu/cs/

[10] Juels, A., Minimalist cryptography for low-cost RFID tag. In conference on Security in communication Networks 2004

[11] Dimitrios, T., A Secure and Efficient RFID Protocol that could make Big Brother(partially) Obsolete, IEEE Computer Society Press

[12] Duc et al., Enhancing Security of EPCglobal Gen-2 RFIDTag against Traceability and Cloning, SCIS 2006

Glossary

RFID	Radio Frequency Identification
RF	Radio Frequency
DoS	Denial of Service
EPC	Electron Product Code
PRNG	Pseudo Random Number Generator
CRC	Cyclic Redundancy Code
PRF	Pseudo Random Function
RNG	Random Number Generator