박 사 학 위 논 문

Doctoral Thesis

RFID 태그를 위한 암호 프로토콜에 관한 연구

A Study on Cryptographic Protocols for RFID Tags

당 누엔득 (鄧 遠 德 Dang, Nguyen Duc)

정보통신공학과

Department of Information and Communications Engineering

한 국 과 학 기 술 원

Korea Advanced Institute of Science and Technology

2010

RFID 태그를 위한 암호 프로토콜에 관한 연구

A Study on Cryptographic Protocols for RFID Tags

A Study on Cryptographic Protocols for RFID Tags

Advisor : Professor Kim, Kwangjo

by

Dang, Nguyen Duc
Department of Information and Communications Engineering
Korea Advanced Institute of Science and Technology

A thesis submitted to the faculty of the Korea Advanced Institute of Science and Technology in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Information and Communications Engineering

Daejeon, Korea
2009. 12. 01.
Approved by

Professor Kim, Kwangjo
Advisor

RFID 태그를 위한 암호 프로토콜에 관한 연구

당 누엔득

위 논문은 한국과학기술원 박사학위논문으로 학위논문심사 위원회에서 심사 통과하였음.

2009년 12월 01일

심사위원장 Kwangjo Kim (인)

심사위원 Daeyoung Kim (인)

심사위원 Soontae Kim (인)

심사위원 Byoungcheon Lee (인)

심사위원 Doo Ho Choi (인)

DICE 20045329 당 누엔득. Dang, Nguyen Duc. A Study on Cryptographic Protocols for RFID Tags. RFID 태그를 위한 암호 프로토콜에 관한 연구. Department of Information and Communications Engineering . 2010. 97p. Advisor Prof. Kim, Kwangjo. Text in English.

Abstract

Radio Frequency Identification (RFID) is an emerging technology to replace the Barcode technology. The technology can be used for many powerful applications including automatic item tracking, smart home appliances, anti-counterfeiting *etc.* The key idea is to attach each and every item with an RFID tag which can be read by RFID readers via radio communication. Each RFID tag is a low-cost device capable of emitting a unique number which will be served as the identification information of an RFID-tagged item in a database at the back-end server.

Unfortunately, a widespread adoption of RFID is uncertain because of its inherent threats in security which includes tag cloning and privacy violation. It turns out that these two security threats come from the very basic operation of an RFID tag, that is to send the identification of an RFID-tagged item (hereafter referred to as *Electronic Product Code* (EPC)) in cleartext. This is an inherent security risk since we depend on the EPC number to recognize a product as genuine or fake. An attacker equipped with a compatible reader can scan many RFID tags to collect a large number of EPC numbers. He then can produce RFID tags which emit exactly the same EPCs he has collected. These tags are called *cloned tags*. The cloned tags can be attached to counterfeited items which should be recognized as genuine items. The core functionality of an RFID tag also raises privacy concern. As each EPC number is unique, an attacker with a compatible reader can recognize and track RFID tags which leads to privacy violation of a person carrying tagged items. Denial or disruption of service might also affect an RFID system. The reasons are two-fold: RFID requires a huge number of tag to be deployed; RFID is a wireless technology at its core and therefore is subject to various sources of interference and jamming.

To deal with the security problems of RFID, the use of cryptographic protocols is required. However, designing cryptographic protocols for RFID tags is a challenging task as a low-cost RFID tag has very limited computational resources. Indeed, it is infeasible to implement current public key cryptographic primitives and block ciphers on low-cost tags. As a result, a new approach to design cryptographic protocols for RFID tags which employ only *lightweight* primitives is required. The most popular lightweight primitive used in designing cryptographic protocols for RFID is hash function.

In this thesis, we aim to solve some of open problems in RFID security. First of all, we propose a lightweight authentication protocol called HB* which is secure against man-in-the-middle attack. HB* is an improved version of another protocol called HB$^+$ proposed by Juels and Weis. However, HB$^+$ is not secure against man-in-the-middle attacks and several attempts to secure HB$^+$ against the said attack have failed. We then address a security weakness against denial-of-service attack (DoS) of many RFID authentication protocols. In particular, we point out that the existing method to authenticate and identify RFID tags may cause the back-end server to do exhaustive search on its database. We solve this problem by using a two-phase authentication method. That is, a tag is first authenticated to verify that it is actually in the database. This phase can be done by an RFID reader. Then, the server only authenticates and identifies tags that are correctly verified by the RFID reader. We apply this method to two well-known RFID authentication protocols called O-FRAP and O-RAP which were proposed by Tri Van Le *et al.* Finally, we try to solve the scalability and security issues of known grouping-proof protocols for RFID. Grouping-proof protocols allow multiple RFID tags to be scanned at once such that their co-existence is guaranteed. One typical application of a grouping-proof protocol is to scan tags that are supposed to stay *together*. We propose a scalable grouping-proof protocol as well as an accompanying security model in which we provide a sound security definition for secure grouping-proof protocols.

Contents

List of Tables

List of Figures

1. Introduction

1.1 An Overview of an RFID System

Radio Frequency Identification (RFID) is a means to auto-identify objects, and assets efficiently and quickly. With RFID technology, RFID tags are attached to consumer items and these tags contain tiny, but durable, computer chips with very small antennas. Passive tags are powered-up from the interrogation Radio-Frequency (RF) signal from an RFID reader whereas active tags are self powered by batteries. The tiny computer chips contain an Electronic Product Code (EPC) number that uniquely identifies the item to which it is attached to, and the antennas automatically transmit this EPC number without requiring line-of-sight scanning, to readers. All the information associated with that EPC number is stored on a network of databases, called the EPC-Information Services (EPC-IS) or the back-end server.

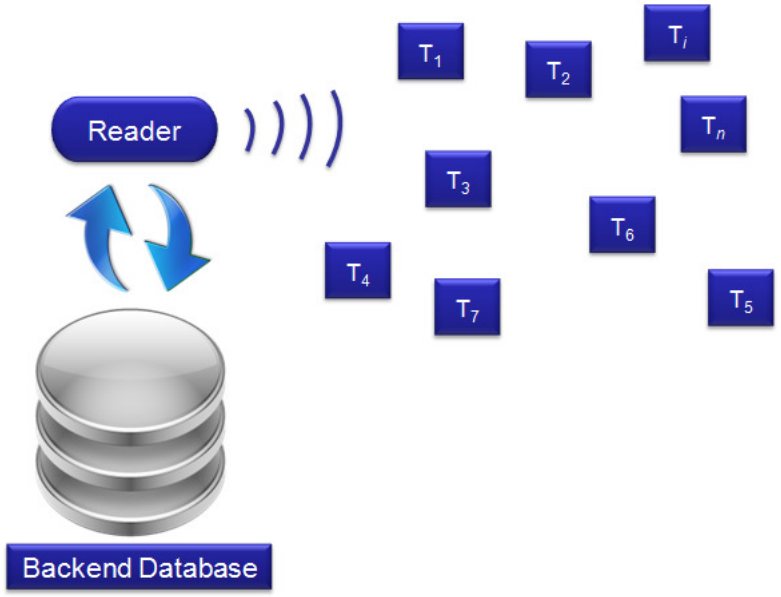

Figure 1.1: Three Components of an RFID System

We illustrate three basic components of an RFID system in Fig. 1.1. In practice, there

is more components in a complete RFID ecosystem. These components include middleware which sits between readers and the back-end server, ONS (Object Naming Serivce) which identifies the database in which information about an object is stored given the object's EPC number.

To design a protocol, including cryptographic protocols for RFID tags, it is important to understand the computational characteristics of an RFID tag. In this thesis, we focus mainly on passive tags which are meant to be deployed at the broadest scale. A passive tag should be very cheap to produce. It is estimated that in the coming years, the cost of a passive RFID tag will be reduced to only a few cents. A typical design of a passive RFID tag is shown in Fig. 1.2. The center of the chip is the processing unit and memory of the tag whereas the outer space is for the antenna.

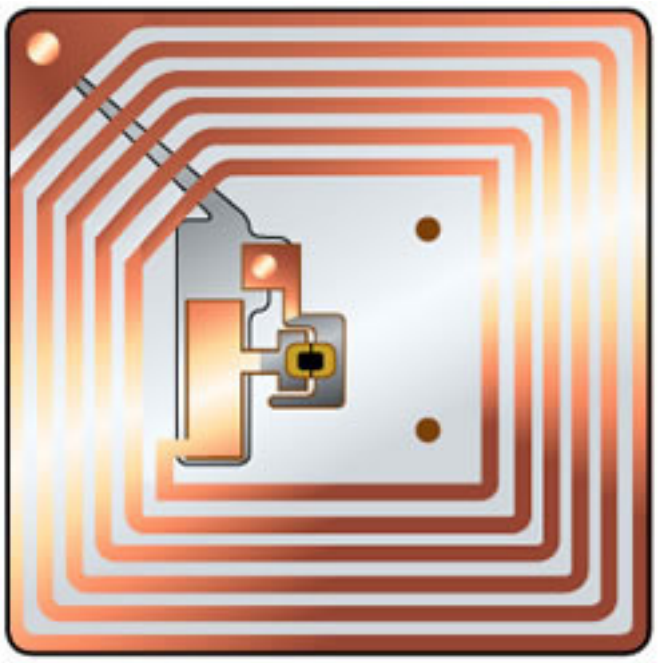

Figure 1.2: A Schematic View of an RFID Tag

The most important standard for passive tags is the EPCglobal Class-1 Gen-2 specification [3]. We refer to RFID tags conforming to the specification as Gen-2 tags. The functionalities of a Gen-2 tag are briefly summarized as follows:

- Gen-2 has a few KB of memory.
- Gen-2 tag communicates with RFID readers in UHF band (800-960 MHz) and its

communication range can be up to 2 10m.

- Gen-2 tag has a Pseudo-Random Number Generator (PRNG) on chip. In addition, it can compute basic Boolean operations and Cyclic Redundancy Code (CRC) for integrity checking purpose.
- A Gen-2 RFID tag can be rendered permanently unusable once it receives the kill command with a valid 32-bit kill PIN (*e.g.*, tag can be killed at the point-of-sale). The goal of turning off an RFID tag is to protect the privacy of the tag holder.
- Read/Write to a Gen-2 RFID tag's memory is allowed only after it is in secure mode (*i.e.*, after receiving access command with a valid 32-bit access PIN).

A few typical applications of RFID technology are described below:

- *Automatic Supply Chain Management:* An RFID tag gives an item an identity just like a barcode. However, RFID tags can be scanned in bulk without human intervention. As a result, one can implement an automatic supply chain management system by attaching RFID tags on all tracked items and placing RFID readers at different check points. The RFID readers can be connected to the back-end server to provide real-time tracking information.
- *Smart Home Appliances:* RFID readers can be integrated into home appliances to provide added benefits to end users. For example, a refrigerator equipped with an RFID reader can scan RFID-tagged items stored inside. Then, the refrigerator accesses to a publicly available or a home server to get various information on each item like the expired date, the item origin, *etc.*
- *Ubiquitous Computing Experience for End Users:* The sheer ubiquitous availability of RFID-tagged items also brings many powerful applications to ubiquitous computing. An end user equipped with an RFID reader (possibly intergraded into his smartphone or PDA) can scan tagged items and collect information about them on-the-go. For instance, when a customer goes shopping in a supermarket, he/she can queries tagged items to get detailed information on the goods, compare the prices, *etc.* On the other hand, an RFID tag can also be embedded into the end user's smartphone. The RFID tag may store information about its owner including personal identity (possibly a pseudo one for privacy reason), banking account, *etc.*. These information might help the user to do some micro payments like bus and subway tickets,

Abbreviation	Description
RFID	Radio Frequency Identification
ONS	Object Naming Service (the equivalent of DNS in an RFID system)
LFSR	Linear Feedback Shift Register
PRNG	Pseudorandom Number Generator
PRF	Pseudorandom Function
MAC	Message Authentication Code
EPC	Electronic Product Code
CA	Certificate Authority
PKI	Public Key Infrastructure
SPN	Substitution & Permutation Network
DES	Data Encryption Standard
AES	Advanced Encryption Standard
ECB	Electronic Codebook (as in block cipher operation mode)
CBC	Cipher-block Chaining (as in block cipher operation mode)
CFB	Cipher Feedback (as in block cipher operation mode)
OFB	Output Feedback (as in block cipher operation mode)
CTR	Counter (as in block cipher operation mode)
RSA	Rivest, Shamir, Adelman (inventor of a public-key encryption scheme)
OAEP	Optimal Asymmetric Encryption Padding
PKCS	Public-Key Cryptography Standard
COA	Ciphertext-only Attack
KPA	Known-Plaintext Attack
CPA	Chosen-plaintext Attack
CCA	Chosen-ciphertext Attack
CCA2	Adaptive Chosen-ciphertext Attack
LPN	Learning Parity with Noise Problem
DoS	Denial of Service

Table 1.1: Abbreviations

movies, *etc.* The surrounding environment can also recognize the user via the embedded tag to provide entrance to buildings, recommendation of available computing services and the likes.

- *Anti-Counterfeiting:* An RFID tag is a computing device and thus capable of storing more information and performing sophisticated scanning protocol than a barcode. An RFID tag can be embedded into bank notes, money papers and passports to prevent counterfeiting. When a tag is scanned, tag-to-reader authentication and vice versa can be performed so that counterfeited tags can be detected. In case of passport, an RFID tag can store not just identity but also biometric information of the owner to provide even stronger anti-counterfeiting. RFID tags also have been used in military in order to identify friend or foe in the battlefield.

1.2 Security Threats to an RFID System

Despite being a fairly new technology and not yet widely deployed, RFID technology has already been the target of some real-world attacks [46]. Ironically, RFID suffers from a number of security threats because of the core functionality of an RFID tag itself, that is to enable automatic item tracking. The security threats include:

- *Tag Cloning*: When queried by an RFID reader, an RFID tag emits a unique number called *Electronic Product Code* (EPC for short). This EPC number serves as product identity which points to an entry in a database of the back-end server. Unfortunately, this is an inherent security risk since we depend on the EPC number to recognize a product as genuine or fake. An attacker equipped with a compatible reader can scan many RFID tags to harvest a large number of EPC numbers. He then can produce RFID tags which emit exactly the same EPCs he has collected. We call this kind of tags *cloned tags*. The cloned tags can be attached on counterfeited items which should be recognized as genuine ones.

- *Privacy Invasion*: The core functionality of an RFID tag also raises privacy concern. As EPC number is unique, an attacker with a compatible reader can recognize and track RFID tags which leads to privacy invasion of people carrying tagged items.

- *Denial/Disruption of Service*: RFID technology is really useful when it is deployed at a very large scale. The hope is to attach to each and every item of human interest

an RFID tag. In this case, the infrastructure for an RFID system has to maintain and process a large amount of data. If a large number of fake tags and even malicious readers are deployed, computational resources can be abused and disruption of service may happen. Another form of denial-of-service attack is to physically interfere (*e.g.,* by using electro-magnetic jamming technique) the communication channel between tags and readers. However, while protocol-level denial-of-service attacks are possible at the broadest scale, physical attacks might be possible at a much smaller scale, *e.g.,* effective against a tag population of less than a hundred tags. Therefore, in this thesis, we only consider attacks and defenses at protocol level.

- *Location-based Attacks (Mafia Fraud/Terrorist Attacks)*: Wireless communication is inherently subject to location-based attacks including so-called *mafia fraud attack* and *terrorist attack* [17]. This type of attacks happen even if cryptographic protocols are used. Mafia fraud attack (sometimes referred to as distance fraud attack) is a man-in-the-middle relay attack. The attacker simply relays messages between two honest parties involved in a protocol and makes the two parties believe that they are in close proximity. The mafia fraud attack is specially effective against RFID because an RFID reader is supposed to scan only tags within its communication range. Terrorist attack is a more sophisticated variation of mafia fraud attack in a sense that an attacker can collaborate with a dishonest party involved in a protocol.

1.3 Security Requirements for an RFID System

In response to the security threats mentioned above, one should implement cryptographic protocols between RFID tag and reader such that it is infeasible for malicious parties to realize the security threats. The desirable security properties of a cryptographic protocol for RFID are described below.

- *Mutual Authentication between Tag and Reader/Back-end Server*: In order to EPC numbers from being harvested by malicious parties, reader-to-tag authentication must be provided. In addition, the server should not waste its computational resource on verifying and identifying fake tags. Therefore, RFID readers should also authenticate tags before forwarding legitimate tags to the server for identification.

- *Privacy Protection:* In order to prevent a tag from being tracked by malicious parties, it is not sufficient to avoid communicating the tag's EPC number in clear text.

Indeed, the information exchanged during different authentication sessions should not help a malicious party to trace a tag. We call such a property *unlinkability*. We refer to a protocol that provides both secure authentication and unlinkability as *a privacy-preserving authentication protocol*. A common approach to provide privacy-preserving authentication is to use pseudonym. More specifically, for each authentication session, a tag uses a different *temporary identity* called pseudonym to communicate with an RFID reader.

- *Forward Security*: As an RFID tag is generally not a tamper-proof device, it can be easily stolen and dissected to reveal secret information stored in the memory of the tag. Many authentication protocols for RFID including [55] have taken this threat into account by providing a security property called *forward security*. In the case of a privacy-preserving authentication protocol, forward security guarantees that all of authentication sessions of a tag happened before the tag's secret is revealed remain unlinkable. In other words, the piracy of the tag is protected up to the point of the loss of the secret information. A well-known method to achieve forward security is to update the secret key frequently (say, after every authentication session). Once a secret key is revealed, the previous authentication sessions that are associated with old and unknown secret keys are unlinkable. Updating secret keys regularly might also have positive impact on providing privacy-preserving as a tag possesses different keys during different authentication sessions. Unfortunately, updating the secret key interactively between a reader and a tag is often subject to de-synchronization of secret, *i.e.,* the attacker can cause the reader and the tag to posses different keys which makes future communication impossible.

- *Secure Key Exchange*: Wireless communication is vulnerable to eavesdropping. To prevent sensitive information like EPC and secret key from being eavesdropped, the information exchanged between a tag and a reader or back-end server should be encrypted. That leads to the need to establish a fresh session key for each interrogation session.

- *Secure Tag Location*: To defeat location-based attacks like mafia fraud attack, it must be possible for two parties involved in a protocol to measure (at least approximately) the distance between them. A common method is to use round-trip time of messages exchanged between two parties to estimate the distance. Brands and Chaum presented such a protocol which they called distance-bounding protocol in

[18].

- *Availability and Dependability:* Considering the huge amount of tags that would be live in a whole RFID ecosystem, a protocol designed for RFID should make it possible to filter out unwanted traffic as early as possible. The back-end server and the middle-ware layer should not be overwhelmed by the amount of illegitimate or unnecessary data. We also want an RFID tag to be correctly identified at the back-end server.

2. An Overview of Cryptographic Research

In this chapter, we briefly summarize some of fundamental concepts in cryptographic research. There are two lines of work in cryptographic research: construction of cryptographic schemes and definition of what we mean by a *secure* cryptographic scheme. Before going into discussing some of these works, we shall define three important concepts in cryptography: negligible function, one-way function and indistinguishability.

The term *negligible* is mentioned in almost all cryptographic papers. Roughly speaking, a function is negligible if it decreases so fast, that is faster than the inverse of any polynomial.

Definition 1 (Negligible Function)**.** *A function $f : \mathbb{N} \to \mathbb{R}$ is said to be a negligible function if for every polynomial in n, poly(n), there exists a positive integer k such that for all $n > k$, we have:*

$$|f(n)| < \frac{1}{poly(n)}$$

The use of polynomial in the above definition provides a convenient way to interpret security analysis of a cryptographic protocol that we will see later on. When evaluating security of a cryptographic scheme, it usually boils down to measuring the success probability of an adversary whose resource (memory size, number of computation steps, number of oracle queries. *etc*) is *polynomially* bounded. If the resulting probability is *negligible*, it implies that even if the adversary repeats the attack in polynomial number of times, the chance of success is still too small to matter. Another frequently used term in cryptographic papers is *infeasible*. By saying a certain task is infeasible, we mean that the task cannot be realized with an efficient (polynomial time) algorithm (and sometimes with an overwhelming probability of success).

Next, we shall define the central concept in cryptography as well as complexity (*i.e.,* the P=NP? question), that is the one-wayness property. Roughly speaking, a function is called one-way it it is easy to compute but hard to invert.

Definition 2 (One-way Function)**.** *A function $f : \{0,1\}^* \to \{0,1\}^*$ is said to be one-way if it satisfies the following conditions:*

1. *There is an efficient algorithm which computes $f(x)$ on input x.*

2. *Given $f(x)$ where x is randomly chosen, it is infeasible to find a pre-image of $f(x)$ with non-negligible probability of success.*

Another important concept that is related to one-way function is trapdoor one-way function. Let's think of scrambling data for example. The process to scramble the data should be easy. On the other hand, it should be hard to de-scramble the scrambled data for any illegitimate party. However, for the legitimate parties, de-scrambling data should not be infeasible.

Definition 3 (One-way Trapdoor Function)**.** *A function $f(x) : \{0,1\}^* \to \{0,1\}^*$ is said to be one-way trapdoor if it is an one-way function except that there exists a trapdoor such that if it is given, f can be efficiently inverted.*

The trapdoor information can be thought of as a secret information so that only parties know the secret information will be able to de-scramble the scrambled data.

Another central concept in cryptography is the notion of indistinguishability or more precisely computational indistinguishability (not visual nor physical indistinguishability).

Definition 4 (Indistinguishability)**.** *We say that two objects,* e.g., *two random processes, are computationally indistinguishable if there is no efficient algorithm to tell them apart with non-negligible probability of success.*

Now, some cryptographic primitives which shall be mentioned throughout this thesis will be introduced. Then, the task of defining the notion of security will be discussed.

2.1 Cryptographic Primitives

2.1.1 Pseudorandom Number Generator

Randomness plays an important role in computer science and even more so in cryptography. Random numbers are usually used to prevent the so-called replay attack. That is, randomness makes it impossible for an attacker to replay previous messages of legitimate parties. In practice, it appears infeasible to generate true random numbers. Hence, we refer to a procedure which generates near true random numbers as *Pseudorandom Number Generator* (or PRNG for short). For the rest of this thesis, when the term *random* is used, it actually means *pseudorandom* unless clear distinction is needed.

There are two approaches in generating pseudorandom numbers: the physical approach and the algorithmic approach. In the physical approach, random bits are collected from various physical random sources including noise from electronic circuit, atmospheric noise, light, human keystroke on keyboard, interrupts in desktop computer, *etc.* The disadvantages of the physical approach include the high cost of implementation and the difficulty to cope with interference and influence from malicious parties. The algorithmic approach does not collect random bits from any source but computes it in a deterministic manner. An algorithmic PRNG is defined as follows.

Definition 5 (Pseudorandom Number Generator (PRNG))**.** *An algorithmic PRNG is an efficient and deterministic algorithm which expands a n-bit random seed to l-bit string such that $l > n$ and the returned l-bit string is computationally indistinguishable from a randomly chosen l-bit string.*

It has been shown that if one-way function exists, then one can construct a PRNG satisfying the above definition based any one-way function. On the practical side, a popular design for an algorithmic PRNG is to employ a *Linear Feedback Shift Register* (LFSR). A LFSR is a shift register which receives input (1 bit at a time) as a linear function of its previous states. The initial state of the register is fed with a random seed. An example of a 4-bit LFSR is depicted in Fig. 2.1.1

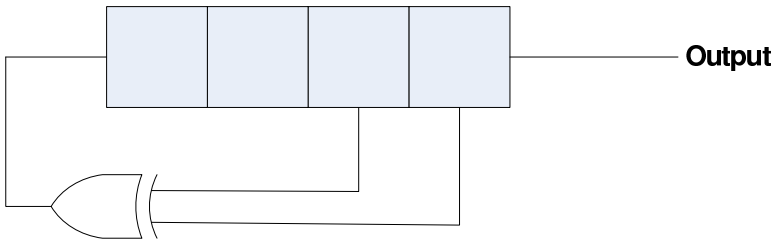

Figure 2.1: An Example of A 4-bit Linear Feedback Shift Register

While the use of LFSR provides highly efficient PRNGs, it has been shown that LFSR-based PRNGs do not achieve computational indistinguishability from the true random. One way to improve the quality of a LFSR-based PRNG is to replace the linear function with a non-linear one. Recent works on PRNG use *block cipher* and *cryptographic hash function*. We shall discuss block cipher and cryptographic hash function below.

2.1.2 Pseudorandom Function

Recall that a true random function takes any input and produces a random output. The idea behind a random function is that instead of providing a random seed (as in the case of a PRNG) whenever a random string is needed, one can give any input and get back a random string. A random function can be implemented as follows: pick a function at random from all possible functions $f : \{0,1\}^n \rightarrow \{0,1\}^n$. The output of f is random because f is chosen at random from $(2^n)^{2^n}$ possible choices. Unfortunately, the number of choices for F is too big. It is impossible to list all $(2^n)^{2^n}$ candidates for f. Therefore, we want something that mimics a true random function but is practical.

Definition 6 (Pseudorandom Function [78])**.** *A pseudorandom function is an efficient and deterministic algorithm which given a randomly chosen n-bit seed, s, and an n-bit argument x, returns a n-bit string, denoted $f_s(x)$, so that output of f_s is computationally indistinguishable from output of a true random function.*

It is well-known that one can construct a pseudorandom function from any PRNG. In practice, *block cipher* and *cryptographic hash function* are also used to build pseudorandom functions.

2.1.3 Cryptographic Hash Function

Hash functions are widely used searching algorithms in which each object is *hashed* into a bit string of fixed length called hash so that it is easy to look up the object given its hash. It turns out hash function is also a powerful tool in cryptography. However, cryptography needs hash functions with much stricter properties than the ones used in searching algorithms. A cryptographic hash function is defined as follows.

Definition 7 (Cryptographic Hash Function)**.** *A function $h : \{0,1\}^* \rightarrow \{0,1\}^n$ is called a cryptographic hash function if the following conditions are satisfied:*

1. *Efficiency: Given x, it is easy to evaluate $h(x)$.*

2. *Pre-image Resistance (One-wayness): For sufficiently large n (say, at least 128), it is infeasible to find x from $h(x)$.*

3. *2nd Pre-image Resistance: Given x_1, it is infeasible to find $x_2 \neq x_1$ such that $h(x_1) = h(x_2)$.*

4. *Collision-free: For sufficiently large n, it is infeasible to find any pair (x_1, x_2) such that $h(x_1) = h(x_2)$.*

It is easy to show that the collision-free property implies the 2nd pre-image resistance and one-wayness. Because of the collision-free property, $h(x)$ is sometimes referred to as *digital fingerprint* or *digest* of x. One typical application of cryptographic hash function is in creating a digital signature. Instead of signing a digital document of arbitrary size, one can sign the document's fingerprint whose size is fixed and small (usually less than 512 bits). Cryptographic hash functions are also widely used for checking the integrity of a message. For example, a message is sent via a network together with its corresponding hash. A receiver of the message can verify the integrity of the received message by checking if the received hash equals the hash of the received message.

Unfortunately, as cryptographic hash function is a special case of one-way functions, it is unclear whether such a function exists. Most of works on cryptographic hash function have focused on constructing functions that come close to have the last three properties of a cryptographic hash function. The most common design for current cryptographic hash functions is the Merkle-Damgard's construction which makes use of a compression function in a chaining manner [6]. A message which is first padded with binary encoding of the length of the message is divided into different blocks of equal size. The first block and an initialization vector (IV) are fed into the first round of compression. The output of the first round together with the second block are again compressed, *etc.* It was shown by Merkle and Damgard that if the compression function is collision-free then the resulting hash function is also collision-free. The construction is depicted in Fig. 2.1.3

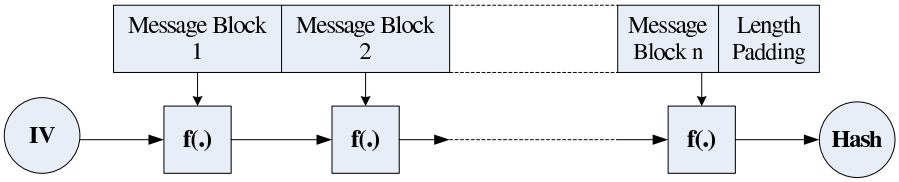

Figure 2.2: Merkle-Damgard's Construction of Cryptographic Hash Function

Popular cryptographic hash functions include MD-2/4/5 by Ronald Rivest; RIPEMD-128/160 by Dobbertin, Bosselaers and Preneel; SHA-1/256/512 by NIST/NSA. In Table 2.1, characteristics of several hash functions are summarized.

Recently, collisions have been found on most of cryptographic hash functions including MD-4, MD-5, SHA-0 and the reduced version of SHA-1 [23, 24, 28, 29, 30, 31, 33, 34].

Hash Function	Block Size (Bits)	Hash Size (Bits)	Year
MD-2	512	128	1989
MD-4	512	128	1990
MD-5	512	128	1992
RIPEMD-128	512	128	1996
RIPEMD-160	512	160	1996
SHA-1	512	160	1993
SHA-256	512	256	2000
SHA-512	1024	512	2000

Table 2.1: Popular Cryptographic Hash Functions

A competition has been organized to design a better hash function which will be named SHA-3. For the moment, SHA-1 (including its variants SHA-256 and SHA-512) is the only hash function that is recommended to use.

2.1.4 Block Ciphers

The need to keep sensitive information hidden from unwanted parties dates back to early human history. It is obvious that encryption receives the most attention in early as well as modern cryptographic research. Block cipher is a kind of algorithm for keeping sensitive information private to legitimate parties who should share a common secret key in advance. A block cipher should be designed in a way that anybody without the knowledge of the secret key cannot obtain any useful information about encrypted messages. A block cipher is formally defined below.

Definition 8 (Block Cipher)**.** *A block cipher consist of two efficient algorithms operated on a key space $\mathcal{K}$, a message space $\mathcal{M}$ and a ciphertext space $\mathcal{C}$ (It is usually the case that $\mathcal{M}$ is the same as $\mathcal{C}$). The two algorithms are defined as follows:*

1. *Encryption function $\mathcal{E} : \mathcal{K} \times \mathcal{M} \to \mathcal{C}$ which takes a secret key k, a plaintext m as the input and produces a ciphertext c.*

2. *Decryption algorithm $\mathcal{D} : \mathcal{K} \times \mathcal{C} \to \mathcal{M}$ which takes a secret key K, a ciphertext c as the input and produces a plaintext m.*

We say that a block cipher is correct if $\forall m \in \mathcal{M}$ and $k \in \mathcal{K}$, we have $\mathcal{D}(k, \mathcal{E}(k, m)) = m$.

Note that, we have not formally defined the requirement that a ciphertext c should not reveal any useful information about the corresponding plaintext m. We shall discuss

this issue in the security definition section later on. Nevertheless, an obvious requirement for a block cipher is that the size of the key and message spaces must be large enough so that exhaustive search is infeasible. Now, we would like to talk about the design of a block cipher. There are two major designs for a block cipher: Feistel's structure and Substitution-Permutation Network (SPN). The latter design is used by most modern block ciphers. The goal of permutation and substitution is to provide two key properties of a *good* cipher proposed earlier by Shannon including confusion and diffusion. An example of a 3-round SPN-based bock cipher is depicted in Fig. 2.1.4 where each S_i is a 4-bit substitution function (also called S-Box which replaces 4-bit input string with a different 4-bit output string), P is a permutation function and K_i is a scheduled key for the i-th round derived from the secret key via a key scheduling procedure.

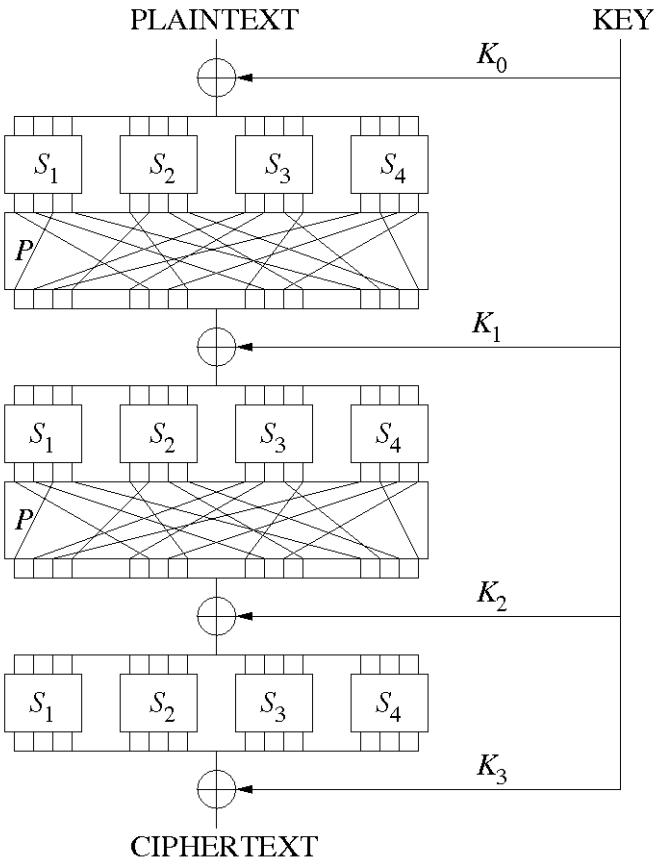

Figure 2.3: An Example of A 3-round SPN-based Block Cipher

When using a block cipher to encrypt a message longer than the size supported by the cipher, one can divide the message into smaller blocks and encrypt each block separately using one secret key. In this case, we say that the block cipher is operated in *Electronic Codebook* mode (or ECB mode for short). However, a message encrypted using ECB mode is vulnerable to re-ordering attack, that is a malicious party can rearrange the order of ciphertext without causing the failure in the decryption process. For example, if the message consists of two blocks, each of two digits: 1020. Then the malicious party can change the order of the ciphertext so that when the ciphertext is decrypted, the resulting plaintext is 2010. To prevent re-ordering of ciphertext and other sophisticated attacks, different modes of operation for block ciphers have been proposed. They include *Cipher-block Chaining* (CBC), *Cipher Feedback* (CFB), Output Feedback (OFB) and Counter (CTR) modes. The encryption and decryption in CBC mode are illustrated in Fig. 2.1.4 and Fig. 2.1.4, respectively.

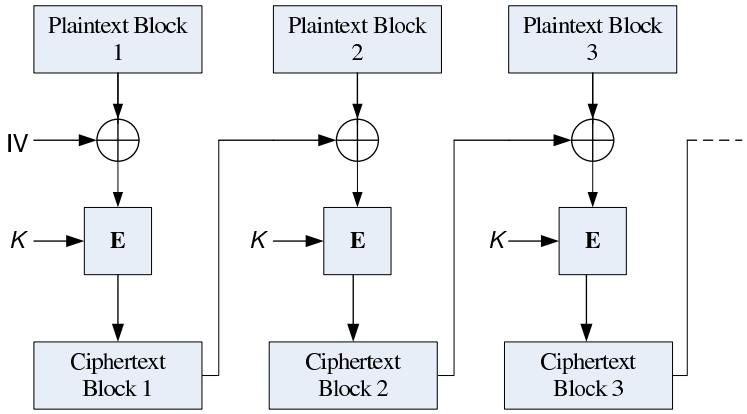

Figure 2.4: Encryption in CBC Mode

Popular block ciphers include DES and its hardened variant Triple-DES, Rijndael and Blowfish. Among these block ciphers, Rjndael is recommended to use as it was the winner of the *Advanced Encryption Standard* (AES) which has replaced the old counterpart *Data Encryption Standard* (DES). The characteristics of these block ciphers are summarized in Table 2.2.

2.1.5 Message Authentication Code

As mentioned earlier, cryptographic hash functions can provide integrity checking service by attaching the hash of a message to the message itself. However, since anyone can com-

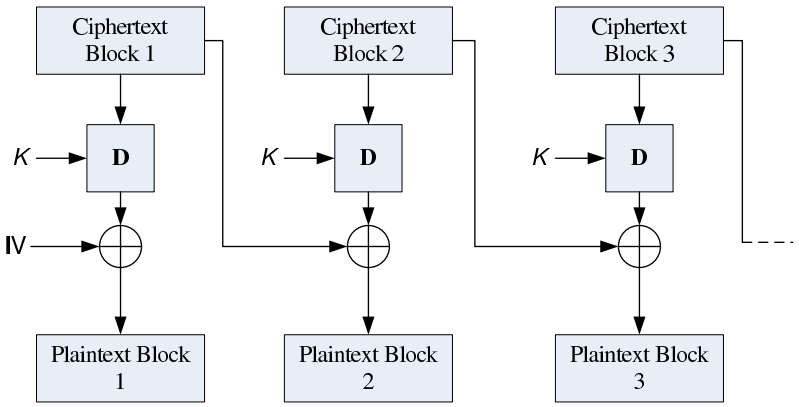

Figure 2.5: Decryption in CBC Mode

Block Cipher	**Key Size (Bits)**	**Block Size (Bits)**	**Number of Rounds**	**Year**
DES	56	64	16	1989
Triple-DES	56/112/168	64	48	1998
AES (Rinjdael)	128/192/256	128	10/12/14	1998
Blowfish	32-448	64	16	1993

Table 2.2: Popular Block Ciphers

pute the hash, it say nothing about the origin of the message. In many applications, a party wishes to know whether another party really send a message assuming the two parties share some secret information. A cryptographic scheme providing such a service is called *Message Authentication Code* (MAC for short). A MAC scheme is formally defined as follows:

Definition 9 (Message Authentication Code)**.** *A MAC scheme consists of two efficient algorithms called MAC-Sign and MAC-Verify. The two algorithms are defined as follows.*

1. *MAC-Sign:* $\mathcal{K} \times \mathcal{M} \to \Sigma$ *which takes a secret key* k *and a message* m *as the input and produces a so-called MAC value* σ *on the message* m*,* $\sigma = \textit{MAC-Sign}(k, m)$*.*

2. *MAC-Verify:* $\mathcal{K} \times \mathcal{M} \times \Sigma \to \{0, 1\}$ *which takes a secret key* k*, a message* m *and a MAC value* σ *as the input and produces either 0 or 1. If* $\textit{MAC-Verify}(k, m, \sigma) = 1$*, we say that* σ *is valid MAC on* m*. Otherwise,* σ *is an invalid MAC on* m*.*

The MAC scheme is correct if $\forall k \in \mathcal{K}$ *and* $m \in \mathcal{M}$*, we have* $\textit{MAC-Verify}(k, m, \textit{MAC-Sign}(k, m)) = 1$*.*

Once again, A formal security definition for a MAC scheme will be discussed later on. Note that, a MAC scheme does not qualify as a digital signature scheme as any party who knows the secret key can compute a MAC on any message (whereas in digital signature, we expect that a signature can be produced by one and only one party).

A common approach to design a MAC scheme is to use a block cipher operated in the CBC mode. However, because of the cost of a block cipher is much higher than that of a cryptographic function and some block ciphers are not royalty-free. A preferable approach to design a MAC scheme is to add a secret key to a cryptographic hash function so that a hash cannot be computed if the key is not given. A successful construction of a MAC scheme in this direction is the HMAC scheme due to Mihir Bellare, Ran Canetti and Hugo Krawczyk [11]. The signing algorithm of HMAC is defined as follows: $\text{HMAC-Sign}(k, m) = h((k \oplus \textit{opad})||h((k \oplus \textit{ipad}))||m))$ where h is a cryptographic hash function and *opad* and *ipad* are two constants ($\textit{opad}$ = 0x5c5c...5c and $\textit{ipad}$ = 0x3636...36 so that the bit lengths of *opad* and *ipad* are the same as that of the secret key k). The verifying algorithm of HMAC is straightforward: another party who knows the secret key k can compute the MAC himself and compare it against the received MAC value. HMAC is now standardized and recommended to use.

2.1.6 Public-Key Cryptography

Public-key cryptography (PKC for short) was invented Whitfield Diffie and Martin Hellman in 1976 [5]. The key paradigm of PKC is that each party in a system owns a pair of keys, one is public and the other is private. Since the initial idea, PKC has been extensively developed and used for many powerful cryptographic applications. In this Section, we overview two main applications of PKC including public-key encryption and digital signature.

Public Key Encryption

Public-key encryption (or asymmetric encryption to distinguish with the symmetric case of block ciphers) allows a party called Alice to send a message to Bob without sharing any secret key in advance. Instead, Alice uses Bob's public key to encrypt her message and then Bob uses his private key to decrypt Alice's message (in such a way that without Bob's private key, any message encrypted using Bob's public key cannot be decrypted). Public-key encryption appears to be superior than block ciphers. However, in practice, s public-key encryption scheme is much slower than block ciphers. Thus, a common scenario in practice is to use a public-key encryption scheme to transfer a secret key first and then using a block cipher with the shared secret key to encrypt messages that the two parties wish to exchange. A public-key encryption scheme is formally defined as follows.

Definition 10 (Public-Key Encryption)**.** *A public-key encryption scheme consist of three efficient algorithms PK-Keygen, PK-Encrypt and PK-Decrypt operated on a public key space $\mathcal{PK}$, a private key space $\mathcal{SK}$, a message space $\mathcal{M}$ and a cipphertext space $\mathcal{C}$. The three algorithms are defined as follows:*

1. *The key generation algorithm PK-Keygen: $\mathbb{N} \rightarrow \mathcal{PK} \times \mathcal{SK}$ which takes a security parameter $k \in \mathbb{N}$ (*e.g., *the bit length of the public and private keys) as the input and returns a (public key, private key) pair for a party which is denoted as $(pk, sk) \in \mathcal{PK} \times \mathcal{SK}$.*
2. *The encryption algorithm PK-Encrypt: $\mathcal{PK} \times \mathcal{M} \rightarrow \mathcal{C}$ which takes a party's public key pk, a plaintext m as the input and produces the ciphertext c.*
3. *The decryption algorithm PK-decrypt: $\mathcal{SK} \times \mathcal{C} \rightarrow \mathcal{M}$ takes a party's private key sk, a ciphertext c as the input and produces a plaintext m.*

We say that a public-key encryption scheme is correct if $\forall m \in \mathcal{M}$ and $\forall (pk, sk) \leftarrow$ PK-Keygen(.), we have PK-Decrypt(sk, PK-Encrypt(pk, m)) = m.

The first public-key encryption scheme was proposed by Ron Rivest, Adi Shamir and Leonard Adelman and thus named RSA. The scheme uses modular exponentiation of large integer. The three algorithms of RSA are described below:

1. Key generation: chooses two distinct random prime numbers p and q of the same bit length. Let $n = pq$ the Euler's totient function $\phi(n) = (p-1)(q-1)$. The returned public key includes n and an integer e such that $1 < e < \phi(n)$ and the corresponding secret key includes n and an integer d such that $ed \equiv 1 \pmod{\phi(n)}$.

2. Encryption: given a message m encoded as an integer such that $1 < m < n$ and a public key $pk = (n, e)$, the ciphertext c is computed by $m^e \pmod{n}$.

3. Decryption: given a ciphertext c and a private key $sk = (n, d)$, the plaintext m is computed by $c^d \pmod{n}$

RSA encryption is correct because $c^d \equiv (m^e)^d \equiv m^{ed} \equiv m^{1+k\phi(n)} \equiv m \pmod{n}$ (for k is some integer and note that $m^{\phi(n)} \equiv 1 \pmod{n}$). The recommended security parameter for RSA scheme is that the bit length of n should be at least 1024 bits. In practice, before a message m is encrypted, it should go through a padding process to improve the security. One of such padding schemes is called *Optimal Asymmetric Encryption Padding* (OAEP) proposed by Mihir Bellare and Philip Rogaway [10]. The combination of RSA encryption and OAEP is known as RSA-OAEP and is standardized in the RSA's public-key cryptography standard No. 1 (PKCS#1) [4]. Other popular public-key encryption schemes include ElGamal's scheme, Rabin's scheme, Cramer-Shoup's scheme and Paillier's scheme.

Digital Signature

Digital signature is perhaps the most powerful application of PKC. The use of two keys for each party enables one to produce a digital signature that resembles a real signature in many aspects including the impossibility to forge a valid signature (also known as unforgeability) and non-repudiation. In addition, digital signature can be used to provide integrity checking service like a MAC scheme. To describe a PKC-based digital signature scheme roughly, Alice uses her own private key to sign a digital message and Bob can verify Alice's signature by using Alice's public key. Since only Alice knows her own private

key, nobody else can sign any message on behalf of Alice. A (public-key) digital signature scheme is formally defined as follows:

Definition 11 (Digital Signature)**.** *A digital signature scheme consist of three efficient algorithms DS-Keygen, DS-Sign and DS-Verify operated on a public space $\mathcal{PK}$, a private key space $\mathcal{SK}$, a message space $\mathcal{M}$, a signature space Σ. The three algorithms are defined as follows:*

1. *The key generation algorithm DS-Keygen: $\mathbb{N} \to \mathcal{PK} \times \mathcal{SK}$ which takes a security parameter $k \in \mathbb{N}$ (*e.g., *the bit length of the public and private keys) as the input and returns a (public key, private key) pair for a party which is denoted as $(pk, sk) \in \mathcal{PK} \times \mathcal{SK}$.*

2. *The signing algorithm DS-Sign: $\mathcal{PK} \times \mathcal{M} \to \Sigma$ which takes a party's private key $sk \in \mathcal{SK}$, a message $m \in \mathcal{M}$ as the input and produces the a signature on m called $\sigma \in \Sigma$.*

3. *The verifying algorithm DS-verify: $\mathcal{PK} \times \mathcal{M} \times \Sigma \to \{0, 1\}$ which takes a party's public key $pk \in \mathcal{PK}$, a message $m \in \mathcal{M}$ and a signature $\sigma \in \Sigma$ as the input and returns either 1 or o to indicate whether σ is valid signature on m (with respect to the public key pk).*

We say that a digital signature scheme is correct if $\forall m$ and $\forall (pk, sk) \leftarrow$ DS-Keygen(.), we have DS-Verify(pk, m, DS-Sign(sk, m)) = 1.

The RSA encryption scheme can turn into a digital signature scheme by using the decryption algorithm as the signing algorithm and the encryption algorithm as the verifying algorithm. In particular, Alice who owns a public key $pk = (n, e)$ and a private key $sk = (n, d)$ can sign a message m by computing $\sigma \equiv h(m)^d \pmod{n}$ where h is a cryptographic hash function. Bob can verify Alice's signature by checking if $\sigma^e \equiv h(m) \pmod{n}$. Similar to RSA encryption, a message m needs to be padded before signing to improve security. A padding scheme for RSA digital signature adopted in PKCS#1 standard is called *Probabilistic Signature Scheme* (PSS) which was proposed by Mihir Bellare and Philip Rogaway [12]. Other popular digital signature schemes include Digital Signature Algorithm (DSA) and its elliptic-curve-based variant ECDSA, ElGamal's signature scheme, Rabin's signature scheme and Schnorr's signature scheme.

Public Key Infrastructure

A problem with PKC is that a party's public key has to be in the public domain. However, there is no way that one can be sure of the actual identity of an owner of a public key just by looking at the public key itself. One way overcome this problem is to have a trusted party to approve a relationship between a public key and a party's real identity. The trusted party uses a digital signature scheme to sign public keys of its registered parties. This kind of trusted party is called *Certificate Authority* (CA) and its signature is called certificate. In practice, we can have many CAs and they can be organized hierarchically. We refer to a system of hierarchical CAs as a *Public Key Infrastructure*.

2.2 Security Definitions and Provable Security

In the previous section, we talked about different cryptographic primitives and different properties that we want from them. we also mentioned some intuitions and guidelines that one should follow in order to achieve the kind of properties we expect. However, we have not discussed how a property of a cryptographic scheme can be verified. It turns out that verification of cryptographic properties, or a more commonly used term *security analysis*, is as important as designing a cryptographic scheme itself. We usually say that a cryptographic scheme or more generally an information system is *secure* if it meets all of desirable (security) properties. In order to rigorously analyze security of a cryptographic protocol, we first need to quantify the term *security* so that a careful measurement of the *security quantity* would tell us whether a cryptographic scheme satisfies a desirable property or not. In this section, we will discuss how security of a cryptographic scheme is understood and quantified.

2.2.1 Definition of Security

Since it would take too much time and space to cover definition of security for all of cryptographic primitives mentioned in the previous section, we will discuss here only security definition for encryption schemes which include both block cipher and public-key encryption schemes. This security definition will be used in the following chapters. An often used term in cryptographic papers is *security model*. Generally, a security model refer a general settings through which a security definition can be given. The two terms, security model and security definition, can be used interchangeably without any confusion because a security model often dictates how a security definition is given and vice versa.

For an encryption scheme, the security goal is to preserve the confidentiality of an encrypted message. A straightforward interpretation of *confidentiality* is that any given ciphertext c which is encrypted using any encryption key does not leak any useful information about the corresponding plaintext m, not even one bit. More analytically speaking, the probability of getting any useful information about the plaintext is negligible (or even better, it is exactly the same as trying to decrypt c with all possible decryption keys in which case the encryption scheme in question achieves the so-called perfect secrecy). Although, that interpretation gives a better understanding of what a secure encryption scheme is, it still does not help much in quantifying the security since there are too many possible encryption/decryption keys and (plaintext, ciphertext) pairs to account for. A solution for that problem is to pick an encryption key at random (in case of bock cipher, encryption key is the same as decryption key and in case of public-key encryption, the decryption key is often derived from the encryption key), then analyze the encryption scheme in question with that one randomly chosen key[1]. In addition, we should also pick a few random (plaintext, ciphertext) pairs to verify the confidentiality property instead of checking the whole message and ciphertext spaces.

Note that, no assumption about the party interested in breaking the security, that is the adversary, has been made. It is obvious that how much information and computational resources available to the adversary greatly affect how successful in breaking the security the adversary would be. As mentioned earlier, the computational resources available to the adversary should be polynomially bounded. In other words, the adversary should be a feasible computational machine. Regarding information for the adversary (beside the internals of the encryption scheme but not the decryption key), it should help the adversary study the behaviors of the encryption scheme in order to violate the confidentiality (the best scenario for the adversary is to discover the decryption key). For an encryption scheme, the following sources of information are available for the adversary (in an increasing order of influence):

- *Ciphertext-only*: In this case, the adversary can obtain only ciphertexts without knowing what plaintexts are used to produce the ciphertexts. The adversary equipped only with this source of information is called *passive adversary*.

- *Known-plaintext*: In this case, the adversary is given both ciphertexts and the corresponding plaintexts.

[1]There might be a pitfall here because we have witnessed that some encryption schemes like DES, RC4 and Blowfish have a few weak keys.

- *Chosen-plaintext*: In this case, the adversary can request ciphertexts of plaintexts that he chooses.

- *Chosen-ciphertext*: In this case, the adversary can request plaintexts of ciphertexts that he chooses. In other words, the adversary is given access to a decryption machine so that he can submit any ciphertext to get the corresponding plaintext.

- *Adaptive chosen-ciphertext*: In this case, the adversary can choose the ciphertexts to be decrypted in an adaptive manner. That is, the adversary can chooses some ciphertexts to be decrypted. After he gets the resulting plaintexts, he can select other sets of ciphertexts to be decrypted.

We also refer to different sources of information described above as different types of attacks that the adversary can perform, *e.g.,* ciphertext-only attack (COA), known-plaintext attack (KPA), chosen-plaintext attack (CPA), chosen-ciphertext attack (CCA) and adaptive chosen-ciphertext attack (CCA2). A question one might ask now is how the adversary can obtain those information and what role these different information sources play in a security analysis. The most well-known method to address that question is to have a game (also called an experiment) between a challenger and the adversary. The challenger is the one to pick an encryption key at random. In case of public-key encryption schemes, the challenger also computes the corresponding decryption key according to the key generation procedure PK-Keygen(.). The encryption key is public and therefore is given to the adversary. Having knowledge of the encryption and decryption keys, the challenger can simulate all information sources described above for the adversary. In particular, the challenger constructs two machines, an encryption one and a decryption one. These two machines are called oracles. The encryption oracle takes a plaintext as its input and return the corresponding ciphertext encrypted using the encryption key chosen by the challenger. Similarly, the decryption oracle decrypts the input using the decryption key picked by the challenger. The interaction between the adversary and these two oracles dictates what kind of attacks the adversary performs. For example, CPA can be simulated by giving the adversary's access to the encryption oracle, but not the decryption oracle. Whereas, CCA can be simulated by giving the adversary's access to the decryption oracle

after the adversary has selected his ciphertexts. Note that, in case of public-key encryption, there is no need to provide an encryption oracle as the adversary can do it himself by using the public key. The challenger is also one to pick a few (plaintext, ciphertext) pairs at random. These pairs will be used to challenge the adversary at the end of the game. The adversary wins the game if he can violate the confidentiality of the challenged (plaintext, ciphertext) pairs.

Now that we have settled on fixing one random key and specifying different types of attack by the adversary in a game between a challenger and the adversary to analyze the security, let's come back to the interpretation of confidentiality. We want to quantify the hardness of learning anything useful about the plaintext from the corresponding ciphertext. It would be great if we can measure the hardness directly, however there some difficulties in doing so as follows:

- The relationship between the ciphertext and the plaintext can be very complex which makes it difficult to measure the hardness to inverse the encryption accurately.

- We do not want the ciphertext to leak any bit of information. However, not all bits of the ciphertext might be equally hard to inverse.

The above difficulties are probably the reason why the security of most block ciphers is usually analyzed in terms of its resistance against some known specific attacks (*e.g.,* differential and linear cryptanalysis). Nevertheless, it is worthy to measure the security against more general and even unknown attacks. Goldwasser and Micali were the first to introduce a way to quantify the security of a public-key encryption scheme by defining two security notions called *GM-security* (also known as polynomial security or ciphertext indistinguishability) and *semantic security* [7]. The two security notions are actually equivalent. Therefore, we will present here the definition for GM-security which is more commonly used in the literature. Roughly speaking, an encryption scheme achieves GM-security if given a ciphertext which is the result of encrypting one of two randomly chosen messages, an adversary cannot decide with non-negligible probability which message is used to produce the ciphertext. The job of a security analysis is to measure that probability and show that it is negligible. The security notion GM-security under chosen-plaintext attack (denoted as IND-CPA) is formally defined via the following game between a challenger and the adversary $\mathcal{A}$:

1. The challenger picks an encryption key at random and constructs the encryption oracle $\mathcal{E}(.)$.

2. $\mathcal{A}$ queries $\mathcal{E}(.)$ with plaintexts of his choice.

3. $\mathcal{A}$ generates two plaintexts m_0 and m_1, and gives the two messages to the challenger. The challenger pick a random bit i and computes $c = \mathcal{E}(m_i)$. c is then given to $\mathcal{A}$.

4. $\mathcal{A}$ outputs his guess for i as i'.

5. $\mathcal{A}$ wins the game if $i = i'$.

Definition 12 (GM-Security (IND-CPA))**.** *An encryption scheme is said to be GM-secure under chosen-plaintext attack if the probability that any polynomial-bounded (polynomial in running time, polynomial in memory and polynomial in number of queries to available oracles) algorithm $\mathcal{A}$ wins the above game is negligible,* i.e., *the difference between guessing i at random and computing i correctly is negligible.*

Similar to the above definition, one can define stronger security notions for an encryption scheme include GM-security against chosen-plaintext attack (IND-CCA) and GM-security against adaptive chosen-plaintext attack (IND-CCA2)[2]. Cramer-Shoup's scheme and RSA-OAEP are two public-key encryption schemes that are known to achieve IND-CCA2. Note that, in case of the IND-CCA2 notion, the adversary is given the decryption oracle even after seeing its challenge, the bit i. However, the adversary might not use c to query the decryption oracle because that would allow the adversary to know which message is used to produce c. It is also important that to note that in order to achieve IND-CPA, IND-CCA or IND-CCA2, an encryption scheme has to have a probabilistic encryption function. That is a single message when encrypted twice will produce two different ciphertexts. The reason that randomized encryption is required is because the adversary can query m_0 and m_1 to the encryption oracle even before seeing the challenge i and then easily check whether c is encrypted using m_0 or m_1.

[2]Stronger notion means that an IND-CCA2 encryption scheme is also IND-CCA, an IND-CCA encryption scheme is also IND-CPA, *etc.*

2.2.2 Computational Hard Problems and Provable Security

Understanding the true meaning of the term *secure* is a subtle process and often takes a lot of time to mature. However, measuring the security quantity, *i.e.,* the success probability of the adversary violating the security, is no less challenging. This is not a surprise since many cryptographic schemes are either directly or indirectly related to the assumption that one-way function exists. Unfortunately, it is still unknown whether one-way function really exists unless P=NP is proved to be wrong. As a result, it appears impossible to measure the success probability of the adversary directly.

In response to the problem, a so-called *reduction* paradigm was proposed. That is, the success probability of the adversary is measured with respect to the probability of solving a hard computational problem. Since the probability of solving a hard problem is believed to be negligible, it is hoped that the probability of success of the adversary is also negligible. In a security analysis, one should construct a polynomial time reduction algorithm which uses the adversary as a subroutine to solve the hard problem (at least with a non-negligible probability of success). Some of computational hard problems commonly used in cryptographic include integer factoring problem, discrete logarithm problem and Diffie-Hellman problem. When breaking the security of a cryptographic scheme is shown to be as hard as solving a hard problem, we usually say that the cryptographic scheme is provably secure. The term *provable security* is quite controversial since the security of the protocol is not actually proved but rather related to something else. Furthermore, some of well-known cryptographic schemes which were shown to be provably secure before have been shown to be otherwise. Nevertheless, reductionism and provable security have contributed a lot of important works which are being used in real-world applications.

Another aspect of the reduction paradigm is to relate breaking security of one cryptographic protocol to that of another protocol. In some cases, this approach might be easier to reduce breaking the security directly to solving some hard problems. However, one must be careful to deal with the *compatibility* of security goals of two cryptographic schemes. For example, it would be meaningless if we want to show that an encryption scheme achieves IND-CCA by reducing its security to another encryption scheme which achieves only IND-CPA. Another application of reduction from one scheme to another is to prove the possibility and impossibility of certain cryptographic tasks. To show that a cryptographic task is theoretically possible, one can use a cryptographic primitive as a building block (often in a black-box manner, that is we do not consider the internals of the primitive as long as it is known to achieve its desirable security properties) and construct a cryptographic scheme satisfying the cryptographic task. Then, it should be shown

that breaking the security of the cryptographic scheme can be efficiently reduced to that of the cryptographic primitive. On the other hand, a similar technique can be used to show that a certain cryptographic task requires the existence of some cryptographic primitives. If one of such cryptographic primitive can not be realized, then we know that it is impossible to realize the cryptographic task (in a sense that we cannot provide provable security for any construction realizing the cryptographic task).

3. Previous Cryptographic Protocols for RFID

In this chapter, we summarize some of the most representative works in RFID security. These works include construction of cryptographic protocols to counter the security threats presented in chapter 1 as well as known issues of those protocols. Other previous works that are related to the main content of this thesis will be discussed later in respective chapters.

One important characteristic of cryptographic protocols for RFID tags is that they should have very low requirement on computational resource. It is because the cost of an RFID tag (especially, a passive one) should be very low. As a result, the computational functionalities built-in a tag is very minimal. Even though there are on-going works to implement PKC-based cryptographic primitives (especially, the elliptic curve-based primitives), these types of cryptographic tools are still beyond the capacity of current low-cost RFID tags, at least in a foreseeable future. Instead, cryptographic protocols for RFID tags should only make use of so-called *lightweight cryptographic primitives* like PRNG, PRF, cryptographic hash functions and probaly block ciphers. We usually call cryptographic protocols for RFID tags as lightweight protocols. To describe the lightweight cryptographic protocols for RFID tags for the rest of this thesis, we will use the notations summarized in Table 3.1.

3.1 Security Features in Gen-2 Specification

The industry recognized the security threats to RFID very early. In this section, we will talk about security features in the most notable industrial standard for RFID tags at the moment, the Gen-2 specification by EPCglobal Inc [3].

3.1.1 Reader-to-Tag and Tag-to-Reader Authentication

The Gen-2 specification does not provide true Tag-to-Reader and Reader-to-Tag authentication. A Gen-2-compliant RFID tag simply backscatters its EPC number once being queried by a compatible reader so that the tag can be identified later at the back-end server. Unfortunately, this clearly leaves Gen-2 tags vulnerable to cloning threats since any compatible reader can harvest EPC numbers. A Gen-2-compliant reader is required

Notation	Description
Ber_η	A Bernoulli distribution with expected value η
D	Tag database at the back-end server.
$G(.), H(.)$	Cryptographic Hash Functions
$f(.)$	Pseudorandom Function
K_i	Secret key of tag $\mathcal{T}_i$
$\text{MAC}_K[.]$	Message authentication code with secret key K
P	A co-existence proof of multiple tags
$\mathcal{R}$	Reader
$\text{SK}_K[.]$	Symmetric encryption with secret key K
TS	Timestamp
$\mathcal{T}_i$	An RFID tag
$\mathcal{V}$	Verifier (Back-end Database)

Table 3.1: Notations

to be authenticated only when it needs to read or write directly from or to a tag's memory. To do so, a reader and a tag share a common 32-bit secret key (called *Access Password*). The reader-to-tag authentication protocol is implemented as follows:

1. The tag must already be selected and identified (*Acknowledged* state). A reader starts by sending a request to the tag (Req_RN command).

2. The Tag responses with 16-bit random number $RN16$.

3. The reader takes the first 16 bits of the access password (MSB first, the second half of access password is used when the reader needs to access the tag again), denoted as $APwd16$, and computes its authentication token as $t = APwd16 \oplus RN16$.

4. Once receiving the reader's authentication token t, the tag computes $APwd16' = t \oplus RN16$. If $APwd16'$ does not match with the tag's version of the access password, the tag rejects the reader. Otherwise, the reader is successfully authenticated (*Open* or *Secured* state).

The above protocol is clearly not secure against eavesdropping, *i.e.,* passive adversaries. An eavesdropper can listen to the communication channel between the tag and the reader and collect t and $RN16$. A half of the access password is then revealed by computing $t \oplus RN16$. The Gen-2 specification recommends that reading and writing to a tag's memory should be done in a physically secure location (so that eavesdropping is not possible).

However, the assumption that the protocol is carried out in a secure location is strong. Indeed, if reading and writing can be done in a secure location, there is no need to implement any authentication.

3.1.2 Privacy Protection by Disabling Tags

The Gen-2 specification proposes a rather conservative method to provide privacy protection, that is to permanently disable a tag, *e.g.*, at the point-of-sale in a supermarket. A Gen-2-compliant tag can be *killed* after receiving a kill command. A reader-to-tag authentication protocol similar to the one described above must be successfully completed before the tag accepts the kill command. Indeed, the authentication protocol is carried out twice, each using one half of a 32-bit kill password.

While disabling a tag is obviously an effective countermeasure against illegal tracking, it is arguably over-killed. In many scenarios like tracking animal, smart home appliances, *etc*, a tag should not be permanently disabled. Furthermore, in case of supply chain management, a tag is still likely helpful in many ways after the item is purchased (*e.g.*, for warranty purpose).

3.2 Ohkubo-Suzuki-Kinishita Protocol

One of the most famous protocols for RFID is the Ohkubo-Suzuki-Kinishita protocol [21] which has generated a significant number of followed-up papers. The protocol assumes that a tag can compute two cryptographic hash functions, $G(.)$ and $H(.)$. A tag i is given an initial EPC number s_i^1 which is also stored in the database at the back-end server. After each interrogation session, the EPC number is updated in a hash chaining fashion, that is $s_i^{k+1} = H(s_i^k)$. The goals of updating the EPC numbers are two-fold:

- To provide privacy protection by using a different EPC number in each authentication session.

- To provide forward-security as it is infeasible to compute previous EPC number from the current EPC number due to the pre-image resistance property of a cryptographic hash function.

During the k-th authentication session, a tag computes its authentication token r_i^k as the hash of its current EPC number, *i.e.,* $r_i^k = G(s_i^k)$. To verify a tag, the server starts

from the initial EPC of all tags in the database and compute $G(s_i^1)$, $G(s_i^2)$, $\cdots$, $G(s_i^k)$ for $i = 1, 2, \cdots, n$ until a match is found. The Ohkubo-Suzuki-Kinishita protocol provides privacy-preserving, forward security and tag-to-server authentication. However, it does not provide server-to-tag authentication. In addition, the server has to go through the whole tag database and compute the hash chains to identify a tag. This makes the server an attractive target for denial-of-service attacks.

3.3 Distance-bounding Protocols for RFID

RFID protocols are inherently insecure against mafia fraud attack which was suggested by [17]. It does not matter what type of cryptographic protocols is used, an attacker can always relay messages between a reader and a tag which is not in the communication range of the reader (and therefore is not supposed to be scanned). The attack is illustrated in Fig. 3.1.

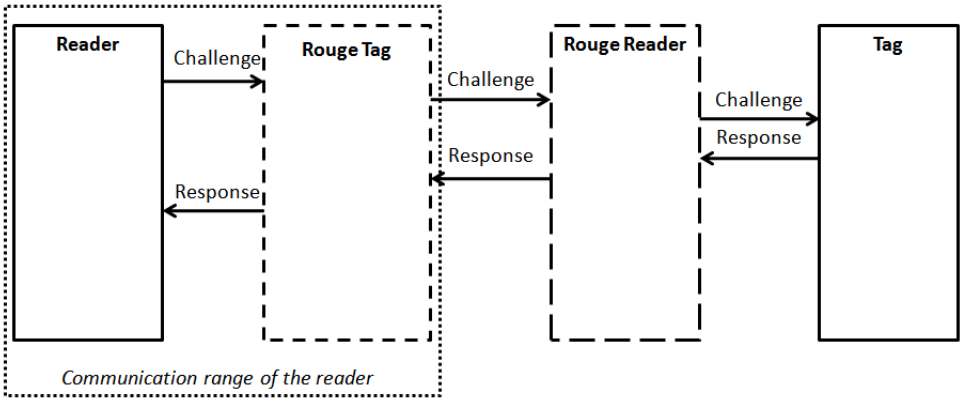

Figure 3.1: Mafia Fraud Attack on RFID Protocols

Brands and Chaum were the first to propose a practical countermeasure against the mafia fraud attack [18]. Since the mafia fraud attack is about faking the location, it is necessary to verify the location of each party involved in a protocol. However, there is no way to measure the distance between two autonomous parties. Therefore, Brands and Chaum suggested that round-trip time can be used to approximately measure the distance. While using round-trip time is not a new idea, Brands and Chaum pointed out that the round-trip time should be as short as possible and the measurement should be repeated multiple times to improve accuracy. Brands and Chaum called their countermea-

sure distance-bounding protocols.

Hancke-Kuhn's distance-bounding protocol is the first protocol which addresses mafia-fraud-attack against RFID protocols. The key idea is to repeat a simple authentication step multiple times so that each step can be complete in a very short time. Let t_{max} be the maximum time taken by one simple authentication step. A tag is accepted only if each simple authentication step completes successfully and within t_{max} amount of time, *i.e.*, $\Delta t_j \leq t_{max}$. The protocol between a tag $\mathcal{T}_i$ and a reader $\mathcal{R}$ is illustrated in Fig. 3.2.

$\mathcal{R}(K_i)$		$\mathcal{T}_i(K_i)$
$N_1 \in_R \{0,1\}^l$	$\xrightarrow{N_1}$	
	$\xleftarrow{N_2}$	$N_2 \in_R \{0,1\}^l$
	$\{H\}^{2n} = f(K_i, N_1, N_2)$	
	$\{v^0\} = H_1 \|\| H_2 \|\| \cdots \|\| H_n$	
	$\{v^1\} = H_{n+1} \|\| H_{n+2} \|\| \cdots \|\| H_{2n}$	
	for $j = 1$ **to** n **do**	
$C_j \in \{0,1\}$		
Start clock	$\xrightarrow{C_j}$	
		$R_j = v_j^{C_j}$
Stop clock	$\xleftarrow{R_j}$	
Verify R_j		
Verify $\Delta t_j \leq t_{max}$		

Figure 3.2: Hancke-Kuhn's Distance-bounding Protocol

Note that, recently Chandran *et al.* showed that it is impossible to securely measure the distance between two autonomous parties in any manner. Their result confirms that the use of distance-bounding protocols only provide practical defense against mafia fraud attack.

4. Security Definitions for Cryptographic Protocols for RFID Tags

A proper security definition is essential to understand and analyze the security of a cryptographic scheme and it is no exception to a cryptographic protocol for RFID tags. The most important part of a security model is to rigourously define what we means by a *secure system*. In case of RFID, we need to define secure mutual authentication, privacy-preserving, forward security and secure key exchange so that the definition correctly captures our intuition about security properties of a secure protocol for RFID tags. We briefly summarize two security models for RFID below.

4.1 Security Definition for Authentication Protocol

Authentication is not a new kind of cryptographic protocol. In fact, authentication protocols have been studied extensively resulting in a lot of concrete constructions as well as a mature security definition. We review here the security definition for a secure authentication protocol for two parties sharing a common secret key [25]. The definition is certainly applicable to cryptographic protocols for RFID that provide only one-way authentication like the HB^+ protocol. As usually, one should define a set of oracles which simulate behaviors of two parties involved in an authentication protocol.

- The reader oracle $\mathcal{R}(.)$ simulates the behaviors of a reader. Since the reader wishes to authenticate a tag, $\mathcal{R}(.)$ generally takes no input and simply returns a challenge.

- The tag oracle $\mathcal{T}(.)$ simulates the behaviors of a tag. Generally, $\mathcal{T}(.)$ takes the reader's challenge as the input and returns the tag's response.

- The result oracle result(.) determines whether a tag is successfully authenticated by a reader or not. This oracle should take all messages exchanged between the reader and the tag as its input and return either 1 or 0 where 1 indicates authentication is successful and 0 means otherwise.

Again, different types of attacks can be classified based on how an adversary interacts with the above oracles.

- Passive attack: in this type of attack, the adversary is given access to only $\mathcal{R}(.)$.

- Active attack: the adversary is given access to both $\mathcal{R}(.)$ and $\mathcal{T}(.)$.

- Man-in-the-middle attack: the adversary is given access to all three of above oracles.

As we will point out in chapter 5, an authentication HB^+ protocol is shown to be not secure against man-in-the-middle attack. Indeed, the access to the result oracle result(.) plays an important role in the success of of the attack since it let the adversary know whether a modification to the reader's challenge a affects the outcome of the verification. On the other hand, HB^+ is provably secure against active attacks. This emphasizes the point we want to make here that the availability of additional information is very important to the adversary. In [25], the authors referred to security against active attack and man-in-the-middle attack as that in *detection security model* and *prevention security model*, respectively.

For an authentication protocol, the goal of the adversary is to impersonate a tag. More specifically, the adversary should compute an alternative secret key such that this key can used to be successfully authenticated to the reader. An authentication protocol is said to be secure if the success probability of the adversary is negligible. Let $\mathcal{O} \subset \{\mathcal{R}(.), \mathcal{T}(.), \text{result}(.)\}$ be the set of oracles available to the adversary, the security definition for an authentication protocol can be given via the following game between a challenger and an adversary:

1. The challenger picks a secret key K at random and set up oracles in $\mathcal{O}$ for the adversary.

2. The adversary interacts with oracles to collect information. This is called the querying phase.

3. To prepare for the challenge phase, the challenger gets a fresh challenge c from the reader oracle $\mathcal{R}(.)$. In case $\mathcal{R}(.)$ requires some input, the challenger should collects

that input from the adversary. Then, c is given to the adversary.

4. The adversary can still interact with oracles in $\mathcal{O}$ except that using c to query the tag oracle is prohibited.

5. Finally, the adversary outputs a secret key K' and build a tag oracle called $\mathcal{T}'(.)$. The adversary wins the game if $\text{result}(c, \mathcal{T}'(c)) = 1$. In other words, the adversary can use K' to impersonate the tag successfully.

Definition 13 (Secure Authentication)**.** *An authentication protocol is said to be secure under certain attacks specified by oracles available in $\mathcal{O}$ if the probability that any polynomial time adversary $\mathcal{A}$ wins the above game is negligible.*

4.2 Vaudenay's Security Model for RFID

Vaudenay presented an RFID-specific security model for RFID in [57]. The Vaudenay's model is a classical type of a security model in a sense that a number of oracles to provide information for the adversary are specified and the security is defined via a game between a challenger and the adversary. In the Vaudenay's model, there are one reader (*i.e.,* the back-end server and the readers are seen as a single entity) and a tag population. An RFID authentication protocol is viewed as a collection of the following algorithms:

- SetupReader(.) is an efficient algorithm which takes a security parameter s as the input and returns a public key K_P and a secret key K_S for the reader.

- SetupTag(.) is an efficient algorithm which takes a security parameter s, the reader's public key K_P and an object identity ID as the input and returns a secret key and the initial internal state for a tag.

- An efficient two-party protocol Π between a tag and a reader such that at the end of the protocol (assuming that the reader has been set up properly and tag population has been created) the reader outputs either $\perp$ or ID of the tag.

Definition 14 (Correctness of an RFID Authentication Protocol)**.** *An RFID authentication protocol is said to be correct if, for a negligible probability, the reader's output is* $\perp$ *and the tag is illegitimate, or* ID *and the tag* ID *is legitimate.*

In order to provide a realistic security definition for RFID protocols, Vaudenay observed that in practice, it is impossible for the adversary to access to all tags available at once. Therefore, in the security model, the adversary should be allowed to access to some tags at once. Vaudenay took this observation into account by introducing two special oracles called DrawTag(.) and FreeTag(.). All oracles are defined below:

- CreateTag(b, ID): This oracle allows the adversary to create either a legitimate tag using the SetupTag(.) algorithm ($b = 1$) or a illegitimate one ($b = 0$). The resulting tag has an identity ID.

- DrawTag(D): This oracle draws n tags from the tag population according to a probability distribution D. The drawn tags can be either legitimate or illegitimate tags. Note that, in order to make sense in defining a security notion for privacy-preserving, even a tag is drawn twice, it should be given two different IDs.

- FreeTag(ID): This oracle release a drawn tag with ID and render it unreachable.

- Launch(.): This oracle runs a new instance π of the protocol Π. π is returned to the adversary.

- SendReader(m, π, m'): This oracle replaces a message m sent to the reader with m' in an protocol instance π.

- SendTag(m, π, m'): This oracle replaces a message m sent to a tag with m' in an protocol instance π.

- Result(π): This oracle returns 0 if at the end of an instance protocol π the reader outputs $\perp$ and 1 if otherwise.

- Corrupt(ID): This oracle returns the current state of the tag ID.

As usual, one can classify different types of attacks based on how the adversary interacts with the above oracles. Vaudenay distinguished the following types of attacks on RFID protocols:

- *Weak attack*: In this type of attack, the adversary is not given access the Corrupt(.) oracle.

- *Narrow attack*: In this type of attack, the adversary is not given access to the Result(.) oracle.

- *Forward attack*: In this type of attack, the adversary can use the Corrupt(.) oracle only once.

- *Strong attack*: In this type of attack, the adversary can call all oracles in any fashion.

- *Destructive attack*: This one is similar to the strong attack except that the adversary is not allowed to interact with a tag after corrupting the tag.

These narrow attack can be combined with other types of attacks resulting in new attacks like narrow-strong, narrow-destructive, narrow-forward and narrow-weak attacks. The security definition for a secure RFID authentication protocol against strong attack is given as follows:

Definition 15. *Secure RFID Authentication An RFID scheme is said to provide secure authentication if for any polynomial-bounded adversary, the the following probability is negligible: there exists one protocol instance π launched by the adversary in which the reader identified an uncorrupted legitimate tag* ID *but there was not matching conversation with this tag.*

The above definition implies that if the reader authenticates and identify a legitimate tag but never actually communicates with it, then the adversary must have impersonated

the tag. However, it is still quite vague since there is no clear way to quantify the security. In addition, the definition only accommodate tag-to-reader authentication but not the other way around.

The security notion for privacy protection in RFID protocols is is also given in [57]. We recall the definition below.

Definition 16. *Privacy-Preserving RFID Protocol Consider an adversary working in two phases: the querying phase in which the adversary interacts with a set of oracles and the analysis phase without any oracle access. Before entering the second phase, the adversary receives his challenge as a set of tags drawn from the* DrawTag(.) *oracle. At the final step, the adversary should output either true or false. The adversary wins if his output is true. An RFID protocol provides privacy protection if all polynomial-bounded adversaries are trivial in a sense that the adversary can make no effective use of protocol messages.*

The above security definition for privacy allowed Vaudenay to prove some interesting possibility and impossibility results as follows.

- A pseudorandom function is sufficient to construct a secure privacy-preserving authentication protocol for RFID under weak attack.

- An RFID protocol that achieves narrow-strong privacy can be used to construct a secure key agreement protocol. In other words, a secure key agreement protocol is a minimal requirement to build a narrow-strong private RFID protocol.

- A protocol that achieves strong privacy is impossible to realize.

4.3 Security Model for RFID in Universal Composable Framework

Universal Composable Framework [64] is a security model whose goal is to ensure that a secure protocol should remain secure even when running in a complex system. Essentially, a security model of this kind should define a so-called *ideal functionality* in which a cryptographic task is implemented assuming that a trusted third party exists. In an ideal functionality, each party (including the adversary attacking the cryptographic task) who wishes

to achieve his desired security goals only communicates with the trusted party. The security is defined as the indistinguishability between the ideal functionality and a real-world protocol. In [55], an ideal functionality for a secure RFID protocol called $\mathcal{F}_{aauth}$ which defines mutual authentication, privacy-preserving and forward security was presented. Before describing $\mathcal{F}_{aauth}$, we summarize the following notations used here:

- $\mathcal{A}$: the adversary who communicates directly with the ideal functionality instead of intercepting with other parties in the system.

- P: an party which can be either a tag or the server. Like the Vaudenay's model, the authors of $\mathcal{F}_{aauth}$ also considered the back-end server and the reader as a single entity.

- *type*(P): the type of a party P which indicates whether P is a tag or the server.

- *sid*: sub-session identifier. In an ideal functionality, the whole lifetime of a protocol is called a session and one instance of the said protocol (in the view of one party) is called a sub-session. Each sub-session is uniquely identified with a *sid*.

- *active*(P): a collection of identifiers for preceding incomplete sub-sessions involving P.

- *state*(P): Internal state of a party P.

We now recall the definition of $\mathcal{F}_{aauth}$ given in [55]. Essentially, an ideal functionality maintains a database and implements a number of interfaces for other parties to call. The following interfaces are defined for $\mathcal{F}_{aauth}$.

- INIT: A party P (a tag or the server) can call this interface to initiate a protocol instance, *e.g., a sub-session*. If P is not corrupted, $\mathcal{F}_{aauth}$ generates an unique sub-session identifier *sid*, record INIT(*sid*, P) in its database and send INIT(*sid*, *type*(P), *active*(P)) to the adversary. The reason that the ideal functionality send *type*(P) instead of P itself to the adversary is to protect the privacy of P. The *active*(P) This interface is illustrated in Fig. 4.3.

- ACCEPT: This interface is used to request a party to be authenticated. As in practice, an active adversary can intercept the communication channel between tags and readers and modify communicated messages at his will, $\mathcal{F}_{aauth}$ takes this fact into account by letting the adversary to decide which entity can be authenticated. The adversary does so by calling ACCEPT(*sid*, *sid*). $\mathcal{F}_{aauth}$ checks its database to see if INIT(*sid*, P) and INIT(*sid'*, P') exist. If so, $\mathcal{F}_{aauth}$ replaces the two records with PARTNER(*sid'*, P', *sid*, P) and sends ACCEPT(P') to P (*i.e.,* P' is now authenticated to P). Else if there is a record PARTNER(*sid*, P, *sid'*, P') (*i.e.*,, P has been authenticated to P'), $\mathcal{F}_{aauth}$ removes the record and sends ACCEPT(P') to P.

- IMPERSONATE: This interface is used by the adversary to impersonate a corrupted party P'. The adversary can do so by calling IMPERSONATE(*sid*, P'). $\mathcal{F}_{aauth}$ checks if INIT(*sid*, P) is in its database and P' is corrupted, If so, $\mathcal{F}_{aauth}$ removes the found record and sends ACCEPT(P') to P.

- CORRUPT: The adversary can corrupt a party P by calling CORRUPT(*sid*). If there is record INIT(*sid*, P) or PARTNER(*sid*, P, *sid'*, P') in $\mathcal{F}_{aauth}$'s database such that P is not the server (*i.e.,* only tags can be corrupted), $\mathcal{F}_{aauth}$ marks P as corrupted and remove *state*(P).

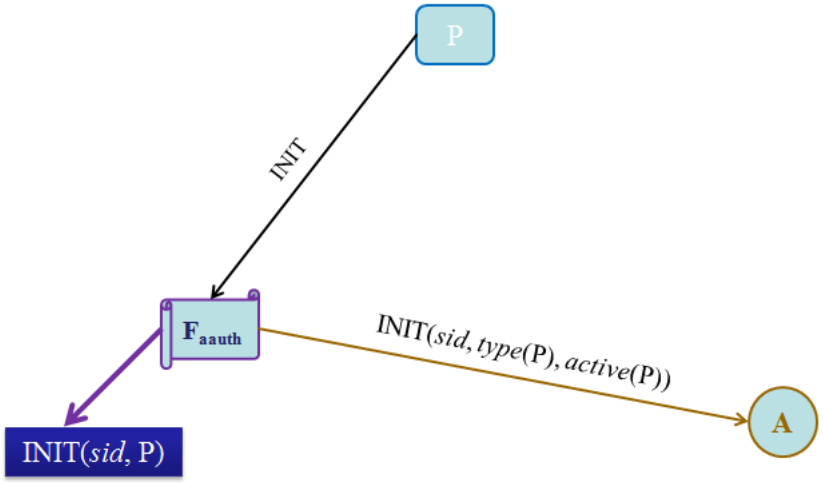

Figure 4.1: INIT interface of $\mathcal{F}_{aauth}$

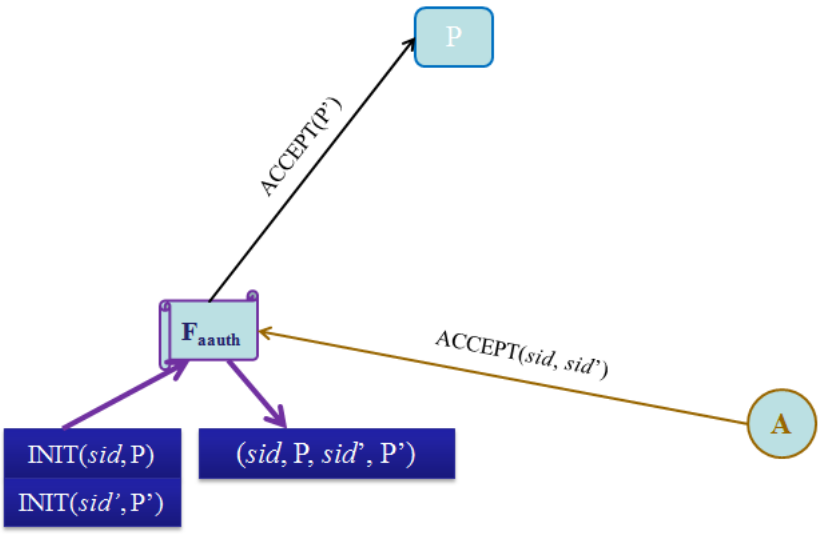

Figure 4.2: ACCEPT interface of $\mathcal{F}_{aauth}$

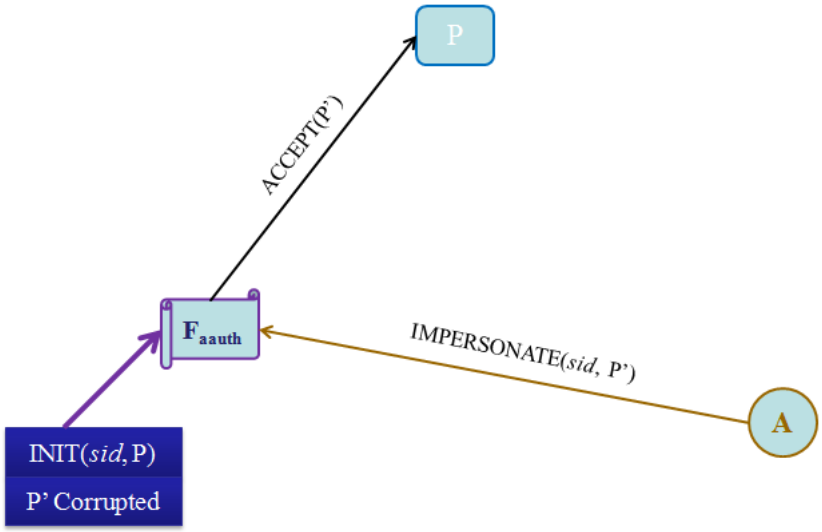

Figure 4.3: IMPERSONATE interface of $\mathcal{F}_{aauth}$

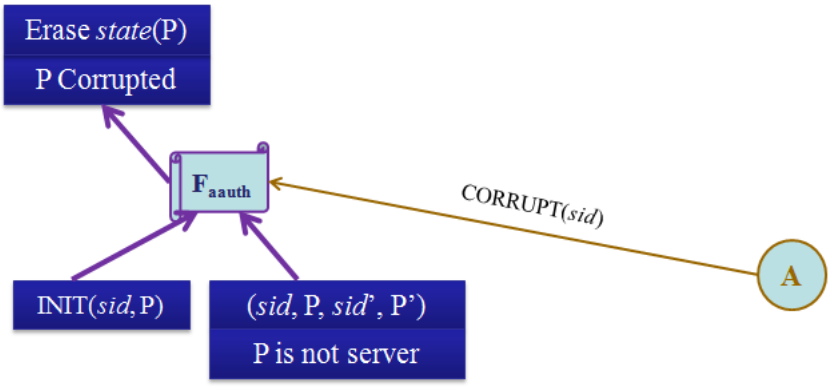

Figure 4.4: CORRUPT interface of $\mathcal{F}_{aauth}$

5. HB*: Securing HB+ Against Man-in-the-middle Attacks

HB+ is a very lightweight protocol, yet achieves provable security based on a well-studied hard problem [25]. However, the fact that HB+ is not secure against man-in-the-middle attack might disqualify it from being used in critical applications. Many attempts [49, 59, 68, 67] to secure HB+ against man-in-the-middle attack have failed. In this chapter, we present HB* which is an improved version of HB+ and show that HB* is secure against man-in-the-middle attack. In addition, HB* does not suffer from incompleteness problem.

5.1 HB and HB+ Protocols

HB+ by Juels and Weis is arguably one of the most interesting authentication protocols for RFID tags because the protocol is very efficient to implement on low-cost hardware. In addition, the security of HB+ is based on a well-studied hard problem called *Learning Parity with Noise* (LPN for short). The LPN problem is new in cryptography but better known in machine learning researches which has shown that LPN is a NP-hard problem. The origin of HB+ can be traced back to the Hopper and Blum's paper [16]. Hoppper and Blum presented two provably secure human authentication protocols, one of which depends on the intractability of the LPN problem (and usually referred to as the HB protocol). The HB+ protocol provides only tag-to-reader authentication but does not consider neither privacy protection nor forward security.

5.2 Binary Inner Product and Learning Parity with Noise Problem

The HB protocol family involves the computation of binary inner product of two k-bit numbers. The operation is defined as follows: given two k-bit number $a = (a_0 a_1 ... a_{k-1})_2$ and $x = (x_0 x_1 ... x_{k-1})_2$, the binary inner product of a and x, denoted as $a \cdot x$ is computed as follows:

$$a \cdot x = (a_0 \wedge x_0) \oplus (a_1 \wedge x_1) \oplus \cdots \oplus (a_{k-1} \wedge x_{k-1})$$

This binary inner product operation can be carried out easily by low-cost devices and even human. It is easy to see that binary inner product operation follows distributive law: $(a_1 \oplus a_2) \cdot b = (a_1 \cdot b) \oplus (a_2 \cdot b)$.

In [9], Goldreich and Levin proved that binary inner product outputs a hardcore bit and constructed a pseudorandom number generator from an one-way function and binary inner-product.

As we mentioned earlier, the security of a cryptographic scheme is often related to solving some hard computational hard problems. Therefore, it is important to look for some potential hard problems and study them carefully. *Learning Parity with Noise* (LPN) problem is a well-known hard problem based on binary inner product operation. LPN problem was originated from machine learning area. Roughly speaking, the problem is about finding an hidden value x from noisy sample data. Each sample is collected as a binary inner product of x and a random number a. If there is no noise in sample data, one can easily compute x by solving a system of linear equationsh with k unknowns as k bits of x. However, it usually the case in machine learning algorithms that the collected data is noisy. In case of LPN problem, it means that instead of getting $(a \cdot x)$, one gets $(a \cdot x) \oplus 1$. LPN problem is formally defined as follows.

Definition 17 (LPN Problem)**.** *Let* $\mathbf{M}$ *be a random* $q \times k$ *binary matrix. Let* $\eta \in (0, \frac{1}{2}]$ *be a noise factor and* $\mathbf{v} = (v_0, v_1, \cdots, v_{r-1})^T$ *be a noise vector of* q *dimensions whose each member is generated independently according to noise factor* η*,* i.e.*,* $\mathbf{Pr}(v_i = 1) = \eta$*. Choose a random* k*-bit secret* x *and compute vector* $\mathbf{z} = (\mathbf{M} \cdot x) \oplus \mathbf{v}$*. Given only* $\mathbf{M}$*,* $\mathbf{z}$ *and* η*, compute* x*.*

LPN problem has been extensively studied in [13, 14, 15, 16]. Those results showed that LPN problem is likely an intractable problem. To solve LPN problem as defined above, the best known algorithm by Blum *et al.* has sub-exponential complexity of $2^{O(\frac{k}{\log k})}$.

Note that, the above definition does not require η to be strictly less than $\frac{1}{2}$. Such a requirement is needed for HB and HB$^+$ because if $\eta = \frac{1}{2}$, the computation of the response z is essentially same as guessing it naively at random. Indeed, for HB and HB$^+$, η should be relatively small, *e.g.,* less than $\frac{1}{4}$. Recently, some negative results on the intractability of the LPN problem were presented in [47, 69] by exploiting that property. Their algorithms can solve the LPN problem for a small η value as in the case of the HB and HB$^+$ protocols. As we shall see later, our proposed protocol is the only LPN-based protocol immune against this attack. In fact, the recommended noise factor for our protocol is $\frac{1}{2}$ as LPN problem generally becomes harder when η gets closer to $\frac{1}{2}$.

In [41], the authors showed that if LPN problem is hard, a $(k+1)$-bit string $(a, (a \cdot x) \oplus v)$

is indistinguishable from a true random $(k+1)$-bit string. In fact, Katz and Shin used this result to give two elegant security proofs for HB and HB$^+$. We restate here the result in [41] for our definition of LPN problem.

Lemma 1. *Under the assumption that LPN problem is hard, the* $(k+1)$*-bit string* $(a, w = \mathcal{B}(a,x) \oplus v)$ *appears as a true random* $(k+1)$*-bit string where* a *and secret* x *are two random* k*-bit numbers and* v *is a random bit such that* $Prob(v = 1) = \eta$ *with* $\eta \in (0, 1)$*.*

5.2.1 HB Human Authentication Protocol

HB is the first cryptographic application of the binary inner product operation proposed by Hopper and Blum [16]. In the HB protocol, a person (denoted as $\mathcal{H}$) wishes to be authenticated by a machine (denoted as $\mathcal{C}$). They share a k-bit long secret x. The protocol consists of several executions of a basic challenge-response protocol which is described in Fig. 5.1.

Human(x)		**Machine**(x)
$v \leftarrow \text{Ber}_\eta$		
	$\xleftarrow{\quad a \quad}$	$a \in_R \{0,1\}^k$
$z = (a \cdot x) \oplus v$	$\xrightarrow{\quad z \quad}$	
		Verify $z = a \cdot x$

Figure 5.1: Basic authentication protocol of HB

The basic protocol starts with $\mathcal{C}$ sending a k-bit random challenge a to $\mathcal{H}$. $\mathcal{H}$ then computes an 1-bit response z as the binary inner product between a and x. Before sending z to $\mathcal{C}$, $\mathcal{H}$ decides whether to flip the value of z depending on the probability $\eta \in (0, \frac{1}{2})$. The probability η is fixed and z is flipped independently for every protocol round. We can say that, the noise bit v is drawn from Ber_η which is a Bernoulli distribution with an expected value η. Sending a noisy response z will prevent an eavesdropper who captures k instances of the basic protocol, *i.e.,* k different pairs (a, z), from recovering the secret x through Gaussian elimination (k bits of x are k unknowns and each pair of (a, z) constitutes an equation with k unknowns). $\mathcal{C}$ verifies $\mathcal{H}$ by counting the number of correct responses in q rounds of the basic protocol. Due to the effect of η, the genuine human should send roughly ηq false responses. Therefore, $\mathcal{C}$ accepts $\mathcal{H}$ only if it receives about ηq false responses from $\mathcal{H}$. However, this is also the source of the so-called incompleteness problem of HB. Because $\mathcal{C}$ cannot be sure about the exact number of false responses from $\mathcal{H}$, it is not possible to authenticate $\mathcal{H}$ with 100% certainty. Therefore, even a legitimate

$\mathcal{H}$ who follows the protocol correctly can still be rejected by $\mathcal{C}$. A method recommended in [25] is to restrict the number of false responses strictly less ηq. But this acceptance criteria clearly does not solve the problem completely.

HB is shown to be secure against eavesdropping attack. In other words, an eavesdropper observing messages exchanged between $\mathcal{H}$ and $\mathcal{C}$ has a negligible chance of impersonating $\mathcal{H}$. More specifically, an eavesdropper $\mathcal{A}$ obtains r pairs of (a, z) and tries to deduce a k-bit number x' which can be used to successfully impersonate $\mathcal{H}$. Let $\mathbf{M}$ be a $q \times k$ binary matrix such that each row of $\mathbf{M}$ is a k-bit challenge a sent by $\mathcal{C}$. Let us view x' as a column vector of dimension k and r responses z observed by $\mathcal{A}$ as vector $\mathbf{z}$. Then $(\mathbf{M} \cdot x') \oplus \mathbf{z} = \mathbf{v}$ where $\mathbf{v}$ is a column vector of dimension r where each '1' bit in $\mathbf{v}$ corresponds to one incorrect response sent by $\mathcal{H}$. In order to be accepted by $\mathcal{C}$, the Hamming weight of $\mathbf{v}$, denoted as $\mid \mathbf{v} \mid$ has to be less than or equal to ηr. The problem of finding such x' is exactly one instance of the LPN problem. Note that, as pointed out by Katz and Shin in [41], finding x' is essentially equivalent to finding x itself. Therefore. the HB protocol is provably secure against eavesdropping attack under the assumption that the LPN problem is intractable.

5.2.2 HB$^+$ Authentication Protocol

Since HB is so efficient that even human can execute the protocol, it is tempting to use HB in low-cost devices like RFID tags. However, HB cannot be directly used for RFID tag authentication since it is not secure against active attacks. More specifically, an attacker can retransmit the same challenge a for one protocol session then he can learn a noise-free value of $(a \cdot x)$, *i.e.*, one equation with k unknowns as k bits of x. By applying such attack with k linear independent values of a, the attacker can recover x using Gaussian elimination. Juels and Weis solved this problem by proposing an enhanced protocol called HB$^+$ [25]. In the HB$^+$ protocol, an RFID tag (denoted as $\mathcal{T}$) plays a role as a human and an RFID reader (denoted as $\mathcal{R}$) plays a role as a machine. Comparing to the HB protocol, $\mathcal{T}$ and $\mathcal{R}$ share an additional k-bit secret y. To prevent a malicious reader from extracting the secrets stored in tag's memory, $\mathcal{T}$ first selects a random k-bit number b called blinding factor and sends it to $\mathcal{R}$. This blinding factor effectively eliminates the threat of revealing a tag's secret key to malicious readers. A detail description of HB$^+$ protocol is given in Fig. 5.2.

In [25, 41], HB$^+$ is showed to be secure against active attacks (*i.e.,* a reader can be malicious) under the assumption the LPN problem is hard. Unfortunately, HB$^+$ still has the incompleteness problem like the HB protocol because noise is applied to the response

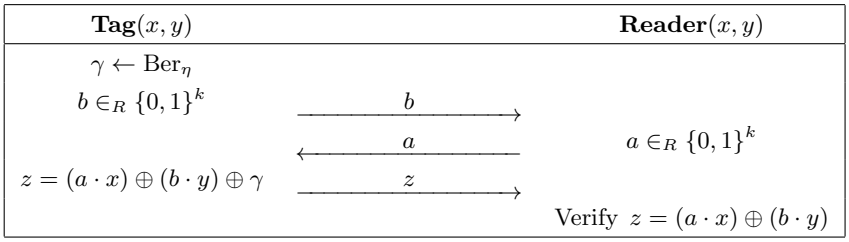

Figure 5.2: Basic authentication protocol of HB$^+$

z.

5.2.3 Man-in-the-middle Attack on HB$^+$

In [26], Gilbert *et al.* presented a very effective man-in-the-middle attack on HB$^+$, which could allow an attacker to discover the secrets x and y. The attack is also called GRS attack and requires an attacker to intercept the challenge a sent by $\mathcal{R}$ and replace it with $a' = a \oplus \delta$. $\mathcal{T}$. The tag computes the response z using a' and we have:

$$z = (a' \cdot x) \oplus (b \cdot y) \oplus v = ((a \oplus \delta) \cdot x) \oplus (b \cdot y) \oplus v = (\delta \cdot x) \oplus (a \cdot x) \oplus (b \cdot y) \oplus v$$

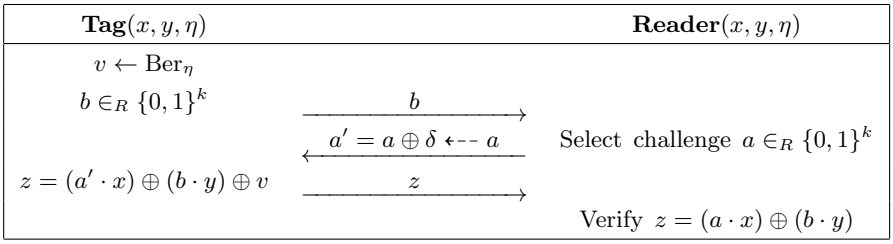

Figure 5.3: Man-in-the-middle attack on HB$^+$

The attacker then uses the same δ for r different challenges in one session of the protocol. And if $\mathcal{R}$ accepts $\mathcal{T}$, with high probability, $\delta \cdot x = 0$ since δ does not change the value of the correct response $z = (a \cdot x) \oplus (b \cdot y) \oplus v$. Otherwise, it is likely that $\delta \cdot x = 1$. By collecting k linear independent δ, the attacker can discover x using Gaussian elimination. The attack is illustrated in Fig. 5.3.

5.3 The HB* Authentication Protocol

Key Idea. We now present an improved variant of HB$^+$ protocol called HB* which can resist against man-in-the middle attacks including GRS attack. We observe that in the

HB$^+$ protocol, the response z is always computed by associating the secrets x and y with the challenge a and the blinding factor b, respectively. This partly helps the GRS man-in-the-middle attack presented above because an attacker knows that his modified challenge a' will be counted toward x. Note that, in terms of security, there is no distinction between the roles of x and y. Therefore, it is possible to eliminate GRS attack by randomly swapping the roles of x and y when computing the response z. Furthermore, both the tag and the reader should fairly involve in such process so that if either party acts maliciously, security of the protocol will not be compromised. However, we do not want to use extra cryptographic primitives like block ciphern or cryptographic hash functions to achieve our goal. The reason is that it is desirable to preserve the efficiency of HB$^+$ and base security of HB* solely on the LPN problem.

Construction.. In the new protocol, there are 4 k-bit secrets, x, y, r and t shared by the tag and the reader. The new secrets r and t will be used to securely communicate 2 bits between the tag and the reader. The key idea is to embed a bit γ into a pair (a, w) where a is a random k-bit number and $w = (a \cdot s) \oplus \gamma$. If γ is generated at a fixed probability, then a collection of (a, w) form an instance of the LPN problem. Under the assumption that the LPN problem is computationally hard, the pair (a, w) appears as a random $(k+1)$-bit string [41]. Therefore, γ can be securely communicated via (a, w). A detailed description of the basic protocol of HB* is given in Fig. 5.4.

Tag(x, y, r, t)		**Reader**(x, y, r, t)
		$\gamma \in_R \{0,1\}$
		$a \in_R \{0,1\}^k$
	$\xleftarrow{\quad a, w \quad}$	$w = (a \cdot r) \oplus \gamma$
$\gamma' \in_R \{0,1\}$		
$b \in_R \{0,1\}^k$		
$w' = (b \cdot t) \oplus \gamma'$		
If $\gamma' = (a \cdot r) \oplus w$		
$\quad z = (a \cdot x) \oplus (b \cdot y)$		
Otherwise,		
$\quad z = (b \cdot x) \oplus (a \cdot y)$	$\xrightarrow{\quad b, w', z \quad}$	
		If $\gamma = (b \cdot t) \oplus w'$,
		$\quad$ verify $z = (a \cdot x) \oplus (b \cdot y)$
		Otherwise,
		$\quad$ verify $z = (b \cdot x) \oplus (a \cdot y)$

Figure 5.4: Basic authentication protocol of HB*

In the basic authentication protocol of HB*, the role of x and y are swapped according to γ and γ' sent by the tag and the reader, respectively. Unlike the HB$^+$ protocol, $\mathcal{T}$ is accepted after r rounds of the basic authentication protocols only if all of q responses are correctly verified. This is because we no longer need to apply noise to the response z as the change in how z is computed for each basic protocol round already does the job. This property is called *perfect completeness* and is another advantage of HB* comparing to HB$^+$ and HB. In case of HB$^+$, even though a genuine RFID tag follows the protocol properly, there is still a chance it is not accepted by the RFID reader. This is clearly not desirable in practical applications. Another big difference between HB* and HB$^+$ is that the basic authentication protocol of HB* is only a 2-round protocol. In fact, the basic authentication protocol of HB$^+$ can also be 2-round by allowing the tag to send the blinding factor b together with the response z. However, the security proof provided by Juels and Weis requires that the blinding factor b must be sent before the challenge a. Note that, in HB*, the two noise values γ and γ' are chosen at random. Therefore, (a, w) and (b, w') form two instances of the LPN problem with the noise factor $\frac{1}{2}$. As the LPN problem is most intractable with true random noise, HB* is immune against various algorithms to solve instances of the LPN problem with small noise factor reported in [47, 69]. We can also consider other variants of the LPN problem which are possibly harder than the one defined in the previous chapter. For example, it is possible to use the noise factor as a secret or a variable value in HB*.

We also want to note that, HB* can also be used as an implicit authenticated key exchange protocol. As for each round of HB*, the reader securely transmit one bit to the tag and vice versa. Therefore, the tag and the reader can use $\gamma \oplus \gamma'$ as one bit of their session key. This key can be used to securely communicate data later, *e.g.,* by using one-time pad.

Computational cost and features of HB$^+$, HB*, Trusted-HB in [68] and HB# [67] are compared in the Table 5.3.

5.4 Security of HB* against Man-in-the-middle Attacks

In this section, we will show that HB* is secure against GRS attack [26] as well as general man-in-the-middle attacks. First of all, we can see that a direct application of the GRS attack does not work for the HB* protocol. It is because an attacker who intercepts $\mathcal{R}$'s challenge a and replaces it with $a' = a \oplus d$ does not know which secret (either x or y) is associated with d. Therefore, the attacker cannot compute neither x nor y by using

Table 5.1: Comparision of HB$^+$, HB*, Trusted-HB and HB#

	HB$^+$	HB*	Trusted-HB	HB#
Key Length (in bits)	$2k$	$4k$	$2k$ + hash	$2(k+q-1)$
#Inner-Product	$2q$	$4q$	$2q$	$2q$
Additional Primitives	No	No	Hash Function	Toeplitz matrix
Resistant against MIM	No	Yes	Broken	Broken
Completeness	No	Yes	No	No
Key Exchange	No	Yes	No	No

Gaussian elimination.

Since the secret r and t determine how the response z is computed, the attacker might attempt to recover r and t first. His first option is to look at (b, w') and (a, w). However, these two pairs constitute two valid instances of the LPN problem with noise factor $\frac{1}{2}$. Therefore, it is hard to recover the secret t from (b, w') and r from (a, w). Moreover, (b, w') and (a, w) originate from a single entity, $\mathcal{T}$ and $\mathcal{R}$, respectively. This property inherently implies the impossibility of man-in-the-middle attack. This is fundamentally different from the HB and HB$^+$ protocols because in these two protocols, an instance of the LPN problem is formed by information exchanged between two parties, and thus makes man-in-the-middle attack impossible.

We now show that HB* is secure against man-in-the-middle attack. First of all, we show that the pair (a, w) and (b, w') securely communicate γ and γ', respectively. As attacker cannot make adaptive queries on (a, w) (and (b, w'), it is enough to show that the encryption of bit γ (and γ') is secure against ciphertext only attack.

Theorem 1. *The encryption algorithm $\mathcal{E}_x(\gamma) = (a, w = (a \cdot x) \oplus \gamma)$ is GM-secure against ciphertext-only attack (IND-COA) under the assumption the LPN problem is hard.*

Proof. Be definition, GM-security means the attacker cannot distinguish between $\mathcal{E}_x(\gamma_b)$ and $\mathcal{E}_x(\gamma_{1-b})$ with γ_0 and γ_1 are two known plaintexts and b is a randomly chosen bit. Since the attacker only knows the ciphertexts, *i.e.,* (a, w), this proof is a direct consequence of the **Lemma** 1. □

As no noise is applied to the response z, we shall show that z appears as a random bit so that by observing z, an attacker cannot decide how z is computed. Fortunately, as Goldreich and Levin proved that binary inner product operation outputs a hard-core bit, *i.e.,* it is hard to guess it if given only one input. Therefore, z is also a hard-core

bit as it is XORing two other hardcore bits. This implies z appears as a random bit and statistically, attackers learns no useful information by observing z.

Now, let's consider general man-in-the-middle attacks against HB* in which an attacker tries to modify the reader's challenge (a, w) or the tag's response (b, w', z) in order to learn one of secret keys, (r, t, x, y). As we has proved earlier, (a, w) and (b, w') are two secure ciphertexts of γ and γ', respectively. In addition, it is impossible for a man-in-the-middle attacker to uncover neither r nor t if the LPN problem is hard. Therefore, it is reasonable to assume that the attacker does not learn anything useful by trying to detect how z is computed. We can also see that the attacker is not likely to learn anything useful if it tries to modify the reader's w and the tag's (w', z). Based on above observation, we can consider only a special class of man-in-the-middle attacks on HB* in which the attacker tries to modify only the reader's a and the tag's b in order to uncover either x or y. This class of attacks can be seen as a generalized GRS attack. First of all, we observe that in order to deduce one bit of either x or y, the attacker needs to modify the bits of a and/or b at the same position. This is because of the nature of the binary-inner product which consists of a *bitwise* AND and then XOR operations. Therefore, it is sufficient to consider the attack to uncover only one bit of either x or y based on different options for the attacker in modifying the corresponding bits of either a or b, *i.e.,* a_i and b_i. Let's assume that the attacker want to uncover the bits x_i and y_i of the two secrets x and y, we can see that only when $x_i = y_i = r_i = t_i = 0$, the response z will not be changed by modifying a_i and b_i (thus allow the attacker to detect this case by seeing if the tag is accepted by the reader when a_i and b_i are intercepted and modified). In other cases, the response z will vary which makes it impossible for the attacker to learn any useful information. We conclude that HB* is secure against man-in-the-middle attacks including GRS attack if secret keys are chosen such that there is no bit position i satisfying $x_i = y_i = r_i = t_i = 0$.

5.5 Implementation Issue of HB* Protocol

Given that the current best algorithm to solve LPN problem has its computational complexity of $2^{O(\frac{k}{\log k})}$, the recommended value for k, *i.e.,* bit length of x, y, r and t, is at least 128. Therefore, HB* protocol requires a tag to store at least $128 \times 4 = 512$ bits. This storage requirement is certainly not beyond the capacity of current low-cost RFID tags like Gen-2 tags. To implement the protocol, a tag also needs some buffer memory to store temporary data. However, a binary inner product operation can be implemented with just a 4-bit buffer memory and another $2k$-bit memory to store two inputs.

HB* protocol might also provide privacy protection. It is because each tag's output includes (b, w', z) in which (b, w') appears as a true random $(k+1)$-bit string. The privacy-preserving property is guarenteed even if each tag is not assigned an unique set of secret keys, (r, t, x, y). This might help in reducing storage at the back-end server as well as the cost of looking up a tag in the tag database.

6. Defending RFID Authentication Protocols against DoS Attacks

In this chapter, we point out that two popular RFID protocols known as O-FRAP and O-RAP protocols are vulnerable against DoS attacks. We then propose two improved protocols called O-FRAP$^+$ and O-RAP$^+$ to counter the DoS attack. Comparing to O-FRAP, O-FRAP$^+$ introduces insignificant computational overhead but requires much less memory storage at the server side. Last but not least, the two improved protocols are shown to be at least as secure as their original counterparts. The countermeasure proposed here can also be applied to most of RFID authentication protocols to prevent DoS attacks.

6.1 O-FRAP and O-RAP Protocols

O-FRAP which stands for *Optimistic Forward secure RFID Authentication Protocol* is an authentication protocol in which a back-end server authenticates and identifies RFID tags. Each tag is numbered from 1 to n and all of tag information are stored in a database D at the back-end server. Note that, the RFID reader is omitted in the description of O-FRAP as it essentially just plays the role of an intermediate party who relays messages exchanged between a tag and the server. A detail description of O-FRAP is given in Fig. 6.1.

We now discuss how O-FRAP achieves unlinkability and forward security. Each tag shares with the server a secret key denoted by k_{tag}. To protect a tag against malicious tracking, for each authentication session, a tag uses a randomly chosen number r_{tag} as its pseudonym. The tag pseudonym is stored in both the memory of the tag and the server's database D. The goal is to use the pseudonym to index D and quickly look up information on a tag given its pseudonym (the $D.query(.)$ procedure).

To achieve forward security, k_{tag} is updated after every successful authentication session both at the tag and the server sides. In O-FRAP, the tag updates its key in the last round only if it successfully verifies the server. Therefore, an active attacker can intercept and modify the server's authentication token causing the tag fails to verify the server and not to update its key. To prevent de-synchronization attack, the server keeps two versions of secret key for each tag in its database, a previously-used key (denoted by k_{tag}^{prev}) and a

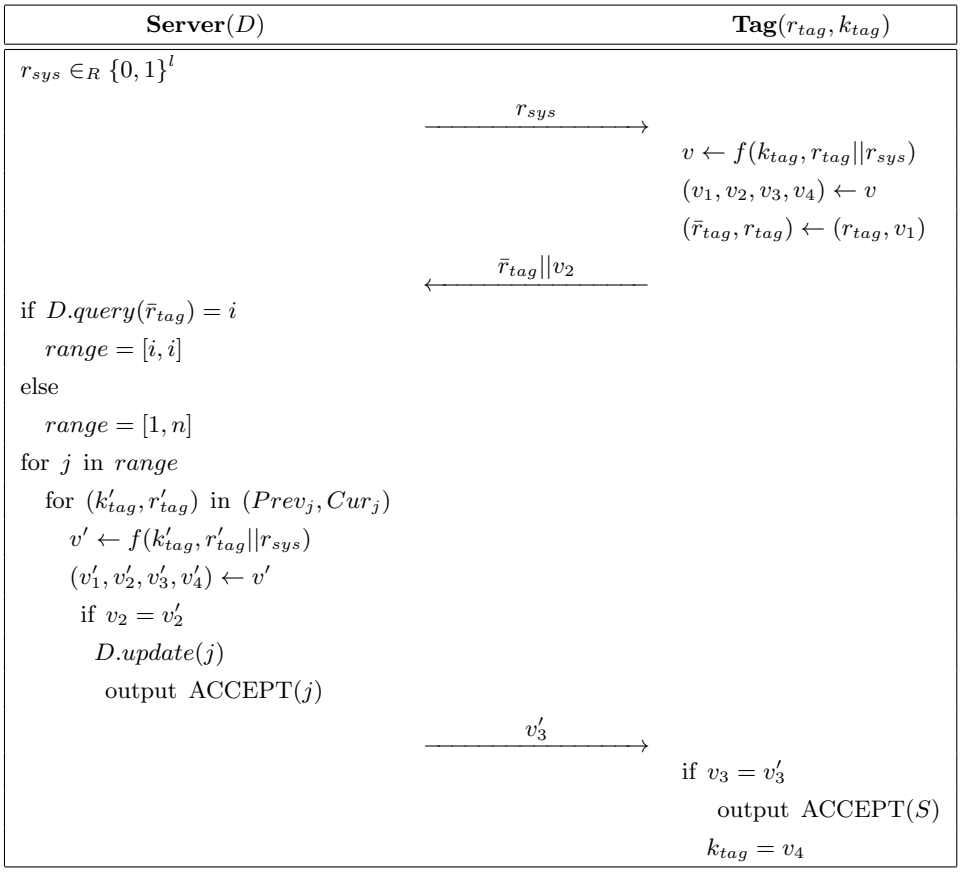

Figure 6.1: The O-FRAP Protocol

currently-used one (denoted by k_{tag}^{cur}). The server updates the two keys in the $D.update(.)$ procedure as follows:

- If the tag is authenticated with k_{tag}^{cur}, the server does: $k_{tag}^{prev} = k_{tag}^{cur}$ and $k_{tag}^{cur} = k_{tag}^{new}$ where k_{tag}^{new} is a newly generated key.

- If the tag is authenticated with k_{tag}^{prev}, the server preserves k_{tag}^{prev} and lets $k_{tag}^{cur} = k_{tag}^{new}$. The server does not update k_{tag}^{prev} because an active attacker can cause de-synchronization of secret by modifying the server's authentication token v_3' in two consecutive sessions.

The server also maintains two versions of pseudonym for each tag. Each entry in D which corresponds to one tag is indexed by two pseudonyms. The two pseudonyms are also updated in the same fashion as the secret keys are. We denote $Prev_i$ and Cur_i as two instances of tag information, each of the form (Secret Key, Pseudonym), for a tag numbered i in D.

O-RAP which stands for *Optimistic RFID Authentication Protocol* is a simplified version of O-FRAP which appeared in [73]. O-RAP is essentially O-FRAP but without a key updating procedure. As a result, O-RAP does not provide forward security and the back-end server does not need to store two versions of key for each tag.

Note that, Khalil and Raphael pointed out that the forward security of O-FRAP$^+$ can be violated because the tag outputs ACCEPT before updating its secret key. Therefore, if an attacker corrupts the tag just before the tag's secret key is updated, the immediate previous authentication session can be linked to the current session. We also would like to remark that in O-FRAP, the tag updates its secret key only if the server is authenticated successfully. This potentially defeats forward security because an attacker can modify v_3' to cause the server authentication to fail (which leads to the tag not to update its secret key). Another weakness of O-FRAP noticed in [66] is that the privacy-preserving property can be violated. More specifically, an attacker can trick a tag into updating its pseudonym but not its secret key in one session and then he will be able to trace the tag in the immediate following session. Note that, the two attacks presented in [66] are not quite practical as only two consecutive sessions can be linked.

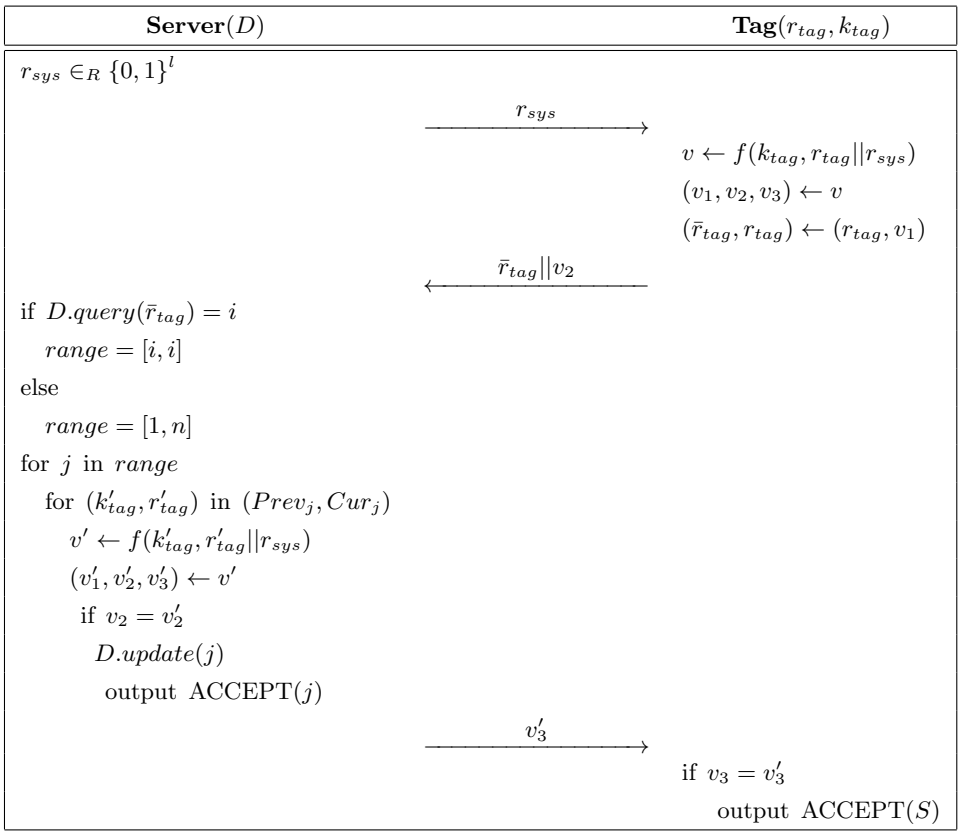

Figure 6.2: The O-RAP Protocol

6.2 Denial-of-Service Attack on O-FRAP and O-RAP

We now present a DoS attack on O-FRAP. The attack also works on O-RAP as the two protocols share the same design. In O-FRAP, the server will scan through the whole tag population in D if it fails to single out one tag in D given a pseudonym of a tag, $\bar{r}_{tag}$. An attacker can exploit this property by sending a bogus $\bar{r}_{tag}$, say $\bar{r}^*_{tag}$ to the server and thus abuses the computational resources of the server. A widespread presence of fake tags can make the problem even more serious. One may argue that if the server fails to locate a single tag in D, it means that an attack is detected. In addition, the fact that the server tries to match a tag with all the available tags in its database is simply to make the protocol complete. However, O-FRAP and O-RAP were designed to function like that for a different reason which was not mentioned in [55, 73]. The actual reason is that it is straightforward to cause de-synchronization of pseudonym between a tag and the server. As a tag always updates its pseudonym regardless of being queried by a legitimate or malicious reader, the pseudonym can be easily de-synchronized by an attacker who sends arbitrary query requests to the tag. In order to accommodate an RFID tag whose pseudonym has been de-synchronized, the server needs to match that tag with each and every entry in its database. The attack is illustrated in Fig. 6.3 where an attacker queries the tag with r^*_{sys} to cause de-synchronization of pseudonym. Then, both the tag and the attacker can cause the server to scan through the whole database D. We also want to remark that the de-synchronization of pseudonym always implies that the pseudonym at the tag is ahead of the pseudonym at the server. Therefore, keeping a pseudonym used in one of previous session and indexing D with this value are useless.

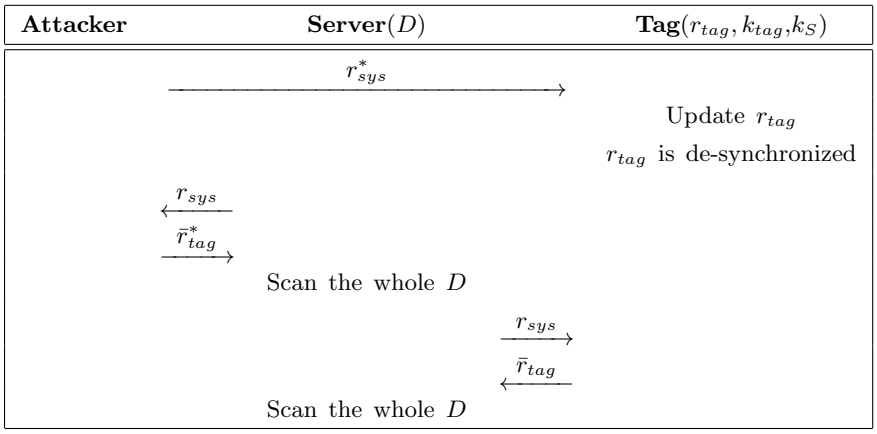

Figure 6.3: Denial of Service Attack on O-FRAP and O-RAP

There are several ways to prevent the above attack as follows:

- An RFID tag should avoid updating its pseudonym and terminate an authentication session if it is queried by an unknown server. Note that, in order to provide unlinkability, the pseudonym of a tag should not be sent before the server is authenticated.

- If an attacker attempts to send an invalid pseudonym, the server should be able to detect it and take appropriate measures, *e.g.,* stop the authentication session and examine the tag in a physically secure location. In case of O-FRAP, the server cannot distinguish whether a tag in question suffers from de-synchronization of pseudonym or its pseudonym has been actively modified by the attacker.

As we will see below, the two enhanced protocols, O-FRAP$^+$ and O-RAP$^+$, take both of the above approaches into account.

6.3 O-FRAP$^+$ and O-RAP$^+$ Protocols

We now present the first improved protocol, O-FRAP$^+$, which aims to solve the security issues with O-FRAP mentioned above. First, we shall discuss how to address those security issues and then describe the construction of O-FRAP$^+$.

Main Idea. In order to prevent an RFID tag from updating its pseudonym accidentally, the server needs to be authenticated first. This cannot be done in O-FRAP as the server has to look up a secret key shared with an unknown tag before it can compute the authentication token v_3'. Note that, there is no loss of security if a tag uses one key to authenticate the server and another key to prove its identity. Therefore, we can use another common and fixed key to authenticate the server. This key can be common for all tags or a local group of tags (*e.g.,* the tag database is partitioned and distributed) so that the server does not have to search for this key first. Let's call this key k_S. Authenticating the server first has another benefit as the tag can now update its secret key before the server. The de-synchronization attack can still be possible but in this case the problem is much easier to handle without the need for storing two version of keys for each tag. To detect whether a pseudonym has been tampered with before reaching the server, a tag can attach to its pseudonym an integrity-checking message with the secret key k_S. In other words, the tag is authenticated by the server in two steps: first, the server verifies that the tag is in its database with k_S; then the tag is identified with its own secret key

k_{tag}. In practice, a reader can use the secret key k_S to filter out fake tags. Only tags that pass the first authentication step using k_S (*i.e.*, tags that are actually in the database) can be forwarded to the back-end server. Then, the back-end server will use the second authentication step to authenticate and identify the tags.

Construction. In O-FRAP$^+$, each tag shares with the server two keys, a common secret key k_S and a private secret key k_{tag}. The database D is indexed with only currently-used pseudonyms of tags. The protocol consists of 4 rounds roughly described as follows:

- Round 1: The server broadcasts its querying request r_{sys}.
- Round 2: The tag then challenges the server with t_{sys}.
- Round 3: The server sends its response which will be verified by the tag.
- Round 4: After authenticating the server, the tag updates its pseudonym and secret key. Then it sends its old pseudonym and its authentication token to the server so that the server can authenticate and identify the tag. Note that, in response to the attack in [66], the tag should updates its secret key and pseudonym before accepting the server.

A detail description of O-FRAP$^+$ is given in Fig. 6.4 where a C language convention *return* statement is used instead of *output* used in the description of O-FRAP. Note that, O-FRAP$^+$ is a 4-round protocol for a purely practical reason. In practice, an RFID reader is usually the one to initiate an authentication session. In addition, the first message (*Query Request*) can be a broadcast message to all tags within the communication of the RFID reader without any effect on the security of the protocol.

We now discuss the key updating procedure at the server side. After successfully verifying that the tag is in D by using k_S, it is likely that the $D.query(.)$ will succeed and return one entry in D. Let $instance(i)$ be a (Secret Key, Pseudonym) = (k_{tag}, r_{tag}) pair of the tag i stored in D. Then, the server can authenticate the tag with the key k_{tag}. However, it is still possible to cause de-synchronization of k_{tag} in O-FRAP$^+$. By modifying v_2, an attacker can cause tag authentication with k_{tag} to fail which results in the server not to update its version of k_{tag}. Note that, the de-synchronization of secret in O-FRAP$^+$ is very different from the same problem in O-FRAP. In O-FRAP, the de-synchronization of

k_{tag} means the tag still keeps a key used in one of previous authentication sessions while the server keeps the currently-used key. Whereas, in O-FRAP$^+$, if de-synchronization of k_{tag} occurs then the server keeps one of the old keys while the tag has the latest key. Furthermore, even though k_{tag} is inconsistent between the server and the tag, the server can still locate the candidate tag in its database. It is clearly not the case in O-FRAP. This is why we do not need to store two versions of secret key for each tag in D. To accommodate a tag whose k_{tag} has been de-synchronized, we update k_{tag} in a chaining fashion, *i.e.,* $k_{tag}^{new} = f(k_{tag}^{cur})$[1]. The server can try a new $k_{tag} = f(k_{tag})$ to re-authenticate the tag once it fails to authenticate the tag with its current version of k_{tag}. The number of times the server tries in this scenario is up to a specific deployment of O-FRAP$^+$.

Using the same approach described above, we can also secure O-RAP against the denial-of-service attack presented in this paper. We call O-RAP$^+$ as the secure version of O-RAP. Note that, O-RAP does not have a key updating procedure. Therefore, there is no key updating procedure as well as *goto* statement to handle the de-synchronization of secret problem in O-RAP$^+$. O-RAP$^+$ is illustrated in Fig. 6.5.

6.4 Security Analysis and Comparison

Secure Mutual Authentication. We can see that O-FRAP$^+$ and O-RAP$^+$ use the same mechanism to provide mutual authentication as in O-FRAP and O-RAP. More specifically, an authentication token is composed of a random nonce and the output from the function $f(.)$ with the shared secret key and the random nonce as the input. The only difference is that the keys to authenticate the server and a tag are different in O-FRAP$^+$ and O-RAP$^+$. In the following theorem, we prove that O-RAP$^+$ is at least as secure as O-RAP in terms of authentication. The relationships between O-FRAP$^+$ and O-FRAP can be proved similarly.

Theorem 2. *Suppose that there exists an adversary $\mathcal{A}^{O-RAP^+}$ that impersonates either a tag or the server in the O-RAP$^+$ protocol with success probability ϵ. One can construct another adversary $\mathcal{A}^{O-RAP}$ that impersonates either a tag or the server in the O-RAP protocol with success probability at least ϵ.*

Proof. In order to impersonate the server in the O-RAP$^+$ protocol, given a challenge t_{sys}, the adversary $\mathcal{A}^{O-RAP^+}$ should be able to compute an authentication token u' such that $u' = f(k_S, r'_{sys}||t_{sys})$ where r'_{sys} is chosen by the adversary himself. Since the sever authentication token in the O-RAP protocol is derived from $v \leftarrow f(k_{tag}, r_{tag}||r_{sys})$ in which

[1]If the output length of $f(.)$ is longer than l, we can take the first l bits the output of $f(.)$.

the adversary $\mathcal{A}^{O-RAP}$ can choose r_{sys}. Therefore, the adversary $\mathcal{A}^{O-RAP}$ can makes use of the adversary $\mathcal{A}^{O-RAP^+}$ by feeding the challenge r_{tag} to the adversary $\mathcal{A}^{O-RAP^+}$. Then when the adversary $\mathcal{A}^{O-RAP^+}$ returns (u, r_{sys}), $\mathcal{A}^{O-RAP}$ picks $\mathcal{A}^{O-RAP^+}$'s r_{sys} as its own r_{sys} and lets $v = u$ which should be a valid authentication token. Note that, since the actual authentication token of the sever in the O-RAP protocol is v_3. Therefore the bit length of v_3 is smaller than the bit length of u which is the server's authentication token in the O-RAP$^+$ protocol. We conclude that impersonating the server in the O-RAP$^+$ protocol is at least as hard as that in the O-RAP protocol.

The case of tag impersonation can be proved similarly. Thus the proof is complete. □

Privacy-Preserving. Both O-FRAP$^+$ and O-RAP$^+$ are secure against the tracing attack presented in [66]. The reason is that the tag pseudonym and secret key are updated at the same time. Therefore, an attacker cannot cause a tag to update its pseudonym but not its secret key. Note that, the tag pseudonym is emitted only after the server is verified so not updating the pseudonym after every session does not make a tag vulnerable to tracing attack. In addition, the fact that a tag stops the protocol prematurely (*i.e.,* the server is not successfully authenticated) might have positive impact on privacy protection in practice because it limits the ability of an attacker to detect the presence of RFID tags. For instance, a malicious party may attempt to look for a particular RFID tag embedded in a passport to determine the nationality of the passport holder. However, because the tag terminates the protocol early, the malicious party might not have enough information on the protocol signature to decide the presence or origin of the tag. Once again, we prove the relationship between O-RAP$^+$ and O-RAP in term of privacy protection here.

Theorem 3. *Suppose that there exists an adversary $\mathcal{A}^{O-RAP^+}$ that tracks a tag in the O-RAP$^+$ protocol with success probability ϵ. One can construct another adversary $\mathcal{A}^{O-RAP}$ that track another tag in the O-RAP protocol with success probability at least ϵ.*

Proof. This theorem can be proved in the same way as the Theorem 2. In particular, if an adversary can track a tag in the O-RAP$^+$ protocol by observing the tag's output or querying the tag itself, a similar adversary can be constructed to track a tag in the O-RAP protocol. Note that, the use of the fixed key k_S, which is a key different of O-RAP$^+$ comparing to O-RAP, does help the adversary to track a tag because if the tag's challenge t_{sys} is randomly chosen, the tag's output with respect to k_S is also random because $f(.)$ is a pseudo-random function. □

Forward Security. O-FRAP$^+$ prevents the attack in [66] by requiring a tag to update its secret key before accepting the server as authenticated. Furthermore, in O-FRAP$^+$, an

attacker cannot cause a tag not to update its secret key even in case k_S is corrupted. It is different from O-FRAP where an attacker can modify the server authentication token v_3' so that a tag will not update its secret key. As updating secret is required to achieve forward security, O-FRAP$^+$ is at least as secure as O-FRAP.

Resistant against DoS Attack. O-FRAP$^+$ and O-RAP$^+$ prevent fake tags from abusing the server's computational resources by verifying the integrity of pseudonyms. In addition, an attacker cannot cause a tag to update its pseudonym leading to inconsistent pseudonyms between tags and the server (which makes the server to scan through the whole tag database when querying tags in O-FRAP and O-RAP protocols).

Comparison. The comparison in terms of required computational resource and security features of O-FRAP, O-FRAP$^+$ and O-RAP$^+$ is given in Table 6.1.

Table 6.1: Comparison of O-FRAP, O-FRAP$^+$ and O-RAP$^+$

	O-FRAP	O-FRAP$^+$	O-RAP$^+$
No. of $f(.)$ Evaluation	Tag: 1 Server: 1	Tag: 4 Server: 4	Tag: 3 Server: 3
Key Length	Tag: $2l$ bits Server: $4ln$ bits	Tag: $3l$ bits Server: $(2n+1)l$ bits	Tag: $3l$ bits Server: $(2n+1)l$ bits
Mutual Authentication	Yes	Yes	Yes
Privacy Protection	Weak	Strong	Strong
Resistant against DoS	No	Yes	Yes

6.5 Implementation Issues of O-FRAP$^+$ and O-RAP$^+$ Protocols

O-FRAP$^+$ and O-RAP$^+$ Protocols can be integrated into a RFID system such that a reader perform the first phase of authentication (*i.e.,* verifying a tag's pseudonym) and the second phase of authentication is done at the back-end server (with the reader as an intermediate party). This is a key part to prevent malicious parties from abusing the server's computational resources.

A common concern about RFID protocols that provide privacy-preserving by using pseudonym is how to populate a tag database when a large batch of tags are provided. We can address this problem by having tags to operate in a special mode called *access mode* which is much like the access mode of Gen-2 tags. In the access mode, a reader is

given access to the entire memory of a tag so that it can collect all tag information to populate the tag database. The reader must be first authenticated to do so and in case of O-FRAP$^+$ and O-RAP$^+$, a tag can use the key k_S, which is common for all tags and known to the reader in advance, to authenticate the reader.

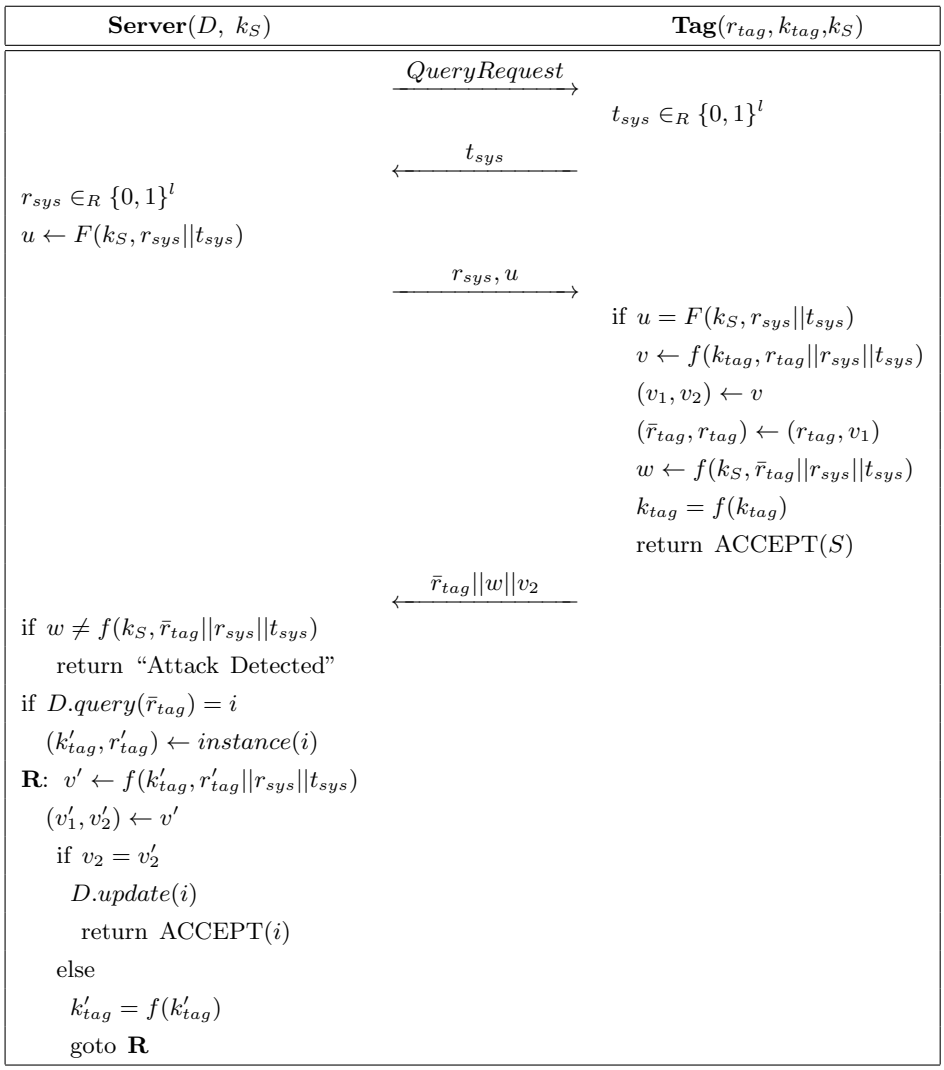

Figure 6.4: The O-FRAP$^+$ Protocol

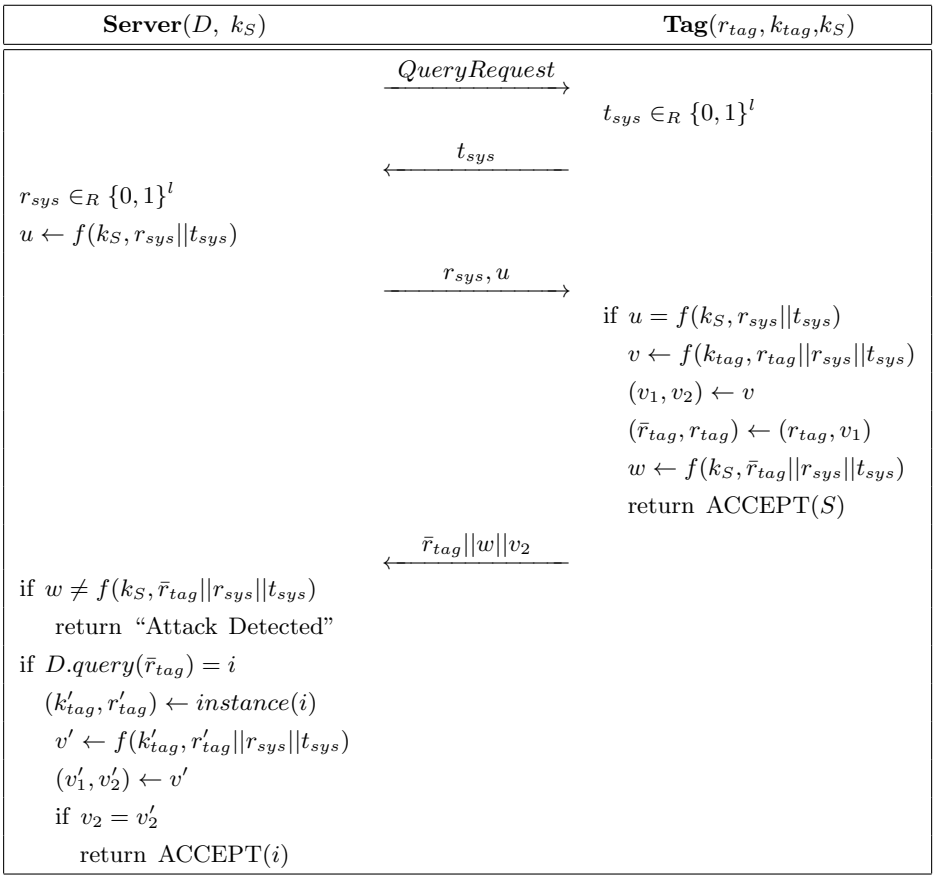

Figure 6.5: The O-RAP$^+$ Protocol

7. A Scalable Grouping-Proof Protocol

In this chapter, we first point out that all of the previous grouping-proof protocols in [27, 36, 53, 54, 65] suffer from a scalability problem. More specifically, a reader has to relay messages from one tag to another tag which makes it difficult to scan a large number of tags at the same time. Our proposed protocol aims to solve this problem by removing the need to relay messages among tags.

7.1 Grouping-Proof Protocols for RFID

Grouping-proof protocols allow multiple RFID tags to be scanned at once such that their co-existence is cryptographically guaranteed. One typical application of a grouping-proof protocol is to scan tags that are supposed to stay *together*. For example, RFID tags attached on different parts of a car should be located near each other. Juels [27] proposed the first protocol of this kind which is called yoking-proof. The protocol allows an RFID reader to produce a co-existence proof of two RFID tags. Subsequent protocols which enhance the security and support simultaneous scanning of more than two tags appeared in [36, 53, 54, 65]. We briefly summarize these protocols below.

7.1.1 Yoking-Proof

Yoking-proof [27] enables an RFID reader to produce a proof that two RFID tags are present within the communication range of the reader. The proof can then be verified by the verifier which knows secret keys of the two tags. In the yoking-proof protocol, a tag proves its presence by signing a random number generated by another tag presence in the communication range of the reader at the same time. A message authentication code (MAC for short) algorithm can be used as a signing mechanism. The reader is in charge of forwarding the random numbers and collecting the MACs to form a proof of co-existence. The resulting co-existence proof can be verified by checking the validity of the MACs. The protocol proceeds as follows:

1. $\mathcal{R} \rightarrow \mathcal{T}_1$: request.
2. $\mathcal{T}_1 \rightarrow \mathcal{R}$: $\mathcal{T}_1, r_1$ where r_1 is chosen at random.

3. $\mathcal{R} \rightarrow \mathcal{T}_2$: r_1.

4. $\mathcal{T}_2 \rightarrow \mathcal{R}$: $\mathcal{T}_2, r_2, m_2 = \text{MAC}_{K_2}[r_1]$ where r_2 is chosen at random.

5. $\mathcal{R} \rightarrow \mathcal{T}_1$: r_2.

6. $\mathcal{T}_1 \rightarrow \mathcal{R}$: $m_1 = \text{MAC}_{K_1}[r_2]$.

7. $\mathcal{R} \rightarrow \mathcal{V}$: $P = (\mathcal{T}_1, r_1, m_1, \mathcal{T}_2, r_2, m_2)$.

An illustration of yoking-proof is also given in Fig. 7.1.

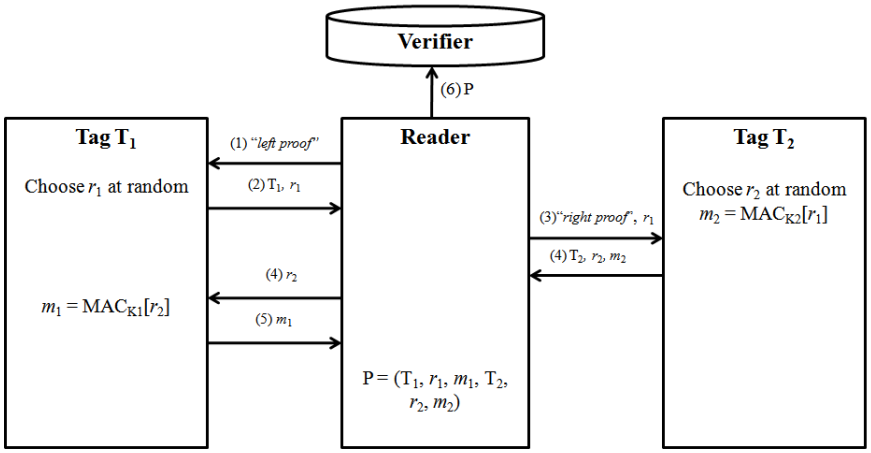

Figure 7.1: Yoking-Proof for RFID Tags

7.1.2 Saitoh-Sakurai's Grouping-Proof Protocol

Saitoh and Sakurai [36] showed that yoking-proof is vulnerable to replay attack. The reason is that the two messages m_1 and m_2 are not guaranteed to be generated in the same session. As a result, an attacker can reuse m_2 in another session which results in a forged proof. To prevent the attack, Saitoh and Sakurai proposed a timestamp-based yoking-proof which requires an online verifier to issue a timestamp TS for each session. TS is included in each co-existence proof and needs to be signed by both $\mathcal{T}_1$ and $\mathcal{T}_2$. The online verifier accepts a proof only if it is received within the expected lifespan of one interrogation session. The protocol is illustrated in Fig. 7.2

In [36], the authors also proposed another protocol which allows the simultaneous scanning of more than two tags. The protocol is called grouping-proof and requires an additional entity called pallet tag. The pallet tag has more computational resource than an

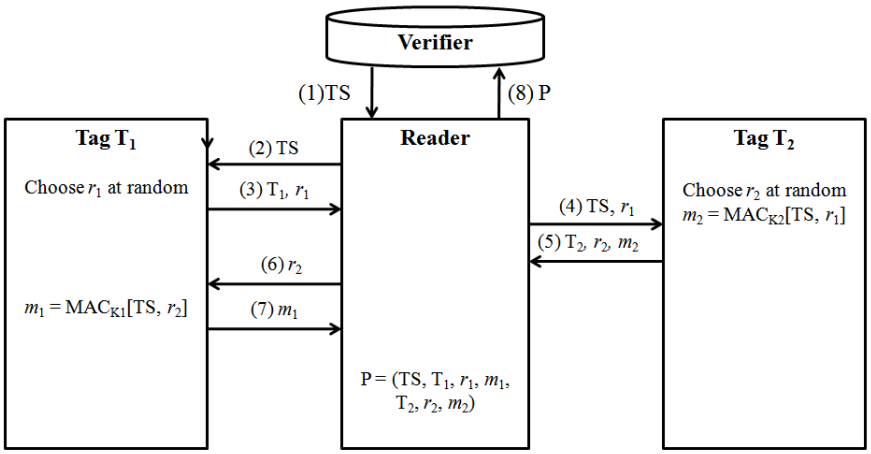

Figure 7.2: Timestamp-based Yoking-Proof for RFID Tags

RFID tag and acts as a representative of all RFID tags that are in the same package with the pallet tag.

1. $\mathcal{V} \rightarrow \mathcal{R}$: TS.
2. $\mathcal{R} \rightarrow \mathcal{T}_1, \mathcal{T}_1, \cdots \mathcal{T}_n$: TS.
3. $\mathcal{T}_i \rightarrow \mathcal{R}$: m_i =MAC$_{K_i}$[TS], for $i = 1, 2, \cdots, n$.
4. $\mathcal{R} \rightarrow$ Pallet Tag: TS, $m_1, m_2, \cdots, m_n$.
5. Pallet Tag $\rightarrow \mathcal{R}$: C$_P$ = SK$_K$[TS, $m_1, m_2, \cdots, m_n$].
6. $\mathcal{R} \rightarrow \mathcal{V}$: P =(TS, C$_P, \mathcal{T}_1, \mathcal{T}_2, \cdots, \mathcal{T}_n$).

The proof P is subject to timestamp verification by the online verifier in order to prevent replay attacks. Then, the co-existence proof is verified by checking the validity of each m_i. The protocol is illustrated in Fig. 7.3

7.1.3 Other Variants of Yoking-Proof

Piramuthu proposed another variant of yoking-proof which does not use timestamp to prevent replay attack [53]. The main idea is to let the tag $\mathcal{T}_1$ sign both its own random number r_1 and the tag $\mathcal{T}_2$'s MAC m_2. As a result, neither of two MACs m_1 nor m_2 can be reused in another session. The protocol is illustrated in Fig. 7.4

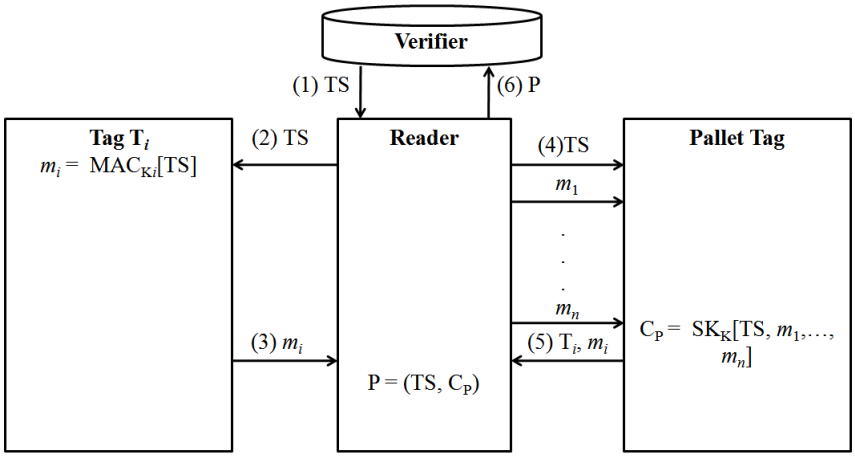

Figure 7.3: Saitoh-Sakurai's Grouping-Proof Protocol

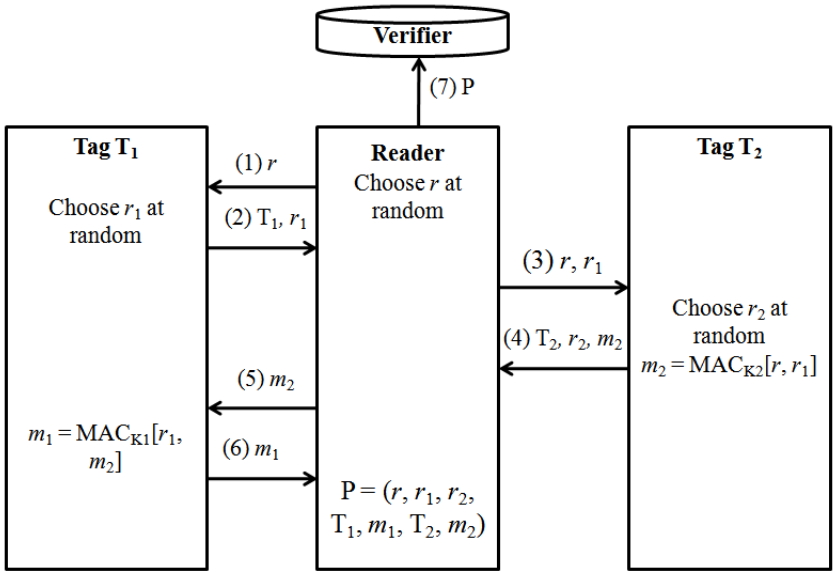

Figure 7.4: Piramuthu's Variant of Yoking-Proof

Lin *et al.* pointed out that Piramuthu's protocol may suffer from interference problem when multiple readers are represent. More specifically, when a tag is queried by two readers at the same time, the tag might have problem determining what messages to sign if no proper session management is present. Lin *et al.* also showed that the timestamp-based yoking-proof presented in [36] is indeed not secure against replay attack. An attacker can query the tag $\mathcal{T}_1$ with many different timestamp values in the future. Then, the responses from $\mathcal{T}_1$ can be used to query the tag $\mathcal{T}_1$ which is in a different location and at different times. To counter the problem, Lin *et al.* proposed another variant of timestamp-based yoking proof in which the verifier encrypts a timestamp value before sending to a reader. The protocol is illustrated in Fig. 7.5

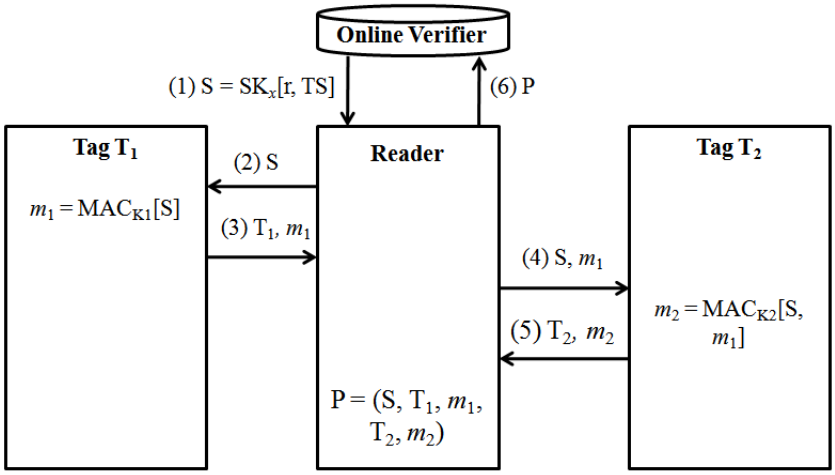

Figure 7.5: Lin et al.'s Variant of Yoking-Proof

Lin *et al.* also proposed another grouping-proof protocol for any number of tags but without using a pallet tag. The protocol uses a method called timestamp-chaining. That is, to produce a co-existence proof of n tags, the first tag signs the hashed timestamp value TS_1 from the reader and the i-th tag signs the hash of timestamp TS_i and the $(i-1)$-th tag's MAC. The reader is in charge of forwarding the hashes and assigning proper values for each timestamp.

7.1.4 Burmester et al.'s Grouping-Proof Protocol

Burmester *et al.* proposed two grouping-proof protocols which employ a different approach comparing to other previous works [65]. In particular, in the Burmester *et al.*'s protocols,

a tag does not use MAC to sign its challenge. However, in order to produce a co-existence proof of two tags, Burmester *et al.*'s protocols assume that the two tags share a group id (denoted as gid) and a common secret key (denoted as K_g). Each tag also maintains a counter variable c such that c is increased by 1 after each successful protocol session. We describe below only one protocol in [65]. The other protocol has the same design but provides privacy protection by updating the group id after each session.

1. $\mathcal{R} \rightarrow \mathcal{T}_1, \mathcal{T}_2$: r_{sys} chosen at random.
2. $\mathcal{T}_1, \mathcal{T}_2 \rightarrow \mathcal{R}$: gid.
3. $\mathcal{R} \rightarrow \mathcal{T}_1, \mathcal{T}_2$: $\mathcal{T}_1$ and $\mathcal{T}_2$ are linked.
4. $\mathcal{T}_1 \rightarrow \mathcal{R}$: c, r_1 where $r_1 || s_1 = f(r_{sys}, c, K_g)$.
5. $\mathcal{R} \rightarrow \mathcal{T}_2$: r_1, c.
6. $\mathcal{T}_2 \rightarrow \mathcal{R}$: t_2, s_2 if $r_1 = r_2$ where $r_2 || s_2 = f(r_{sys}, c, K_g)$ and $t_2 = f(r_2, c, K_2)$. If $r_1 \neq r_2$, $\mathcal{T}_2$ terminates the protocol.
7. $\mathcal{R} \rightarrow \mathcal{T}_1$: s_2.
8. $\mathcal{T}_1 \rightarrow \mathcal{R}$: t_1 if $s_1 = s_2$ where $t_1 = f(r_1, c, K_1)$. $\mathcal{T}_1$ also update its counter value $c = c + 1$. If $s_1 \neq s_2$, $\mathcal{T}_1$ terminates the protocol.
9. $\mathcal{R} \rightarrow \mathcal{V}$: $P = (r_{sys}, gid, c, r_1, t_1, r_2, t_2)$.

The above protocol is illustrated in Fig. 7.6.

7.2 Security Issues of Previous Grouping-Proof Protocols

An important part of designing a secure protocol is to define a security model in which the term *secure* correctly captures our intuition about real-world security of the protocol. We argue that this task has not been done adequately in previous works. In particular, no previous work addresses *mafia fraud attack* presented in [17]. Mafia fraud attack is simply a relay attack in which an attacker relays messages exchanged between a reader and tags. All of grouping-proof protocols for RFID are insecure against this attack because the attacker can relay messages exchanged between a reader and tags that are out of the communication range of the reader. The result is an invalid proof that contains tags not in the communication range of the reader at the time of interrogation. A generic mafia fraud attack on all grouping-proof protocols is illustrated in Fig. 7.2.

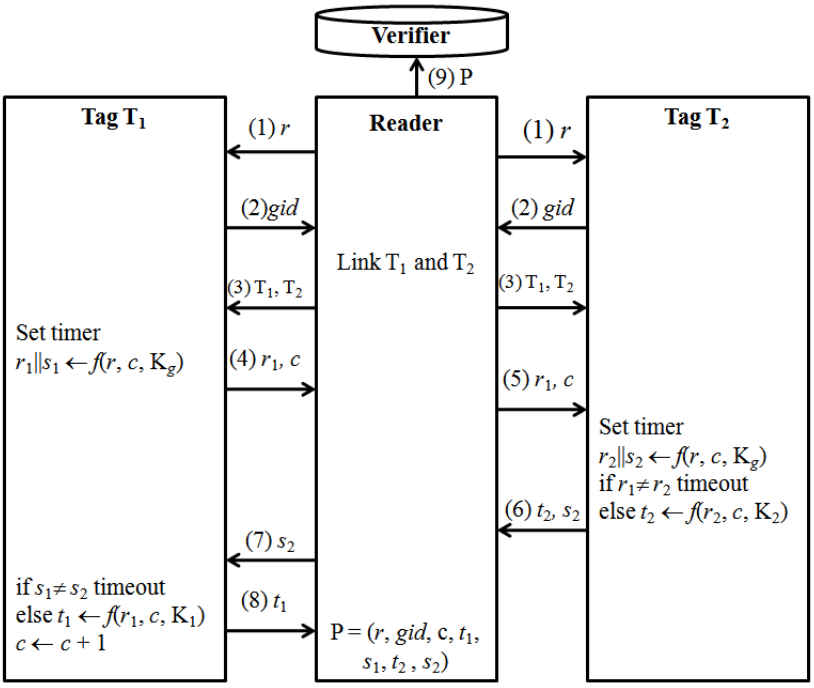

Figure 7.6: Burmester et al.'s Variant of Yoking-Proof

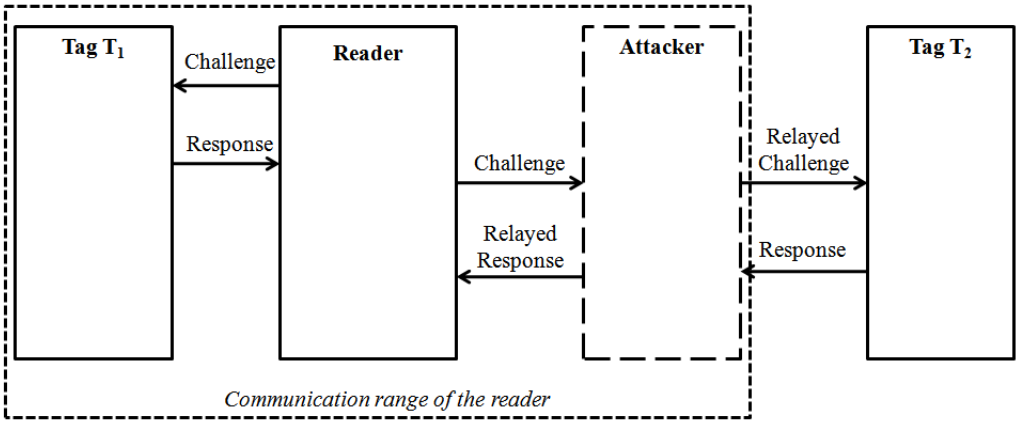

Figure 7.7: Generic mafia fraud attack on grouping-proof protocols

A specific mafia fraud attack on Lin *et al.*'s variant of yoking-proof is illustrated Fig. 7.8.

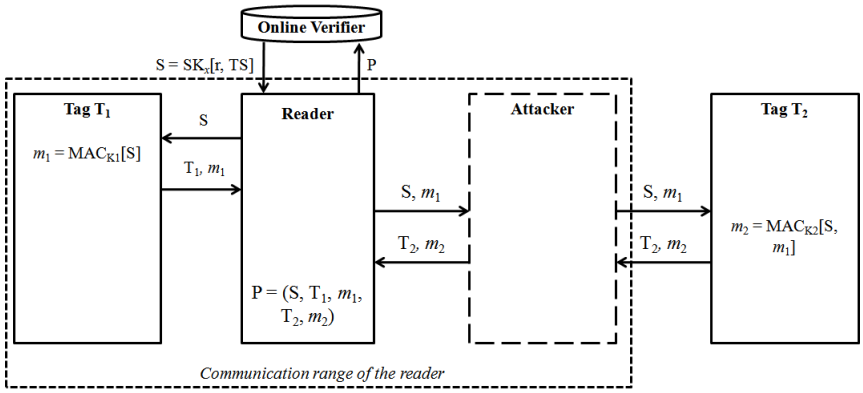

Figure 7.8: Mafia fraud attack on Lin *et al.*'s protocol

Other grouping-proof protocols can be attacked similarly. We think that a security model that does not address this issue cannot be a proper security model for grouping-proof protocols because it would be impossible to prove the security. In practice, we can somehow limit this attack by using timestamp so that a relay attacker does not have enough time to relay messages out of the communication range of the reader. Indeed, some of previous protocols [36, 54] make use of timestamp which is actually used to defeat replay attack. However, this prevention method works only if an interrogation session lasts as short as possible. Since a reader has to relay messages among tags in previous protocols, a protocol session can be prolonged which makes mafia fraud attack more feasible. Therefore, it is also important to solve the scalability problem in order to mitigate mafia fraud attack. Note that, the use of timestamp does not mean that we do not need to take mafia attack into account when defining a security model for grouping-proof protocols. After all, mafia fraud attack is always possible, at least from the theoretical point of view. In the light of a recent work [75] by Chandran, it appears that constructing a protocol that is provably secure against mafia fraud attack might be impossible. What we can do is to implement certain practical countermeasures like using a distance-bounding protocol. Another issue when defining a security model for grouping-proof protocol is that the verifier has no knowledge of what or how many tags are actually in the communication range of a reader. Therefore, we cannot achieve security at all if a reader is allowed to behave maliciously in an arbitrary way. For example, a reader can deliberately avoid scanning some tags resulting in an invalid co-existence proof. In this chapter, we present

a security model for secure grouping-proof protocols which takes the all of above issues into account. In particular, we put the following assumptions in our security model:

- Relaying messages out of the communication range of a reader is not possible. We address this assumption by restricting the adversary's access to the tag oracle during the last phase of an experiment in which the adversary interacts with a set of oracles, receives a challenge and attempts to solve the challenge. We shall discuss this in more details below.

- The reader is trusted to execute the protocol fruitfully but it may report an invalid co-existence proof to the verifier. In particular, before reporting a valid proof to the verifier, a dishonest reader may try to remove a tag from the proof, replace a tag in the proof with another tag or add another tag to the proof. In practice, the protocol can be implemented in a tamper-proof chip whereas a proof is assembled and sent to the verifier by the reader in software (and therefore is subject to malicious behaviors of a reader).

It is important to note that, none of previous protocol appears to be secure in a weaker assumption. In this chapter, we propose a grouping-proof protocol for RFID by using a (n, n)-secret sharing scheme (also referred to as *unanimous consent control* in [77]). The goal of using a (n, n)-secret sharing scheme in our protocol is to let n tags sign n different challenges. The n challenges are n shared secrets of a number which is randomly chosen by the verifier. The threshold property of a (n, n)-secret sharing scheme guarantees that n signed challenges are *tied together*.

7.3 Scalability Issue of Previous Grouping-Proof Protocols

The design of yoking-proof and timestamp-based yoking proof suffers from a serious scalability issue. The reason is that a reader needs to relay messages from one tag to another so that a tag can sign random numbers that were generated by other tags. As a result, if the reader wants to produce a co-existence proof of n tags, it will need to relay $n(n-1)$ messages among n tags. The number of relaying messages can be reduced to $(n-1)$ if a proof is constructed in a chaining fashion. That is, the first tag signs the second tag's random number. The second tag signs the third tag's random number and so on. However, this approach might be subject to replay attack if a protocol is not designed carefully.

Let's assume that a tag $\mathcal{T}_i$ appears in two chaining proofs. Using $\mathcal{T}_i$ as a connector, an attacker might try to connect the first half of the first proof with the second half of the second proof to produce a forged proof. Nevertheless, this is a significant communication overhead compared to the traditional method of scanning one tag at a time which requires no message to be relayed by the reader. This same problem also appears in other variations of yoking-proof and in [53, 54, 65].

The grouping-proof protocol by Saitoh and Sakurai does not use the same design of yoking-proof. However, it requires a pallet tag which is capable of performing symmetric encryption. This increases the cost of multiple scanning of tags and might not be flexible in practice. For example, in a retail store, items that are scanned at a point-of-sale usually do not have an accompanying pallet tag. In addition, the reader still needs to relay messages from all tags to the pallet tag. In order to scan n tags at once, the reader needs to relay n messages to the pallet tag.

As we pointed out earlier, the lifespan of one protocol session may affect the resilience of a grouping-proof protocol against mafia fraud attack. We think that it is important to solve the scalability problem of previous grouping-proof protocols, for the sake of not only performance but also security.

7.4 Security Model for Secure Grouping-Proof Protocols

It is important that before designing a protocol to secure certain cryptographic tasks, one should clearly define the meaning of the term *secure*. In this paper, We present a security model for a secure grouping-proof protocol for RFID tags which addresses mafia fraud attack and the level of trust on an RFID reader. We then define what a secure grouping-proof protocol for RFID tags means. Our security model is a conventional security model in a sense that the adversary is given access to a set of oracles and the term *secure* is defined via a game between a challenger and the adversary. In [65], the authors proposed another security model for secure grouping-proof protocol in the *Universal Composable Framework* (UC framework for short). A protocol which is secure in UC framework is guaranteed to remain secure even when running as a component of a large system. The most important part of a security model in the UC framework is the *ideal functionality* which is a trusted party implementing the required cryptographic task. The ideal functionality defined in [65] is called $\mathcal{F}_{group}$ which interacts with different involving parties via 5 interfaces: activate, initiate, link, complete and verify (whereas involving parties do not interact with each other directly). Interested readers are referred to [65] for the descrip-

tion of each of $\mathcal{F}_{group}$'s interface. The problem with $\mathcal{F}_{group}$ is that there is no condition for a tag to call $\mathcal{F}_{group}$'s initiate. Indeed, only tags within the communication range of a reader are qualified to make the initiate calls to $\mathcal{F}_{group}$. Unfortunately, *the communication range of a reader* is not modeled in $\mathcal{F}_{group}$. That is probably why the full security proofs for two protocols in [65] are not yet available. Note that, this does not mean security proofs for lightweight authentication protocols for RFID are invalid. In case of an authentication protocol, the goal of the adversary is to impersonate a tag. Simply relaying message between a legitimate tag and a reader does not qualify as impersonation. It is also worth mentioning that most of previous grouping-proof protocols employ timestamp which makes it difficult to rigorously analyze their security. Indeed, it is better to avoid using a physical object in the description of a protocol but embed it in the security model or assumption.

We now describe the security model for a secure grouping-proof protocol. First of all, it should be realized that for a grouping-proof for RFID tags protocol, the primary goal of an adversary is to inject some tags (possibly genuine) into a valid co-existence proof while the tags are not actually in the communication range of the reader. In addition, the adversary might also want to remove some tags from a valid co-existence proof. It is also assumed that the reader can behave maliciously but does execute the protocol correctly. When reporting a co-existence proof to the verifier, a malicious reader may try to replace some tags in the proof with different tags, add a tag to the proof or remove a tag from the proof. One can obtain a stronger security notion by allowing a malicious reader to deviate from the protocol in any fashion. However, it is impossible to achieve security because the verifier has no knowledge of what and how many tags are actually in the communication range of the reader. The malicious reader can violate the security by deliberately not scanning some tags. This issue also appears in all of the previous protocols in [27, 36, 53, 54, 65]. Indeed, the timestamp-chaining protocol by Lin *et al.* is vulnerable to malicious behaviors of a reader even if the reader is trusted to execute the protocol correctly. The reason is that before reporting a co-existence proof of n tags to the verifier, the malicious reader can remove some tags at the end of the timestamp chain from the proof without invalidating the proof.

A set of oracles that provide information to the adversary are defined as follows:

- The reader(.) oracle: This oracle simulates a reader during a protocol session. That it, it returns the reader's challenge to a tag.

- The corrupt-reader(.) oracle: This oracle corrupts a reader and returns the current

state of the reader. The adversary is also allowed to control the reader after this oracle is called.

- The tag(.) oracle: This oracle simulates a tag during a protocol session. That is, it returns the tag's response given a challenge from a reader.

- The verify(.) oracle: This oracle takes a co-existence proof P as input and returns 1 if P is valid and 0 otherwise.

We now define the security notion for a secure grouping-proof protocol via the following game between a challenger and an adversary.

1. The challenger first sets up the verifier and a reader and tags to prepare for the game.

2. In the first phase of the game, the adversary collects information via 4 oracles: reader(.), tag(.), corrupt-reader(.) and verify(.). These oracles are simulated by the challenger.

3. In the second phase of the game, the challenger gives the adversary a valid proof P of n tags as a challenge. The adversary's goal is to either remove a tag from P or add a new tag to P or replace a tag in P with a different one. In this phase, the adversary is also given access to the corrupt-reader(.) oracle after the challenge proof P is constructed. However, the tag(.) oracle is not provided to the adversary after the adversary has seen P. This is to reflect our assumption that relay attack is not possible. The adversary should output a new proof P' which satisfies one of its goals.

4. The adversary wins the game if verify(P') returns 1. That is, P' is a valid co-existence proof.

Definition 18. *A grouping-proof protocol is said to be secure if the winning probability of the adversary in the above game is negligible.*

7.5 A Scalable Grouping-Proof Protocol From Secret Sharing

We now propose our grouping-proof protocol for multiple RFID tags which does not suffer from the scalability problem. We use a (n, n)-secret sharing scheme to stop messages being relayed among tags. A (n, n)-secret sharing scheme allows one to *split one secret* x into n of so called shared secrets such that x can only be reconstructed from the shared secrets only if all of n shared secrets are provided. This property is used in our proposed protocol so that each tag can sign its own random number to prove its existence. The random numbers are shared secrets generated by a secret sharing scheme. If the original secret generated by a verifier can be recovered from signed shared secrets that were backscattered by tags, then the proof of co-existence of tags is verified. A (n, n)-secret sharing scheme can be implemented as follows:

- Given a secret x, a dealer chooses $(n-1)$ random numbers $y_1, y_2, \cdots, y_{n-1}$ as the first $(n-1)$ shared secrets.

- The last shared secret y_n is computed by $y_n = x \oplus y_1 \oplus y_2 \oplus \cdots \oplus y_{n-1}$.

It is easy to see that the above protocol achieves perfect security since it is impossible to recover x without any of $y_1, y_2, \cdots$ or y_n. In addition, for each randomly chosen x, a shared secret of x is also random. This property is important to prevent replay attack as a shared secret is used as a challenge in our proposed protocol. We now describe our grouping-proof protocol below.

1. $\mathcal{V} \rightarrow \mathcal{R}$: x chosen at random. The verifier also sets a time-to-live on x such that a co-existence proof associated with x must be received within the lifespan of x (which should be approximately the time taken by one interrogation session of a reader).
2. $\mathcal{R} \rightarrow \mathcal{T}_i$: x, y_i for $i = 1, 2, \cdots, n$ where $y_1, y_2, \cdots,$ and y_n are n shared secrets of x.
3. $\mathcal{T}_i \rightarrow \mathcal{R}$: $\mathcal{T}_i, m_i = \text{MAC}_{K_i}[y_i, x]$, for $i = 1, 2, \cdots, n$.
4. $\mathcal{R} \rightarrow \mathcal{V}$: $P = (\mathcal{T}_1, y_1, m_1, \mathcal{T}_2, y_2, m_2, \cdots, \mathcal{T}_n, y_n, m_n)$.
5. $\mathcal{V}$: The verifier verifies a proof P by checking if P is received within the lifespan of $x = y_1 \oplus y_2 \oplus \cdots \oplus y_n$ and each m_i is valid MAC of the tag $\mathcal{T}_i$ on (x, y_i).

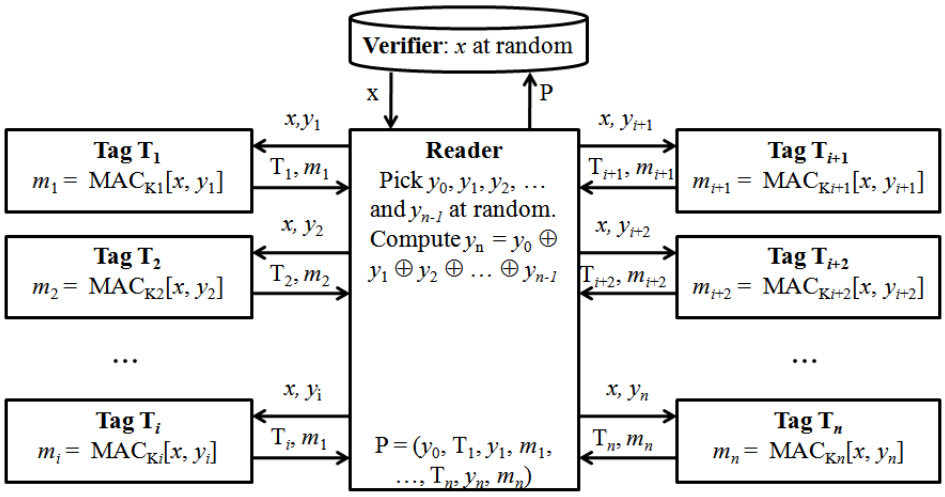

Figure 7.9: The proposed grouping-proof protocol

The above protocol is also illustrated in Fig. 7.5.

Remark. Note that, it is important to stress that we do use timestamp in our protocol to prevent a malicious reader from abusing x *i.e.*, the malicious reader can take x and use shared secrets of x on different tags at different locations and times). However, the way which timestamp is used in our protocol is very different from previous protocols. More specifically, we do not use timestamp as a challenge to a tag. Instead, only the verifier maintains timestamp for each interrogation session. This allows us to leave "time-to-live of x" to the security model. Indeed, the fact that a co-existence proof must be received within the lifespan of x fits in the assumption that a reader always executes the protocol correctly until reporting a proof to the verifier. Therefore, we can ignore the use of timestamp in the security proof of our protocol.

We now analyze the success probability of an adversary attacking our protocol. The probability is measured in terms of the success probabilities of adversaries attacking the underlying MAC and secret sharing schemes in the following theorem.

Theorem 4. *Let α be success probability of an adversary attacking the underlying MAC scheme. Let ϵ be the success probability of an adversary that attacks our proposed grouping-proof protocol, then we have:*

$$\epsilon = O\left(\alpha + 2^{-\frac{l}{2}}\right)$$

where l is the bit length x and d is the number of tags in the tag database.

Proof. Let $\mathcal{A}$ be the adversary that attacks our proposed grouping-proof protocol. Given a challenge $P = (\mathcal{T}_1, y_1, m_1, \mathcal{T}_2, y_2, m_2, \cdots, \mathcal{T}_n, y_n, m_n)$ and let $x = y_1 \oplus y_2 \oplus \cdots \oplus y_n$, $\mathcal{A}$ wants to achieve one of the following goals:

- Construct a co-existence proof $P' = (\mathcal{T}_1^*, y_1^*, m_1^*, \mathcal{T}_2^*, y_2^*, m_2^*, \cdots, \mathcal{T}_n^*, y_n^*, m_n^*)$ such that $\{\mathcal{T}_1, \mathcal{T}_2, \cdots, \mathcal{T}_n\} \neq \{\mathcal{T}_1^*, \mathcal{T}_2^*, \cdots, \mathcal{T}_n^*\}$; $y_1^* \oplus y_2^* \oplus \cdots \oplus y_n^* = x$; and $m_1^*, m_2^*, \cdots$ and m_n^* are valid MACs of $\mathcal{T}_1^*, \mathcal{T}_2^*, \cdots$ and $\mathcal{T}_n^*$ on $(y_1^*, x), (y_2^*, x), \cdots$ and (y_n^*, x), respectively. In other words, $\mathcal{A}$ succeeds when it can replace at least one tag that is actually in the communication range of the reader by another tag. We call this type of adversary Type-I adversary.

- Construct a co-existence proof $P' = (\mathcal{T}_1^*, y_1^*, m_1^*, \mathcal{T}_2^*, y_2^*, m_2^*, \cdots, \mathcal{T}_{n-1}^*, y_{n-1}^*, m_{n-1}^*)$ such that the cardinality of $\{\mathcal{T}_1, \mathcal{T}_2, \cdots, \mathcal{T}_n\} \setminus \{\mathcal{T}_1^*, \mathcal{T}_2^*, \cdots, \mathcal{T}_{n-1}^*\}$ is 1; $y_1^* \oplus y_2^* \oplus \cdots \oplus y_{n-1}^* = x$; and $m_1^*, m_2^*, \cdots$ and m_{n-1}^* are valid MACs of $\mathcal{T}_1^*, \mathcal{T}_2^*, \cdots$ and $\mathcal{T}_{n-1}^*$ on $(y_1^*, x), (y_2^*, x), \cdots$ and (y_{n-1}^*, x), respectively. In other words, the adversary can remove a tag from P. We call this type of adversary Type-II adversary.

- Construct a co-existence proof $P' = (\mathcal{T}_1, y_1^*, m_1^*, \mathcal{T}_2, y_2^*, m_2^*, \cdots, \mathcal{T}_n, y_n^*, m_n^*, \mathcal{T}_{n+1}^*, y_{n+1}^*, m_{n+1}^*)$ such $y_1^* \oplus y_2^* \oplus \cdots \oplus y_n^* \oplus y_{n+1}^* = x$; and $m_1^*, m_2^*, \cdots, m_n^*$ and m_{n+1}^* are valid MACs of $\mathcal{T}_1, \mathcal{T}_2, \cdots, \mathcal{T}_n$ and $\mathcal{T}_{n+1}^*$ on $(y_1^*, x), (y_2^*, x), \cdots, (y_n^*, x)$ and (y_{n+1}^*, x), respectively. In other words, the adversary can add the tag $\mathcal{T}_{n+1}^*$ to P. We call this type of adversary Type-III adversary.

In the first phase of the attack, $\mathcal{A}$ are given access to three oracles: the tag(.) oracle, the reader(.) oracle and the verify(.) oracle. The corrupt-reader(.) oracle is not needed as $\mathcal{A}$ can eavesdrop x itself from challenges sent to tags (except that $\mathcal{A}$ can control the reader after seeing the challenge P, however this does not affect the analysis here). The tag(.) oracle is essentially a MAC oracle as it outputs MAC on an input value together with a tag ID. In the second phase of the attack, the adversary can only control the reader after seeing the challenge P. No oracle access is given in this phase. As usual, we limit the number of calls to oracles and running time of the adversary to be polynomial in security parameters. We analyze the success probability of each type of the adversary below.

Type-I Adversary: We distinguish two cases of Type-I adversary as follows:

- **Case 1**: none of (y_i^*, x) for $i = 1, 2, \cdots, n$ has not been asked to the tag(.) oracle. In this case, $\mathcal{A}$ is essentially a MAC forger with m_i^* is a forged MAC. Indeed, if $\mathcal{A}$

can forge a MAC, then it is obvious to attack the proposed grouping-proof protocol by constructing a forged MAC on one of (y_i, x) for $i = 1, 2, \cdots, n$ such that the forged MAC is a valid MAC of a tag not in $\{\mathcal{T}_1, \mathcal{T}_2, \cdots, \mathcal{T}_n\}$. Therefore, the success probability of $\mathcal{A}$ is bounded by the success probability of the MAC adversary.

- **Case 2**: at least one of (y_i^*, x) for $i = 1, 2, \cdots, n$ has been asked to the tag oracle. We only consider the case that the adversary try to replace one tag in P with another tag. But it can be easily generalized to the case of replacing more than one tag. Since $\mathcal{A}$ is not supposed to forge a MAC (otherwise, it is easier to attack by executing the scenario of the adversary in the first case) and the $\mathsf{tag}(.)$ oracle is not provided in the second phase of the attack, $\mathcal{A}$ can only hope that its query to the tag oracle with (y_i^*, x) results in $(\mathcal{T}_i^*, m_i^*)$ such that $\mathcal{T}_i^*$ is not among $(\mathcal{T}_1, \mathcal{T}_2, \cdots, \mathcal{T}_n)$ and y_i^* constitutes a valid shared secret. However, because the underlying (n, n)-secret sharing scheme is perfectly secure and x is randomly chosen for each session, y_i^* has to be one of $y_1, y_2, \cdots, y_n$. In other words, $\mathcal{A}$ succeeds only if one of the pairs (y_i, x) for $i = 1, 2, \cdots, n$ has been queried to the $\mathsf{tag}(.)$ oracle in the querying phase such that the returned tuple $(\mathcal{T}_i^*, m_i^*)$ satisfies the adversary's goal. As shared secrets are randomly distributed and there are $(d - n)$ candidate tags for $\mathcal{T}^*$, the success probability of $\mathcal{A}$ is $\frac{d-n}{d} 2^{-\frac{l}{2}}$.

Type-II Adversary: Using the same analysis for Type-I adversary, we can see that the best option that adversary can succeed is to forge a MAC. For example, if the adversary wants to remove $\mathcal{T}_n$ from P, it can forge a MAC of $\mathcal{T}_{n-1}$ on $(y_{n-1} \oplus y_n, x)$. The resulting proof P' is $(\mathcal{T}_1, y_1, m_1, \mathcal{T}_2, y_2, m_2, \cdots, \mathcal{T}_{n-1}, y_{n-1}^*, m_{n-1}^*)$ where $y_{n-1}^* = y_{n-1} \oplus y_n$ and m_{n-1}^* is the forged MAC. Otherwise, the adversary would have to hope that $(y_{n-1} \oplus y_n, x)$ was queried to the $\mathsf{tag}(.)$ oracle during the querying phase. To conclude, the success probability of Type-II adversary is bounded by $\alpha + \frac{n}{d} 2^{-\frac{l}{2}}$.

Type-III Adversary: The success probability of Type-III adversary can also be analyzed similarly. In particular, if the adversary can forge a MAC, he can add a tag $\mathcal{T}_{n+1}^*$ to P by forging two MACs of $\mathcal{T}_n$ and $\mathcal{T}_{n+1}^*$ on (y_n^*, x) and (y_{n+1}^*, x), respectively, such that $y_n^* \oplus y_{n+1}^* = y_n$. The forged proof P' is $(\mathcal{T}_1, y_1, m_1, \mathcal{T}_2, y_2, m_2, \cdots, \mathcal{T}_n, y_n^*, m_n^*, \mathcal{T}_{n+1}^*, y_{n+1}^*, m_{n+1}^*)$ which should be correctly verified by the verifier. Therefore, we can obtain the success probability of Type-III adversary as $\frac{1}{2}\alpha + \frac{d-n}{d} 2^{-\frac{l}{2}}$.

Combing the success probabilities of three types of the adversary, we complete the proof. □

Theorem 4 suggests that if the underlying MAC scheme is secure, *i.e.,* α is negligible, and l is long enough, then the success probability of an adversary attacking our proposed grouping-proof for RFID tags protocol is negligible. We conclude that the proposed grouping-proof scheme is secure.

We now compare our proposed scheme with previous protocols with respects to performance assuming that we want to produce a co-existence proof of n tags.

Protocol	**Number of Relaying Messages**	**Cost of Generating Proof**
Yoking-Proof	$n(n-1)$	$2n(n-1)$ MACs
2nd Protocol [36]	n	n MACs 1 Encryption
Protocol in [53]	$n(n-1)$	$2n(n-1)$ MACs
1st Protocol [54]	$n(n-1)$	$2n(n-1)$ MACs 1 Encryption
1st Protocol in [65]	$n(n-1)$	4n(n-1) $f(.)$ Evaluations
Proposed Protocol	0	n MACs

Table 7.1: Comparison of Performance

7.6 Countermeasure against Mafia Fraud Attack for Grouping-Proof Protocols

In order to prevent mafia fraud attack on grouping-proof protocol, we should prevent tag location from being forged. One can use a distance-bounding protocol to verify the tag location by measuring time taken by one querying session. One of such protocol is discussed in section 3.3. However, it is not sufficient to measure the time taken by one protocol session. We need to measure the time taken to query each individual tag. We can implement the countermeasure as follows:

1. $\mathcal{R}$: Start clock.

2. $\mathcal{R} \rightleftarrows \mathcal{T}_1$: Query the first tag.

3. $\mathcal{R}$: Stop clock and include $\mathcal{T}_1$ in the co-existence proof only if $\mathcal{T}_1$'s response is received within a pre-defined amount of time.

4. $\mathcal{R}$: Start clock

5. $\mathcal{R} \rightleftarrows \mathcal{T}_2$: Query the second tag.

6. $\cdots$

8. Conclusion and Future Work

RFID security is a relatively new research area. Within less than a decade, a large number of research papers dealing with security issues of RFID technology have appeared. This thesis deals with some of open problems in RFID security. The contribution of this thesis is three-fold:

- We have presented HB* protocol which can prevent GRS as well as general man-in-the-middle attacks. Our proposed protocol can be seen as a combination of two instances of the HB protocols and one instance of the HB$^+$ protocol. HB* offers a number of advantages over other improved HB$^+$ protocols presented in [67, 68] including: The security of HB* does not rely on any additional primitive; The design of HB* enables the use of the strongest instance of the LPN problem, *i.e.,* the noise factor is exactly $\frac{1}{2}$; HB* is perfectly complete; and HB* can be used as an authenticated key exchange protocol. Since HB* is secure and efficient, it can be used to provide authentication for low-cost devices including RFID tags.

- We have showed that the provably secure RFID authentication protocol O-FRAP and its simplified version O-RAP are vulnerable to DoS attack. In particular, we pointed out that it is trivial to abuse the back-end server's computational resources in O-FRAP and O-RAP. We then presented two improved protocols of O-FRAP and O-RAP which we call O-FRAP$^+$ and O-RAP+, respectively. The most important point in the design of O-FRAP$^+$ and O-RAP$^+$ is that the server should be authenticated first by using a fixed secret key. By doing so, it is possible to not only prevent DoS attack but also remove the need for storing two versions of secret key and pseudonym for each tag in the server's database. Our approach of two-phase authentication can be applied to many other RFID authentication protocols which use a similar design to O-FRAP (that is, the tag is authenticated and identified first). In some applications where mutual authentication between tag and server/reader is not needed, using two-phase authentication can still help a reader to prevent unwanted information to travel to the back-end server.

- We have presented a grouping-proof for RFID tags protocol based on secret sharing.

Our proposed protocol solves the scalability issue of previous protocols by removing the need to relay messages among tags by a reader. We also define a security model for a secure grouping-proof protocol. The proposed security model deals with the case of untrustful readers in a proper way. In particular, we cannot assume that a reader is totally untrustful. Instead, it should be assumed that a reader is trusted to execute a grouping-proof protocol correctly but may behave maliciously when reporting a co-existence proof of tags to the verifier. We also address the impact of mafia fraud attack on the security of a grouping-proof protocol. Finally, we showed that the proposed grouping-proof protocol satisfies the security notion defined in this thesis.

Even though many works have been done to counter security threats to RFID technology, many issues are still unsolved and some others need further investigation. In our opinion, some of outstanding issues are:

- *Lack of research on lightweight cryptographic primitives*: Most of cryptographic protocols for RFID make use of a pseudorandom number generator. Yet, almost no work has focused on designing and analyzing a pseudorandom number generator for RFID tags. The same problems applies to other cryptographic primitives including symmetric encryption and MAC.

- *Study on possibility and impossibility of certain cryptographic tasks for RFID*: All of authentication protocols for RFID that provide forward security employ the so-called *interactive key-evolving protocol* to update the secret key at the tag and the server/reader. That is, the tag and the reader both updates their shared secret keys at the end of each authentication session. However, updating the key interactively often leads to de-synchronization of the shared secret key. As pointed out in [70], many forward secure RFID authentication protocols are subject to the de-synchronization attack (which may imply the loss of forward security). We suspect that it is impossible to realize a robust interactive key-evolving protocol that is secure against de-synchronization of secret. Another interesting problem to investigate is the possibility of secure grouping-proof protocols with an off-line verifier. We also suspect this kind of protocol might be impossible to realize.

- *Study on new security models for RFID*: Known security models for RFID like Vau-

denay's model ignore the RFID reader as a party of an RFID system. In particular, the level of trust on the RFID reader should play an important role in analyzing the security of a protocol. We argue that this might lead to significant separation between theoretical security and real-world security. We need to develop a new security model or extend existing models that take the RFID reader as an indispensable entity of an RFID system (rather than combine it with the back-end server).

요 약 문

RFID 태그를 위한 암호 프로토콜에 관한 연구

RFID (Radio Frequency Identification)는 모든 사물에 저렴하고 무선으로 읽고 쓸 수 있는 태그 (RFID 태그)를 부착해 공급망관리 (supply chain management)와 같은 새로운 응용 분야를 이끄는 전도유망한 기술이다. 각각의 RFID 태그는 태그가 부착된 사물에 대한 자세한 정보를 나타내기 위해 고유한 문자열로 구성된 개체 식별자를 저장한다. 이러한 특성으로 인해 RFID는 공급망관리만이 아니라 개별 사용자들의 주변 환경에 위치한 다양한 사물을 식별 및 추적할 수 있게 해준다. 아이러니하게도 RFID의 고유한 특징이 RFID가 실생활에 적용되는데 걸림돌이 될 수 있는 보안 취약점을 유발한다.

- *Tag Cloning*: RFID가 널리 사용된다면, 판매상품과 같은 주변 사물을 인지하기 위해 RFID에 의존하게 된다. 그러나 RFID에 저장된 개체 식별자는 RFID 리더와 태그 간의 통신 도청 혹은 호환성이 있는 RFID 리더를 통해 손쉽게 획득 가능하다. 따라서, RFID 태그는 복제되어 위조된 상품에 부착되어 진품으로 여겨질 수 있다.

- *사용자 프라이버시 침해*: 개체 식별자의 고유성과 폭넓은 유효성은 최종 사용자의 프라이버시를 침해할 수 있다. RFID 태그가 널리 보급되어 대다수의 사람들은 RFID 태그가 부착된 다수의 제품을 소지하면, 호환성이 있는 RFID 리더를 소지한 악의적인 공격자로 하여금 개별 사용자가 소지한 제품을 식별하거나 개별 사용자의 위치를 추적할 수 있다.

이러한 보안 취약점을 해결하기 위해 RFID 리더와 태그 통신뿐만 아니라, RFID 기술의 모든 계층에 암호프로토콜을 통합해야 한다. 이를 위해 본 논문에서는 3가지 암호프로토콜을 제안한다. 먼저, 중간자 공격에 내성을 지니며, 경량화된 인증 프로토콜 HB*를 제안하였다. 둘째, Tri Van Le *et al.*이 제안한 O-FRAP와 O-RAP가 DoS 공격에 취약한 점을 지적한 다음 해당 프로토콜들을 2단계 인증 기법을 통해 개선하였다. 이는 다수의 인증 프로토콜들과 유사하게 O-FRAP와 O-RAP도 RFID 태그 식별을 위해 Back-end 서버가 자신의 데이터베이스 전수 검사하기에 DoS 공격에 악용될 수 있다. 이를 해결하기 위해 RFID 태그는 Back-end 서버의 데이터베이스에 존재여부를 인증하고 Back-end 서버는 RFID 리더에 의해 태그가 정확하게 확인된 여부를 인증하고 식별하도록 하였다.

마지막으로 한번의 스캔을 통해 다수의 태그가 동일한 장소에 있는지를 파악하는데 활용되는 grouping-proof 프로토콜의 확장성 이슈를 해결해 새로운 grouping-proof protocol을 제안하였으며, 해당 프로토콜의 보안 모델을 만들어 제안된 프로토콜의 안전성을 분석하였다.

References

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

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

[3] EPCglobal Inc, *Class 1 Generation 2 UHF Air Interface Protocol Standard*, Version 1.2.0, Available at `http://www.epcglobalinc.org/standards/uhfc1g2`.

[4] RSA Laboratories, *PKCS #1: RSA Cryptography Standard*, Available at `http://www.rsa.com/rsalabs/node.asp?id=2125`.

[5] Whitfield Diffie and Martin Hellman, *New directions in cryptography*, IEEE Transactions on Information Theory 22, 644-654, 1976.

[6] Ralph C. Merkle, *Secrecy, authentication, and public key systems*. Stanford Ph.D. Thesis, Available at `http://www.merkle.com/papers/Thesis1979.pdf`, 1979.

[7] Shafi Goldwasser and Silvio Micali, *Probabilistic encryption*, Journal of Computer and System Sciences, 28:270-299, 1984.

[8] Whitfield Diffie, *The first ten years of public-key cryptography*, Proceedings of the IEEE 76 (1988), 560-577, 1988.

[9] Oded Goldreich and L.A. Levin, *Hard-core Predicates for Any One-Way Function*,In the Proceedings of The 21st ACM Symposium on Theory of Computation, pages 25-32, 1989.

[10] Mihir Bellare and Philip Rogaway, *Optimal Asymmetric Encryption - How to encrypt with RSA*, In the Proceedings of Eurocrypt'94, Springer-Verlag, LNCS 950, pp. 92-111, 1995.

[11] Mihir Bellare, Ran Canetti and Hugo Krawczyk, *Keying Hash Functions for Message Authentication*, In the Proceedings of CRYPTO'96, Springer-Verlag, LNCS 1109, pp. 1-15, 1996.

[12] Mihir Bellare and Philip Rogaway, *The exact security of digital signatures: How to sign with RSA and Rabin* , In the Proceedings of Eurocrypt'96, Springer-Verlag, LNCS 1070, pp. 399-416, 1996.

[13] Johan Hastad, *Some Optimal Inapproximability Results*, In the Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing, ACM Press, pp. 1-10, May, 1997.

[14] Michael Kearns, *Efficient noise-tolerant learning from statistical queries*, In the Journal of ACM, Volume 45, Issue 6, ACM Press, pp. 983-1006, November, 1998.

[15] Avir Blum, Adam Kalai and Hal Wasserman, *Noise-tolerant Learning, the Parity Problem, and the Statistical Query Model*, In the Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, ACM Press, pp. 435-440, 2000.

[16] Nicholas Hopper and Manuel Blum, *A Secure Human-Computer Authentication Scheme*, In the Proceedings of ASIACRYPT'01, Bart Preneel (Ed.), Springer-Verlag, LNCS 2248, pp. 149-153, 2001.

[17] Yvo Desmedt, *Major security problems with the Unforgeableable (Feige)-Fiat-Shamir proofs of identiy and how to overcome them*, In SecureCom'88, pp. 15-17, 1988.

[18] Stefan Brands and David Chaum, *Distance-Bounding Protocols*, In Advances in Cryptology EUROCRYPT'93, Volume 765 of Lecture Notes in Computer Science, Springer-Verlag, pp. 344-359, 1994.

[19] Stephen Weis, *Security and Privacy in Radio Frequency Identification Devices*, Master Thesis, Available at `http://theory.lcs.mit.edu/~sweis/masters.pdf`, May 2003.

[20] Ari Juels, Ronald Rivest, and Michael Szydlo, *The Blocker Tag: Selective Blocking of RFID Tags for Consumer Privacy*, In V. Atluri (ed.) 8th ACM Conference on Computer and Communications Security, ACM Press, pp. 103-111, 2003.

[21] Miyako Ohkubo, Koutarou Suzuki, and Shingo Kinoshita, *Efficient Hash-Chain Based RFID Privacy Protection Scheme*, In the Proceedings of International Conference on Ubiquitous Computing, Workshop Privacy, September 2004.

[22] Stephen Weis, S. Sarma, Ronald Rivest, and D. Engels, *Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems*, Proc. of the 1st Security in Pervasive Computing, LNCS 2802, pp. 201-212, 2004.

[23] Eli Biham and Rafi Chen, *Near-Collisions of SHA-0*, In the Proceedings of CRYPTO'04, Springer-Verlag, LNCS 3152, pp. 290-305, 2004.

[24] Antoine Joux, *Multicollisions in Iterated Hash Functions. Application to Cascaded Constructions*, n the Proceedings of CRYPTO'04, Springer-Verlag, LNCS 3152, pp. 306-316, 2004.

[25] Ari Juels and Stephen Weis, *Authenticating Pervasive Devices with Human Protocols*, In the Proceedings of CRYPTO'05, Victor Shoup (Eds.), Springer-Verlag, LNCS 3261, pp. 293-308, 2005.

[26] Henri Gilbert, Matthew Robshaw and Hervé Silbert, *An Active Attack Against HB^+ - A Provably Secure Lightweight Authentication Protocol*, Available at `http://eprint.iacr.org/2005/237`.

[27] Ari Juels, *Yoking-Proofs for RFID Tags*, In the Proceedings of First International Workshop on Pervasive Computing and Communication Security, IEEE Press, pp. 138-143, 2004.

[28] Xiaoyun Wang, Dengguo Feng, Xuejia Lai and Hongbo Yu, *Collisions for Hash Functions MD4, MD5, HAVAL-128 and RIPEMD*, Available at `http://eprint.iacr.org/2004/199`.

[29] Xiaoyun Wang, Hongbo Yu, Yiqun Lisa Yin, *Efficient Collision Search Attacks on SHA-0*, In the Proceedings of CRYPTO'05, Springer-Verlag LNCS 3621, pp. 1-16, 2005.

[30] Xiaoyun Wang, Yiqun Yin and Hongbo Yu, *Finding Collisions in the Full SHA-1*, In the Proceedings of CRYPTO'05, Springer-Verlag LNCS 3621, pp. 17-36, 2005.

[31] Xiaoyun Wang, Yiqun Yin and Hongbo Yu, *Collision Search Attacks on SHA-1*, Available at `http://theory.csail.mit.edu/yiqun/shanote.pdf`, 2005.

[32] Arjen Lenstra, Xiaoyun Wang and Benne de Weger, *Colliding X.509 Certificates*, Available at `http://eprint.iacr.org/2005/067`, 2005.

[33] Xiaoyun Wang and Xuejia Lai, *Cryptanalysis of the Hash Functions MD-4 and RIPEMD*, In the Proceedings of Eurocrypt'05, Springer-Verlag LNCS 3494, pp.18-36, 2005.

[34] Xiaoyun Wang and Hongbo Yu, *How to Break MD5 and Other Hash Functions*, In the Proceedings of Eurocrypt'05, Springer-Verlag, LNCS 3494, pp.1-18, 2005.

[35] Gerhard P. Hancke and Markus G. Kuhn, *An RFID distance bounding protocol*, In the 1st International Conference on Security and Privacy for Emergin Areas in Communications Networks (SECURECOMM'05), IEEE Computer Society, pp. 67-73, 2005.

[36] Junichiro Saitoh and Kouichi Sakurai, *Grouping-Proofs for RFID Tags*, In the Proceedings of AINA International Conference, IEEE Computer Society, pp. 621-624, 2005.

[37] Gildas Avoine and Philippe Oechslin, *A Scalable and Provably Secure Hash-Based RFID Protocol*, In the Proceedings of Workshop on Pervasive Computing and Communications Security - PerSec'05, March 2005.

[38] Gildas Avoine, Etienne Dysli, and Philippe Oechslin, *Reducing Time Complexity in RFID System*, In the Proceedings of Selected Areas in Cryptography (SAC)'05, Bart Preneel and Stafford Tavares (Ed.), Springer-Verlad, LNCS 3897, pp. 291–306, 2005.

[39] D. Molnar, A. Soppera and D. Wagner, *A Scalable, Delegatable Pseudonym Protocol Enabling Ownership Transfer of an RFID Tag*, In the Proceedings of Selected Areas in Cryptography (SAC)'05, Bart Preneel and Stafford Tavares (Ed.), Springer-Verlad, LNCS 3897, pp. 276–290, 2005.

[40] Tassos Dimitriou, *A Lightweight RFID Protocol to Protect against Traceability and Cloning Attacks*, In the Proceedings of SecureComm'05, September 2005.

[41] Jonathan Katz and Ji Sun Shin, *Parrallel and Concurrent Security of the HB and HB$^+$ Protocols*, Available at `http://eprint.iacr.org/2005/461.pdf`.

[42] D. Carluccio, K. Lemke, and C. Paar. *Electromagnetic side channel analysis of a contactless smart card: First results*, In the Proceedings of Workshop on RFID and Lightweight Crypto (RFIDSec05), 2005.

[43] S. Chaumette and Sauveron. *An efficient and simple way to test the security of Java Cards*, In WOSIS 2005, 3rd International Workshop on Security in Information Systems, April 2005. Miami, Fl., USA, April 2005.

[44] S. Karthikeyan, M. Nesterenko, *RFID security without extensive cryptography*, In the Proceedings of SASN'05, ACM Press, pp. 63-67, 2005.

[45] Ari Juels, *RFID Security and Privacy: A Research Survey*, In the Journal of Selected Areas in Communication (J-SAC), 24(2): pp. 381-395, February 2006.

[46] Ilan Kirschenbaum and Avishai Wool, *How to Build a Low-Cost, Extended-Range RFID Skimmer*, Available at `http://eprint.iacr.org/2006/054`.

[47] Marc P.C. Fossorier and Miodrag J. Mihaljevic and Hideki Imai and Yang Cui and Kanta Matsuura, *A Novel Algorithm for Solving the LPN Problem and its Application to Security Evaluation of the HB Protocol for RFID Authentication*, Available at `http://eprint.iacr.org/2006/197`.

[48] H.-Y. Chien, *Secure Access Control Schemes for RFID Systems with Anonymity*, In the Proceedings of 2006 International Workshop on Future Mobile and Ubiquitous Information Technologies (FMUIT'06), 2006.

[49] J. Bringer, H. Chabanne, and E. Dottax, *HB++: A Lightweight Authentication Protocol Secure against Some Attacks*, In the Proceedings of IEEE International Conference on Pervasive Services, Workshop Security, Privacy and Trust in Pervasive and Ubiquitous Computing, 2006.

[50] Pedro Peris-Lopez, Julio Cesar Hernandez-Castro, Juan M. Estevez-Tapiador, and Arturo Ribagorda, *LMAP: A real lightweight mutual authentication protocol for low-cost RFID tags*, In the Proceedings of RFIDSec'06, 2006.

[51] Pedro Peris-Lopez, Julio Cesar Hernandez-Castro, Juan M. Estevez-Tapiador, and Arturo Ribagorda, *EMAP: An efficient mutual authentication protocol for low-cost RFID tags*, In the Proceedings of IS'06, LNCS 4277, Springer-Verlag, pp. 352-361, 2006.

[52] Pedro Peris-Lopez, Julio Cesar Hernandez-Castro, Juan M. Estevez-Tapiador, and Arturo Ribagorda, *M2AP: A minimalist mutual-authentication protocol for low-cost RFID tags*, In the Proceedings of UIC'06, LNCS 4159, Springer-Verlag, pp. 912-923, 2006.

[53] Selwyn Piramuthu, *On Existence Proofs for Multiple RFID Tags*, In the Proceedings of ACS/IEEE International Conference on Pervasive Services, IEEE Computer Society, pp. 317-320, 2006.

[54] Chih-Chung Lin, Yuan-Cheng Lai, J. D. Tygar, Chuan-Kai Yang and Chi-Lung Chiang, *Coexistence Proof using Chain of Timestamps for Multiple RFID Tags*, In the Proceedings of APWeb/WAIM International Workshop, Springer-Verlag LNCS 5189, pp. 634-643, 2007.

[55] Tri Van Le, Mike Burnmester and Breno de Medeiros, *Universally Composable and Forward Secure RFID Authentication and Authenticated Key Exchange*, In the Proceedings of the 2nd ACM Symposium on Information, Computer and Communications Security, pp. 242-252, March 2007.

[56] H. Y. Chien, *SASI: A New Ultralightweight RFID Authentication Protocol Providing Strong Authentication and Strong Integrity*, IEEE Transactions on Dependable & Secure Computing, pp. 337-340, 2007.

[57] Serge Vaudenay, *On Privacy Models for RFID*, In the Proceedings of ASIACRYPT'2007, LNCS 4833, Springer-Verlag pp. 68-87, 2007.

[58] M. Hutter, S. Mangard, and M. Feldhofer. *Power and EM Attacks on Passive 13.56 MHz RFID Devices*, In the Proceedings of the Workshop on Cryptographic Hardware and Embedded Systems (CHES 2007), LNCS 4727, pp. 320-333, Springer-Verlag, 2007.

[59] J. Munilla and A. Peinado, *HB-MP: A Further Step in the HB-Family of Lightweight Authentication Protocols*, Computer Networks, doi:10.1016/j.comnet.2007.01.011, 2007.

[60] H. Y. Chien, C. H. Chen, *Mutual authentication protocol for RFID conforming to EPC class-1 generation-2 standards*, Computer Standards & Interfaces, vol. 29, pp. 254-259, 2007.

[61] Y. Oren and A. Shamir. *Remote Password Extraction from RFID Tags*, IEEE Transactions on Computers, 56(9):1292-1296, 2007.

[62] T. Plos, *Susceptibility of UHF RFID Tags to Electromagnetic Analysis*, CT-RSA 2008, LNCS 4964, pp.288-300, 2008.

[63] Selwyn Piramuthu, *Lightweight Cryptographic Authentication in Passive RFID-Tagged Systems*, IEEE Transactions on Systems, Manegement and Cybernetics, Part C 38(3), pp. 360-376, 2008.

[64] Ran Canetti, *Obtaining Universally Composable Security: Towards the Bare Bones of Trust*, Available at `http://eprint.iacr.org/2007/475`.

[65] Mike Burmester, Breno de Medeiros Rossana Motta. *Provably Secure Grouping-proofs for RFID tags*, In the 8th Proceedings of International Conference on Smart Card

Research and Advanced Applications (CARDIS 2008), IFIP WG 8.8/11.2, September 8-11, 2008.

[66] Khaled Ouafi and Raphael C.-W. Phan, *Traceable Privacy of Recent Provably-Secure RFID Protocols*, In the Proceedings of ACNS 2008, Springer-Verlag LNCS 5037, pp. 479-489, 2008.

[67] Henri Gilbert, Matthew J.B. Robshaw and Yannick Seurin, *HB#: Increasing the Security and Efficiency of HB$^+$*, Available at *http://eprint.iacr.org/2008/028*.

[68] Julien Bringer and Herve Chabanne, *http://eprint.iacr.org/2008/042*, Available at `http://eprint.iacr.org/2008/042`.

[69] Jose Carrijo, Rafael Tonicelli and Hideki Imai, *A Novel Probabilistic Passive Attack on the Protocols HB and HB$^+$*, Available at `http://eprint.iacr.org/2008/231`.

[70] Ton Van Deursen and Sasa Radomirovic, *Attacks on RFID Protocols*, Available at `http://eprint.iacr.org/2008/310`.

[71] Zbigniew Golebiewski and Krzysztof Majcher and Filip Zagorski and Marcin Zawada, *Practical Attacks on HB and HB+ Protocols*, Available at `http://eprint.iacr.org/2008/241`.

[72] Pedro Peris-Lopez, Julio Cesar Hernandez-Castro, Juan M. Estevez-Tapiador, and Arturo Ribagorda, *LAMED : A PRNG for EPC Class-1 Generation-2 RFID Specification*, Computer Standards & Interfaces, Volume 31, Issue 1, pp. 88-97, January 2009.

[73] Mike Burnmester, Tri Van Le, Brene De Medeiros and Gene Tsudik, *Universally Composable RFID Identification and Authentication Protocols*, In the ACM Transactions on Information and Systems Security, Vol. 12, No. 4, Article 21, April 2009.

[74] Chong Hee Kim and Gildas Avoine, *RFID distance bounding protocol with mixed challenges to prevent relay attacks*, Available at `http://eprint.iacr.org/2009/310`.

[75] Nishanth Chandran, Vipul Goyal, Ryan Moriarty and RafailOstrovsky, *Position Based Cryptography*, In the Proceedings of CRYPTO'09, Springer-Verlag LNCS 5677, pp. 391-407, 2009.

[76] Karl Koscher, Ari Juels, Vjekoslav Brajkovic and Tadayoshi Kohno, *EPC RFID Tag Security Weaknesses and Defenses: Passport Cards, Enhanced Drivers Licenses, and Beyond*, In the Proceedings of ACM CCS'09, ISBN: 978-1-60558-894-0, pp. 33-42, 2009.

[77] Alfred Menezes, Paul C. van Oorschot, and Scott A. Vanstone, *Handbook of Applied Cryptography*, Available at `http://www.cacr.math.uwaterloo.ca/hac/`.

[78] Oded Goldreich, *Modern Cryptography, Probabilistic Proofs and Pseudorandomness*, Published by Springer, ISBN: 3-540-64766-X.

[79] Alfred Menezes, *Elliptic Curve Public Key Cryptosystems*, Published by Kluwer Academic, ISBN: 0-7923-9368-6.

[80] Douglas R. Stinson, *CRYPTOGRAPHY: Theory and Practice*, 2nd Edition, Published by CRC Press, ISBN: 0-8493-8521-0.

[81] Williams Stalling, *CRYPTOGRAPHY AND NETWORK SECURITY: Principles and Practice*, 2nd Edition, Published by Prentice-Hall, ISBN: 0-13-869017-0.

[82] Donald E. Knuth, *The Art of Computer Programming: Seminumerical Algorithms*, Vol. 2, 3rd Edition, Published by Addison-Wesley, ISBN: 0-201-89684-2.

감사의 글

First of all, I would like to thank my advisor, Prof. Kwangjo Kim, for his guidance and support throughout the course of my study in KAIST. The freedom to choose my own research topic that Prof. Kim gave me pays an important part in the successful completion of this thesis. I also would like to thank other members of my PhD defense committee including Prof. Daeyoung Kim of KAIST, Prof. Soontae Kim of KAIST, Prof. Byoungcheon Lee of Joongbu University and Dr. Doo Ho Choi of ETRI. Their careful proof-reading and many insightful technical comments have made signficant improvement to this thesis in terms of both editorial and technical quality. I am also grateful to Prof. Jung Hee Cheon of Seoul National University for giving me important advices that helped me though both master and PhD courses. My gratitude also goes to Dr. Ton Quoc Binh of BaoViet Bank for his support and encouragement.

I also had privileged of working with many kind and enthusiastic people here in CAIS lab at KAIST. I have learnt a lot from them in many things ranging from culture to gadgets, etc. It was my big embarrassment that I could not master Korean language during my time here in Korea. There were too many occasions in which my lack of Korean proficiency would have put me in trouble. However, thank to unconditional help from my fellow Korean students, I have lived a trouble-free life in Korea. I would like to personally thank Hyunrok Lee, Jangseong Kim, Kyusuk Han, Zeen Kim, Divyan, Vo Duc Liem, Sungmok Shin, Imsung Choi, Myounghan Yoo, Junhyun Yim, Hyeran Mun, Hyewon Park, Hanyoung Noh, Minhae Kwak, Sungchul Heo, Sungjune Yoon, Sungbae Ji, Youngjoon Seo, Shangshin Lee, Jaemin Park, Jeongkyu Yang, Kangseok Kyu and CAIS lab's secretary Ms. Hyun-Kyung Park.

I have been also very lucky to be surrounded by a great community of Vietnamese students here in KAIST-ICC. There were of course unforgettable memories of many lengthy soccer matches between the "young" team and the "old" team. I have been a proud member of both teams and owned my reputation as "the stone" (true to its meaning, I'd say I am not that quick but it is hard to get around me). The next thing that I will remember the most is the amazing number of talented chefs here in VietICU. I think I was a little too eager to join any party organized by VietICU members. My bad! But, I want to say thank to all VietICU members for making my life in Korea so much more enjoyable than it should. I wish you all great successes in your future endeavors.

Last but not least, I want to express my deepest gratitude to members of my family including my father, my mother, my sister, my brother, my in-laws, and most of all, my wife, Ha Thao, and my son, Viet Anh. There were many times when my papers were rejected and I wanted to quit, their love and constant encouragement always reminded me that I could do it if I really want it. This thesis is a sole effort of mine. But, in a very real sense, it would not have happened if my family was not always there for me. I therefore dedicate this thesis to all of my family members.

Curriculum Vitae

Name	:	Dang Nguyen Duc
Date of Birth	:	May 28, 1977
Birthplace	:	Hoa Binh, Thuong Tin, Ha Tay, VIETNAM
Domicile	:	105-B1D, Thanh Cong Street, Ba Dinh District, Hanoi, VIETNAM
Address	:	R504, 119 Munjiro, Yusong-gu, Daejeon, 305-732 KOREA
E-mail	:	nguyenduc@kaist.ac.kr/nguyenduc@icu.ac.kr

Educations

1995. 9. – 2000. 5. Computer Science, Hanoi University of Technology, VIETNAM (B.S).

2001. 6. – 2003. 8. Computer Science, Information and Communications University (now part of KAIST), REPUBLIC OF KOREA (M.S).

2004. 3. – 2009. 12. Computer Science (Information Security), Department of Information and Communications Engineering, KAIST, REPUBLIC OF KOREA (PhD).

Career

1999. 7. – 2001. 5. Software Developer at HiPT Co., Hanoi, Vietnam.

2001. 8. – 2001. 12. Teaching Assistant for ICE0721 (*Elliptic Curve Cryptography*) by Prof. Jung Hee Cheon at Information and Communications University (now part of KAIST).

2003. 2. – 2003. 6. Teaching Assistant for ICE1210 (*Algorithm Design and Analysis*) by Prof. Pandu Rangan at Information and Communications University (now part of KAIST).

2003. 9. – 2004. 2. Software Developer at HiPT Co., Hanoi, Vietnam.

2005. 6. – 2005. 8. Teaching Assistant for ICE0715 (*Mobile Computing*) by Prof. Chansu Yu at Information and Communications University (now part of KAIST).

2005. 8. – 2005. 12. Teaching Assistant for ICE0715 (*Programming Fundamentals II - C++*) by Prof. Lee Sai Peck at Information and Communications University (now part of KAIST).

2006. 2. – 2006. 6. Teaching Assistant for ICE1311 (*Automata and Theory of Computation*) by Prof. Ji-Ae Shin at Information and Communications University (now part of KAIST).

2006. 2. – 2006. 6. Teaching Assistant for ICE0124C (*Programming Fundamentals I - Java*) by Prof. Lee Sai Peck at Information and Communications University (now part of KAIST).

2006. 7. – 2008. 6. Research Assistant in *Research on ID-based Cryptography Technique and Applications for 4G Mobile Network Project* Funded by Samsung Electronics.

2006. 8. – 2006. 12. Instructor for ICE0125B (*Programming Fundamentals II - C*) at Information and Communications University (now part of KAIST).

2007. 2. – 2007. 6. Instructor for ICE0124A (*Programming Fundamentals I - Java*) at Information and Communications University (now part of KAIST).

2007. 8. – 2007. 12. Instructor for ICE0125C (*Programming Fundamentals II - C*) at Information and Communications University (now part of KAIST).

2008. 1. – 2008. 12. Research Assistant in *Research on RFID Security & EPCglobal RFID Standardization Project* Funded by ETRI.

2008. 2. – 2008. 6. Instructor for ICE0124C (*Programming Fundamentals I - Java*) at Information and Communications University (now part of KAIST).

2008. 2. – 2008. 6. Teaching Assistant for ICE1311 (*Automata and Theory of Computation*) by Prof. Ji-Ae Shin at Information and Communications University (now part of KAIST).

2008. 8. – 2008. 12. Instructor for ICE0125A (*Programming Fundamentals II - C*) at Information and Communications University (now part of KAIST).

2008. 8. – 2008. 12. Teaching Assistant for ICE1200B (*Data Structures*) by Prof. Ji-Ae Shin at Information and Communications University (now part of KAIST).

2009. 1. – 2009. 12. Research Assistant in *Research on RFID Security & Next Generation RFID Standard Project* Funded by ETRI.

Publications

1. **Dang Nguyen Duc**, Hyunrok Lee and Kwangjo Kim, *Toward Designing Provably Secure Cryptographic Protocols for RFID Tags*, Auto-ID Lab Whitepaper Series, Available at `http://www.autoidlabs.org/rssdetail/dir/article/1/322/`.
2. **Dang Nguyen Duc**, Hyunrok Lee and Kwangjo Kim, *Enhancing Security of Class I Generation 2 RFID against Traceability and Cloning*, In Auto-ID Lab Whitepaper Series on Networked RFID Systems and Lightweight Cryptography Raising Barriers to Product Counterfeiting, Springer Berlin Heidelberg, ISBN: 978-3-540-71641-9.
3. **Dang Nguyen Duc** and Kwangjo Kim, *On the Security of RFID Group Scanning Protocols*, IEICE Transaction on Information and Communications Systems, Vol. E93-D, No. 3, Mar. 2010.
4. **Dang Nguyen Duc** and Kwangjo Kim, *Defending RFID Authentication Protocols against DoS Attacks*, Elsevier's Journal of Computer Communications (Under Review).
5. **Dang Nguyen Duc** and Kwangjo Kim, *Security Analysis of A Remote User Authentication Protocol by Liao and Wang*, Available at `http://eprint.iacr.org/2009/610`.
6. **Dang Nguyen Duc** and Kwangjo Kim, *A Secure Lightweight Authentication Protocol Based on Hard Learning Problem*, Elsevier's Journal of Computer Standards & Interfaces (Under Review).
7. **Dang Nguyen Duc** and Kwangjo Kim, *Grouping-Proof Protocol for RFID Tags: Security Definition and Scalable Construction*, To Appear in The Proceedings of SCIS 2010 (Abstract Only), 2010.
8. **Dang Nguyen Duc**, Divyan M. Konidala, Hyunrok Lee and Kwangjo Kim, *Open Issues in RFID Security*, RFID Security and Cryptography 2009 (Invited Paper).
9. **Dang Nguyen Duc** and Kwangjo Kim, *How to Exchange Secret on Low-cost Devices*, Triangle Symposium on Advanced ICT 2008 (TriSAI 2008), Oct. 6-9, 2008.
10. Divyan M. Konidala, Hyunrok Lee, **Dang Nguyen Duc** and Kwangjo Kim, *Security and User Privacy for Mobile-RFID Applications in Public Zone*, Triangle Symposium on Advanced ICT 2008 (TriSAI 2008), Oct. 6-9, 2008.

11. **Dang Nguyen Duc** and Kwangjo Kim, *Securing HB+ against GRS Man-in-the-Middle Attack*, In Proceedings of SCIS 2007, Abstracts pp.123, Jan. 23-26, 2007.
12. **Dang Nguyen Duc** and Kwangjo Kim, *Human Authentication Protocol for Distributed Computing Environment*, In Pre-Proceedings of WISA 2006, pp.367-372, Aug. 28-30, 2006.
13. **Dang Nguyen Duc**, Jaemin Park, Hyunrok Lee and Kwangjo Kim, *Enhancing Security of EPCglobal Gen-2 RFID Tag against Traceability and Cloning*, In Proceedings of SCIS 2006, Abstracts pp.97, Jan. 17 20, 2006.
14. **Dang Nguyen Duc**, Zeen Kim and Kwangjo Kim, *A New Provably Secure Transitive Signature Scheme*, In Proceedings of SCIS 2005, Jan.25 28, 2005.
15. Divyan M. Konidala, **Dang Nguyen Duc**, Dong-man Lee and Kwangjo Kim, *A Capability-based Privacy-preserving Scheme for Pervasive Computing Environments*, In Proceedings of IEEE PerSec'2005, pp.136-140, Mar. 8 12, 2005.
16. **Dang Nguyen Duc**, Kyusuk Han, Zeen Kim and Kwangjo Kim, *A New Transitive Signature Scheme based on RSA-based Security Assumptions*, In Proceedings of Industrial and Short-Papers Track in ACNS2005, pp.165-175, Jun. 7 10, 2005.
17. **Dang Nguyen Duc**, Jung Hee Cheon and Kwangjo Kim, *A Forward-Secure Blind Signature Scheme Based on the Strong RSA Assumption*, In Proceedings of ICICS'03, Springer-Verlag LNCS 2836, pp.11-21, Oct.10 13, 2003.
18. **Dang Nguyen Duc** and Kwangjo Kim, *Grouping-Proof Protocol for RFID Tags: Security Definition and Scalable Construction*, In the Proceedings of CISC-Winter'09, 2009 (Excellent Paper Award).
19. **Dang Nguyen Duc** and Kwangjo Kim, *O-FRAP+: Enhancing Security of a Forward secure RFID Authentication Protocol*, In the Proceedings of CISC-Summer'09, pp.406-410, 2009.
20. **Dang Nguyen Duc** and Kwangjo Kim, *A Lightweight Key Agreement Protocol Based on LPN Problem*, In the Proceedings of CISC-Winter'07, vol.17, no.2, pp.709-712, 2007.
21. **Dang Nguyen Duc** and Kwangjo Kim, *Secure HB+ against Man-in-the-middle Attacks*, In the Proceedings of CISC-Winter'06, pp.265-272, 2006.
22. **Dang Nguyen Duc**, Jaemin Park, Hyunrok Lee and Kwangjo Kim, *A Simple Secure Communication Protocol for RFID Devices*, In the Proceedings of CISC-Winter'2005, pp.254-259, 2005.

23. **Dang Nguyen Duc** and Kwangjo Kim, *Forward-Secure Blind Signature Scheme Based on the Strong RSA Assumption*, In the Proceedings of CISC-Summer'03, pp.21-25, 2003 (Excellent Paper Award).