2015-02-03

Secure Distance Bounding

Zheng, Xifan

UNIVERSITY OF CALGARY

Secure Distance Bounding

by

Xifan Zheng

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

CALGARY, ALBERTA

January, 2015

Abstract

Location (or distance) information of a device plays a significant role in current location-based systems. How to determine the location of a device or verify the location claims made by a device is challenging, as devices are untrusted and may have an incentive to claim a false location. In secure localization and positioning system, the trusted verifier(s) interact with the untrusted prover to determine its location or validate its location claim. In this thesis, we mainly focus on one of the prominent areas of such systems: distance bounding. Distance (upper) bounding (DUB) allows a verifier to verify whether a proving party is located within a certain distance bound. DUB protocols have many applications in secure authentication and location-based services.

This thesis has two main contributions. The first is that we consider the dual problem of distance lower bounding (DLB), where the prover proves it is outside a distance bound from the verifier. We motivate this problem through a number of application scenarios and model security against distance fraud (DF), Man-in-the-Middle (MiM), and collusion fraud (CF) attacks. We prove impossibility of security against these attacks without making physical assumptions. We propose approaches to the construction of secure protocols under reasonable physical assumptions and give detailed design of a DLB protocol with security analysis using our proposed model. This is the first treatment of the DLB problem in the untrusted prover setting with a number of applications, raising new research directions and opportunities in location based services. We discuss our results and propose directions for future research.

One of the main assumptions which DUB protocols rely on is that the time that the prover spends in receiving the challenge, processing, and sending the response is negligible compared to the propagation time of the signal between the prover and verifier. This strict requirement poses difficulties on the implementation of DUB protocols and limits the possible

development of applications for distance bounding as well. The second contribution in this thesis is that we design a novel one-round DUB protocol that uses one-way transmission time to estimate the distance instead of round-trip time, so that the assumption of negligible processing time is not required any longer. In order to prove the security, we formalize the notion of time in a distributed environment with adversarial users. In this model, time is implemented by a trusted party broadcasting unpredictable timestamps at a high frequency. We show that the timestamp is proved to be fresh and unpredictable. We then extend the time model to formalize DUB protocols and define corresponding attacks. Finally, we prove the security of our proposed distance bounding protocol and discuss potential issues when implementing such protocol.

Besides these two main contributions, we also have the following two contributions: (1) we identify and analyze a new attack: false rejection attack, which poses serious threat to proximity-based authentications that uses distance bounding protocol for proximity evaluation; (2) We investigate the feasibility of replay attack in context-based proximity authentication using real-world data.

Acknowledgements

Foremost, I would like to express my sincere gratitude to my advisor Dr. Rei Safavi-Naini for the continuous support of my study and research. Doing research is a long journey, but, her patience, motivation, enthusiasm, and immense knowledge lead me to enjoy this journey. She gave me the freedom to choose research topics that I am interested in and always encouraged me to think out of box. Her guidance helped me in all the time of research and writing of this thesis. I enjoyed a lot when working under her supervision and I am very pleased with the results we have achieved together. I could not have imagined having a better supervisor and mentor for my M.Sc study.

I would like to thank my committee members, Professor Gerard Lachapelle and Professor Mea Wang, for attending my defense and providing insightful comments. And I gratefully acknowledge the Department of Computer Science, University of Calgary, iCore (Informatics Circle of Research Excellence), and AITF (Alberta Innovates Technology Futures) for providing me the financial assistance to support my M.Sc work.

My sincere thanks also goes to Dr. Hadi Ahmadi and Dr. Mohsen Alimomeni who provided me valuable comments during my work. I also want to thank my colleagues who gave me guidance, support, and company, especially Deb Angus who always shared her interesting experience with me and designed my ISPIA profile page.

I am grateful to my friends: Qiao Wang, Yuan Gao, Yumeng Wang, Peng Chen, Xiaoyang Liu, Tian jin, Ocean, Yuki, Liang Chen, Yang Liu, Muyun Cao, Wanting Mao who gave me unlimited support, courage, and happiness during my stay in Calgary. I would never forget the beautiful moments I shared with them.

Last but not least, I would like to thank my parents Lixia Liu and Minfeng Zheng, for giving birth to me and supporting me spiritually throughout my life. It is because of their unflagging love and unconditional support, I can be myself and pursue my dream.

Table of Contents

List of Tables

List of Figures and Illustrations

Chapter 1

Introduction

Nowadays, many smart devices are capable of using various positioning techniques, such as Global Positioning System (GPS) [41] and cellular network [19] to determine the position of a device. This leads to the rapid development of location-based applications (LBS) that provide useful services and applications based on the location information[1].

A drawback of current positioning framework is that it is the client itself that performs the positioning. Hence, it is difficult for other entities to securely determine the exact location of the client device or verify the correctness of a location claim that is made by a client device. Many attacks have been implemented to show this possibility of cheating on the location [78, 42], which is problematic to many LBS applications. For example, in many LBS applications, proximity to certain locations is the prerequisite to some form of advantages (e.g., customers earn reward points when visiting coffee shops or obtain free coupons when checking in at a restaurant's location). Hence, there is an incentive for users to engage in location cheating [40] by lying about their location to the verifier. Such location cheating enables malicious users to obtain unqualified benefits without physically showing up at the venue. Malicious users can even repeat location cheating to stack all benefits together. Therefore, secure positioning technique is in urgent demand for LBS verifier to verify the correctness of the location claims that clients have made. Location information has also been extensively used in peer-to-peer scenarios. For example, a LBS social network system [17] uses location information to evaluate the proximity of two users. Users can only connect to each other when they are within certain proximity level. The other example is that the mobile device can control their visibility to other devices based on the location information. In [55], the author describes a system in which a device only reveals its information to those

[1]The term location used in this thesis includes all relative information, such as distance to a certain venue.

peer devices that are present at the same location. Therefore, devices also need a secure way to verify if a location claim that is made by a peer device is in fact genuine.

Secure localization and positioning systems allow users to securely verify the location claims that are made by other users. The goal of such systems is to securely establish a location of the sender or the receiver that is involved in the protocol. The estimation of physical location is computed based on physical properties of the signal or the transmission channel: Message Received Signal Strength (RSS) [26], Time of Flight (TOF) [61], Angle of Arrival (AOA) [57], or a combination of the above[31]. In all cases, there is a trusted infrastructure that verifies the location or distance claim of other entities, by executing a pre-determined protocol. This infrastructure usually consists of a number of trusted verifiers with known locations. The prover is a device which establishes its location or distance to the trusted verifier by participating in an interactive protocol with verifiers. Most protocols consist of a challenge-response phase, in which the verifier sends a specific challenge and expects a valid response from the prover. The verifier then validates the location or distance claim that is made by the prover based on the response and physical properties associated with it (e.g., RSS, TOF, AOA).

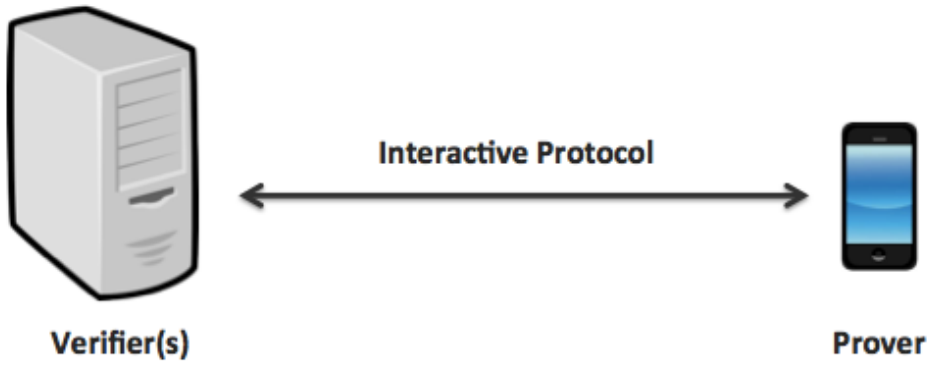

Figure 1.1: Secure Positioning System

1.1 Distance Bounding Protocol

Distance bounding (DB) is one of the prominent research areas in secure localization and positioning system. Typically, distance bounding protocols allow the verifier to ensure that the prover is within a certain distance[2] from the verifier. However, its original proposal aims to prevent relay attack instead of providing location information to the verifier. Brands and Chaum [11] introduced the notion of distance bounding (DB) to prevent relay attack in authentication protocols by using the location as an unforgeable attribute of the prover. Brands and Chaum's DB protocol estimates the distance through measuring the round-trip time of challenges and responses that travel at the speed of light. Relying on the fact that information cannot travel faster than light, we can ensure to obtain the upper bound of the distance between two parties. Hence, relay attack is prevented by forcing the adversary to come closer to the victim, which increases the possibility of being detected.

A number of distance bounding protocols have been proposed in recent years [11, 39, 72, 66, 45, 2, 10]. The proposed protocols are in the same form of challenge and response phase, and use time-of-flight of the message to estimate the distance, while they differ in terms of the performance and security guarantee they provide. There are also distance bounding protocols that use other signal properties, such as received signal strength, and angle of arrival for time estimation. However, we mainly focus on the class of distance bounding protocol that uses time-of-flight (TOF) in this thesis.

Generally, the security and precision of all distance bounding protocols that use time-of-flight for distance measuring rely on the following assumptions:

- Line of sight propagation. One of the main assumptions that is made for distance bounding protocol is that the prover and verifier are in line of sight, whereby radio waves travel in a straight line.

- Free space (without multipath effect). Even with line of sight propagation, the travel

[2]This distance normally is a short distance from few centimetres to a few meters

time measurement may still be biased by multipath propagation which occurs when signals bounce from neighbouring surfaces. Multipath effects are avoided in a free space environment, such as vacuum.

- Negligible processing time. The other important assumption on which the security of distance bounding relies is that the time that the prover spends for processing is negligible compared to the propagation time of the signal between the prover and the verifier. Processing time includes the time for receiving the challenge, processing and transmitting the response.

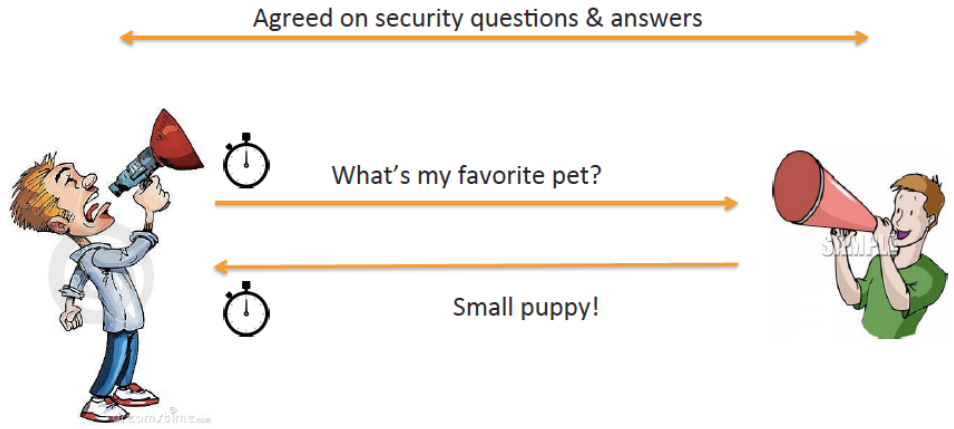

Figure 1.2: An example of distance bounding in real world

Distance bounding protocol is a challenge-response protocol, in which a challenge is sent by the verifier and the prover is required to provide a valid response. The round-trip time of challenge and response is used to measure the distance between the prover and the verifier. The verifier's challenges are unpredictable to the prover and the prover's responses are determined by the challenge. Hence, it is impossible for the prover to reply before receiving the challenge. The prover, therefore, cannot pretend to be closer to the verifier than it really is. In order to guarantee the security of distance bounding protocols, electromagnetic signals which travel at the speed of light are used and therefore the measured distance is the upper bound of the real distance. The intuition of distance bounding protocols can be

illustrated using a real world example. Imagine that Alice and Bob want to measure the distance between them (as shown in Figure 1.2). They first agree on a set of security questions and answers, which are only known by them. At the time of measurement, Alice starts the stopwatch and shouts at Bob one of the security questions. Once Bob hears the question, he shouts back with the corresponding answer immediately. Alice stops the stopwatch once she hears the response from Bob. The security questions and answers guarantee that it is Bob who is involved in the measurement procedure. The round-trip time can be used to calculate the upper bound of distance between Alice and Bob.

Distance bounding protocols have been widely used for many localization and authentication scenarios. For instance, in [30], authors showed the feasibility of relay attack on Passive Keyless Entry and Start (PKES) system and authors proposed to incorporate distance bounding protocol to enhance the security of PKES systems. A proximity based access control scheme for implantable medical devices using distance bounding protocol is proposed in [65] and therefore the access to the certain resources is determined by the proximity of the devices. In [71], the author proposed to use multiple verifiers to triangulate the location of the prover with distance bounding protocol resulting in a location verification scheme. Distance bounding protocols are also appropriate for Radio-Frequency Identification (RFID) authentication [73, 3].

The security of distance bounding protocols was normally evaluated by analyzing its resilience to three types of attacks: distance fraud, mafia fraud, and terrorist fraud - which were awarded their names by historical reasons. In *distance fraud attack*, the malicious prover acts alone to convince the verifier that he is at a shorter distance than reality. In *mafia fraud attack*, the prover is honest but is a victim of an adversary who tries to shorten the distance between the prover and verifier by interfering with the communication. In *terrorist fraud attack*, the dishonest prover colludes with another adversary closer to the verifier to convince the verifier that he is at a shorter distance. In [18], the authors proposed a new type of attack

called *distance hijacking*. In this attack, a dishonest prover exploits one or multiple honest parties to provide a verifier with false information about the distance between itself and the verifier. So far, it is assumed that the distance bounding protocol that is resilient against these four attack types can be considered secure. One should note that the adversaries of the above attacks all aim to shorten the distance.

Note that in all distance bounding protocols we discussed before, users obtain advantage when they are close to the verifier (within a certain distance bound). Such distance bounding protocols obtain an upper bound of distance between the prover and the verifier and adversaries aim to shorten the distance. However, in this thesis, we identify that there is another class of scenarios where a prover wants to prove its distance to a verifier is higher than a bound in order to obtain the advantage. Such distance bounding protocol should result in a lower bound of the distance between the prover and the verifier and adversaries aim to extend the distance. In order to distinguish this new problem from the conventional distance bounding problem, we refer the former one that obtains the upper bound of the distance between the prover and the verifier to distance upper bounding (DUB) and the proposed one to distance lower bounding (DLB). DLB problem naturally arises in application scenarios where privileges are given when a requester is far away from a provider.

When using round-trip time of the challenge and the response to estimate distance in distance bounding protocols, one of the main assumptions is that the processing time, which includes the time for receiving the challenge, processing and sending the response, is negligible compared to the propagation time between the prover and the verifier. However, it is hard to achieve such strict requirement in practice using today's state-of-the-art hardware, which acts as a major obstacle when implementing distance bounding protocols. So far, only several implementation prototypes have been proposed using analog processing [64] or the hybrid approach of analog and digital processing [63]. When using electromagnetic signals that travel at the speed of light in the measurement, even 1 ns of processing delay results

in an inaccuracy of $3 \times 10^8 \times 1 \times 10^{-6} \div 2 = 150m$. On the other hand, if ultrasound signals which travel at 330 m/second are used, the protocol will be vulnerable to wormhole attack [16], in which the adversary uses electromagnetic signals to relay messages in order to shorten the round-trip time of the message. Hence, the strict requirement of processing time not only puts difficulties on the implementation of DB protocols, but also limits the possible development of DB applications. In this thesis, we will address this problem by proposing a new notion of time and designing distance bounding protocol accordingly.

In the thesis, we proposed two distance bounding protocols and provided security analysis in an ideal environment, in which the signal transmitted between the prover and verifier is the line of sight propagation, all parties are located in open space and the processing time is negligible compared to the propagation time. However, in practice, because of the restriction of the environment, these assumptions will rarely be true. Hence, the travel time measurement will be biased by multipath propagation, as well as non line of sight propagation. In addition, it is hard to achieve negligible processing time with current technology. Hence, when implementing such a system for measuring the distance between transmitters and receivers, one must take these phenomena into account. Additionally, depending on the different accuracy and ranging requirement of the application envisaged, extensive experiments should be conducted to evaluate the effects that are caused by the different environment.

1.2 Proximity-based authentication

Proximity-based authentication refers to a class of authentication mechanisms that not only require the user to prove its identity by holding the corresponding secret key, but also require the user to come close to the verifier when authenticating. This is an effective way to defeat Man-in-the-middle (MiM) attack, because the attacker is forced to come close the verifier and so his probability of being detected is significantly increased. Besides defeating MiM attack, proximity-based authentication is widely applied to different applications scenarios.

For example, access control [33, 65] can be deployed based on proximity-based authentication (also called proximity-based access control), to prevent unauthorized access. Considering an access control system for a secure room in a building, the user who holds the correct identity card gets authenticated by approaching the reader in order to get access to the room. When applying this to mobile devices, proximity-based authentication offers an attractive way to bootstrap secure communication channels between devices [44, 82]. For example: users may place their mobile phones next to each other to establish a secure communication for transferring documents; or users may put their mobile devices close to their laptops or other large displays to mirror the screen. Talking to contactless payment system[48, 88], proximity-based authentication improves its security by requiring multi-factor authentication.

Normally, proximity estimation is achieved by three different ways. The first approach is to explore the hardware properties of RFID (Radio-Frequency Identification) device [27]. Generally, most RFID tags do not have local power supply. Instead, tags are powered by electromagnetic induction from magnetic fields produced near the reader. Hence, proximity assumption is established when the RFID tag is activated. However, this approach is vulnerable to relay attack, in which the adversary comes close to the victim with a reader to activate the tag and relay all messages to the other reader to get authenticated. Several works have realized this attack in practice [36, 30]. Hence, proximity evaluation is now conducted by the following two approaches.

One approach utilizes distance bounding protocols [39, 64, 66, 10] to evaluate proximity[3]. Although this poses challenge to the implementation as distance bounding protocol requires fast processing time in the range of nanoseconds, it provides a better security guarantee compared to a context-based approach. We refer this approach to the DB-based approach. In this thesis, we identify an overlooked attack, named false rejection attack, that poses significant threat to DB-based proximity-based authentication protocols. As a result, the

[3]In the latter design of distance bounding, authentication is integrated with distance estimation and therefore the standalone distance bounding can be treated as proximity-based authentication

adversary can easily prevent an honest prover from getting authenticated and this will lead to significant consequences.

Context-based approach achieves proximity evaluation by asking users to sense the context information and compare the result. A number of schemes have been proposed, which utilize different sources of context information including: ambient light [35], acoustic environment [80, 35], temperature, humidity, air pressure [70], and Radio Frequency (RF) based signal such as WiFi and Bluetooth [80, 44, 82]. In these approaches, many protocols do not even require users to pre-register in the system, and the credential in this case becomes the location only. The prover and the verifier sample their environment via sensors at the same time and compare the result to see if two parties are in proximity. If the measurements are similar enough, we say two parties are close. We refer this approach to context-based approach. In this thesis, we investigate the class of context-based proximity authentication that utilizes properties of 802.11 signal to determine proximity. We prove that without carefully choosing the source of context information, such an approach is vulnerable to replay attack.

1.3 Contributions

1.3.1 Distance Lower Bounding

Distance (upper) bounding (DUB) protocols have been widely studied in recent years: a *verifier* V interacts with a *prover* P to obtain assurance that the prover is at a distance at most B from the verifier in wireless environments. To estimate distance, secure DUB protocols measure the round-trip time of a *fast-exchange challenge-response protocol*, using electromagnetic (EM) communication and assuming constant speed (speed of light) for EM signals.

In this thesis, we consider the dual problem of *distance lower bounding* (DLB) where a prover P wants to prove its distance from a verifier V is higher than a bound B. The DLB

problem naturally arises in application scenarios where privileges are given based on the distance of a requester to a provider. For example, a company offers unrestricted Internet access to games and entertainment software to employees when they are outside of main office area of the company campus (e.g, Google campus), and restricted access when employees are within the main office area. Here the requirement is for employees to prove they are outside the main working area (shown in Figure 1.3). A second scenario is when the parking lot is divided into zones and the parking charge depends on the distance of the car to the main point of interest (e.g. discounted rate will be given if users park their car further away from the shopping mall entrance). In both cases once the privilege is granted based on the distance, one needs to use monitoring mechanisms such as continuous authentication to ensure that the user stays within the claimed area. Embedding such authentication in streaming services such as games or music is straightforward. For the latter scenario, one can use random scanning of the area to ensure correct claim. Although determined users may be able to bypass the authentication, but they will be inconvenient (e.g. move the car frequently) and also have to accept the risk of detection and penalty.

Figure 1.3: Application Scenario of Distance Lower Bounding

The main contributions of this work are as follows: we first motivate and initiate the study of distance lower bounding (DLB) problem in a setting where the prover is untrusted. We then construct a security model for the DLB problem and define three broad classes

of attacks: *distance fraud (DF)* where the prover is malicious and acts alone to enlarge its distance to the verifier; *collusion fraud (CF)* where the prover is malicious and has a collaborating helper that it uses to enlarge its distance to the verifier ; and finally *Man-in-the-middle (MiM) attack* where the prover is honest and is the victim of an external attacker who aims to enlarge the distance between the honest prover and the verifier. We prove that security against any of these attacks without making any *physical assumption*[4] is impossible. In particular, a fully malicious prover can *always* succeed in the DF attack, and an external attacker (without the cryptographic credentials) can always jam-and-delay the signal between the verifier and the prover and succeed in the MIM attack. This also implies that a malicious prover that has a helper (CF) will always succeed. Fourth, we study reasonable assumptions under which there is a solution to the problem, and construct a secure DLB protocol under specific assumptions. We analyze security of our DLB protocol against DF, MiM and CF attacks. We also estimate time, memory and energy requirements of our protocol and conclude with open questions and directions for future research. This work has been published and presented at the 2014 International Conference on Information and Communications Security (ICICS 2014) [90].

1.3.2 Model Time and Novel Distance Bounding using Timestamps

In this work, we focus on the problem of distance upper bounding. Given the requirement of fast processing and the complexity of RF ranging systems, distance upper bounding protocols have been challenging to implement. So far, only a few designs have been proposed and implemented [64, 63]. Furthermore, the strict restriction of processing time limits the possible developments of distance upper bounding applications. For example, frequency hopping [74] is a promising way to defeat false rejection attacks on distance bounding (see Chapter 5 for details). However it cannot be directly applied to the existing distance bounding scheme as the additional time consumed by frequency hopping will significantly impact the

[4]Physical assumptions include limited access to the device hardware, and/or the communication channel.

accuracy of distance bounding. The reason why this limitation exists comes from the fact that distance bounding protocol measures distance based on the round-trip time (RTT) of message transmission, which includes both transmission time and processing time. The processing time here refers to the time that the prover takes to receive the challenge, process, and send the response. In the setting of untrusted prover, the verifier is not able to isolate the processing time from RTT. Therefore, fast processing on the prover side is necessary in order to approximate to the real transmission time.

In this thesis, we address this problem by measuring the distance using the one-way transmission time ($OwTT$). Intuitively, one can obtain the one-way transmission time ($OwTT$) by asking the sender to include its sending time t_s in the message. Then the receiver can obtain the receiving time t_r upon receiving the message and calculate the transmission time by $t_t = t_r - t_s$. However, this simple idea requires: (1) the sender and receiver's time are synchronized; and (2) the sender is honest and would not lie about its sending time. Apparently, achieving the latter requirement is not easy, as the prover can be malicious, and even colluding with external adversaries in the adversarial setting of DUB protocols. Therefore, we introduce Global Counter (GC), a trusted party that broadcasts unpredictable timestamps at a high frequency. The idea of GC is inspired by the prototype source that is implemented by NIST for public randomness [58]. In this prototype, a randomness beacon broadcasts full-entropy bit-string in block of 512 bits for every 60 seconds. With GC, the receiver has assurance to obtain the upper bound of the transmission time of the message that is sent by any sender. Note that the malicious prover is only interested in shortening its distance to the verifier in DUB (through shortening the transmission time). Hence, the malicious prover that aims to extend its distance to the verifier is not considered in this work. But one can find how to prevent such an attack in [90].

Although time plays an important role in various cryptographic protocols, for example to guarantee freshness in authentication protocols or to limit lifetime for public keys, few

works have discussed the problem of modelling time formally. Barbosa and Farshim [4] have extended the existing authentication framework to capture the use of timestamps. In [68], Schwenk has started the first discussion of modelling time in Turing Machine environment for the purpose of guaranteeing the freshness in cryptographic protocols. This requires that the adversary could not use a timestamp that is in the past to get accepted by the protocol. In our case, besides guaranteeing the freshness, we use time values for a different purpose: to calculate the upper bound of the transmission time of a message. This requires timestamps to be unpredictable in the sense that the adversary could not forge a valid future timestamp before seeing it. This fundamental difference reveals that we could not simply use the time model in [68] for our work.

The main contributions of this work are as follows: We first motivate and initiate the study of distance upper bounding protocol using one-way transmission time, which is different from all previous distance upper bounding protocols. We propose a new time model that not only guarantees the freshness, but also allows users to calculate one-way transmission time of the message, which can be used in Turing Machine-based implementations. Based on this time model, we then formalize distance bounding protocol and define the resistance to three general attacks: distance fraud (DF), Man-in-the-middle (MiM) attack and collusion fraud (CF). We then proposed a novel one-round distance bounding protocol that obtains the upper bound of the distance based on the one-way transmission time, in which processing time is no longer a limitation. Finally, we prove the security of our proposed protocol and discuss issues when implementing such protocol in practice.

1.4 Other Contributions

1.4.1 False Rejection Attack on Distance Bounding

All distance-bounding protocols focus on preventing adversaries or dishonest provers from making the verifier believe he is at a closer location than he actually is. However, the facts

that the adversary can easily delay the round-trip time to make the honest prover appear at a farther place or tamper the response to fail the challenge-response phase, were overlooked. As the result of a failure of protocol, the honest prover who locates close to the verifier, will not be able to get authenticated or access to the resources he should have accessed. In addition, the prover and the verifier will not be aware that the attack is happening. With such a denial-of-service-like attack, proximity-based authentication will suffer from the false rejection. We therefore define such attack in distance bounding protocol, which is mounted by tampering or delaying the challenge-response communication of distance-bounding protocol, as false rejection (FJ) attack.

One should notice that tampering or delaying the challenge and response is not a security threat to distance- bounding protocol that aims to estimate the distance only. This is because the purpose of such distance-bounding protocol is to obtain the upper-bound of distance between prover and verifier. Hence its security guarantee is that there should not be a prover making the verifier believe he is at a closer location than he actually is. Tampering or delaying the challenge or response only leads to the fact that the prover appears farther than its physical location, which is not desirable for attackers or dishonest provers. So false rejection attack does not violate the security guarantee of distance-bounding protocol.

However, when distance bounding protocol is used for proximity-based authentication (or authentication is naturally incorporated in the DB protocol), this becomes a significant threat. Imagine the following scenario: the database in an office building employs distance-bounding protocol for proximity-based authentication to prevent man-in-the-middle attack. Only the employee who is close to the building can get authenticated and gain access to this database. However, the adversary can easily make the distance-bounding protocol fail by tampering challenge-response protocol or make the verifier believe this honest prover is located at a farther place by delaying the challenge or response. As a result, an authorized employee who actually locates close to the building would not be able to get access to

the database, which leads to false rejection. The other example would be using distance bounding protocol in implantable medical device [65]. Mounting false rejection attack in this scenario may lead to serious health issues of users, even leading to death. In addition, unlike traditional easily detected denial-of-service (DoS) attacks which are conducted by simply blocking the the whole communication, false rejection attack will not be aware by either the prover nor verifier. Hence, false rejection attack becomes a more significant threat to those application scenarios using distance-bounding protocol for proximity-based authentication.

In this thesis, we first identify the overlooked problem when applying distance bounding protocol to proximity-based authentication and define a new class of attack - false rejection attack. We analyze the scope of this attack and its relation to other attacks in distance bounding protocol. This type of attack poses a serious threat in many proximity-based application scenarios. Second, we formally define this attack using the same notations from Boureanu's model for distance bounding [8]. Third, we examine the existing distance bounding protocol and advise possible solutions to prevent false rejection attack.

1.4.2 Feasibility of Replay Attack in Context-based Proximity Authentication

By utilizing the context data sensed by both the verifier and the prover, one can achieve proximity-based authentication [80, 35, 44, 82]. We refer this class of authentication scheme to context-based proximity authentication. In this approach, the prover and verifier sample their context via sensors to determine their proximity. The source of context information used varies depending on the design: ambient light [35], acoustic environment [80, 35], temperature, humidity, air pressure [70] and Radio Frequency based signals such as WiFi and Bluetooth [80, 44, 82].

While context-based proximity authentication has broad applications, the choice of the appropriate source of context information has significant impact to its security. In this thesis, we investigated how replay attack can impact context-based proximity authentication in the scenario that the prover is not fully trusted. We show the feasibility of such attack on real

data towards the design that uses 802.11 frame (WiFi) as context information and advise the countermeasures to defeat such attack.

1.5 Overview of Thesis

The structure of this thesis is as follows. Chapter 2 introduces the relevant background information and an overview of related work on distance bounding and proximity authentication. Chapter 3 presents the problem of distance lower bounding, and its model, assumptions, description and security analysis. Chapter 4 presents a novel distance bounding protocol that utilizes time model to eliminate the use of round-trip time as other distance bounding protocols. Chapter 5 identifies the false rejection attack in distance bounding that applied to proximity authentication and Chapter 6 investigates the feasibility of replay attack in context-based proximity authentication. Chapter 7 summarizes the thesis and presents directions for future work.

1.6 Publications

Parts of the work presented in this thesis is based on articles I coauthored. In particular, the thesis was based on the following publications:

- Xifan Zheng, Rei Safavi-Naini and Hadi Ahmadi, "Distance Lower Bounding", In proceedings of International Conference on Information and Communications Security (ICICS), 2014

- Xifan Zheng and Rei Safavi-Naini, "Distance Bounding using Global Counter", Draft

Chapter 2

Background

This chapter discusses the necessary background knowledge related to distance bounding, different formalizations and attack models that are considered in the literature and some of relevant work. Distance bounding protocol is first proposed to prevent relay attacks in authentication. This can be achieved by using the location as an unforgeable attribute of the prover. The relay attack is a man-in-the-middle attack scenario, where an adversary sits between two parties and relays messages (and also alters messages when necessary) on its own behalf. The consequence of the relay attack is that the adversary identifies himself to one party using the identity of the other party without being noticed by either party. Relay attack is a significant threat to many application scenarios of authentication protocols. For instance, passive keyless entry and start (PKES) system is widely used in modern cars. However, [30] shows that it is possible for an attacker to open a car by relaying the communication between the key and the car. Another example is contactless credit-card system; a malicious payment terminal can relay signals to a fake card, which is trying to buy something more expensive [23].

To defeat relay attacks, Brands and Chaum [11] introduced the first notion of distance bounding (DB) protocol. Brands and Chaum's DB protocol estimates the distance by measuring the round-trip time of challenge and response messages that is transmitted at the speed of light. To guarantee the security of distance bounding protocol, some physical assumptions about the signal propagation are required, which are discussed in section 2.1.1. By relying on the fact that information cannot travel faster than light, one is sure to obtain the upper bound of the distance between two parties. Hence, relay attack is prevented by forcing the adversary to come closer to the victim.

The security of distance bounding protocols is normally evaluated by analyzing its resilience to three types of attacks: distance fraud, mafia fraud and terrorist fraud, which are named as such for historical reasons. In a *distance fraud attack*, the malicious prover acts alone to convince the verifier that he is at a shorter distance than he really is. In a *mafia fraud attack*, the prover is honest, but is a victim of an adversary who tries to shorten the distance between the prover and verifier by interfering with the communication. In a *terrorist fraud attack*, the dishonest prover colludes with another adversary that is closer to the verifier in order to convince the verifier with a shorter distance. In [18], the authors proposed a new type of attack called *distance hijacking*. In a *distance hijacking attack*, a dishonest prover exploits one or multiple honest parties to provide a verifier with false information about the distance between itself and the verifier. So far, it has been assumed that distance bounding protocols that are resilient against these four attack types can be considered secure. A more detailed illustration of different attacks against distance bounding protocol is provided in section 2.1.3.

A number of distance bounding protocols were proposed in recent years [11, 39, 72, 66, 45, 2, 10]. The proposed protocols differ in terms of the performance and security guarantees they provide. Detailed designs of prominent distance bounding protocols in the literature will be explained and their pros and cons will be discussed in section 2.2.

In order to provide provable security towards distance bounding protocols, researchers have proposed different model to formalize distance bounding protocols [1, 24, 29, 54, 8]. These models will be compared in section 2.3.

Distance bounding protocols have been widely used for proximity based authentication (e.g., passive keyless entry and start system in modern cars [30]), proximity based control (e.g., implantable medical devices [65]), and Radio-Frequency Identification (RFID) authentication [73, 3]. So far, several distance bounding protocols have been implemented using analog processing [64] or the hybrid approach of analog and digital processing [63].

Although the notion of time is used in distance bounding protocols which measure round trip time to estimate the upper bound of distance, modelling time is unnecessary as clock reading is only done by the trusted verifier. However, in the work presented in Chapter 4, modelling time is necessary for proving the security because the untrusted prover is involved in the time measurement procedure. Related work of modelling time will be discussed in section 2.4 to provide preliminary knowledge.

2.1 Distance Bounding Protocols

A distance bounding (DB) protocol obtains an upper bound on the distance between a prover and a verifier. Secure DB protocols [39, 64, 66, 10] estimate the distance by measuring the round-trip time of simple challenge-response messages. The three commonly considered attacks on DUB protocols are, *distance fraud, mafia fraud and terrorist fraud*. In [18], the authors proposed a new type of attack called *distance hijacking*, as an extension of distance fraud.

Brandand Chaum proposed the first pre-commitment DUB protocol[11]. Hancke and Kuhn then proposed a pre-computation DUB protocol[39] that can be easily implemented on RFID devices. Singelee et. al [72] considered distance bounding in noisy environments and proposed a protocol that can tolerate noise. None of the above protocols is secure against terrorist fraud.

Protocols with claimed security against terrorist fraud include [66, 45, 2, 63]. These protocols use different approaches to provide this security. Reid et. al [66] proposed to combine the response with the long term secret key so that knowing the whole response table enables the helper to recover the secret key and impersonate the prover in the future. Kim et. al [45] noted that Reid et. al's protocol will be insecure if the adversary can see the result of the protocol. Avoine et. al [2] used threshold secret sharing to prevent terrorist fraud.

Recently, a number of new attacks against protocols that were believed to be secure, have been proposed[18, 7, 38, 51]. These attacks have generated further interest in formalizing security of distance upper bounding protocols. In [2] a semi-formal model for secure distance-bounding is given; [29] and [28] give a formal model capturing resistance to terrorist fraud; [18] proposes a detailed list of known attacks against distance bounding protocols and gives a formal security model for multi-party settings; and [8] provides a security model in terms of resistance to three general classes of attacks that include other known attacks as special cases. This last approach will be followed.

2.1.1 General assumptions for distance bounding

Generally, the security and precision of all distance bounding protocols that use time-of-flight for distance measuring rely on the following assumptions:

- Line of sight propagation. One of the main assumptions that is made for distance bounding protocol is that the prover and verifier are in line of sight, whereby radio waves travel in a straight line. In distance bounding protocols, the round-trip time of the message is used to estimate the distance between two parties. In general, time-of flight only represents the distance between transmitter and receiver, when line of sight propagation is the basis for the measurement. While it is sufficient to just have any type of propagation channel to enable communication or measuring a time-of-flight, this does not necessarily result in a proper way of obtaining the correct distance; because any obstruction between the transmitting antenna and the receiving antenna will block the signal. Hence travel time measurements for determining the distance between pairs of transmitters and receivers generally require line of sight propagation for accurate results. This is important when one considers that the speed of propagation of EM waves is 300,000 km per second; small timing delays can result in large distance errors when conversion from time to distance is performed.

- Free space (without multipath effect). Even with line of sight propagation, the travel time measurement may still be biased by multipath propagation which occurs when signals bounce from neighbouring surfaces. ice. Multipath effects are avoided in a free space environment, such as vacuum, which does not occur on earth. Although one could always use the signal that arrives earliest for measuring, that signal may still be biased by multipath; also, as the line of sight propagation transmits fastest, the effects of multipath may results in destructive interference, in which a annihilation (cancelling) signal is generated and will lead to a serious signal degradation at the receiver [79].

- Negligible processing time. The other important assumption on which the security of distance bounding relies is that the time that the prover spends for processing is negligible compared to the propagation time of the signal between the prover and the verifier. Processing time includes the time for receiving the challenge, processing and transmitting the response.

2.1.2 How do distance bounding protocols work

The goal of a distance bounding protocol is to allow a verifier obtain the upper bound of its distance to the prover. Although many distance bounding protocols have been proposed, they all follow the same pattern: use the time-of-flight to measure the distance. Normally, a distance bounding protocol has three phases. In the first initialization phase, the prover and verifier compute or commit some nonces that will be used in the following protocol. In the time-critical challenge-response phase, multiple rounds are involved. In each round, the verifier sends out the challenge and the prover needs to provide the correct response according to the value of challenge. The verifier will measure the time between sending out the challenge and receiving the response. In the final verification phase, the verifier needs to verify the correctness of the response and calculates the upper bound of the distance if the

verification is successful. The upper bound is calculated by the round-trip time multiplied by the speed of light. Figure 2.1 shows the distance bounding protocol proposed by [10]. At the end of a successful run distance bounding protocol, the verifier outputs 1; otherwise, the verifier outputs 0. The output may or may not be sent to the prover depending on the protocol specification.

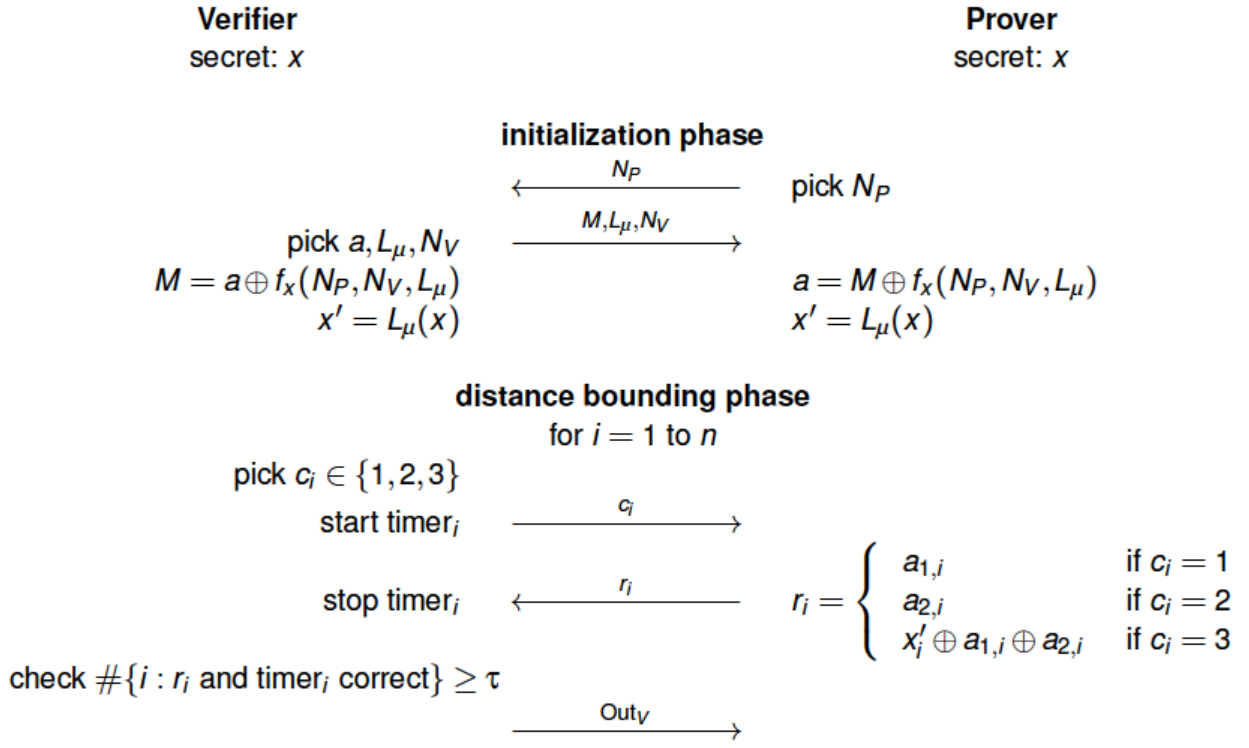

Figure 2.1: SKI Distance bounding protocol, taken from [10]

2.1.3 Attacks against Distance Bounding protocols

Three different types of attack are traditionally considered when analyzing distance bounding protocols: distance fraud, mafia Fraud and terrorist Fraud. All attacks that fall into one of these three classes have a similar goal to make the verifier believe that the prover P is physically closer to the verifier V than it really is, whether the prover is honest or not.

In a distance fraud [11] attack, a dishonest prover P will try to shorten the distance measured by the verifier. This type of attack only involves the dishonest prover himself,

without colluding with any other parties. The typical example of a distance fraud attack is that a protocol allows the prover to send out the response before he receives the challenge messages, which shortens the round trip time measured by verifier. Distance fraud attack is the fundamental attack that needs to be considered when designing a distance bounding protocol; almost every existing protocol promises to protect against this type of attack.

Mafia fraud[21], also known as relay attack, is conducted by an external attacker, who tries to shorten the measured distance between an honest prover and verifier. In this type of attack, the attacker acts as both the verifier and the prover to relay the challenges and responses, without the prover or the verifier noticing the attack. In order to shorten the measured distance, the attacker should be located very close to the verifier.

The third attack type is Terrorist fraud [21]. In this kind of attack, a dishonest prover collaborates with external attackers to convince the verifier that he is closer than he really is. Basically, in terrorist fraud attack, a dishonest prover, who is far from the verifier, exploits the helper, who is actually located close to the verifier, to complete the distance measurement. In this way, the verifier will mistakenly conclude that the dishonest prover is located at a closer location.

Based on three historical attack types, authors of [18] proposed a new attack type that was overlooked before: the distance hijacking attack. In a distance hijacking attack, a dishonest prover P exploits one or multiple honest parties to provide a verifier V with false information about the distance between P and V. One should be aware of the difference between the distance hijacking attack and the terrorist fraud attack: in a terrorist fraud attack, the attack is conducted by colluding with other parties; however, in a distance hijacking attack, the parties are all honest, except attacker himself. Normally, distance hijacking attack is conducted by allowing other honest provers to complete the distance bounding protocol as usual, and then replacing the signature messages with the attackers signature.

There is also impersonation fraud [3], which is proposed as a variant of Mafia fraud. In

impersonation fraud, an external attacker tries to impersonate the prover in order to make the verifier accept.

2.1.4 Exhaustive attack classification framework

By examining all attack types that were discussed before, it can be observed that the main difference among these attack types lies in the parties that carry out the attacks and their mutual relationships with other involved parties. In order to examine whether or not these attack types cover all possible attacks towards distance bounding protocols, [18] provides a comprehensive attack classification for a better understanding. By intuition, this attack classification is a sequence of case distinctions based on three attributes of attacks on distance bounding protocols: whether the prover is honest, whether the prover is the only party involved in the attack, and if not, whether the other involved parties are honest. By considering these three attributes, [18] defines four attack types to cover all possible attacks.

As shown in Figure 2.2, two main cases are first distinguished. If the prover P is honest, then P is not the attacker; therefore an external attacker is trying to shorten the distance. This type of attack would be called external distance fraud. In the second case, if P is not honest, it is then distinguished whether or not P is the only party involved in the attack. If yes, it must be and attacker trying to shorten the distance between itself and V. This type of attack is called lone distance fraud. If there are other parties involved, two final cases can be distinguished. If all of the other parties are dishonest and collaborating, then the attack is assisted distance fraud attack. If all other parties are honest, then the attack is called distance hijacking.

In order to better understand this attack framework, an analysis about the well-known Brands and Chaum protocol [11] is provided in order to see how to determine the existence of these attacks and their countermeasures. The Brands and Chaum protocol is shown in Figure 2.3.

External distance fraud (mafia fraud) occurs if the prover is not able to authenticate the

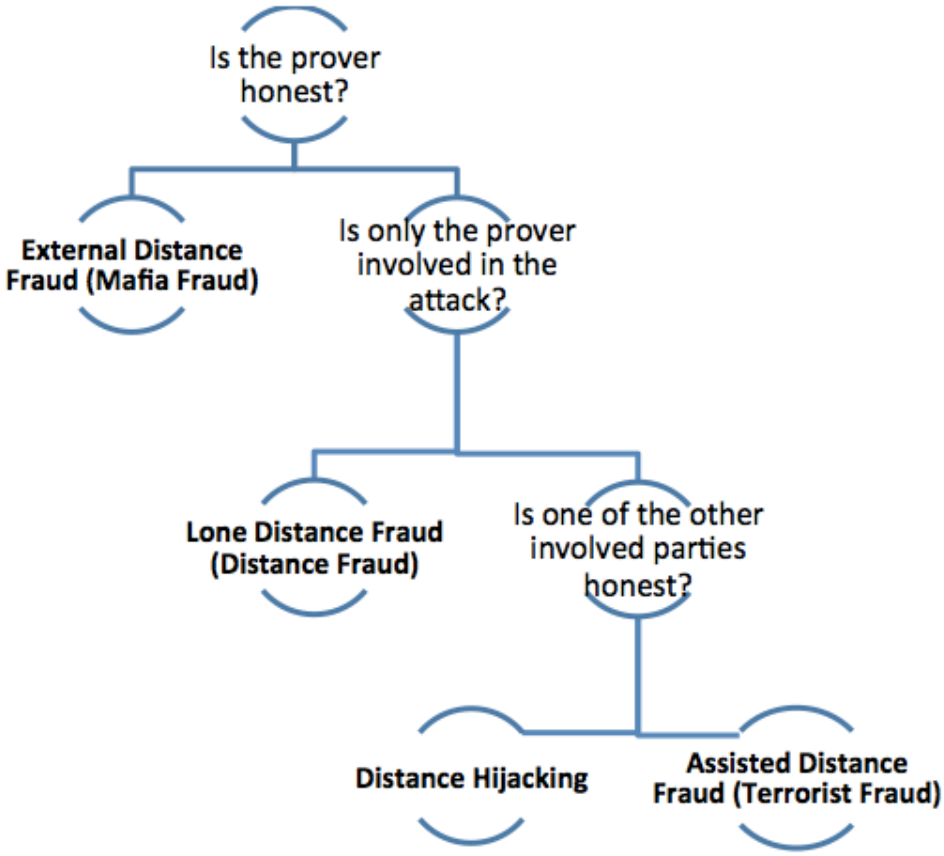

Figure 2.2: Exhaustive attack classification

verifier or the verifier cannot guarantee the provers identity. In this way, the attacker can pretend to be the verifier and then relay all information to the real verifier as the prover. The Brands and Chaum protocol can prevent this type of attack because it introduces the signature scheme to verify the identity of prover. Hence, the countermeasure for this type of attack is to provide the authentication mechanism for either prover or verifier.

Lone distance fraud is a relatively simple attack. This type of attack can be conducted if the round trip time can be shortened by the malicious prover himself. For example, if a protocol allows the prover to send a response before he it receives the challenge, then the prover can send out the response in advance to make the verifier believe the prover is closer than it really is. The Brands and Chaum protocol can also prevent this type of attack because the protocol requires the response to be the calculation of challenge. Hence, the

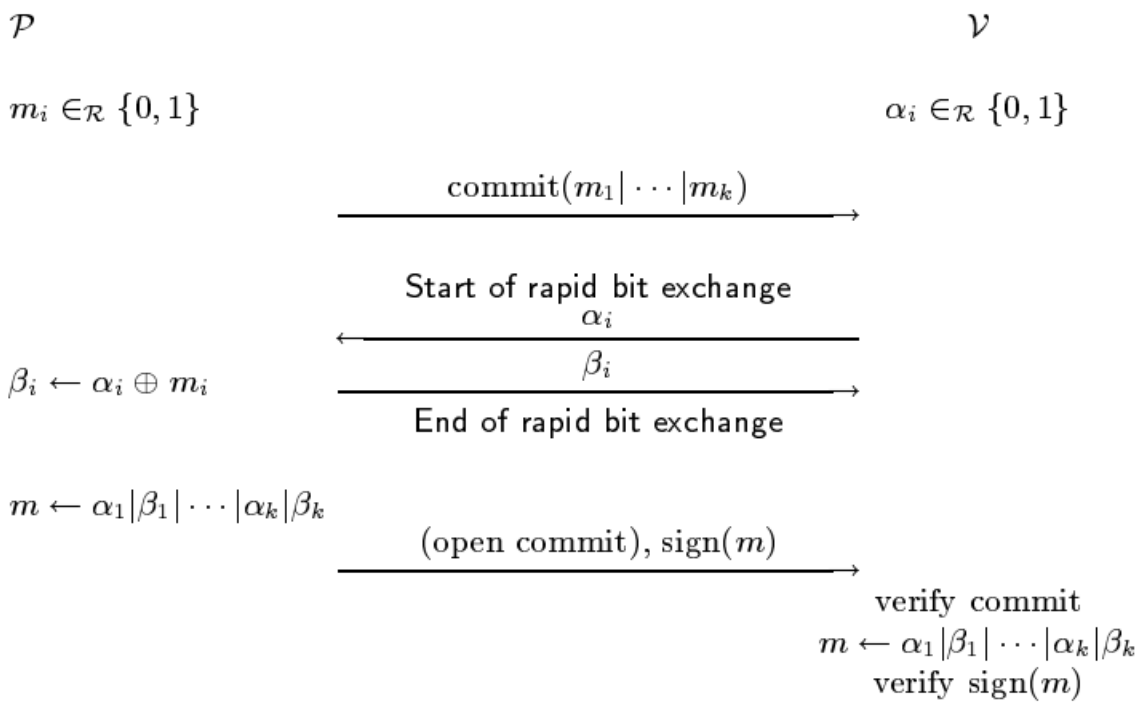

Figure 2.3: Brands and Chaum protocol, taken from[11]

countermeasure for this type of attack is to require that the prover can only send out a response after it receives a challenge.

Distance hijacking exists because the finalizing phase (proof of identity) does not have a cryptographic relationship with the previous two phases. In such a way, the attacker can allow the honest prover complete the first two phases and then use his own identity for the finalizing phase. The Brands and Chaum protocol cannot prevent this type of attack because the attacker can easily replace the honest provers signature with his own. Hence, potential countermeasures include either making the responses of different prover distinguishable or including the identity of prover in the first commitment phase.

Assisted distance fraud attack exists because the three phases in the Brands and Chaum distance bounding protocol are separated, and the round trip time is only measured at the second phase. Hence, the dishonest prover can exploit the other party, who is located closer to the verifier, to help it complete the second phase. Now, all countermeasures for the assisted distance fraud attack are based on the assumption that the dishonest prover would

not like to share his long term secret with other parties for some reason. Hence, involving the long term secret in the second phase can effectively prevent this type of attack.

Although this framework provides an exhaustive analysis of all attack types in distance bounding protocol, it is however difficult to formally define attacks by following this approach. [20] factors all these common threats into three possible attack types.

2.1.5 Attacks defined in Boureanu et al. model

In the Boureanu et al. model[8], all attack types are factored into three possible attack types. These attacks are considered in multi-party environments. This thesis will follow this approach.

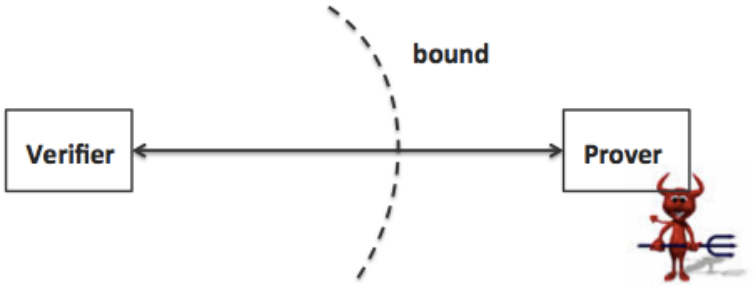

Figure 2.4: Distance fraud (DF) attack

Distance fraud (DF) (shown in figure 2.4) is similar to the classical notion, but here concurrency is considered. Hence, this definition includes other possible provers and verifiers. As a result, this generalized distance fraud also includes distance hijacking.

A Man-in-the-Middle (MiM) (shown in figure 2.5) attack is mounted by an external attacker, who does not have a secret key in the system. The attacker is allowed to have a learning phase by interacting (observing) with many provers and verifiers. Then, the attack phase contains far away targeted provers, verifiers, and possibly many other honest provers and verifiers. The attacker succeeds if the verifier accepts the far away prover. This generalized MiM attack includes mafia fraud and impersonation attacks.

Collusion fraud (CF) (shown in figure 2.6) is defined as an attack in which the helper

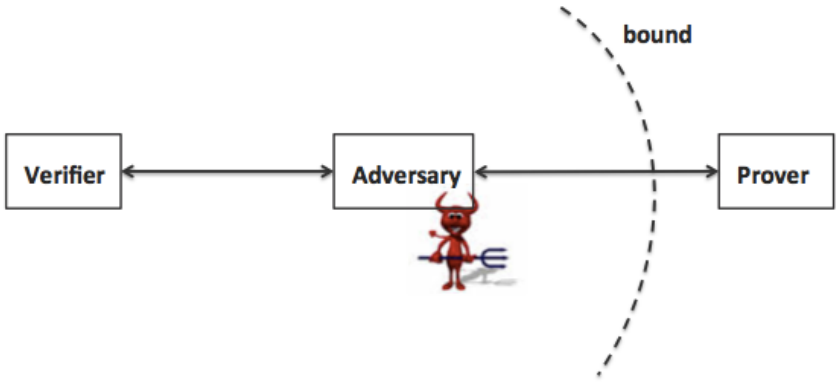

Figure 2.5: Man-in-the-middle (MiM) attack

(external attacker) helps a distant prover to make the verifier accept. This might occur in the presence of many other honest participants. However, the restriction of this attack is that it should not lead to future MiM attacks towards the malicious prover (i.e., the adversary should not be able to gain an advantage allowing subsequent MiM attack through collusion fraud). This generalized definition of collusion fraud includes terrorist fraud.

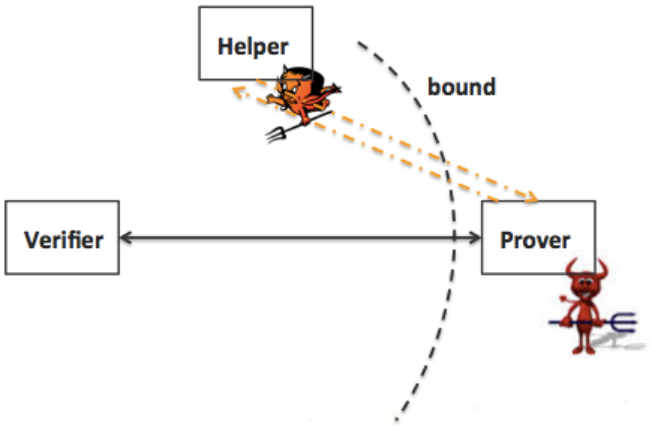

Figure 2.6: Collusion fraud (CF) attack

2.1.6 Distance Bounding in Practice

Distance bounding protocols are normlly evaluated in an ideal environment, in which the signal transmitted between the prover and verifier is the line of slight propagation, all parties are located in open space and the processing time is negligible compared to the propagation

time. However, in practice, because of the restriction of the environment, these assumptions will rarely be true. Hence, the travel time measurement will be biased by multipath propagation, as well as non line of sight propagation. In addition, it is hard to achieve negligible processing time with current technology. Hence, when implementing such a system for measuring the distance between transmitters and receivers, one must take these phenomena into account. In addition, depending on the different accuracy and ranging requirement of the application envisaged, extensive experiments should be conducted to evaluate the effects that caused by the different environment.

Processing time is an important factor that can significantly affect the precision of the result of distance bounding protocols. Processing time consists of the time for receiving, processing and sending. Depending on the different implementation, the processing time of the real distance bounding system varies. The best result one can achieve in the current implementation of analog systems is 1 ns [64], which is converted to around 0.15 meter precision in a round-trip approach, by using a scheme that is called Challenge Reflection with Channel Selection (CRCS). The processing time is minimized here since this implementation eliminates the need for signal conversion and demodulation as it does not require the received challenges being interpreted by the prover. However, this result is achieved in the setting that the transmitter and receiver is connected with a cable. Hence, one should expect a larger processing time for a more practical implementation.

Although analog implementation provides tight security guarantee and precision, many distance bounding protocols require the analog signal that carries the verifiers challenge be converted to a digital signal (ADC) for further processing (e.g, XOR function). Equally, in order to transmit the response back, the prover needs to convert the digital signal back to analog signal (DAC). These steps, including signal detection, ADC/DAC conversion and signal modulation/demodulation significantly increase the processing time. To our best knowledge, the fastest implementation requires about 170 ns [67], which results in a 25.5

meter accuracy.

2.2 Prominent Constructions of DB protocols in the literature

2.2.1 Timeline of research result about DB protocols

In order to provide a comprehensive and organized view of important literature in the area of DB, a timeline of different DB protocols has been provided as follows.

In 1993, Brands and Chaum proposed the very first pre-commitment DB protocol[11], in which the adversary has success probability of $(1/2)^n$ in distance fraud, $(3/4)^n$ in mafia fraud and 1 in terrorist fraud.

In 2005, Hancke and Kuhn proposed a pre-computation DB protocol[39] that can be easily implemented in RFID devices. An adversary has a success probability of $(1/2)^n$ in distance fraud, $(3/4)^n$ in mafia fraud, and 1 in terrorist fraud as well.

In 2006, realizing the previous protocol gave adversaries high success probability when mounting Mafia attacks, Munilla proposed to use pre-agreed "void challenges" [54] to reduce the success probability of mafia attack to $(1/2)^n$ by preventing the adversary from asking the honest prover. It is hard to implement this idea in RFID devices however.

In 2007, Reid et al proposed to intermingle the response string with the long term secret key to prevent terrorist fraud[66]. In such a way, if the prover gives the response string to the colluding party, the colluding party will be able to impersonate the prover in the future.

In 2009, Kim et al found out that Reid's protocol suffers from non-narrow MiM attack[45] if the adversary can access the return channel. The idea is that the adversary can flip certain challenge bit and give it to the prover to get the response bit, and then flip the response and send it back to the prover. By observing the result, it can obtain one bit of the long term key.

In 2011, Avoine et al proposed to use secret sharing to prevent this type of non-narrow MiM attack [2] by splitting the long term key into different shares and making them become

possible responses. Now the terrorist fraud is still discouraged.

In 2012, Boureanu et al found that a dishonest prover can choose a trapdoor nonce to bias the output distribution of the pseudorandom function[7] in order to gain an advantage for a distance fraud attack.

In 2012, Hancke found that in a noisy environment, a dishonest prover can make the colluding party respond to enough challenges correctly without leaking the long term secret, which makes Terrorist fraud possible[38].

In 2013, Boureanu et al proposed the SKI protocol[10] to fix the pseudorandom function problem and prevent terrorist fraud in noisy environments.

A detailed analysis and discussion of prominent DB protocols is given in the following section.

2.2.2 Brands and Chaum's DB protocol

We first look at the Brands and Chaum protocol [11], which is the first distance bounding protocol proposed for preventing relay attacks. In order to determine a practical upper-bound on the physical distance between two parties, the delay between sending out a challenge bit and receiving back the corresponding response bit is measured. This serves as the basic scheme for the following distance bounding protocols.

In Brands and Chaum construction (shown in Figure 2.3), two phases are involved. The first phase is the rapid bit exchange phase, in which the verifier and the prover generate random bit strings $C = C_1C_2...C_n$ respectively and $R = R_1R_2...R_n$ respectively and then exchange these random bits by sending challenges and responses for n rounds. In each round, the verifier transmits one challenge bit C_i, and the prover responds immediately with the response R_i. The round-trip delay between sending challenge C_i and receiving response R_i is measured by the verifier. After all n bits have been exchanged, the second verification phase begins. The prover completes the protocol by signing the concatenations of transmitted challenges and responses (the transcript of the communications $C_1|R_1|...|C_n|R_n$) using the

shared secret key. With this final message, the prover proves to the verifier that not only it is the prover itself involved in the protocol, but also provides a guarantee for the verifier that the challenges and responses are not being altered. The verifier verifies the signature and the transcript, as well as the upper bound of distance. The verifier accepts if, and only if, the prover is nearby and the received signature is a correct signature of the communication transcript.

For applications where the prover is not trusted and may send responses out too soon to shorten the distance, the above protocol does not work because the response bit that the prover sends to the verifier does not depend on the challenge bit that the verifier sends to the prover. Therefore, if the prover knows at what time the verifier will send out challenge bits, it can send out the response before actually receiving the challenge to make the verifier accept regardless of the distance to the verifier. In order to prevent this type of attack, Brands and Chaum described a protocol variant in which the prover commits to the random bit strings $M = M_1 M_2 ... M_n$, using a secure commitment scheme (e.g., by transmitting a secure hash value of $M = M_1 M_2 ... M_n$). During the rapid bit exchange phase, the verifier is required to generate response $R_i = C_i \oplus M_i$ by XOR-ing each challenge bit C_i with the corresponding bit M_i. In this way, the response depends on the challenge. Finally, the prover reveals M and signs the communication transcript. The commitment on M prevents the prover from sending some random bit R_i before receiving the challenge bit and then setting $M_i = R_i \oplus C_i$ after receiving C_i.

Although the above variant prevents distance fraud and mafia fraud, Brands and Chaum left out the resistance to terrorist fraud for future work. In the other word, neither variants of Brands and Chaum's protocol stop the prover from colluding with an external attacker, who is located closer to the verifier, to shorten the distance between the prover and the verifier.

2.2.3 Hancke and Kuhn's DB protocol

Later on, because of the emerging need for proximity authentication techniques for RFID (Radio-Frequency Identification) devices, Hancke and Kuhn[39] proposed light-weight distance bounding protocols that can be easily implemented on RFID devices to defeat relay attack. This new protocol is based on ultra-wideband pulse communication and is aimed at being implemented for simple, asynchronous, low-power hardware, such as passive low-cost tokens.

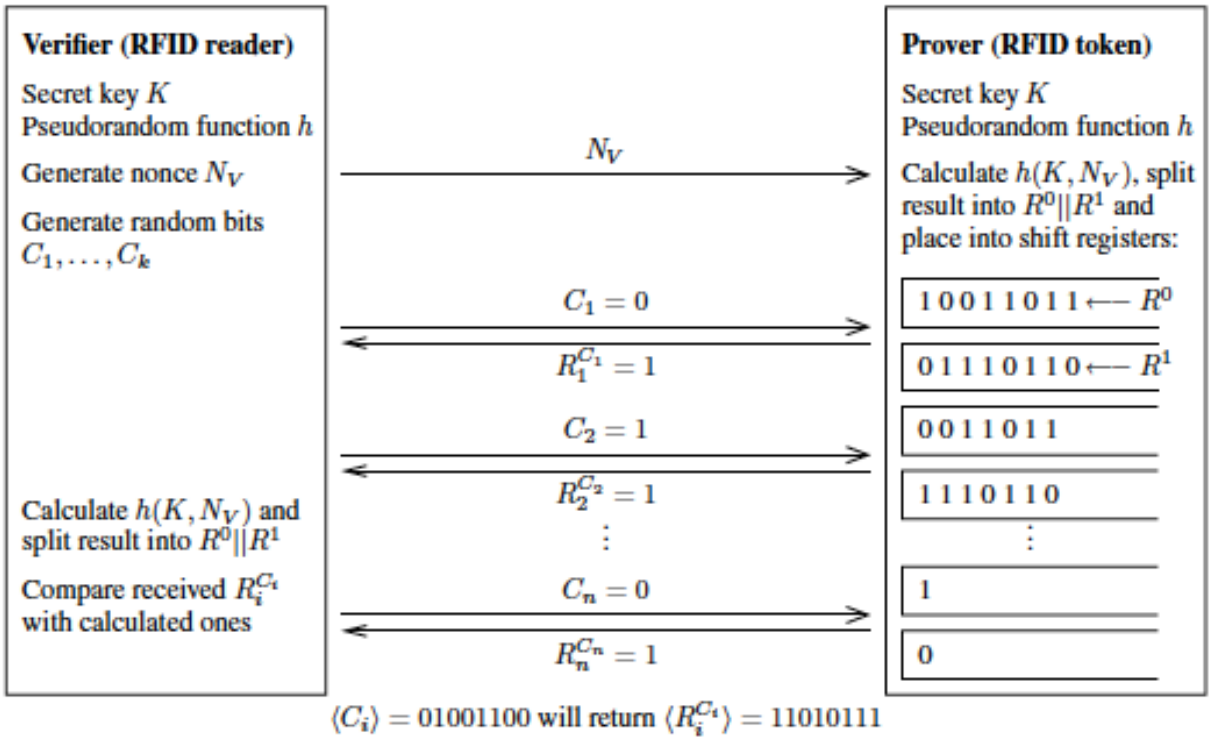

Figure 2.7: Hancke and Kuhn's RFID distance bounding protocol, taken from[39]

In the Hancke and Kuhn protocol, the responses are pre-generated in the sense that the verifier and prover derived the response table at the beginning of the protocol. Here, two parties use a symmetric key k and a pseudorandom function h to derive the response table. The verifier first generates a random nonce N_V and sends it to the prover. The prover calculates $h(K, N_V)$ and splits the result into $R^0||R^1$, which denotes a response table. Then, the distance bounding phase proceeds in n rounds. In each round, the verifier selects a

random challenge $c_i \in 1, 2$, and sends it to the prover. The prover is expected to reply with the corresponding bit in the response table (the ith bit of R_{c_i}) immediately. The verifier measures the round-trip communication time and rejects it if the prover took too long to respond or the response is incorrect.

Compared to the Brands and Chaum protocol [11], the Hancke and Kuhn protocol precomputes the response table in advance. In this way, the commitment (and the open commitment) procedure is eliminated to reduce cryptographic computation of the protocol. Because the response depends on the challenge, it is hard for the prover to shorten the distance by sending responses in advance. Also, because the response table is only known to the party who holds the secret key, a mafia attacker cannot send a response correctly without having the secret key. However, this protocol is not designed to resist to Terrorist fraud. This is because the initialization phase (the phase for both parties to generate the response table) is not time-critical; the malicious prover can give its response table $R^0||R^1$ to the helper, who is located closer to the verifier and lets the helper respond to all challenges. Clearly, the helper only obtains the response table in this way, and no information about the long term secret key of the prover is leaked.

2.2.4 Reid et al. DB protocol

Realizing distance bounding protocol's vulnerability to terrorist fraud, Reid et al [66] proposed the first protocol that is resistant to terrorist fraud. The idea is to prevent the malicious prover from giving the response table to the helper by intermingling the long term secret key with the response table. In this way, once the malicious prover gives its response table to the helper to succeed, the helper is able to succeed in Mafia fraud with the leaked information about the long term secret key. This is achieved by having R_1 derived from the initial nonce that sent by the verifier and setting R_2 to $R_1 \oplus k$. So, a malicious prover that provides both R_1 and R_2 to an helper would also leak k.

However, this approach has several drawbacks. First of all, as mentioned in [51], this

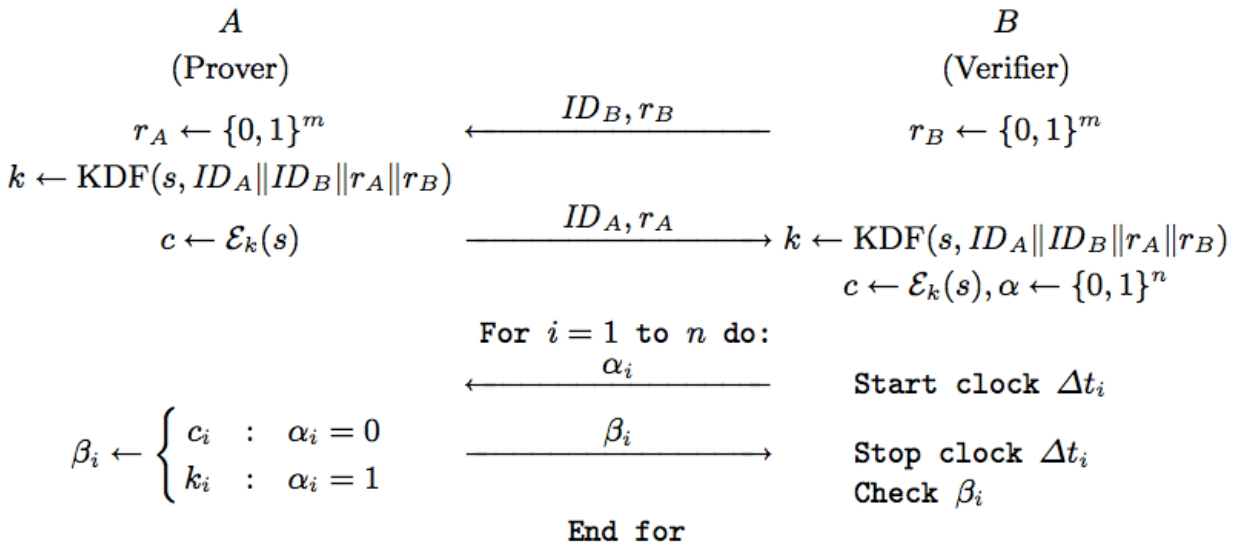

Figure 2.8: The first distance bounding protocol that is resistant to Terrorist Fraud, taken from[66]

approach strictly requires the nonce to be one-time use. Otherwise, it would leak some sensitive information. More importantly, when the return channel is accessible to the adversary, the adversary can see the result of protocol (accept or reject), and this approach is vulnerable to a so-called non-narrow MiM attack [45]. This attack works as follow: when the adversary relays communication between the verifier and the honest prover, it flips the challenge c_i to be c_i'. According to the response from the honest prover, the adversary learns the correct response for c_i'. The adversary then sends the response with the random selected bit to the verifier and learns the correct response to c_i by observing the final output of the verifier (acceptance or rejection). Now, the adversary learns ith bit of R_1 and R_2 and is able to deduce k_i. It can then repeat this for each i and obtain the secret key k. As such, the adversary can succeed in impersonating the prover.

2.2.5 Avoine et al. (TBD protocol)

In [2], Avoine et al. proposed to use secret sharing to prevent this type of non-narrow MiM attack. This is achieved by splitting the secret key into three different shares and

making them become possible responses. The concept behind this is that three responses become the shares of a threshold secret sharing scheme, so that any two shares alone leak no information about the secret. Following this idea, challenges are now chosen from three possible values $c_i \in 1, 2, 3$. The response to challenge $c_i \in 1, 2, 3$ becomes R_1, R_2, and $R_1 \oplus R_2 \oplus k$, respectively. In each round, the adversary can obtain up to 2 responses (with the returning channel). Knowing 2 out of 3 shares, however, would not give any advantages to an adversary. In this way, Terrorist fraud and non-narrow MiM attacks can be prevented.

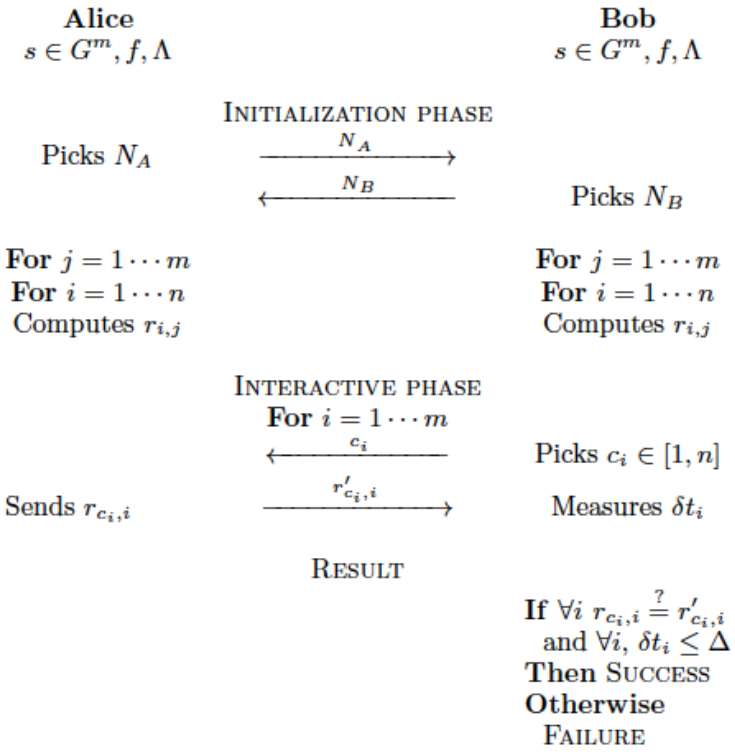

Figure 2.9: TBD protocol, taken from[2]

However, the security of TBD protocol (and other distance bounding protocols use PRF to generate response table) is broken by [7]. As shown in [7], one can make the protocol insecure by constructing PRFs artificially to bias the output distribution of PRF. A malicious prover can construct specific PRF by PRF programming and choose a specific nonce to generate predictable response table in the initialization phase in order to execute distance fraud. This trapdoor is fixed by using PRF masking[7]. In this solution, the response table

is actually generated by the trusted verifier and then masked by the output of PRF. In this way, manipulation of PRF will not affect the distribution of response tables.

2.2.6 Boureanu, Mitrokosta and Vaudenay (SKI protocol)

All previous distance bounding protocols consider noise communication environments only. However, Hancke[38] found that, with the help of noisy environments, the dishonest prover can make the helper respond to enough challenges correctly without leaking long term secret key. In other words, Terrorist fraud could be possible due to noise in distance bounding protocols. The idea of this attack is to take advantage of noisy environments. When there is noise, the correctness of the response is subject to noise. In order to keep the completeness property of distance bounding protocol, a linear number of errors needs to be tolerated, depending on the noise level. Therefore, instead of requiring all responses to be correct, only t out of n rounds need to be correct for a successful run of the protocol. However, noticed by[38], the malicious prover can actually take advantage of this to reveal the response to the helper for t out of n rounds, which will only leak t bits of secret key k to the helper. By fixing the selection of t out of n, the malicious prover will succeed in terrorist fraud in every run of the protocol without further leaking the secret key.

In order to solve all of the previously mentioned vulnerabilities, the SKI protocol combines the advantage of the Hancke and Kuhn protocol [39] and the Avoine et al protocol [2], which results in a light-weight distance bounding protocol that is secure against terrorist fraud with the idea secret sharing. At the same time, multiple challenges and responses decrease the success probability of distance fraud and mafia fraud. Also, several improvements need to mention here: (1) This trapdoor of PRF[7] is fixed by using PRF masking. In this solution, the response table is actually generated by the trusted verifier and then masked by the output of PRF. In this way, manipulation of PRF will not affect the distribution of the response table. (2) In the initialization phase, a linear transformation is used to be applied on the secret x, before using x in the response function, in order to protect against a terrorist fraud

observed by Hancke [38].

2.3 Formalizing Distance Bounding

As one can see, distance bounding protocols have attracted a lot of attention from researchers. Unfortunately, most of them only come with designs and lack formal approaches. This leads to inaccurate analysis and unfair comparisons between protocols. Avoine et al first gave a complete but informal model for distance bounding protocol in [1]. It aims to improve analysis of distance bounding protocols. In this work, the security of distance bounding is defined as the combination of authentication and distance-checking. This framework not only defined a thorough terminology about frauds, adversaries, and provers, which clarified many misleading terms, but also explored the adversary's capabilities and restrictions to define adversary's impact on the protocol. Another contribution of this work is an analysis of the Munilla-Peinado protocol [54] to further illustrate the framework. However, this framework failed to state its assumptions clearly and became informal compared to other works.

Another promising model for distance bounding protocols is proposed by Durholz et al in [24]. This model formally defines security against the attack types mentioned above and evaluates this framework by assessing the security of the Kim-Avoine distance bounding protocol [45]. However, in this model, time (or distance) is modelled in an implicit way, while time (or distance) is a major component of the distance bounding protocol. Also, as mentioned in [29], the definition of attacks in this model seems too strong; therefore, many protocols are proven to be insecure under this model.

Recently, Boureanu et al [8] introduced a formal model of distance bounding protocols based on a Turing Machine model. In this model, parties are modelled as Turing Machines and the distance bounding protocol is modelled as the interaction between Turing Machines. The security of the distance bounding protocol is evaluated through its resistance to different attacks. Three general classes of attacks are defined here to include all of the

above-mentioned attacks. Boureanu et al then prove the security of SKI protocol [10] and extended it to be the first distance bounding protocol that enjoys provable security. In this thesis, this approach is followed in order to construct a model for different protocols.

2.3.1 Turing Machine environment

A Turing Machine is a mathematical model of a computer, and any computable algorithm can be implemented using a Turing Machine. Most cryptographic protocols use Turing Machine in their models as mathematical proof techniques because a Turing Machine is a mathematical object. The definition of Turing Machine is as follows:

Definition 1 (Turing Machine (taken from [68]) *A turing Machine M is a tuple $M = (Q, \Gamma, b,$*

$\Sigma, \delta, q, F)$, where Q is a finite, non-empty set of states, Γ is the tape alphabet, $b \in \Gamma$ is the blank symbol, $\Sigma \subseteq \Gamma - \{b\}$ is the input alphabet, $q \in Q$ is the initial state, $F \subseteq Q$ is the set of final or accepting states, and $\delta : (Q - F) \times \Gamma \to Q \times \Gamma \times \{L, R\}$ is a partial function called the transition function. A Turing Machine is started by setting its internal state to q, writing the input string on the (infinite) tape, and positioning its read/write head on the first input symbol. As long as no accepting state is reached, $\delta(s, q) = (s', q', m)$ is evaluated, where s is the actual state of the Turing Machine, and q is the symbol read by the read/write head from the tape. As a result of this evaluation of δ, the internal state of M is set to s', the symbol q' is written on the tape by overwriting q, and the read/write head either moves right ($m = R$) or left ($m = L$).

2.4 Modeling the time

Although time plays an important role in various cryptographic protocols, for example to guarantee freshness in authentication protocols or to limit lifetime for public keys, few works discussed the problem of modelling time formally. In this work, time will be modelled and

the security of a distance bounding protocol that utilizes this time model will be proved. Thus, some review on related work about time models will be provided.

2.4.1 Timestamps

[49] provides an overview of the usage and security problems of timestamps. In most of the literature, timestamps are used for timestamping documents to guarantee that the document exists at a certain time. This is important for patens, contracts, wills, and other legal documents, which critically depend on the date on which they were signed. Digital timestamp systems were first introduced by Haber and Stornetta[34]. Typically, digital timestamp systems require a central trusted authority for generating the timestamp. Moran et al [53] uses a different model for timestamping documents by assuming a unique random string is broadcasted during each time period. Timestamps are also used in authentication protocols to guarantee the freshness of messages to prevent replay attacks.

2.4.2 Modelling Time for Authenticated Key Exchange Protocols

In [4], a generic modelling technique that can capture the use of timestamps was introduced. Authors applied this technique to two different popular models (the Bellare-Rogaway approach and the Canetti-Krawczyk approach) to analyze the authentication and key agreement protocols utilizing time stamps. In this work, time was modelled by providing parties with a local internal clock, which was incremented by sending TICK requests. However, in terms of Turing Machine model, there is no mathematically sound definition for using local clocks as part of Turing Machines.

In order to better accommodate Turing Machine environments, [68] modelled time as a global counter to guarantee freshness in cryptographic protocols. This approach allows using Turing Machines as computational models while staying as close as possible to the well-established definition of secure authentication protocols. In this work, time is defined as a global counter that provides a time reference to all parties in the system. If a fresh

message has to be sent, the sending Turing Machine requests a timestamp $t_s = (t, aux)$ from the global counter. Upon reception of such request, the global counter increase its local counter ($t \leftarrow t + 1$) and returns the timestamp. When a message is received, the receiving Turing Machine compares the time value in the timestamp to its local counter to determine the freshness of the message. The local counter is updated once the receiving Turing Machine confirms that the message it received is a fresh one.

2.5 Proximity-based authentication

Proximity-based authentication refers to a class of authentication mechanism that not only requires the user to prove its identity by holding the corresponding secret key, but also requires the user to come close to the verifier when authenticating. This is an effective way to defeat Man-in-the-middle (MiM) attacks because the attacker is forced to come close the verifier; therefore the probability of being detected is significantly increased. Besides defeating MiM attacks, proximity-based authentication is widely applied to different application scenarios. For example, access control [33, 65] can be deployed based on proximity-based authentication (also called proximity-based access control) to prevent unauthorized access. Considering an access control system for secure room in a building, the user who holds the correct identity card is authenticated by approaching the reader in order to gain access to the room. When applied to mobile devices, proximity-based authentication offers an attractive way to bootstrap secure communication channels between devices [44, 82]. For example, users may place their mobile phones next to each other to establish a secure communication for document transfers or users may put their mobile devices close to their laptops or other large displays to mirror the screen. In terms of contactless payment systems [48, 88], proximity-based authentication can improve security by requiring multi-factor authentication.

Normally, proximity estimation in proximity-based authentication is achieved in three

different ways. The first approach is to explore the hardware properties of an RFID (Radio-Frequency Identification) device [27]. Generally, most RFID tags do not have local power supplies. Instead, tags are powered by electromagnetic induction from magnetic fields produced near the reader. Hence, proximity assumption is established when the RFID tag is activated. However, this approach is vulnerable to relay attacks, in which the adversary comes close to the victim with a reader to activate the tag and relay all messages to the other reader to get authenticated. Several works have realized this attack is practical[36, 30]. Hence, proximity evaluation is now conducted by the following two approaches.

The context-based approach achieves proximity evaluation by asking users to sense the context information and compare the results. A number of such schemes have been proposed, including utilizing ambient light [35], the acoustic environment [80, 35], temperature, humidity, air pressure [70], and Radio Frequency based signals such as WiFi and Bluetooth [80, 44, 82]. In these approaches, the prover and the verifier sample their context via sensors at the same time and compare the result to see if two parties are in the proximity to one another. If the measurements are similar enough, we say the two parties are close. This approach is defined as the context-based approach.

The other approach utilizes distance bounding protocol [39, 64, 66, 10] to evaluate proximity. Although this imposes challenges for implementation because distance bounding protocol requires fast processing time in the range of nanoseconds, it provides better security guarantee compared to context-based approaches. This approach is defined as DB-based approach.

Chapter 3

Distance Lower Bounding

3.1 Introduction

Distance (upper) bounding (DUB) protocols [11, 39, 72, 66, 45, 2, 10] have been widely studied in recent years: a *verifier* V interacts with a *prover* P to obtain assurance that the prover is at a distance at most B from the verifier. To estimate distance, secure DUB protocols measure the round-trip time of a *fast-exchange challenge-response protocol*, using electromagnetic (EM) communication and assuming constant speed for EM signals. We refer to protocols that use this method for distance estimation, as class $\mathcal{E}F$ (i.e., EM fast-exchange). In this chapter, we consider the dual problem of *distance lower bounding* (DLB) where a prover P wants to prove its distance from a verifier V is higher than a bound B. DLB problem naturally arises in application scenarios where privileges are given based on the distance of a requester to a provider. For example a company offering unrestricted Internet access to games and entertainment software to employees when they are outside of main office area of the company campus (e.g, Google campus), and restricted access when employees are within the main office area. Here the requirement is for employees to prove they are outside the main working area. A second scenario is when the parking lot is divided into zones and the parking charge depends on the distance of the car to the main point of interest (e.g. discounted rate will be given if users park their car further away from the shopping mall entrance). In both cases once the privilege is granted based on the distance, one needs to use monitoring mechanisms such as continuous authentication to ensure that the user stays within the claimed area. Embedding such authentication in streaming services such as games or music is straightforward. For the latter scenario, one can use random scanning of the area to ensure correct claim. Although determined users may be able to bypass the

authentication, but they will be inconvenient (e.g. move the car frequently) and also have to accept the risk of detection and penalty.

A first approach to solving DLB problem would be to use a DUB protocol. However although a successful run of a DUB protocol proves that the prover is close to the verifier, its failure does not say anything about the distance of the prover. This is because none of the DUB protocols protect against distance enlargement attack [14] where the malicious prover enlarges the distance by delaying the response. (Other applications of DUB protocol, such as using DUB protocols with multiple verifiers for secure positioning[14] will also be vulnerable to distance enlargement attack.) A second approach would be to use Global Positioning System (GPS)[41] to determine the location of the user. However one needs to trust the GPS measurements, which is known to be vulnerable to attacks such as GPS spoofing attack [84] where fake satellite signals is used to modify the GPS location data. This solution also results in privacy loss and so one needs to consider privacy enhancing GPS solutions, requiring extra infrastructure.

Attacks on DLB protocols depend on the application scenario. In Section 3.3.1 we consider attacks that are applicable in the above application scenarios and argue that they are parallel to the attacks on DUB protocols. DUB protocols have been analyzed against three broad classes of attacks[83]: *distance fraud (DF)* where the prover is malicious and wants to shorten its distance to the verifier; *collusion fraud (CF)* where the prover is malicious and has a helper that would assist them to shorten its distance to the verifier ; and finally *Man-in-the-middle (MiM) attack* where the prover is honest and is the victim of an external attacker who aims to shorten the distance between the honest prover and the verifier. These classes of attack include impersonation attack, Mafia fraud and Terrorist fraud, that are traditionally considered for DUB protocols. We show that the above attacks are also applicable to DLB and capture the main DLB attacks.

The solution to the DLB problem depends on the trust assumption. The DLB problem

Table 3.1: Impossibility result of DB protocols with different trust assumptions

Trust	DLB problem	DUB problem
Trusted prover	Possible (e.g., secure ranging[79])	Possible[1] (e.g., DB [10])
Fully untrusted prover[2]	Impossible (Section 3.3.2)	
Partially trusted prover[3]	Possible (Section 3.5, Π_{DLB-BM})	

in a setting that *both the prover and the verifier are trusted*, has been considered in [79]. In this chapter, we consider a setting where *the prover is untrusted* and the verifier is trusted.

DUB protocols have been primarily designed when an untrusted prover interacts with a trusted verifier. Unlike DUB problem, provide a solution to DLB problem under the same assumptions. In Section 3.3.2, we prove that it is impossible to have secure DLB protocol if the prover is untrusted and have full control of its computing device (hardware and software), allowing them to deviate arbitrarily from the protocol. One however can have secure protocols by making assumptions on the malicious prover's access to the device and/or the communication channel. Table 3.1 summarizes trust assumptions in DUB and DLB.

Our contribution. We motivate and initiate the study of distance lower bounding (DLB) problem in a setting where the prover is untrusted. We construct a security model for DLB problem and define three broad classes of attacks: *distance fraud (DF)* where the prover is malicious and acts alone to enlarge its distance to the verifier; *collusion fraud (CF)* where the prover is malicious and has a collaborating helper that it uses to enlarge its distance to the verifier ; and finally *Man-in-the-middle (MiM) attack* where the prover is honest and is the victim of an external attacker who aims to enlarge the distance between the honest prover and the verifier. We prove that security against any of these attacks without making any *physical assumption*[4] is impossible. In particular, a fully malicious prover can *always* succeed in the

[1]DUB protocols with fully untrusted prover are secure for all types of trust assumptions.

[2]The malicious prover who has unrestricted control of hardware *and* software of the prover device.

[3]The malicious prover who has restricted control of the hardware of the prover device, but can run malicious software on the device.

[4]Physical assumptions include limited access to the device hardware, and/or the communication channel.

DF attack, and an external attacker (without the cryptographic credentials) can always jam-and-delay the signal between the verifier and the prover and succeed in the MIM attack. This also implies that a malicious prover that has a helper (CF) will always succeed. Fourth, we study reasonable assumptions under which there is a solution to the problem, and construct a secure DLB protocol under specific assumptions. We analyze security of our DLB protocol against DF, MiM and CF attacks. We also estimate time, memory and energy requirements of our protocol and conclude with open questions and directions for future research. In Section 3.4 we give a more general approach to constructing DLB using software attestation systems.

Organization. Section 3.2 breifly introduces the related work. Section 3.3 introduces the model and impossibility results. Section 3.5 describes our approach and proposed protocol. Section 3.6 provides the security analysis. Section 3.7 presents relevant practical considerations and Section 3.8 concludes the paper.

3.2 Related Work

There is a large body of research on secure positioning and distance estimation problem, including distance bounding protocols[11, 39, 10], positioning techniques[14, 41] and secure ranging protocols[79]. As we argued earlier, these approaches are not directly applicable to the DLB problem in the setting that the prover is not trusted. GPS systems use a set of satellites signals to determine the location and are designed for non-adversarial setting, and so GPS systems are vulnerable to signal spoofing attacks[84]; DUB protocols protect against malicious provers trying to shorten the distance, but are in general vulnerable to the distance enlargement attack[14]; secure positioning systems use a DUB protocol with multiple verifiers to triangulate the prover's location, but is also vulnerable to distance enlargement attack, making positioning an insecure approach for DLB; and secure ranging systems only consider non-adversarial setting as well.

To our knowledge this is the first paper to study DLB problem in a setting where the prover is not trusted. Our approach to defining attacks, distance estimation, and design of the protocol is inspired by the large body of literature on DUB, in particular, [83] for formalization of attacks, and [39, 10] for the design of the protocol. The use of bounded-memory assumption for the prover's device in the context of secure code update had been considered in [59].

3.2.1 Boureanu et. al framework

Boureanu et. al [10, 9, 8] showed security weaknesses of a number of protocols and proposed a security driven design for a class of protocols called SKI, which builds on the previous key papers including [66, 2]. Our distance estimation sub-protcol follows the design approach of SKI with security only for non-noisy environments. This allows a simplified design.

3.2.2 Bounded memory and bounded retrieval model

Memory bounded adversaries in cryptography were first considered in information theoretic setting [12]. A related model is Bounded-Retrieval Model (BRM) where the adversary is limited in the number of bits that it can access. BRM [25, 22, 15] is a well established leakage model where leakage parameter is an arbitrary and independent parameter in the system design showing the absolute amount of leakage (to the adversary) that the system can tolerate.

3.3 DLB - Model and Impossibilities

We consider a multi-party system where a party U is modelled by a probabilistic polynomial time (ppt) interactive Turing machine (ITM) with a location loc_U, and some pre-shared keys. Messages that are sent from one location to another, travel at the speed of light and the time taken for travel can be used to determine the distance between the two locations. We

assume parties will receive privileges based on their pre-shared secrets and their locations. A party can be a *prover* or a *verifier*. A *prover* P engages in a two-party protocol with a *verifier* V, to prove the claim that its distance to the verifier satisfies certain bound. Honest parties run predefined algorithms for their side of the interaction.

The verifier V is always honest. The prover however may be malicious, in which case it is denoted by P^*. A malicious prover deviates from the protocol to make incorrect distance claim and access privileges that they are not entitled to. The experiment is expanded to include an external adversary $\mathcal{A}$ who interacts with the parties in the system according to its defined abilities as stated below.

A protocol instance defines an experiment denoted by $exp = (P(x; r_P) \leftrightarrow \mathcal{A}(r_A) \leftrightarrow V(y; r_V))$ where r_P and r_V are random coins, and x and y are secret keys of the prover and the verifier, respectively. At the end of a protocol instance V has an output Out_V which is 1 or 0, showing acceptance or rejection of the DLB protocol, respectively. The prover does not have an output.

A participant in an experiment has a *view* consisting of all its inputs, coins, and messages that it can see. The attacker $\mathcal{A}$ may interact with multiple P's and V's, and its view will include all these interactions. Throughout the paper $\Pr_r[event : experiment]$ denotes the probability of the *event* for the *experiment*, where r denotes that random coins used in the experiment.

Definition 2 (DLB Protocol) *A Distance Lower Bounding (DLB) protocol is a tuple* (Gen, P, V, B) *where* $(x, y) \leftarrow Gen(1^s, r_k)$ *is a randomized key-generation algorithm that takes security parameter* s *and randomness* r_k *and outputs keys* x *and* y*;* $P(x; r_P)$ *is the prover's ppt ITM that takes a secret key* x *and some randomness* r_P*;* $V(y; r_V)$ *is the verifier's ppt ITM taking secret-key* y *and randomness* r_V*, and* B *is a distance-bound. It satisfies two properties:*

- *Termination:* $(\forall s)(\forall R)(\forall r_k; r_V)(\forall loc_V)$ *if* $(.; y) \leftarrow Gen(1^s; r_k)$ *and* $(R \leftrightarrow V(y; r_V))$

is an execution of the protocol between the verifier and any (unbounded) prover algorithm, V halts in $Poly(s)$ computational steps;

- *p-Completeness:* $(\forall s)(\forall loc_V; loc_P$ *such that* $d(loc_V; loc_P) \geq B)$ *we have*

$$Pr_{r_k;r_P;r_V}\left[Out_v = 1 : \begin{array}{l}(x;y) \leftarrow Gen(1^s; r_k)) \\ P(x; r_P) \leftrightarrow V(y; r_V)\end{array}\right] \geq p$$

3.3.1 Attacks on DLB Protocols

We consider three classes of attacks: distance fraud (DF), man-in-the-middle attack (MiM), and collusion fraud (CF). In DF, P^* with $d(P^*, V) < B$ wants to convince V that its distance is at least B. In MiM attack, an external attacker who does not have the secret key interacts with multiple P's and V's, and finally succeeds in taking the role of a prover in a protocol instance (See Figure 3.1). In CF, P^* colludes with a helper to claim a longer distance to V (See Figure 3.2). The collusion should not leak the prover's secret key to the helper. The formal definitions of the attacks are below.

Distance fraud (DF) attack. In this attack, a dishonest prover who is closer than the distance (bound) B, wants to convince the verifier that it has a distance at least B.

Definition 3 (DF-resistance) *A DLB protocol* Π *is* α*-resistant to distance fraud if* $(\forall s)(\forall P^*)(\forall loc_v)$ *such that* $d(loc_v, loc_{p*}) \leq B)(\forall r_k)$*, we have*

$$Pr_{r_v}\left[Out_v = 1 : \begin{array}{l}(x,y) \leftarrow Gen(1^s; r_k) \\ P^*(x) \leftrightarrow V(y; r_v)\end{array}\right] \leq \alpha,$$

where P^* *is a dishonest prover. Because of the concurrent setting we effectively allow polynomially bounded number of* $P(x')$ *and* $V(y')$ *close to* $V(y)$ *with independent* (x', y')*.*

Distance hijacking. Definition 3 captures *distance hijacking attack* [18] against DLB protocols. In this attack P^* who is at distance $< B$, uses DLB communications of unaware honest provers at a distance $\geq B$ to claim a distance $\geq B$.

Man-in-the-middle (MiM) attack. In an MiM attack an external adversary who does not have the secret-key of the protocol, will first interact with honest provers and verifiers, and then takes the role of a prover in a protocol instance with the verifier (See Figure 3.1).

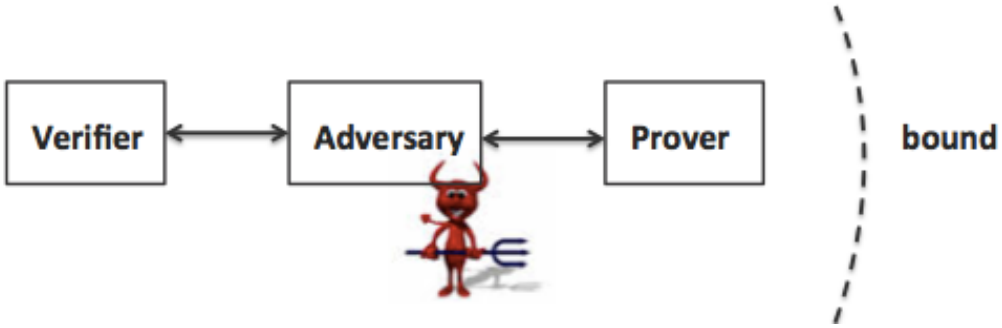

Figure 3.1: MiM attack in DLB

Definition 4 (MiM-resistance) *A DLB protocol* Π *is* β*-resistant to MiM attack if for all* s*, for all polynomially bounded* m, ℓ, z*, for all polynomially bounded* (A_1, A_2) *and for all locations that satisfy* $d(loc_{P_j}, loc_V) < B$*, where* $j \in \{m+1, \cdots, \ell\}$*, and we have,*

$$Pr_{r_v}\left[\begin{array}{ll} & (x,y) \leftarrow Gen(1^s) \\ Out_v = 1: & P_1(x)...P_m(x) \leftrightarrow A_1 \leftrightarrow V_1(y)...V_z(y) \\ & P_{m+1}(x)...P_\ell(x) \leftrightarrow A_2(View_{A_1}) \leftrightarrow V(y) \end{array}\right] \leq \beta.$$

Here probability is over all random coins of the protocol, and $View_{A1}$ *is the final view of* A_1*. The definition allows polynomially bounded number of* $P(x')$*'s,* $P^*(x')$*'s, and* $V(y')$*'s with independent* (x', y')*, anywhere.*

In this definition the attacker can have a learning phase during which it interacts with m honest provers and z verifiers. It then uses its view in the attack phase and engages in $\ell - m$ protocol instances between honest provers and the target verifier.

Mafia fraud and impersonation attack. Definition 4 is general and covers Mafia fraud and impersonation attack as special cases. In Mafia fraud, there is no learning phase. The attacker interacts with an honest prover and attempts to make the verifier output accept.

That is, $m = z = 0$ and $\ell = 1$ in the attack phase. In impersonation attack the attacker uses multiple, possibly concurrent, interactions with the verifier to make the verifier output 1.

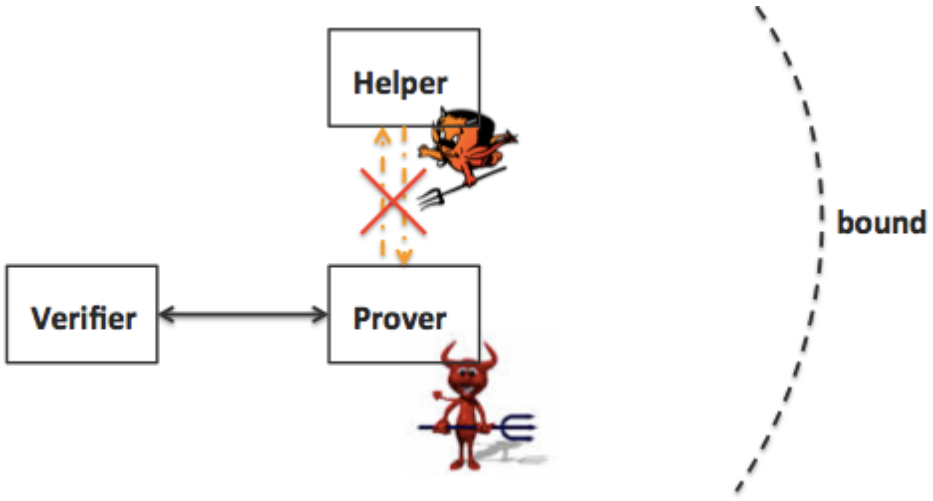

Figure 3.2: Collusion fraud in DLB

Definition 5 (CF-resistance) *A DLB protocol* Π *is* (γ, η) *resistant to collusion fraud if* $(\forall s)(\forall P^*)(\forall loc_{v_0})$ *such that* $d(loc_{v_0}, loc_{P^*}) < B$, *for all ppt* A^{CF} *if we have,*

$$Pr_{r_v}\left[Out_{v_0} = 1 : \begin{array}{l}(x,y) \leftarrow Gen(1^s) \\ P^*(x) \leftrightarrow A^{CF} \leftrightarrow V_0(y)\end{array}\right] > \gamma,$$

then there exists an extended[5] *MiM attack with* m, ℓ, z, P_i, V_j, *and an attacker* (A_1, A_2) *such that,*

$$Pr\left[Out_v = 1 : \begin{array}{l}(x,y) \leftarrow Gen(1^s) \\ P_1^{(*)}(x)...P_m^{(*)}(x) \leftrightarrow A_1 \leftrightarrow V_1(y)...V_z(y) \\ P_{m+1}(x)...P_\ell(x) \leftrightarrow A_2(View_{A1}) \leftrightarrow V(y)\end{array}\right] > \eta.$$

Here $P^{(*)}$ *is a prover that is either* P *or* P^*. *We have* $d(loc_{P_j}, loc_V) < B$, *for all* $j \in \{m+1 \cdots \ell\}$.

[5] *Because learning phase allows interaction with* P^*.

Collusion fraud implies that if a malicious prover P^* who is close to V_0 can succeed in extending the distance with the help of A^{CF}, then there is an adversary (A_1, A_2) who will succeed in an (extended) MiM attack where multiple instances of $P^{(*)}(x)$, denoting honest or dishonest prover are used during the learning phase.

Terrorist fraud. In Terrorist fraud, P^*, with $d(P^*, V) < B$, gets aid from a helper who does not have the secret key, to succeed in an instance of the protocol with the verifier. Definition 5 captures terrorist fraud as a special case by letting $m = z = \ell = 1$, by simply allowing A_1 to run A^{CF} and succeed in impersonation and making V to accept.

3.3.2 Impossibility results

We consider protocols in $\mathcal{EF}$. Let C denote speed of light, t_c and t_r denote the verifier's clock readings, when the challenge is sent and the response is received, respectively. If the received response is correct, the verifier calculates $T_\Delta = t_r - t_c$ to estimate the distance of the prover. Let T_{proc} denote the *processing time* of the prover. The verifier estimates the prover's distance D as

$$D = \frac{(T_\Delta - T_{proc})C}{2}. \tag{3.1}$$

In DF, considering a malicious prover with full (hardware and software) control over the proving device. The malicious prover at close distance D is a registered user of the system and knows the credential to calculate correct responses to the verifier challenges. To claim a longer distance, the prover simply modifies the execution to add appropriate delay by tampering with the hardware/software. This results in the impossibility of DF resistance. Without assuming any restrictions, a MiM adversary can easily succeed in DLB by sitting between the verifier and an honest prover and launching a jam-and-delay attack to add appropriate delay. Hence, we have impossiblity of MiM resistance. The impossibility of resistance against CF follows immediately from the above impossibility results for DF and MiM resistance. We can state this even more generally: Any setting (set of assumptions)

that makes DLB security against DF or MiM impossible will make it impossible to resist against CF. The reason is CF attackers can make prearrangements to simulate a DF attack (e.g., without helper being engaged in the DLB instance) or a MiM attack (e.g., by the dishonest prover acting as an honest one). These statements are formalized below:

Theorem 1 (Impossibility Result) *1. Any DLB protocol in $\mathcal{EF}$ is vulnerable to DF if P^* has full (hardware and software) control over the prover's device.*

2. No DLB protocols in $\mathcal{EF}$ can provide β-resistance with $\beta < 1$ to MiM attack launched by an external attacker who can jam and delay messages to (or from) the prover.

3. For any DLB protocol in $\mathcal{EF}$, P^* can succeed in CF with probability 1 and negligible key leakage to the helper, if the helper has full access to the communication channels with P^*, and P^* has full control over the prover's device. The result holds even if communication is only allowed in one direction between the prover and the helper.

The proof sketch of Theorem 1 is as follows:

Proof sketch of Theorem 1:.

For (1), assume a malicious prover (who can calculate correct responses to the verifier challenges) at $D < B$. To claim a longer distance $D+D'$, the prover modifies the execution to add appropriate delay by tampering with the hardware/software and responds after $2D'/C$ second(s). The attack succeeds with probability 1.

For (2), A MiM attacker can use the following strategy: upon receiving a message from one party, the adversary jams the signal to prevent it from being received by the other, and later forwards it with appropriate delay. The theorem holds irrespective of any hardware assumption (e.g. bounded memory) on the prover's device, and succeeds even if jam and delay can be applied in one direction only.

For (3), note that CF resistance requires both DF resistance and MiM resistance: A CF attacker can simply simulate a successful DF attacker by simply ignoring the helper; or it

Table 3.2: DLB security against the three attacks in different settings.

Attacks	Assumptions			
	No Assumption	Prover's device	Communication	Combined
		[BM]	$[\overline{\mathsf{OC}}]$	$[BM + \overline{\mathsf{OC}}]$
DF-security	×	✓	×	✓
MiM-security	×	×	✓	✓
CF-security	×	×	×	✓

can also simulate a successful MiM attacker, by allowing P^* in the CF attack to run the algorithm of P, and the helper in CF to run the algorithm of the MiM adversary, A^{MiM}.

3.3.3 Restricted DF, MiM, and CF

To remove the above impossibility results, we must use reasonable assumptions (restrictions) on the adversary's control of the device and/or the communication channel. We refer to attacks under these conditions as restricted DF, MiM and CF (rDF, rMiM and rCF), to emphasize extra assumptions are needed.

Notations. We use PD to denote the prover's device, and $\text{rX}^{[Y]}$ to denote restricted version of attack X, where $X \in \{DF, MiM, CF\}$ and restrictions are stated in Y. For example, $\text{rDF}^{[BM]}$ refers to the restricted DF attack, under the restriction that PD has bounded memory.

Table 3.2 summarizes our impossibility results and shows assumptions that are used in our construction in Section 3.5. The assumptions that we use for security against rDF are, (i) P^* cannot access (read or write) the PD 's read-only memory (ROM), and (ii) PD has *bounded memory (BM)*. Note that the first assumption still allows P^* to inject malicious codes into the device writable memory (RAM), and modify correct execution of the protocol. The bounded memory assumption is a well-established model in cryptography [12], and has also been used in the design of security systems [59]. To achieve the security against rMiM and rCF, in addition to the above assumptions, we require the helper to have no *On-line Communication (OC)* with the prover during the fast-exchange phase. In Section

3.5, we present a DLB protocol that provides security against rDF, rMiM and rCF under the above assumptions. Note that one may achieve rDF, rMiM and rCF resistance using other assumptions that restrict the prover and the helper. For example instead of assuming a root of trust on the PD, one may establish a dynamic root of trust using software attestation. We give a software attestation-based DLB protocol in Section 3.4. This protocol also requires no online-communication assumption for security against all attacks. We also provide an overview of security analysis and implementation challenges of the protocol.

3.4 Software attestation based DLB protocol

In the setting of $\text{rDF}^{[\text{SA}]}$, we do not assume a root of trust on the PD. Instead, we establish a dynamic root of trust, and guarantee correct execution of the fast-exchange phase, as part of the DLB protocol. In [69], a software-based *externally verifiable code execution (EVCE)* system (called Pioneer) is proposed to assure correct code execution over an untrusted device. It guarantees that a piece of code, referred to as*target executable*, executes untampered from possible malicious codes that may reside on the device, by dynamically establishing a *root of trust* and secure execution environment, on the device. The protocol $\Pi_{DLB-EVCE}$ uses EVCE in each fast-exchange round.

The fast exchange phase uses the *externally verifiable code execution (EVCE)* system proposed in [69]. In EVCE, the goal is to ensure that the correct target code is invoked, and the execution is untampered in the sense that, other than performing denial of service attacks, no malware that may exist on the computing device can interfere with the execution of the target code. In this model, no root of trust on the device is assumed: the verifier establishes an isolated execution environment on the device using the verification function, and uses checksum and time to ensure the correct execution of the verification function; it then calculates the keyed-hash value of the target code and sends it to the verifier, assuring the integrity of the target executable. By giving the verification function and the target

code the highest privilege level of the CPU, the execution of the verification function and the target executable will be atomic, and no untrusted code can execute before the above two codes complete their execution. EVCE requires the target executable code to be self-contained. To satisfy this requirement, we propose protocol $\Pi_{DLB-EVCE}$ that uses EVCE in each round of DLB.

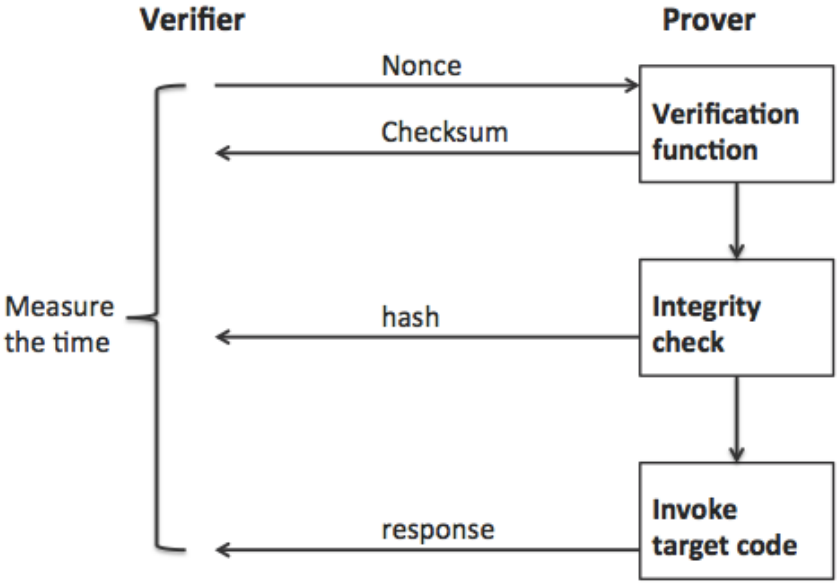

Figure 3.3: Software attestation based DLB protocol

$\Pi_{DLB-EVCE}$. The initialization and verification phases of this protocol are as in Section 3.5.1. Figure 3.3 shows a fast-exchange round of the protocol. In each round, the verifier sends a nonce that invokes the verification function of the prover. The verification function computes the checksum over itself, and sets up an isolated execution environment. The keyed-hash value of the target executable is calculated for the purpose of integrity check, and then the target code is invoked. The target code consists of the function used for finding the response to the nonce, and sending the response to the verifier. The time T_Δ is measured from the time that the nonce is sent, until the time that the response is received. The round trip time is then calculated as $T_\Delta - T_{proc}$, where T_{proc} consists of the time used for, (i) finding the checksum of the verification function, (ii) integrity check of the target code, and (iii)

finally its invocation. The main drawback of the protocol however, is the accuracy of the distance estimation that depends on correct estimation of T_{proc}. A correct estimation of T_{proc} in turn depends on the correct estimation of these three components, and so high accuracy would be hard to achieve.

Security analysis of $\Pi_{DLB-EVCE}$. Because of space, we only give an outline of the security proof. DF-resistance of $\Pi_{DLB-EVCE}$ follows from the correct execution of fast-exchange rounds, which is guaranteed because of the security of software-based attestation EVCE. This means that no delay will be introduced in the execution of the protocol, and at the end of the fast-exchange phase, the distance will be correctly estimated. EVCE also requires a trusted network to eliminate proxy attacks where the computing device asks a faster computer (proxy) to compute the checksum on its behalf, and so for this construction only rMiM$^{[\overline{\mathsf{OC}}]}$ and rCF$^{[\mathsf{SA},\,\overline{\mathsf{OC}}]}$ can be considered. That is, any other restrictive assumptions for rMiM and rCF must include no-online communication. Security against rMIM and rCF follows from secure generation and sharing of the response table that ensures that the response bits cannot be guessed by outsiders (See Section 3.6).

3.5 DLB protocol Constructions

Assumptions and attack model. We assume that the PD is a bounded memory device, which has a protected memory (ROM), and a writable memory (RAM) with (fixed) L bit size. Without loss of generality, we consider RAM as an array indexed from 1 to L. The DLB protocol code is stored partly in ROM, denoted by DLB_{ROM}, and partly in RAM, denoted by DLB_{RAM}. We assume V has a shared key with the PD, and holds the same DLB code. The secret key of PD is stored in ROM and is accessible only to the code in ROM. We assume communication channel is noise free, although our results are extendable to noisy communication by applying similar methods as [72]. The adversary may store and run

arbitrary malicious code on the RAM of the PD, but is not able to tamper the hardware of the device.

Approach. Using equation 3.1, P^* at distance D can always delay the response by $2D'/C$ second(s) to claim a longer distance $D + D'$. Let T_{max} denote the maximum expected response generation (processing) time by the verifier. (This can be estimated for example, by measuring the processing time of a set of functional devices, and choosing T_{max} larger than all the measured times.) Knowing that $0 \leq T_{proc} \leq T_{max}$, the verifier uses the round-trip time T_Δ to obtain the following distance bounds.

$$D \geq D_{lower} = \frac{(T_\Delta - T_{max})C}{2} \tag{3.2}$$

We propose a protocol that assumes *bounded memory for* PD and enables V *to force an upper bound on the delay introduced by* P^*.

DLB protocol outline. Π_{DLB-BM} consists of three phases: *(i) Initialization:* during which the prover and verifier exchange nonces, and use them together with their pre-shared secrets to compute a shared response table. *(ii) Fast-exchange:* that consists of n challenge-response rounds, each round consisting of two consecutive challenges, followed by the two corresponding responses. *(iii) Verification:* during which the verifier checks the received responses to distance estimation and erasing sequence, and accepts if they satisfy the required conditions.

Our protocol referred to as Π_{DLB-BM}, is composed of two basic protocol blocks, Π_1 for distance estimation and Π_2 for secure memory erasure, shown in Figure 3.4. The combination effectively upper bounds the delay that the malicious prover can introduce. Π_1 is a pre-computation challenge-and-response distance estimation protocol [2] and Π_2 is a secure erasure protocol [59] (see Section 3.2). A challenge-response round in Π_1 is used for distance estimation. A challenge-response round in Π_2 however is used to refresh part of the memory that will not be required for the future rounds.

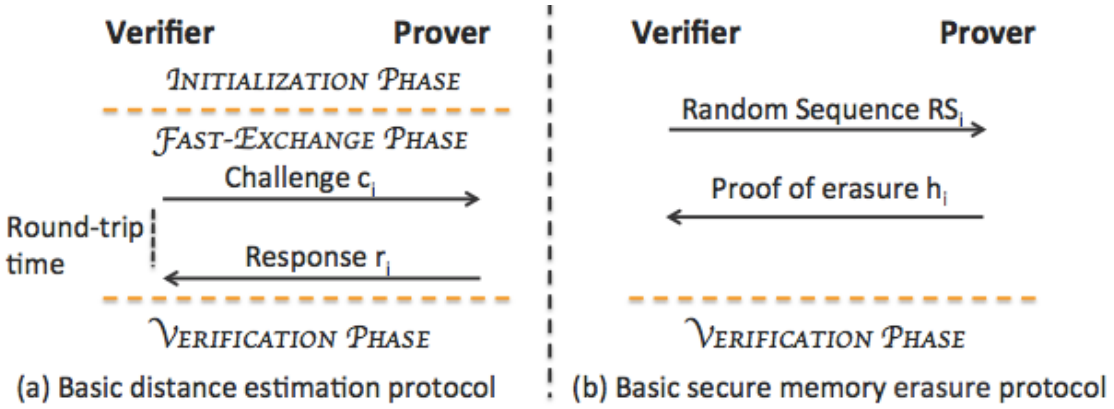

Figure 3.4: Basic building blocks Π_1(a) and Π_2(b)

In each round the verifier sends the challenge of the distance estimation protocol immediately followed by an erasing sequence of length z_i. The prover is expected to respond first to the distance estimation challenge, and then respond to the erasing sequence. The round-trip time is measured beginning at sending out the challenge and ending at receiving the response. The length z_i of the erasing sequence in round i is chosen such that storing it requires overwriting all memory (possibly including any malicious code) except the information required for distance estimation responses in rounds $\geq i+1$. This ensures that the prover cannot delay its response more than the time needed to receive $z_i - 1$ bits of the erasing sequence: The z_i-th bit will wipe out the response bit of the current challenge.

3.5.1 The protocol Π_{DLB-BM}

The secret key consists of two binary strings, x, and $\hat{x}$ in $\{0,1\}^\ell$ respectively. When clear from context, we refer to each string as *key* also. The protocol uses a secure Pseudo Random Function (PRF) $f_x : \{0,1\}^{2k} \rightarrow \{0,1\}^{2n}, x \in \{0,1\}^\ell$, and a secure keyed-hash function $(H_{\hat{x}})_{\hat{x}\in\{0,1\}^\ell} : \{0,1\}^* \rightarrow \{0,1\}^b$. Figure 3.5 shows the messages communicated in the three phases of the protocol.

Phase 1: Initialization phase. The prover generates a k-bit nonce N_p and sends it to the verifier. The verifier selects a k-bit nonce N_v and a $2n$-bit random string A, calculates

$M = A \oplus f_x(N_p, N_v)$, and sends (M, N_v) to the prover. With this information, the prover decrypts M to retrieve $A = M \oplus f_x(N_p, N_v)$ and stores it in memory. A is the *response table* that will be used by the prover to respond to challenges in Phase 2. Considering $A = (a_{(1,j)}, a_{(2,j)})$, where $j = 1 \cdots n$, as a sequence of n bit pairs, we define a third string $a_{(3,j)} = a_{(1,j)} \oplus a_{(2,j)} \oplus x$. $a_{(3,j)}$ is computed at run time from the response table and so is not stored in memory.

Phase 2: Fast-exchange phase. This phase proceeds in n consecutive challenge-response rounds. In each round $1 \leq i \leq n$, the verifier chooses a random challenge $c_i \in \{1, 2, 3\}$ and sends it to the prover, immediately followed by a random erasing sequence RS_i of length z_i. In section 3.5.2, we will discuss how z_i is determined. The role of RS_i is to prevent P from delaying the response to extend its distance. On receiving the challenge c_i, the prover will retrieve the response $r_i = a_{(c_i,i)}$. When $z_i - 1$ bits of RS_i are received, the prover must send r_i to avoid it being overwritten by the final bit of RS_i. The prover must also send the response to the erasing sequence (also referred to as proof of erasure h_i). By correctly designing the computation of the hash, the correct proof of erasure will "prove" that the prover has received and stored the full RS_i and also has kept the code DLB_{RAM} intact (see Section 3.5.2 for details). In addition, the verifier records the time difference $T_{\Delta,i}$ between sending c_i and receiving r_i.

Phase 3: Verification phase. The verifier checks the correctness of response r_i and proof of erasure h_i for all rounds, $i = 1 \cdots n$. It also verifies whether all response times are higher than a threshold θ, determined as follows. Let B denote the distance-bound, and $T(z_i - 1)$ denote the time interval required by the prover to receive $z_i - 1$ bits. The acceptable round-trip time in round i must satisfy $T_{\Delta,i} \geq \theta = \frac{2B}{C} + T(z_i - 1)$, where $C = 3 * 10^8$ is the speed of light (see equation 3.2). The verifier outputs $Out_v = 1$, if and only if all verifications and time checks succeed. Here for simplicity, we have assumed the communication channel is noise free, and so a successful protocol requires all challenges to be correctly responded.

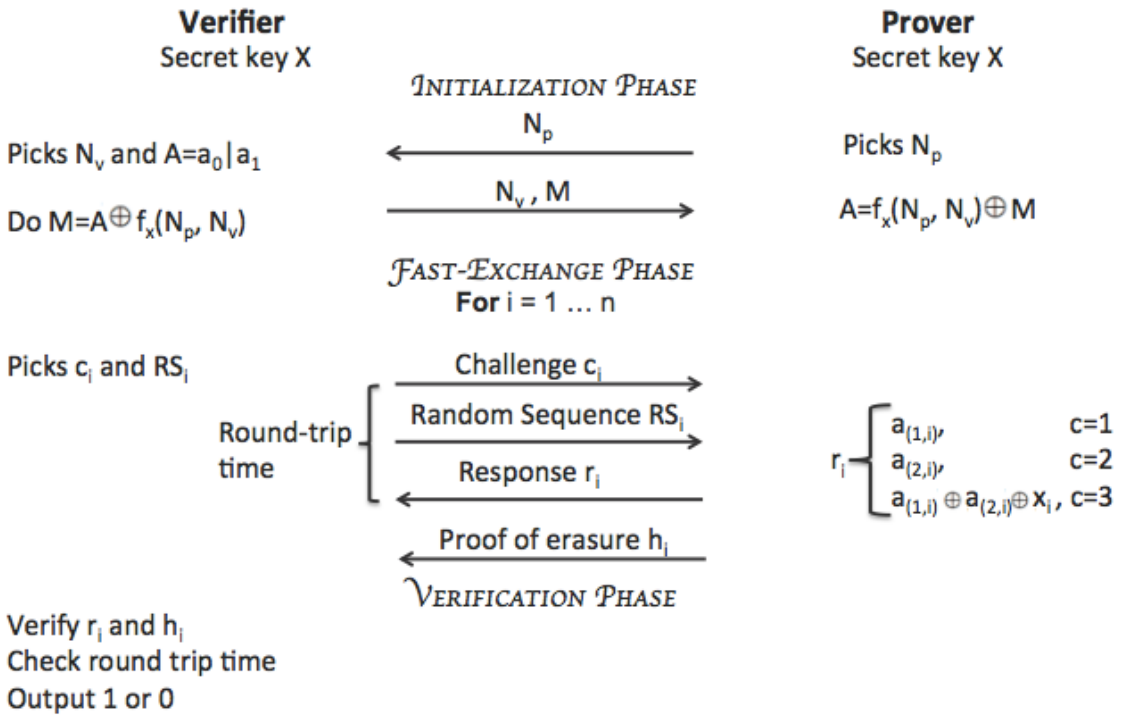

Figure 3.5: Distance lower bounding protocol Π_{DLB-BM}

3.5.2 The design of erasure sequence and its response

In fast-exchange round i, an erasure sequence RS_i is sent to the P, and a correct response is required. RS_i is used to guarantee all the device memory is erased, except DLB code and the part of the memory that is required for future response. The length of the erasing sequence RS_i must be chosen as follows.

Sequence length. If one bit of memory in addition to what is required for future the responses is left out, the dishonest prover can exploit that bit to store the response bit of the current challenge and delay it. In order to erase all but the part of the memory that is required for future response, the length of the erasing sequence RS_i must be chosen as follows.

Let the sizes of the RAM and DLB_{RAM}, be L and λ, respectively. After the initialization phase, the $2n$-bit random table A is stored in the prover's device memory. In Round 1, the erasing sequence RS_1 must erase $L-\lambda-2n$ unused memory, together with $(a_{(1,1)}, a_{(2,1)})$, the two response bits associated with round 1. In each subsequent challenge-response round i,

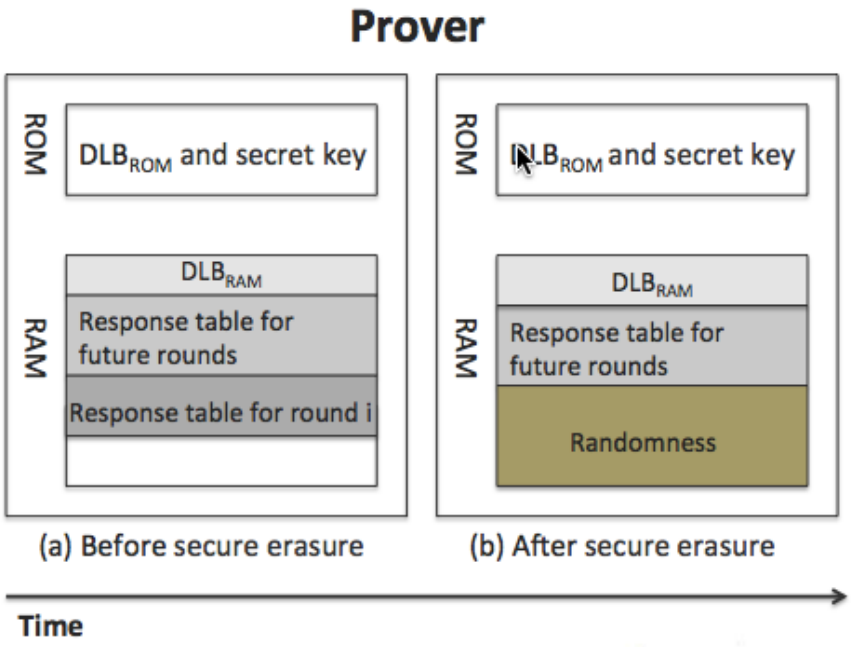

Figure 3.6: Prover's memory during protocol execution

two additional bits $(a_{(1,i)}, a_{(2,i)})$ of A will be used and so the length of the erasing sequence must be increased by two bits. By induction, the random sequence RS_i in round $1 \leq i \leq n$ must have length $L - \lambda - 2(n - i)$. Figure 3.6 shows the state of the prover's memory during protocol execution.

Response to the erasure sequence. The response in round i, denoted by h_i, must guarantee that the PD's memory, contains the sequence RS_i and DLB code DLB_{RAM} in full, and prove that the rest of the memory is erased. We refer to this response as *proof of erasure*, as it is inspired by [59]. To obtain assurance that RS_i is fully stored, it's sufficient to require the prover to return the exact RS_i, starting from the last received bit, to the first one. This however will be expensive from communication and power consumption view point. An efficient approach is to send the cryptographic hash of RS_i. To prevent the prover from calculating the hash value in real-time without storing the whole RS_i, we require the hash function to be applied to the received sequence in the reverse order of arrival. That is, assuming $RS_i = (u_1 \cdots u_{z_i})$, the hash will be applied to $\bar{RS}_i = (u_{z_i} \cdots u_1)$. This

leaves the prover no choice other than waiting for the last bit to arrive, before starting the calculation. To prevent P^* to simply store the required hash value of the code, we use $h_i = H_{\hat{x}}(\bar{RS}_i \parallel DLB_{RAM})$. In this construction, $\bar{RS}_i$ serves as a random nonce, and rules out the possibility of P^* successfully passing the verification without storing the full DLB_{RAM}.

The response calculation must be such that it cannot be delegated to a helper. This requirement is for achieving security against rCF (Section 3.6). We thus use a *keyed-hash message authentication code,* that requires the prover's secret key. The keyed-hash message authentication code uses a suitable cryptographic hash function in a specific structure (e.g. HMAC), to construct a secure MAC, which ensures the keyed-hash value cannot be forged.

3.6 Security analysis of Π_{DLB-BM}

Π_{DLB-BM} protocol uses a PRF and HMAC, and we analyse its security against a computationally bounded adversary. We note that it is possible to construct an information theoretically secure version of this protocol, by replacing the PRF and HMAC with appropriate primitives.

3.6.1 rDF$^{[BM]}$ resistance

In DF, a malicious prover P^* with $d(loc_v, loc_{P^*}) \leq B$, wants to prove, its distance is higher than the bound. To achieve this goal, P^* must send the correct response bit r_i, and the correct proof of erasure h_i, both with sufficient delay, in all rounds $1 \leq i \leq n$, of the fast-exchange phase. Theorem 2 proves that the DF resistance of the DLB protocol Π_{DLB}, assuming that a malicious code of length at least g bits is required.

Theorem 2 *Π_{DLB-BM} is ϵ-resistant to rDF$^{[\mathtt{BM}]}$, with,*

$$\epsilon = \max\left(2^{-(\frac{g}{2})^2}, 2^{-n(n+1)}\right),$$

against any rDF$^{[\mathtt{BM}]}$ attack that requires at least g-bit malicious code, assuming $H_x()$ is HMAC with a suitable cryptographic hash function.

Proof of Theorem 2.

A dishonest prover P^* succeeds if it passes verification of all rounds. We show that for *all* P^*'s possible strategies, its success probability is bounded. A P^*'s strategy σ, is defined by a sequence of actions that it will take over the n rounds. P^* needs a malicious code of size at least g to implement its strategy. The code must be stored in the PD's RAM. In each round, P^* must dedicate g bits of RAM for the malicious code MC, by either over-writing the response table $A^{[i]}$, or RS_i, or DLB_{RAM}, or part of each. Here $A^{[i]}$ is the un-used part of A at the start of round i. It is important to note that success probability of P^* in each round, depends on the action taken in the current round, and all actions taken in all the previous rounds. For example if P^* has overwritten $(a_{(1,i)}, a_{(2,i)})$, during an earlier round j, where $j < i$, then the success probability of producing the correct response to c_i, will be at most $1/2$.

Let $\Pr(Succ^{\sigma}_{DF})$ denote the prover's success probability for a strategy σ (n-round strategy, possibly adaptive) used by P^*. Let S_i denote the event associated with the success in round i, $1 \leq i \leq n$. We have the following:

$$Pr(Succ^{\sigma}_{DF}) = \Pr(\bigwedge_{i=1}^{n} S_i) = \prod_{i=1}^{n} \Pr(S_i | S_{i-1}, \ldots, S_1).$$

Because of the properties of probability, for all rounds i, we have

$$\Pr(S_i | S_{i-1}, \ldots, S_1) \leq 1$$

In round i, the P^*'s device receives a challenge symbol c_i, followed by $L - \lambda - 2(n-i)$ bits of RS_i. The response consists of r_i, and $h_i = H_{\hat{x}}(\bar{RS}_i || DLB_{RAM})$. Because of the unforgeability of HMAC, to calculate h_i, the string RS_i must be fully stored, and DLB_{RAM} must remain intact. If some of these bits, say ℓ, are overwritten, to generate the correct response, the ℓ missing bits can be guessed., with the success probability $2^{-\ell}$.

Let g be even and smaller than the original value of the response table, $g \leq 2n$. In each round, 2 bits of the this table is used and the erasing sequence will be lengthened by 2 bits

to overwrite them. The reduction in the size of the table in each round finally reaches a round $n_0 \stackrel{\triangle}{=} n - \frac{g}{2}$, after which the size of $A^{[i]}, i > n_0$, is less than the malicious code. That is, $2(n-i) < g$ and the length of RS_i satisfies $L - 2(n-i) > L - g$. From round $i > n_0$, to keep the g bit malicious code, some bits from RS_i must be overwritten and this number equals,

$$g - 2(n-i) = g - 2(n_0 + \frac{g}{2} - i) = 2(i - n_0).$$

This leads to a success chance of $2^{-(2(i-n_0)+1)}$ in calculating h_i in round i. The overall success chance is given by,

$$\begin{aligned} \Pr(Succ^{\sigma}_{DF}) &\leq \prod_{1 \leq i \leq n_0} 1 \times \prod_{n_0+1 \leq i \leq n} 2^{-(2(i-n_0))} \\ &= 2^{-\left(\sum_{i=n_0+1}^{n} 2(i-n_0)\right)} = 2^{-\left(\sum_{i'=1}^{n-n_0} 2i'\right)} = 2^{-(\frac{g}{2})(\frac{g}{2}+1)} < 2^{-(\frac{g}{2})^2}. \end{aligned}$$

If g is odd:. An argument similar to the above, shows that the prover needs to drop $2(i - n_0) - 1$ bits in rounds $i > n_0 =\stackrel{\triangle}{=} n - \frac{g+1}{2}$. The success probability is thus obtained as,

$$\begin{aligned} \Pr(Succ^{\sigma}_{DF}) &\leq 2^{-\left(\sum_{i=n_0+1}^{n} 2(i-n_0)-1\right)} = 2^{-\left(\sum_{i'=1}^{n-n_0} 2i'-1\right)} \\ &= 2^{-(\frac{g+1}{2})(\frac{g+1}{2}+1)}.2^{\frac{g+1}{2}} = 2^{-(\frac{g+1}{2})^2} < 2^{-(\frac{g}{2})^2}. \end{aligned}$$

If $g \geq 2n$. Here, the prover needs to drop some bits of the erasing string RS_i in all rounds $1 \leq i \leq n$; in other words, $n_0 = 0$ and the prover's success chance is,

$$\Pr(Succ^{\sigma}_{DF}) \leq \prod_{1 \leq i \leq n} 2^{-(2i)} = 2^{-\left(\sum_{i=1}^{n} 2i\right)} = 2^{-n(n+1)}.$$

This means that the success probability of P^* in *any strategy* is bounded, and the proof is complete.

3.6.2 rMiM$^{[\overline{\text{OC}}]}$ resistance

In rMiM$^{[\overline{\text{OC}}]}$, the adversary cannot send or receive signal to, or from the prover during the fast-exchange phase of the target instance. It however has full communication power during

other phases. We do allow the adversary to jam communications between the verifier and provers in all phases of the protocol (including fast exchange phase).

Theorem 3 *The DLB protocol Π_{DLB-BM} is β-resistant to rMiM$^{[\overline{\texttt{OC}}]}$ attack with $\beta = 2^{-l}$, by choosing $b > \frac{l}{n} - 1$.*

Proof outline of Theorem 3.

- If the target DLB instance does not include a prover, the MiM attack is the same as an impersonation attack. To succeed in this attack, the attacker needs to guess the secret key or guess all responses correctly. It can be seen (and formally proved) that the initialization phase, and the prover's responses in the fast-exchange phase, do not leak prover's secret key, resulting in the success probability equals to 2^{-2l}, assuming $n(b+1) > 2l$.

- Consider the case that the target instance includes a prover. The adversary can succeed if it can jam the messages from prover to the verifier, and send its own response. The adversary can only see the initialization phase, and not the prover's messages in the fast exchange phase of the target DLB instance.

  Thus again, the adversary needs to either guess the key, or guess all challenge and proof-of-erasure values, for all rounds. The probability of the attack will thus be no more than 2^{-2l}.

3.6.3 rCF$^{[\texttt{BM}, \overline{\texttt{OC}}]}$ resistance

Providing rCF security requires security against rDF and rMiM, and so their associated assumptions. We consider rCF$^{[\texttt{BM}, \overline{\texttt{OC}}]}$, and show (i) this is a stronger attack than rDF$^{[\texttt{BM}]}$ (see following for details), and (ii) Π_{DLB-BM} is secure against this attack (see Theorem 4 and its formal proof).

Argument for rCF is more powerful than rMiM:. We show rCF$^{[\mathtt{BM},\,\overline{\mathtt{OC}}]}$ is more powerful than rDF$^{[\mathtt{BM}]}$by giving a protocol that is rDF$^{[\mathtt{BM}]}$ resistant, but not rCF$^{[\mathtt{BM},\,\overline{\mathtt{OC}}]}$ resistant. Consider protocol Π^*_{DLB-BM} which is the same as Π_{DLB-BM} in Section 3.5.1, but uses the hash function without the secret key. By using argument similar to Section 3.6.1, we prove Π^*_{DLB-BM} is rDF$^{[\mathtt{BM}]}$ resistant. However now the response can be constructed by anyone. The colluders can plan their responses as follows: P^* will delay the response to the challenge symbol at will, and the helper prepares and sends the response to RS_i. This shows that Π^*_{DLB-BM} is not rCF$^{[\mathtt{BM},\,\overline{\mathtt{OC}}]}$ resistant.

Theorem 4 *The protocol Π_{DLB} is (γ,η)-resistant to rCF$^{[\mathtt{BM},\,\overline{\mathtt{OC}}]}$, with*

$$\gamma \leq \max\{2^{-\frac{g+0.5}{2}\frac{g+1}{2}}, 2^{-(n+0.25)(n+1)}\} \quad \textit{and} \quad \eta = 2^{-l},$$

assuming the malicious code is at least g bits.

Proof of Theorem 4. In the rCF$^{[\mathtt{BM},\,\overline{\mathtt{OC}}]}$, an L-bit-memory malicious prover P^* at distance $D < B$ wants to claim a distance $D \geq B$ by the help of an adversary A^{CF} who cannot communicate with the prover during the fast-exchange phase of DLB. The choice of $\eta = 2^{-l}$ in the theorem implies that the prover cannot leak any part of its key to the helper since any leakage would lead to a nOC-BM-MiM with probability $> 2^{-l}$. We thus study the best success probability ($\Pr(Succ_{CF})$) of nOC-BM-CF assuming no key leakage to the helper. We note that the prover still can leak part of its response table, which is build as a result of initialization: For each round of fast-exchange i, the prover's response bit to the challenge $c_i \in \{1,2,3\}$ is $r_i = a_{c_i,i}$ where $a_{1,i}$ and $a_{2,i}$ come from the PRF output and $a_{3,i} = a_{1,i} \oplus a_{2,i} \oplus x_i$, where x_i is a bit of the secret-key. So, as long as only two out of the three bits $(a_{1,i}, a_{2,i}, a_{3,i})$ are passed by the prover, there is no secret-key leakage and $\eta = 2^{-\ell}$ is satisfied. Hence, without loss of attack success optimality, we let the adversary have exactly two elements (e.g., the first two) from the triple $(a_{1,i}, a_{2,i}, a_{3,i})$ for each and

every round $1 \leq i \leq n$. In analogy to distance fraud, we let P^* store a malicious code of $g > 0$ in RAM and define $n_0 = n - \frac{g-1}{2}$.

Let $\Pr(Succ_{CF}^{\sigma})$ denote the prover's success probability for a strategy σ (n-round strategy, possibly adaptive) used by P^* and A^{CF}. Let S_i denote the event associated with the success in round i, $1 \leq i \leq n$. Similar to the DF-security analysis (proof of Theorem 2), we have

$$Pr(Succ_{CF}^{\sigma}) = \prod_{i=1}^{n} \Pr(S_i|S_{i-1}, \ldots, S_1), \quad \text{where} \quad \Pr(S_i|S_{i-1}, \ldots, S_1) \leq 1.$$

To simplify the proof like that for DF security, we assume without loss of optimality all r_i's are calculated successfully (with probability 1) and only focus on the calculation of h_i's. Since there is no communication between the prover and the adversary during fast exchange, the should have made prearrangements about which of them is in charge of sending which information to the verifier. Although some part of the response table is known by the adversary, this cannot help the adversary calculate hash responses h_i with any better probability (since the secret-key used for MAC is not leaked). This follows that the success chance is maximized of the prover calculates and transmits h_i's. The CF success probability thus will be bounded in the same way as that of DF in Theorem 2:

$$Pr(Succ_{CF}^{\sigma}) \leq \max\left(2^{-(\frac{g}{2})^2}, 2^{-n(n+1)}\right).$$

3.7 Practical consideration

3.7.1 Distance lower bounding

Different from distance upper bounding, distance lower bounding is suited for a scenario in which the prover wants to prove he is outside of a longer distance to the verifier. The typical range of such scenario varies from tens to hundreds of meters and the precision correspondingly varies from meters to tens of meters. For example, imagine that Google uses a distance lower bounding protocol to give its employees full Internet access when they are outside the

range of their working area. Assuming the main working area of Google campus is a building with a radius of 100 meters and the whole campus is a circle area has a radius of 500 meters (shown in Figure 3.7). The reasonable precision for such scenario would be 20 meters (20% tolerance). We now examine how well the assumptions can fit for this environment and the precision the protocol can achieve in practice.

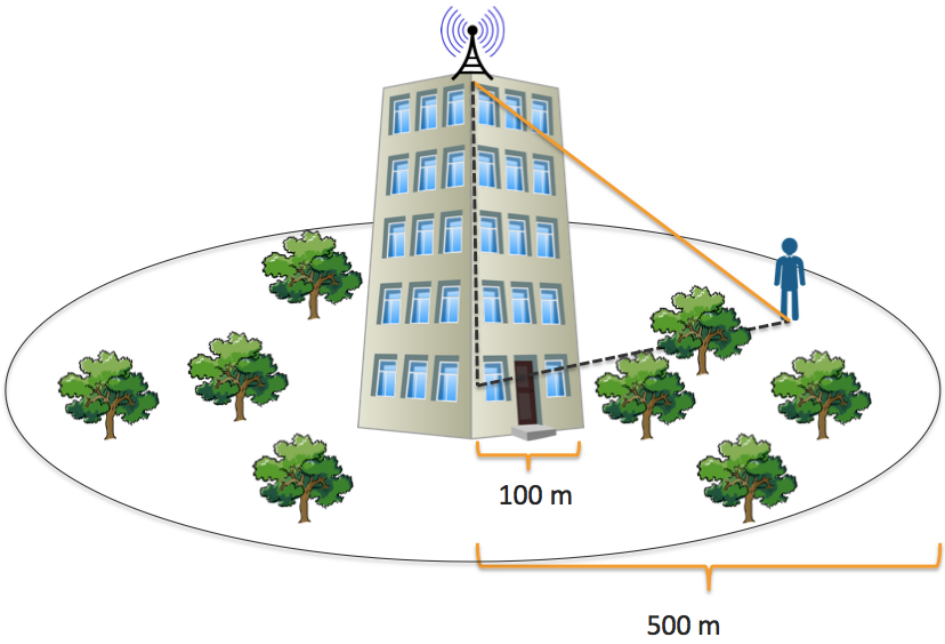

Figure 3.7: Application scenario of distance lower bounding

Line of sight propagation. In distance lower bounding scenario, line of sight propagation could be hard to guarantee for such long distances, even in an open space. There could be trees that block the line of sight signal propagation between the transmitter and receiver. One possible countermeasure would be to increase the height of transmitter and receiver. As shown in Figure 3.7, by increasing the height of the verifier's antenna, one achieve line of sight propagation. While the measured distance is different from the desired horizontal distance, we can easily calculate the horizontal distance using appropriate algorithm if the height difference between the transmitter and receiver is known.

Multipath effect. The effect of multi-path is mitigated here, because of two reasons: First, such a campus is closer to free space in terms of the density and the height of obstacles; secondly the spread of the time delays caused by multipath is generally larger than in indoor environment[52] and it is easier to identify the first arrived signal and somewhat reducing the effect of multipath.

Processing time. Although analog implementation can achieve better accuracy, digital implementation is chosen for two reasons: Firstly, based on the nature of distance lower bounding system, the transmission time is longer and so one does not strictly requires negligible processing time. The accuracy that digital implementation can achieve is acceptable; Secondly, the design of our protocol requires the conversion from analog to digital signal for further processing such as XOR function.

Hence one concludes that for the environment described in the example, it is possible to achieve the required security and accuracy.

3.7.2 Estimation of protocol performance

Security of DLB requires making physical assumptions other that constant speed of electromagnetic signal. Our DLB protocol relies on bounded memory assumption and requires the device memory size to be known. In addition, it assumes the attack is software based using a malicious code that resides in memory. That is all other registers and memories (e.g. ROM, write protected flash memory, etc) are inaccessible to the malicious code. These assumptions are not unrealistic and have been made in previous works [59]. In the following, we discuss the parameter choices of the DLB protocol and analyze its performance with respect to time, energy, and memory consumption on a MicaZ sensor [43] running TinyOS. We require 10^{-6}-resistance to distance and restricted-collusion frauds assuming the length of the malicious code is at least 1 byte.

The computations during the initialization and challenge-response distance estimation are

similar to DUB protocols and so the excellent works on the implementation of DUB protocols [64] can be used to estimate the performance of DLB. Using erasing sequence is however unique to Π_{DLB-BM}. We assume the following parameters: $n = 10$ (number of rounds) and $k = 192$ bit (nonce size). We will use an $HMAC(.,.)$ using SHA1, denoted by HMAC-SHA1, for generating proof-of-erasure in response to RS_i. We have $h_i = HMAC(\hat{x}; \bar{RS}_i||DLB_{RAM})$ of length $b = 160$ bits. HMAC has been proved to be a PRF[5] and so we will also use HMAC (with a different key) as f_x for the generation of the response table. That is, $f_x(N_p, N_v)$ will be the first $2n = 20$ bits of $HMAC(x; N_p||N_v)$.

MicaZ sensor specifications. The device is supplied with two AA batteries and includes an ATMEGA128 microcontroller and a TI-CC2420 radio transceiver. The micro-controller provides 4KB of writable memory (SRAM), with 4KB of EEPROM and 640 KB of write-protected flash memory. The radio transceiver chip works for an RF band of 2.4–2.48 GHz and has 250 Kbps data rate.

Memory consumption.. HMAC-SHA1 takes code size of 4650 bytes [59] for implementation on ROM, and needs around 124 bytes of RAM to load data structures and stack. We assume $l = 128$ bits for the secret key x, and the code size of DLB_{ROM} to be 10KB. Although the EEPROM is only 4KB large, the ATMEGA128 architecture allows for ROM extension via the use of mask ROM, locked flash memory, and fuse bits. Using these extension methods, one can build read-only memory of size 10KB or more. Note that in order to estimate maximum energy consumption, we assume the size of DLB_{RAM} is 0 bits. This will make the erasing sequence the longest, requiring the highest energy during erase phase.

Energy and time consumption. The writable memory in ATMEGA128 (when flash memory is write-protected) is the 4KB SRAM. We used $n = 10$. The erasing sequence has length 32758 bits on average.

Communication costs. The protocol requires the prover to receive N_v, M in initialization

phase and c_i, RS_i in each fast-exchange round, which gives a total of $l_{rx} = len(N_v) + len(M) + 2n + n \times len(RS_i) = 326912$ bits. There is also requirement for sending N_p, r_i's, and h_i's which sums to $l_{tx} = len(N_p) + n + n \times len(h_i) = 1802$ transmission bits. Sending (resp. receiving) a single bit requires $E_{rx} = 2.34\mu J/b$ (resp. $E_{tx} = 4.6\mu J/b$) by the radio transceiver with typical power adjustments [13]. The total communication energy is thus $E_{comm} = l_{tx}E_{tx} + l_{rx}E_{rx} = 773mJ$.

Computation costs. The main part is computing h_i. The remaining computation is negligible. Extrapolating the figures for memory erasure phase in [59] to our 4KB-memory device, we require less than 600 milliseconds time for computing proof of erasure, which is quite practical noting that this is not a time critical operation and will not affect distance estimation. Each HMAC-SHA1 computation uses $3.5\mu J$ per memory byte. Considering the memory size and the number of rounds, the required energy for computation is around $E_{comp} = 3.5 \times 10^{-6} \times 4 \times 2^{10} \times 10 = 140mJ$.

Each AA battery can deliver 1.2 Amperes under an average voltage of 1.2 Volts for one hour [13], implying the power supply of $10,368J$ via the two batteries. This means the proving device can be used for approximately $\frac{10,368}{(773+140)10^{-3}} > 11,000$ runs of DLB protocol before the batteries die. This is quite a reasonable outcome for power consumption. Although for more accurate analysis one must consider idle/sleep mode energy consumption, such considerations will not drastically affect the above results.

3.8 Concluding remarks

We motivated the novel security problem of DLB in the setting that the prover is not trusted, using a number of application scenarios and gave formal security definitions against three general classes of attacks (DF, MiM and CF). We proved that it is impossible to provide security against any of these attacks without making physical assumptions. We showed that an adversary, even if it is computationally bounded, will *always succeed* in DF if it has

unrestricted access to the prover's device (fully untrusted prover), and will always succeed in MiM attacks, if it can control the communication channel between the prover and the verifier. None of these statements are true for DUB protocols: DUB protocols can provide security against malicious prover (with or without the helper) and an MiM attacker who controls the channel between the prover and the verifier. Security of DLB against CF requires restrictions on the adversary's access to the prover device and also the channel between prover and the verifier. These show a fundamental difference between DLB and DUB problems. The only physical assumption in secure DUB protocols is that the speed of the EM signals is constant. In DLB protocols however, in addition to this assumption, one must assume other restrictions on the physical access of the adversary.

Our protocol considers a malicious prover that has restricted access to the prover device, and an external attacker that has restricted access to the communication channel between the prover and the verifier, and provides security against $\text{rDF}^{[\texttt{BM}]}$, $\text{rMiM}^{[\overline{\texttt{OC}}]}$, and $\text{rCF}^{[\texttt{BM},\,\overline{\texttt{OC}}]}$, using reasonable assumptions. Enforcing assumptions in practice however would need special technologies such as targeted jamming [32].

One can replace the above assumptions with other reasonable assumptions. The assumptions are necessary to prevent the adversary from introducing delay in the challenge-response time estimation phase. For example, instead of assuming bounded memory, one can use a software-based *externally verifiable code execution (EVCE)* system such as Pioneer [69], to guarantee that the *target executable* code associated with the distance measurement, is executed without modification by a malicious code that may reside on the device. Using this approach, in each challenge-response round, the device first proves that it has an untampered environment, and then provides the response. Note that EVCE also assumes a trusted network to eliminate proxy attacks. This restriction is inline with the required restrictions in our restricted attacks $\text{rMiM}^{[\overline{\texttt{OC}}]}$ and $\text{rCF}^{[\texttt{SA},\,\overline{\texttt{OC}}]}$. More details on an EVCE based DLB protocol is given in Section 3.4. The important point to note is that *one must restrict the adversary's*

physical access to the environment to achieve any level *of DLB security.*

Our primary application scenarios for DLB in this chapter were examples of proximity-based access control and our considered attacks captured common attacking scenarios in such applications. Other application scenarios in DLB may have different security requirements. Examining these requirements may uncover new requirements and other attacks.

Finally an important open question is to efficiently incorporate DLB in DUB protocols to provide security against distance enlargement attacks.

Chapter 4

Distance Upper Bounding using Timestamps

4.1 Introduction

Distance (upper) bounding (DUB) protocol has been widely researched in recent years since it provides a possible way to defeat relay attack and has wide applications in the area of location-based service (LBS). The traditional design of DUB protocols uses time measurement (time-of-flight of messages) to obtain distance. By assuming the electromagnetic signal travel at speed of light, the verifier can calculate the upper bound of the distance between itself to the prover by measuring the round-trip time (RTT) of a message back and forth between them. The security of DB protocols is normally evaluated by their resistance to three generalized attacks: (1) *distance fraud (DF)* where a malicious prover acts alone to convince the verifier that he is at a shorter distance[1]; (2) *Man-in-the-Middle fraud (MiM)* where an external attacker tries to shorten the distance between the honest prover and the verifier[2]; and (3) *collusion Fraud (CF)* where a malicious prover colludes with an external attacker to shorten its distance to the verifier[3]. Note that the adversary's purpose is only to shorten the distance in all these attacks.

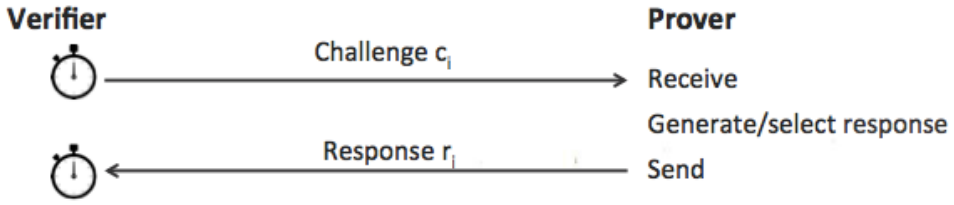

Figure 4.1: Using round trip time (RTT) to measure the distance

However, using RTT to measure distance relies on the assumption that the processing

[1]Distance hijacking attack [18] is included in this generalized attack
[2]Mafia fraud is included in this generalized attack
[3]Terrorist fraud is included in this generalized attack

time[4] is negligible compared to the propagation time in order to obtain accurate distance measurement. This is because RTT includes both propagation time and processing time and it is hard to isolate the processing time from the propagation time. If the verifier overestimates the prover's processing time, the prover is able to shorten the measured distance and pretend to be closer to the verifier. If the verifier underestimates the processing time, the distance bound will be too large and the security guarantee is loosened. This strict requirement poses difficulties on the implementation of DUB protocols since even 1 ns processing delay will enable the malicious prover to cheat up to 15 cm in distance, which does not match the current state-of-the-art hardware. This also limits the possible development of DUB applications since it is not possible to have complicated computations at the prover's side. For example, we identify the false rejection attack that can disrupt the correct execution of DUB protocol and propose using frequency hopping to defeat this attack. This solution, however, is not applicable to the distance bounding schemes that use round-trip time for distance measurement as frequency hopping consumes too much time and makes the result inaccurate. Few implementations of distance bounding protocols [46, 37, 77, 64, 63] have been proposed, but all of them are based on the analog processing or a hybrid of digital and analog processing. To our best knowledge, the most efficient digital implementation of DUB protocol in the open literature [76] requires 170 ns of processing time and thus enables the attacker to cheat on distance up to 25.5 m. In this work, we are going to address the problem: *is it possible to have DUB protocols without using round-trip time?* We will discuss this question and give the design of such distance upper bounding protocol.

Intuitively, in order to eliminate the use of RTT, one can obtain the one-way transmission time ($OwTT$) by asking the sender to include its sending time t_s with the message. And then the receiver can obtain the receiving time t_r upon receiving the message and calculate the one-way transmission time $t_t = t_r - t_s$. However, this simple idea requires: (1) the sender and receiver's time to be synchronized; and (2) the sender is honest and would

[4]The processing time includes the time to receive the challenge, process and send the response.

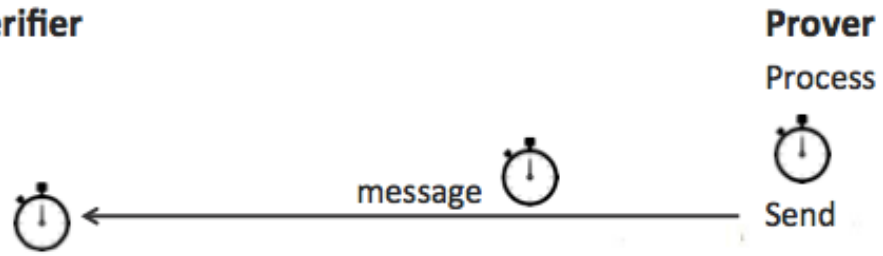

Figure 4.2: Using one-way transmission time ($OwTT$) to measure the distance

not lie about its sending time. Apparently, achieving the latter requirement is not easy, as the prover can be malicious, or even collude with external adversaries in the adversarial setting of DUB protocols. Therefore, we need to find a way to prevent the malicious sender from lying about its sending time, and so the receiver is confident that the real $OwTT$ is obtained. Note that, in DUB protocol, the malicious prover is only interested in shortening its distance to the verifier (through shortening the transmission time). Hence, the malicious prover that aims to extend its distance is not considered in this paper. But one can find how to prevent such distance enlargement attack or obtain the lower bound of distance in [90].

Real Time v.s Timestamp. Time is the measure in real world, in which events are ordered from the past through the present into the future and so the duration of event and intervals between events can be measured accordingly [86]. As one of the fundamental physical quantities, measuring the time (running a clock) always requires measuring some physical parameters, such as the passage of a free-swinging pendulum, or in terms of radiation emitted by caesium atoms. Thus it is hard to model "real time" in a Turing Machine based model [68]. Instead, we want to find an appropriate approximation of real time, which can be modelled in Turing Machine environment, and preserves the important security guarantees[5] offered by the real time. Considering the two requirements we stated above, we require this approximation of real time to be (1) synchronized for all participants in a small area; and

[5]uniqueness, timeliness and etc [50].

(2) unpredictable in terms of future time. Intuitively, synchronization relies on the use of a common time reference and unpredictability involves the use of randomness. We will introduce such implementation of time in the following. In order to distinguish the time values produced by such implementation from real time, we define such time values as *timestamps*.

Modelling Time. Modelling time is not necessary in conventional DUB protocol, since the time is only measured by the trusted verifier. However, in this work, we proposed to measure the one-way transmission time $OwTT$ to obtain the upper bound of the distance. As the untrusted prover is involved in the time measurement, we need to establish the formal model for measuring $OwTT$, in order to conduct formal analysis.

Although time plays an important role in various cryptographic protocols (for example to guarantee freshness in authentication protocols or to limit lifetime for public keys), few works have discussed the problem of modelling time formally. Barbosa and Farshim [4] have extended the existing authentication framework to capture the use of timestamps. Schewenk [68] has started the first discussion of modelling time in Turing Machine environment for the purpose of guaranteeing the freshness in cryptographic protocols. This requires that the adversary not use a timestamp that is in the past in order to get accepted by the protocol. In our case, in addition to guarantee the freshness, we use time values for a different purpose: to find the upper bound of the one-way transmission time of a message. Therefore, it requires timestamps to be unpredictable in the sense that the adversary could not forge a valid future timestamp before seeing it. This fundamental difference reveals that we could not simply use the time model in [68] for our work.

Global Counter. In this paper, we implemented time by proposing a trusted common time reference that broadcasts unpredictable timestamps at a high frequency, which we called Global Counter (GC). With the help of GC, all parties can get the synchronized

timestamps and one can verify the validity of timestamps. The validity here implies that a valid timestamp can only be produced on or after the time it represents. With this model, the receiver is able to calculate the one-way transmission time of a message that is sent by any sender. We then proposed a novel distance bounding protocol that uses this model in the same adversarial setting as previous protocols, and obtains the upper bound of the distance from one-way transmission time.

The idea of GC is inspired by the prototype source which is implemented by NIST for public randomness [58]. In this prototype, a randomness beacon broadcasts full-entropy bit-string in block of 512 bits for every 60 seconds. The beacon is designed to be unpredictable, autonomous and consistent, so that:

- Users cannot predict the bits before they are made available by the source.
- The source is resistant to outside attempts that try to alter the distribution of the random bits.
- A set of users can access to the source in a way that they are receiving the same random bits.

Since the randomness beacon broadcasts unpredictable random bits at fixed frequency (every 60s), it gives an alternative way to measure the elapsed time instead of using a clock. In this work, we try to establish a relationship between time and randomness to achieve unpredictability, and ask every user to prove the time by proving that it has the knowledge of the corresponding randomness. That is, for anyone who wants to prove its time, he/she would need to include the corresponding randomness. Since it is hard to predict randomness (assuming the source uses secure pesudo-random generator), the adversary would not be able to prove its knowledge of an unseen randomness and as a result the adversary can not successfully prove it is at a future time. We refer to this timestamp as the unpredictable timestamp.

Using unpredictable timestamps. In this work, we propose to implement unpredictable timestamps by including randomness to associate time values with randomness. We introduce the trusted party - Global Counter (GC) to allow distributing unpredictable timestamps. And we introduce the formal model of using GC to measure the one-way transmission time of a message. We then extend to time model to construct the formal model of the DUB protocol and construct DUB protocol based on GC. Instead of measuring the round-trip time of the message, only one-way transmission time is measured and used to calculate the distance between the verifier and prover. In conventional DUB protocols, it is not possible to measure the one-way transmission time because the dishonest prover can arbitrarily lie about its sending time to shorten the distance without being detected. In our design, this is not true as it is impossible to predict the future timestamp with high success probability. Hence, the malicious prover can only extend the distance rather than shortening the distance.

The intention is for users to declare their time when sending a message by appending the unpredictable timestamp obtained from the global counter. When a message is received, the receiver obtains another timestamp. The timestamp that is included in the sending message is sent to the global counter to verify its validity. If the verification is successful, the time value included in this timestamp is compared with the one that the receiver obtained to derive an estimation of the elapsed time, which is one-way transmission time (OwTT). Since the timestamp is unpredictable, and if we assume the message travels at the speed of light C, the OwTT we obtained can be used to calculate the upper bound of distance between the sender and verifier by $D_{upper} = OwTT \times C$.

The biggest advantage of this approach is that the processing time is now isolated from the transmission time. In traditional DUB protocols, the time measured by the verifier is the round-trip time, which includes the transmission time and processing time. Hence, no complicated calculation is allowed at the prover's side. Even to apply simple XOR function, it

is not possible to isolate the processing time from the transmission time. With our approach, time is only considered at the time of sending a message. Hence we can have complicated processes (e.g, frequency hopping) before that without affecting the accuracy of the result. The other advantage is that we can have multiple checkpoints at the prover's side. Having these checkpoints actually gives us many possibilities. For example, passive observers can now obtain the distance between itself and the prover and so multiple passive observers can corporate to locate the prover's location even if one observer is compromised; or we can incorporate external verification with DUB protocols.

Traditional DUB protocols also have difficulties in implementation because of the strict requirement of processing time. Since the digital processing requires more steps such as signal detection, ADC/DAC conversion, and signal modulation and demodulation, this increases the processing delay. The existing implementation [64, 63] uses analog or hybrid digital-analog processing to provide tighter security guarantees. By using this approach, one might be able to provide tighter security guarantees for digital processing.

Contribution. The contributions of this work are as follows:

- We motivate and initiate the study of a new approach to construct DUB protocol using one-way transmission time.
- We propose a new time model that not only guarantees the freshness, but also allows one to calculate elapsed time, which can be used in Turing Machine-based implementations.
- We formally define the security of distance bounding protocol based on the time model we proposed.
- We accordingly propose a DUB protocol that relies on one-way transmission time(OwTT), which avoids the drawbacks of traditional DUB protocols.

- We formally prove the security of our DUB protocol and provide a discussion about the implementation of such protocol.

4.2 Model time using timestamps

In previous DUB protocol, the round trip time RTT is measured by the verifier only. In a round of time-critical phase, the verifier records its first local clock reading t_1 at the time of sending the challenge, and its second local clock reading t_2 at the time of receiving the response. The time interval between the verifier sending and receiving message, which is round-trip time, is then calculated as $RTT = t_2 - t_1$. Hence, modelling time is not necessary, as time is only be measured by the trusted verifier.

However, in this work, we propose to use the one-way transmission time to obtain the upper bound of distance. Hence two parties (the sender and the receiver) are both involved in the time measurement. Assuming there exists a global clock, the prover reads the global clock reading t_1, appends it to the message and then sends it to the verifier. Once upon the verifier receives the message, it stores the global clock reading t_2. Ideally, the time interval between the prover sending the message and the verifier receiving the message, which is one-way transmission time ($OwTT$), is calculated by $OwTT = t_2 - t_1$. Since in DUB protocol model (in Section 4.3), the prover could be malicious and so with an untrusted party involving in the time measurement, we need to establish the formal model for securely measuring one-way transmission time. Note that in the DUB protocol, it is assumed that the adversary only aims to shorten the distance to gain advantage. Hence, the adversary would prefer to forge a t_1 that is corresponding to the time after he sent the message instead of before. Hence, our main purpose is to design a way to prevent adversary from attaching a future timestamp to the message.

Although time plays an important role in cryptography, modelling time in Turing Machine environment is still under research. Time values are used for many different purposes in

cryptography: To limit the lifetime of secret keys or to guarantee the freshness of messages. In [4], a generic modelling technique that can capture the use of timestamps was introduced. Authors applied this technique to two different popular models (one is Bellare-Rogaway approach and the other one is Canetti-Krawczyk approach) to analyze the authentication and key agreement protocols utilizing timestamps. In this work, time is modelled by providing parties with local internal clock, which is incremented by sending TICK requests. However, when coming to Turing Machine model, there's no mathematically sound definition of using local clock as part of Turing Machine [68] .

In order to better accommodate Turing Machine environment, [68] modelled time as a global counter to guarantee the freshness in cryptographic protocols. This approach uses Turing Machines as computational model and stay as close as possible to the well-established definition of secure authentication protocols. In this work, time is defined as a global counter that provides a time reference to all parties in the system. If a fresh message has to be sent, the sending Turing Machine requests a timestamp $t_s = (t, aux)$ from the global counter. Upon reception of such request, the global counter increase its local counter ($t \leftarrow t + 1$) and returns the timestamp. When a message is received, the receiving Turing Machine compares the time value in the timestamp to its local counter to determine the freshness of the message. The local counter is updated once the receiving Turing Machine confirms that the message it received is the fresh one.

However, we propose to use time values to calculate the one-way transmission time of a message and therefore measure the distance between two parties. The transmission time of a message plays an important role in cryptographic protocols also, such as distance bounding and software attestation. Note that, time values are used for different purposes in [68] and our work. In the former one, time values are only used to guarantee the freshness. That is the adversary could not use a timestamp that is in the past to get accepted by the protocol. However, in our work, in addition to guarantee the freshness, time values are also used to find

the one-way transmission time, which requires the timestamp to be unpredictable and the adversary could not forge a future timestamp to get accepted. This fundamental difference determines the difference in the design of global counter. In this section, we propose our time model that can be used to measure the elapsed time. In this time model, the time unit is determined by the broadcasting frequency of the global counter and time interval can be obtained by counting how many time units have elapsed. We also formalize a normal protocol execution to measure the one-way transmission time of message and the adversarial capabilities.

4.2.1 Execution Environment

Parties and process oracles. In this execution environment, we have n parties $\{P_1, ..., P_n\}$ and each party P_i runs process oracles π_i^s, $s \in \{1, ..., l\}$. These process oracles and parties are all modelled as Turing Machines. Oracles can be initialized, send and receive messages according to the protocol specification. The parties and process oracles have states$\in$ $\{initialized, finished, accept, reject\}$. In addition, we have two special parties: global counter T and adversary $\mathcal{A}$.

Each party P_i may have a long term key k_i and a local counter T_i. This long term key can be either a public key pair, or a list of $n-1$ symmetric keys shared with other parties. Long term key of party P_i is accessible for cryptographic operations. And the local timer T_i can be accessible by all process oracles of this party.

The local variables (e.g. computed nonces, state values, session keys) are only known to the oracle itself and the party it belongs to. The transcript of all messages sent and received by an oracle π_i^s is stored in local variable $Trans_i^s$.

There is one special party T, acting as global counter. This party broadcasts unpredictable timestamps $t_s = (t, r)$ wirelessly, where t is the time and r is the random string. It is assume that the broadcasted messages arrive to all parties with negligible delay. This assumption is justified by limiting users in an area of relatively small radius compared to

the height of the global counter (see Section 4.5.1 for details).

Definition 6 (Global Counter) *The global counter is a trusted party T that maintains a global count whose value is incremented and broadcasted at the speed of f per seconds. Each broadcasted value is of the form, (r, t) where r is a b-bit random string, and t is the count. We assume broadcasts reach all participants simultaneously* [6]*. Global counter also maintains a query server for validating the timestamp, which accepts the query of timestamps and returns true if it is a valid timestamp, false otherwise.*

The adversary $\mathcal{A}$ is another special party that controls the communication environment: it can corrupt parties and get them to follow its strategy. The adversary tries to break the cryptographic protocol using different strategies. The adversary's capabilities are defined in the Section 4.2.3.

4.2.2 One-way transmission time (OwTT) measuring

Our security definition focuses on a one message interaction defined as follows:

Definition 7 (OwTT measuring procedure) *The OwTT measuring procedure is a one message interaction between two parties. The party who starts this procedure by sending the message is called initiator, and the party who receives the message and measures the one-way transmission time is called responder. After receiving the message, responder calculates the measured OwTT, which is the output of OwTT measuring procedure and is known to responder only. Note that the initiator might be malicious here, but the responder is trusted. Otherwise, the result of measured time is not trustable in any case.*

Retrieve timestamp when sending a message. When the initiator tries to send a message to P_j, its oracle π_i^s picks the latest timestamp $t_{s1} = (r_1, t_1)$ from global counter T. The timestamp will be appended to the message that initiator is trying to send. Then the whole

[6] *this is meaningful is areas of a few kilometres.*

message is sent in an authenticated way.

Calculate one-way transmission time when receiving a message. When the responder receives the message m, its oracle π_j^s obtains the latest timestamp $t_{s2} = (r_2, t_2)$ from global counter T as well. The responder then verifies the message authentication code of m and obtains the timestamp included in the message, which is $t_{s1} = (r_1, t_1)$. The responder sends both t_{s1} and t_{s2} to the query server of global counter to verify their validity. If all verification passes, the responder calculates the elapsed time by the following theorem and updates its local counter.

Theorem 5 *Assuming the global counter broadcasts timestamp at fixed frequency f. The honest initiator obtains the timestamp t_{s1}, whose count is t_1, and sends it to the responder using electromagnetic wave. The responder receives the message, and obtains latest timestamp t_{s2} at the same time, whose count is t_2. After passing the verification, the one-way transmission time (OwTT) is*

$$OwTT \leq (t_2 - t_1)/f$$

Local states of oracles. After being initialized, an initiator oracle π_i^s prepares and sends a message of the form (P_i, P_j, t, r, σ), where P_j denotes the identity of destination party, t and r is obtained by the lastest timestamp broadcasted by global counter T, and σ is the message authentication code. Since the OwTT measuring process is a one-message interaction, π_i^s will immediately switch to finished state, and can no longer be activated. A responder oracle π_j^s will be activated by a protocol message, which has the destination identity of P_j. A series of verifications will be taken as protocol specification and if any of these verifications fails, π_j^s will switch to reject state. If all verifications are passed, π_j^s will switch to accept state with an output value O_t.

Non-authenticated Timestamp. In practice, timestamps may be sent in an authenticated way or be sent unauthenticated. For our model, we choose to have unauthenticated timestamps. The reason is that the only attack that signed timestamps would prevent is Denial-of-Service (DoS) attack, where the adversary would intercept the broadcast and replace the timestamp with random information (invalid timestamp). All parties using the invalid timestamp for OwTT measuring procedure will be rejected. However, our model considers a powerful adversary with complete control of the network. Therefore he can always perform DoS attack by blocking transmissions of some parties. Hence, the power of our adversary is not increased by using unauthenticated timestamps.

Delay-based attack. In practice, the powerful adversary may also delay global counter's broadcasts to make a specific party receiving timestamps slightly later than other parties. When the victim party is the responder, the result is that the OwTT is shortened in this way. Note that, the adversary's goal is to shorten OwTT in this model. This attack can only be mounted by either replaying the historical timestamps or increasing the interval between timestamps. However, our model aims to measure OwTT that is small and therefore the adversary should not bring too much delay, otherwise it will be detected (e.g, the receiving time count is smaller than the sending one). Relay historical timestamps is detectable if each party remains a memory that stores certain amount of timestamps that have been broadcasted in the past. And the responder can also detect if the interval between timestamps is being increased by having a local clock. For this reason, we conclude that any delay of the timestamp broadcast will be detected and it's reasonable to assume that timestamps will reach to each party without delay.

4.2.3 Adversarial Capabilities

The adversary's goal is to shorten OwTT. We assume that the adversary completely controls the network, including the access to the global counter T. He may corrupt one or more parties

and orchestrate an attack strategy that will be followed by the corrupted parties. We assume the global counter T is not corruptible, and its broadcast will always reach all users correctly. We also assume the responder is not corruptible. The adversary can learn session keys or long-term keys from the colluding parties.

In order to clearly state the adversary's capabilities, we model the adversary's behaviour through queries. The *Send* query models that the adversary completely controls the network: All messages are sent to $\mathcal{A}$, who may then decides to drop the message, to store and replay it or to alter and forward it. Thus messages received through a *Send* query are handled by the process oracles exactly like real protocol messages: They may be rejected, answered or they may start or terminate a protocol session. The *Send* messages also enable the adversary to ask any coracle of colluding parties to initiate and run an arbitrary number of protocol instances with his own choice of message, sequential or in parallel. It is assumed that handling messages through *Send* query takes negligible time in order to model a more powerful adversary.

The *Corrupt* query enables A to corrupt with specific parties and A would be able to see all transcripts and local variables of that party. If the corrupted entity has a secret key, it becomes available to the adversary. Note that $\mathcal{A}$ is not allowed to corrupt with global counter or the responder in a time measuring procedure. These queries are formally stated as follows:

- Corrupt(P_i^s): This query is used by the adversary to corrupt a party. The party colluding with adversaries shares the transcript and local variables with A, as well as the secret key. Then this party becomes corrupted party. However, the adversary would not share secret keys among corrupted parties.

- Send(π_i^s, m): The adversary can use this query to make the corrupted party send any message m of its own choice to any oracle π_i^s. The receiving oracle will respond according to the protocol specification.

4.2.4 Properties of Global Counter

Global Counter acts as a global timer in this model. Because each timestamp contains random string and no party can predict a future timestamp without seeing it, timestamps become unpredictable in the model. The properties of global counter is summarized as follows:

- **Unpredictability.** The random string generated by global counter is unpredictable in the sense that users cannot predict random string before it is made to be available to the public. In the other word, the adversary should have small probability in successfully predicting the timestamp that are unseen (corresponding to future time).
- **Accessibility.** The global counter broadcasts authenticated timestamps that are available to users whenever they need, with the guarantee that they all receive the same timestamp. And broadcasts reach all participants simultaneously[7].
- **Autonomy.** The global counter is autonomic so that no adversaries can alter the distribution of the random strings.

According to the above properties, we can have the following theorem.

Theorem 6 *Assuming the global counter uses pseudo-random generator and broadcasts timestamps at frequency f with random string length of l, the adversary has probability ρ at time t to guess any timestamps that is available after time t correctly, where $\rho = 2^{-l}$.*

Proof: Because the global counter uses pseudo-random generator to generate random strings, the probability of guessing an unseen bit is 1/2. Because each random string has length of l bits, the malicious user can only guess unseen random string correctly with probability 2^{-l}.

[7]This assumption is justified in Section 4.5.1

4.3 Distance Bounding Model based on Time Model

The problem we investigated in this work is defined the same as distance upper bounding (DUB) problem. We model DUB protocol based on the above time model. We define the security of DUB protocol by defining its resistance to three general classes of attacks: distance fraud (DF), Man-in-the-middle (MiM) attack and collusion Fraud (CF). We use security games to define each attack and by intuition, our model is equivalent to the well-established distance bounding model [8].

We consider a multi-party system where the party U with a certain location loc_U, is modelled by a polynomially bounded interactive Turing machine (ITM). Some parties have pre-shared secret key and we called them *users*. Some others who do not have secret key are *outsiders*. All parties can access to the global counter. Both *users* and *outsiders* have their own location and location is important to a party in the sense that it determines the timestamp he can obtain in an event. Messages that are sent from one location to another, travel at the speed of light and the time taken for travel can be used to determine the distance between the two locations. *Along the path of the message transmission, parties are able to see different timestamps according to their locations.* The one who is closer to the receiver would see later timestamps. We assume parties will receive privileges based on their pre-shared secrets and their location.

Users in the system could be *prover*, or *verifier*. The adversary $\mathcal{A}$ is a special party that controls the communication environment: it can corrupt users and get them to follow its strategy. The adversary has the same capabilities as we defined in the previous time model. However, since there are two kind of parties involved in this model, the *Corrupt* query is divided into *CorruptU* and *CorruptO* to denote the query for corrupting with users and outsiders respectively. A *prover* P engages in a two-party protocol with a *verifier* V, to prove the claim that its distance to the verifier satisfies certain bound. The prover always acts as the initiator and the verifier always acts as the responder. Honest parties run predefined

algorithms for their side of the interaction. The adversary can only corrupt the prover, and the corrupted prover is denoted as P^*.

A corrupted prover deviates from the protocol to make incorrect distance claim and accesses privileges they are not entitled to. The experiment is expanded to include an external adversary $\mathcal{A}$ who interacts with the parties in the system according to its defined abilities as stated below.

A protocol instance with Global Counter (GC) is a one message protocol, which involved an experiment denoted by $exp = (P^{[GC]}(x) \leftrightarrow V^{[GC]}(y))$, where GC is the Global Counter and x and y are secret keys of the prover and the verifier, respectively. The protocol includes one round of OwTT measure procedures defined above [8]. At the end of a protocol instance, V has an output Out_V, which is 1 or 0, showing acceptance or rejection, respectively. The prover does not have an output. Each protocol instance has access to Global Counter. For an initiator oracle π^s, the timestamp obtained from Global Counter will be used as part of the message be sent. The time required to generate the message is assumed negligible. For a responder oracle, once it is initiated by a message, it reads from the Global Counter, and then continues for the reception and verification process.

A participant in an experiment has a *view* consisting of all its inputs, coins, and messages that it can see. The external attacker $\mathcal{A}$ may interact with multiple Ps and Vs, and its view will include all these interactions. Throughout the paper Pr[*event* : *experiment*] denotes the probability of the *event* for the *experiment*.

In order to define the security of our one-message distance bounding protocol, we first need to define a benign adversary. A benign adversary $\mathcal{A}$ is an adversary that forwards messages *instantaneously* without modifying them. We would like to define correctness of a distance bounding protocol as the probability of a protocol instance outputting 1 for honest prover and verifier whose distance is less than the bound is high, when benign adversary is

[8] For clarification purpose, we only discuss one-round protocol here. But the result can be easily extended for multiple rounds

involved in the protocol instance.

Definition 8 (Distance Bounding Protocol) *A Distance Bounding protocol authenticates a party and its distance to the trusted verifier. The protocol is a tuple (P, V, B), where P is the prover algorithm that takes pre-shared secret key x, V is the verifier algorithm that takes pre-shared secret key y and B is the distance-bound.*[9] *The protocol consists one round of OwTT measuring procedures, in which the prover P corresponds to the initiator and and the verifier V corresponds to the responder. The protocol outputs 1 indicating accept; and output 0 indicating reject otherwise. The protocol satisfies two properties:*

- *Termination: $(\forall loc_V)$, an execution of the protocol starts from the prover sending a message to the verifier and when the verifier receives the message, its internal state changed to finished immediately;*
- *p-Completeness: $(\forall loc_V; loc_P$ such that $d(loc_V; loc_P) \leq B)$ with benign adversary only we have*

$$Pr\left[Out_v = 1 : P^{[GC]}(x) \leftrightarrow V^{[GC]}(y)\right] \geq p$$

4.3.1 Formalization of attacks

In the above definition of distance bounding protocol, authentication and proximity is both required for a successful run of the protocol. In the other word, the prover needs to prove it is within the distance bound *and* hold the correct credential in order to get accepted. The latter requirement is captured by the notion of authentication.

Time-secure Authentication Schewenk [68] have introduced the notion of time-secure authentication protocol in order to define the security of authentication protocol precisely.

[9] *A randomized key generation algorithm $Gen(1^s, r_k)$ that takes security parameter s and randomness r_k, generates secret keys x and y before the protocol execution. When symmetric key scheme is used, x is identical to y.*

We adapt this notion and use it accordingly to define secure authentication of our distance bounding protocol.

Definition 9 (Schewenk Time-secure Authentication) *A protocol* Π *is a* (τ, q, ϵ) *time-secure authentication protocol, if for each adversary* A *that runs in time* τ *and asks at most* q *queries, with probability at least* $1 - \epsilon$*, we have that (1) for each responder oracle that accepts, there is exactly one uncorrupted finished initiator oracle, and (2) for each finished initiator oracle, there is at most one responder oracle that accepts.*

We consider three classes of attacks: distance fraud (DF), man-in-the-middle (MiM) attack, and collusion fraud (CF). In DF, P^*, with $d(P^*, V) > B$, wants to convince V that its distance is less than B. In MiM attack, an external attacker who does not have the secret key, interacts with multiple Ps and Vs, and finally succeeds in taking the role of a prover in a protocol instance. In CF, P^* colludes with a helper (an outsider) to claim a shorter distance to V. The collusion should not leak the prover's secret key to the helper. The formal definitions of the attacks are below.

Experiment Set-up. The following game-based definitions share the same set-up phase as follows: In the beginning of each game, the challenger C sets up a protocol environment with n users $U_1, ..., U_n$ and m outsiders $O_1, ..., O_m$. Users share symmetric secret key with other users, but outsiders does not have secret key. Both users and outsiders have their own location loc_p, which determines the timestamp they can observe during the experiment. The challenger prepares l one-message protocol oracles $\pi^1_{P_i}, ..., \pi^l_{P_i}$ for each party, including users and outsiders. Let $d(loc_{P_i}; loc_{V_j})$ denotes the physical distance between the prover P_i and the verifier V_j. C generates long-lived keys for each pair of users and simulate the Global Counter T. All local variables of oracles are initialized to its initial value at the start point of the game.

Distance fraud (DF) attack In this attack, a dishonest prover who is further than the distance (bound) B, wants to convince the verifier that it has a distance at least B.

Security Game for Distance Fraud (G_{A-1}.) In this game, the adversary A can only ask up to q *CorruptU*, and *Send* queries and A wins the game if there's a responder oracle $\pi^s_{V_j}$ outputting 1 for malicious prover P^*_i where $d(loc_{V_j}; loc_{P^*_i}) > B$.

Definition 10 (DF-resistance) *Let $Pr[G_{A-1}]$ denote the probability that adversary wins in Game G_{A-1}. A distance bounding protocol Π is α-resistant to distance fraud if for each adversary A that runs in time τ and asks at most q quires, in Game G_{A-1} we have*

$$Pr[G_{A-1}] \leq \alpha$$

Distance hijacking. Definition 10 captures *distance hijacking attack* [18] against DLB protocols. In this attack, P^* who is at distance $\geq B$, uses DLB communications of unaware honest provers at a distance $\leq B$, to claim a distance $\leq B$.

Man-in-the-middle (MiM) attack. In the MiM attack, there is an external adversary who does not have the secret-key of the protocol, interacts with honest provers and verifiers, and finally takes the role of a prover in a protocol instance with the verifier.

Security Game for MiM attack (G_{A-2}.) In this game, the adversary A controls the outsider and can only ask up to q *CorruptO* and *Send* queries. A wins the game if (1) there are at least two responder oracles that accept the same message, or if there is a responder oracle that accepts a message from an uncorrupted sender which has not been issued by any sender oracle; or (2) there's a responder oracle $\pi^s_{V_j}$ outputting 1 for honest prover P_i where $d(loc_{V_j}; loc_{P_i}) > B$.

Definition 11 (MiM-resistance) *Let $Pr[G_{A-2}]$ denote the probability that adversary wins in Game G_{A-2}. A distance bounding protocol Π is β-resistant to MiM attack if for each adversary A that runs in time τ and asks at most q quires, in Game G_{A-2} we have*

$$Pr[G_{A-2}] \leq \beta$$

In this definition the attacker can have a learning phase during which it interacts with multiple honest provers and verifiers. It then uses its view in the attack phase, and engages in n protocol instances between honest provers and the target verifier.

Mafia fraud and impersonation attack. Definition 11 is general and covers Mafia fraud and impersonation attack as special cases. In Mafia fraud, there is no learning phase. The attacker interacts with an honest prover and makes the verifier to output accept. In impersonation attack the attacker uses multiple possibly concurrent interactions with the verifier to make the verifier output 1.

Collusion fraud (CF) attack. In collusion fraud, a dishonest prover colludes with a helper to claim a longer distance to the verifier. The collusion should not leak the prover's secret key to the helper. This is captured by the ability of the helper to succeed in a future MiM attack.

Security Game for Terrorist Fraud (G_{A-3}.) In this game, the adversary controls the outsider to help P^* and A can only ask up to q *CorruptO*, *CorruptU* and *Send* queries and A wins the game if (1) there are at least two responder oracles that accept the same message, or if there is a responder oracle that accepts a message from an uncorrupted sender which has not been issued by any sender oracle; or or (2) there's a responder oracle $\pi^s_{V_j}$ outputting 1 for malicious prover P^*_i where $d(loc_{V_j}; loc_{P^*_i}) > B$.

Definition 12 (CF-resistance) *Let $Pr[G_{A-3}]$ denote the probability that adversary wins*

in Game G_{A-3}. A distance bounding protocol Π is (γ, γ')-resistant to collusion fraud if for each adversary A that runs in time τ and asks at most q quires, in Game G_{A-3} we have

$$Pr[G_{A-3}] \geq \gamma$$

then we must have an adversary who has the following success probability

$$Pr[G_{A-2}] \geq \gamma'$$

This definition implies that if a malicious prover P^* who is far from V can succeed in collusion fraud under the help of adversary, then there is an adversary who will succeed in an (extended) MiM attack. In other words colluding attack will not succeed unless some secret information of the malicious prover is leaked.

Terrorist fraud. In Terrorist fraud, P^*, with $d(P^*, V) > B$, gets aid from a helper who does not have the secret key, to succeed in an instance of the protocol with the verifier. Definition 12 captures terrorist fraud as a special case by allowing the collusion fraud adversary play G_{A-2} and succeed in impersonation to make V to accept.

4.3.2 Detailed design of Global Counter (GC)

In the model described above, instead of using clock to measure the time, we try to use Global Counter (GC) to obtain the distance between the prover and the verifier. The unpredictable timestamp is used to replace the clock reading and prevent the adversary from shortening the distance. Because of GC, we are able to measure the one-way transmission time between the prover and the verifier instead of the round-trip time.

The GC is designed to broadcast unpredictable timestamps at a very high frequency. Each timestamp can be seen as a digital symbol, which is a signal with specified duration which presents one or more bits of information in wireless communications. Each symbol is a time value (count) t followed by a l-bits random sequence r. Symbols are separated by the

short space with 0 amplitude and the symbol rate determines the space between individual symbols. The modulation of symbols defines base-band shape, carrier frequency and in which way data is encoded. The preamble provides a way to detect the start of the symbol by using a code sequence with correlation properties. To detect the preamble, the receiver will correlate incoming samples with the known code sequence and ideally only the correct code sequence will trigger the reception process. This is very similar to NIST randomness beacon [58], except GC broadcast (1) at very high frequency; (2) random strings and time values; (3) in a wireless way (i.e. using satellite).

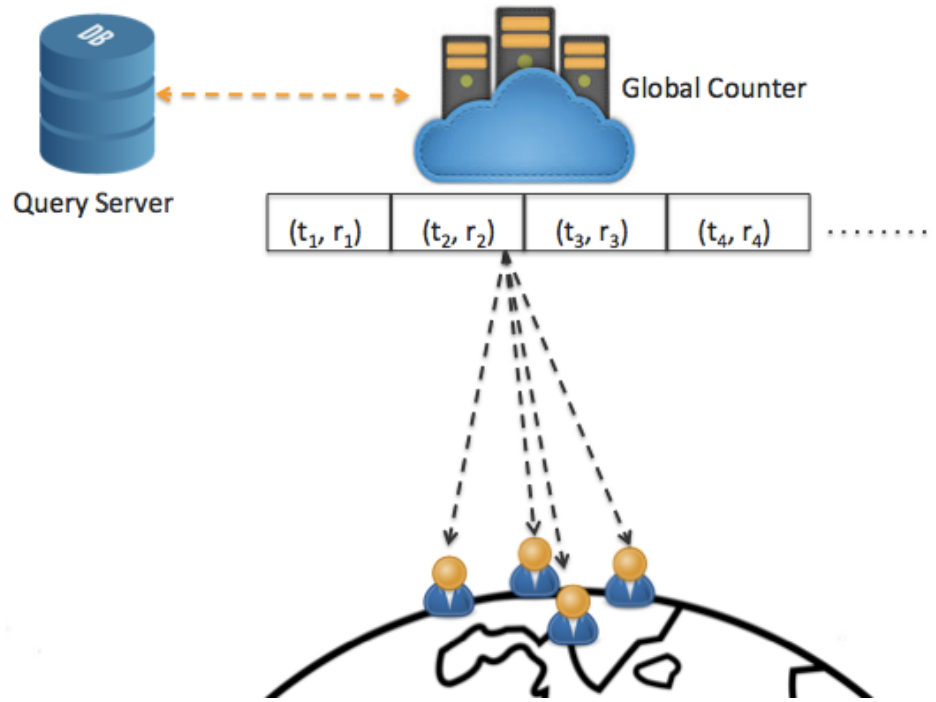

Figure 4.3: Global Counter broadcasts unpredictable timestamps to users

The timestamp broadcasted by GC is unpredictable in terms of future time. Hence, a user (the prover) can send a message with the timestamp to the verifier, to prove that the message is sent at the time that is corresponding to the random sequence. Thanks to its unpredictability, the malicious user can only prove itself to be at current time (by showing the latest available timestamp) or anytime earlier than current time (by showing the timestamp that is available in the past), but not the time that is later than the current time, since the

timestamp is not available yet. This is different from directly reporting the clock reading as timestamp, where the malicious user can report any false clock reading without being detected.

Ideally, for honest prover P and verifier V, the distance between them can be calculated by measuring the one-way transmission time of the message. Assuming GC broadcasts timestamp at a fixed frequency f, one can calculate the elapsed time between two timestamps. Therefore, the verifier can require the prover to obtain the latest random symbol and send it to the verifier, in order to measure the distance.

4.3.3 Protocol description

One-message protocols, such as One Time Password (OTP) protocol, are normally used for authenticating a party. We incorporate the one way transmission time (OwTT) measuring procedure with this kind of one-message authentication protocol to construct a simple distance bounding protocol. Note that this simple design involves only one round of OwTT measuring procedure, while most distance bounding protocols have multiple rounds in time-measuring phase in order to improve security and error tolerance. Also in real-world protocols, generating authentication code takes time and would affect the accuracy of distance bounding protocols. However, for the sake of clarity, we decide to keep our protocol simple at this time and we present the protocol that is more practical in Section 4.5.2.

By having the Global Counter defined above, we are able to design a distance upper bounding protocol that is resistant to distance fraud (DF), Man-in-the-middle attack (MiM) and collusion fraud (CF). The protocol requires the prover to send the message along with the latest timestamp to prove his sending time. The verifier then verifies the timestamp and obtains the elapsed time (one-way transmission time) to calculate the distance. Because the timestamp is unpredictable, so that the malicious prover is not able to shorten the one-way transmission time by attaching a timestamp corresponding to a future time. Thus, DF is prevented. As for MiM and CF, the message is sent in an authenticated way by attaching

message authentication code (MAC), so that unauthorized alteration is prevented. However, authentication requires additional processing time and so deviation occurs. We show how to minimize the effect of authentication in Section 4.5.2.

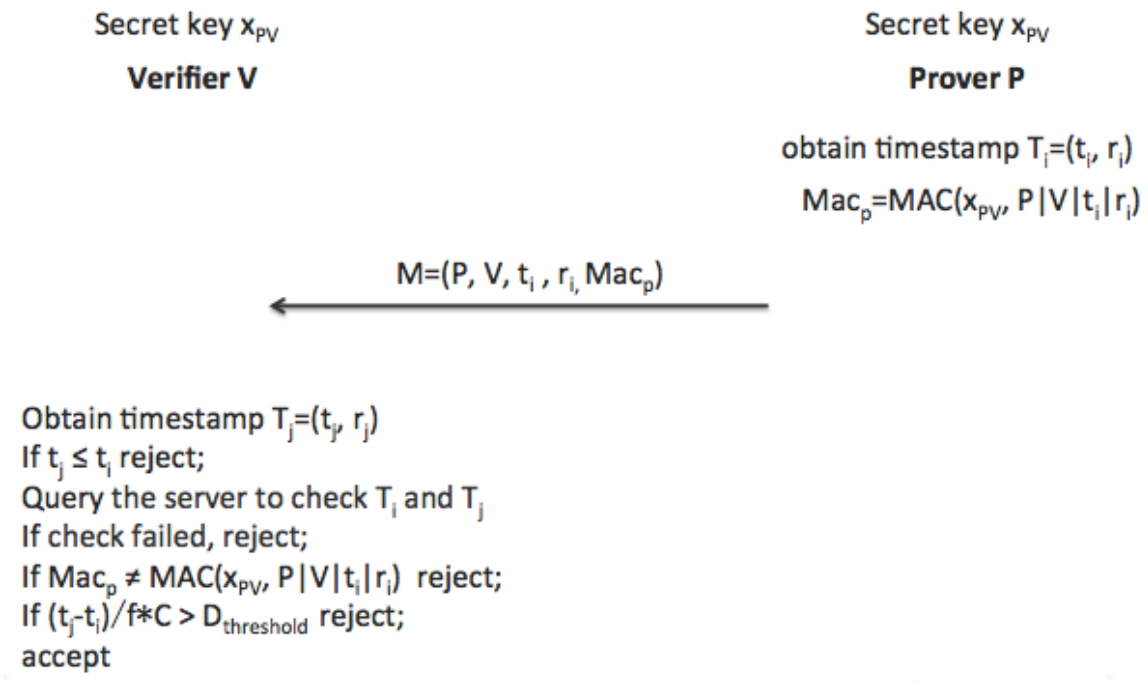

Figure 4.4: Protocol outline of DB-GC protocol

The outline of the protocol is shown in the Figure 4.4. Three parties are involved in the protocol: a GC that broadcasts timestamps at frequency f, a verifier that is trusted and a prover (can be malicious or honest), who would like to prove its distance and identity.

The protocol works as follows:

(1) The protocol starts when the prover obtains the latest timestamp $T_i = (t_i, r_i)$ and constructs message authentication code $Mac_p = (x_{PV}, P|V|t_i|r_i)$, where x_{PV} is the shared secret key between prover P and verifier V. Then a message $M = (P, V, t_i, r_i, Mac_p)$ is sent.

(2) The verifier receives the message and obtains the current timestamp $T_j = (t_j, r_j)$.

(3) The following verifications are then conducted: (i) Verify Mac_p is the correct message authentication code; (ii) Send the request to global counter to verify the validity of timestamp T_i and T_j; (iii) Verify the attached time value t_i is smaller than t_j; (iv) Verify the attached

time value is greater than the local counter; (iv) Compute the distance by

$$D = (t_j - t_i)/f * C$$

where C is speed of light. The verifier accepts if the measured distance D is within the bound B and all verifications are passed; otherwise rejects.

4.4 Security Analysis

Notations: Assuming the GC broadcasts timestamps at a frequency f and each random string in the timestamp has the length of l. The secret key used in the protocol has length of x. And there are in total n users in the system.

4.4.1 Distance fraud (DF) resistance.

In distance fraud, the malicious prover wants to shorten the distance by his own. For distance bounding protocols, all signals are assumed to transmit at speed of light, which means the malicious prover P^* cannot shorten the distance by speeding up the transmission time. However, in traditional DB protocol, P^* can guess the challenge before its arrival and send out the response before receiving the challenge. As the result, the round-trip time of the message will decrease and P^* get accepted if its guess is correct. Considering the challenge is only 1 bit and it is easy for P^* to correctly guess the challenge (with probability 1/2). Therefore, multiple rounds are needed to further decrease the success probability of $P*$.

In our protocol, P^* also needs to guess. P^* needs to guess the timestamp that is corresponding to a future time in order to cheat on the distance. Sending in advance does not work here because the distance is measured by one-way transmission time.

Theorem 7 *$\Pi_{DB-HFRB}$ is ϵ-resistant against any DF attack, where*

$$\epsilon = n^2 \cdot (2^{-l}).$$

Proof: Game G_0 is the original game, where our adversary located outside the distance bound tries to force an oracle to accept a message M^* which is faked. Thus we have

$$\epsilon_0 = \epsilon.$$

In Game G_1, we guess the prover party (initiator) P and the verifier party (responder) V. They share a common symmetric key $k_{P,V}$. Our guess is that the adversary will succeed in making one responder oracle π_v^i a message M^* which is either faked, or has already been accepted by a different oracle. If our guess is wrong, we abort the game, and the adversary looses. According to the fact that in total n parties are in the system, his winning probability is reduced by a factor n. Thus we have

$$n^2 * \epsilon_1 = \epsilon_0.$$

In Game G_2, we abort the game if the adversary $\mathcal{A}$ forges a valid future timestamp or influence the global counter to generate a fake message that includes a valid future timestamp. Since the malicious prover has the secret key, he only needs to forge a valid timestamp that is corresponding to the future time. According to Theorem, we know that the success probability of forging a future timestamp is 2^{-l}, where l is the length of random string of timestamp. Thus we have

$$\epsilon_1 \leq \epsilon_2 + 2^{-l}.$$

Now we have excluded faking the timestamp in this game, the adversary could use a message which were generated by a non-corrupted oracles but only the timestamp is influenced by the adversary, which is impossible according to our definition of global counter. Thus

$$\epsilon_2' = 0.$$

Hence, we conclude that

$$\epsilon = n^2 \cdot (2^{-l}).$$

4.4.2 Man-in-the-middle (MiM) attack resistance.

In MiM attack, the prover P is honest. The adversary $\mathcal{A}$ controls the outsider and wants to shorten the distance between the honest prover P and the verifier V and take that session as his own. In traditional DB protocols, $\mathcal{A}$ normally launch MiM by acting as verifier first to obtain the response and then acting as the honest prover. However, in our protocol, although $\mathcal{A}$ can obtain the message from the honest prover, the message is authenticated by message authentication code in such way that if A wants to replace the timestamp with latest one to shorten the distance, he will also need to know the secret key. Hence, authentication prevents MiM attack.

Theorem 8 *$\Pi_{DB-HFRB}$ is β-resistant against any MiM attack, where*

$$\beta = n^2 \cdot (\epsilon_{MAC}).$$

Proof: Game G_0 is the original game, where our adversary controls outsiders who located within the distance bound and tries to force an oracle to accept a message M^* which is either faked, or has already been accepted by an oracle. Thus we have

$$\epsilon_0 = \beta.$$

In Game G_1, we guess the prover party (initiator) P and the verifier party (responder) V. They share a common symmetric key $k_{P,V}$. Our guess is that the adversary will succeed in making one responder oracle π_v^i a message M^* which is either faked, or has already been accepted by a different oracle. If our guess is wrong, we abort the game, and the adversary looses. According to the fact that in total n parties are in the system, his winning probability is reduced by a factor n^2. Thus we have

$$n^2 * \epsilon_1 = \epsilon_0.$$

In Game G_2, we abort the game if the adversary A forges a valid message authentication code MAC for key $k_{P,V}$. This may happen only with the probability ϵ_{mac}: The simulator

replaces all MAC computations involving the key $k_{P,V}$ with calls to a MAC challenger C_{MAC} which uses a randomly chosen MAC key k; if A forges a valid message authentication code MAC for a fresh message which has not been queried from the MAC challenger, then we have broken the MAC challenge. Thus we have

$$\epsilon_1 \leq \epsilon_2 + \epsilon_{MAC}.$$

Now we have excluded fake message in this game, the adversary could use a message which were generated by a non-corrupted oracles but only the timestamp is influenced by the adversary, which is impossible according to our definition of global counter. The adversary is left with the choice to force an oracle to accept a message M^* that has already been accepted by a different oracle. If A tries to send M^* to any oracle of party $U \neq V$, U will not accept because the target identity is different. If A tries to send M^* to an oracle π_v^i of party V, but M^* has been accepted by another oracle π_v^j of party V, π_v^i will not accept because $t_s \leq t_v$. Thus it holds with probability 1 that the adversary could not make an oracle to accept the message that has been accepted by the other oracle. Thus

$$\epsilon_2' = 0.$$

Hence, we conclude that

$$\beta = n^2 \cdot (\epsilon_{MAC}).$$

4.4.3 Collusion fraud (CF) resistance.

In collusion fraud, $P*$ shorten its distance to the verifier by cooperating with the helper. However, $P*$ will not leak its secret key to the helper, otherwise it's impossible to distinguish the helper from $P*$ and may lead to MiM attack in future. In traditional DB protocols, terrorist fraud is prevented by making the response related to the secret key in such way the only the secret key holder can generate the correct response table. But in our protocol, similar to MiM attack, authentication prevents the helper to replace the timestamp arbitrarily. The

helper is not able to generate the correct message authentication code without the secret key.

Theorem 9 *$\Pi_{DB-HFRB}$ is γ-resistant against any CF attack, where*

$$\gamma = n^2 \cdot (\epsilon_{MAC} + 2^{-l}).$$

Proof:

Game G_0 is the original game, where our adversary makes corrupted outsiders who are located within the bound help the corrupted users to force an oracle to accept a message M^* which is either faked, or has already been accepted by an oracle. Thus we have

$$\epsilon_0 = \gamma.$$

In Game G_1, we guess the prover party (initiator) P and the verifier party (responder) V. They share a common symmetric key $k_{P,V}$. Our guess is that the adversary will succeed in making one responder oracle π_v^i a message M^* which is either faked, or has already been accepted by a different oracle. If our guess is wrong, we abort the game, and the adversary looses. According to the fact that in total n parties are in the system, his winning probability is reduced by a factor n. Thus we have $n^2 * \epsilon_1 = \epsilon_0$.

In Game G_2, we abort the game if the adversary A forges a valid message authentication code MAC for key $k_{P,V}$. This may happen only with the probability ϵ_{mac}: The simulator replaces all MAC computations involving the key $k_{P,V}$ with calls to a MAC challenger C_{MAC} which uses a randomly chosen MAC key k; if A forges a valid message authentication code MAC for a fresh message which has not been queried from the MAC challenger, then we have broken the MAC challenge. Thus we have

$$\epsilon_1 \leq \epsilon_2 + \epsilon_{MAC}.$$

Since we have excluded MAC forgeries in this game, the adversary is left to guess a timestamp (r', t') that is corresponding to the future time and construct a message $M^* =$

$(P, V, r', t', MAC(P, V, r', t'))$. According to Theorem, we know that the success probability of forging a future timestamp is 2^{-l}, where l is the length of random string of the timestamp (r). Thus we have

$$\epsilon_1 \leq \epsilon_2' + \epsilon_{MAC} + 2^{-l}.$$

Now we have excluded fake message in this game, the adversary could use a message which were generated by a non-corrupted oracles but only the timestamp is influenced by the adversary, which is impossible according to our definition of global counter. The adversary is left with the choice to force an oracle to accept a message M^* that has already been accepted by a different oracle. If A tries to send M^* to any oracle of party $U \neq V$, U will not accept because the target identity is different. If A tries to send M^* to an oracle π_v^i of party V, but M^* has been accepted by another oracle π_v^j of party V, π_v^i will not accept because $t_s \leq t_v$. Thus it holds with probability 1 that the adversary could not make an oracle to accept the message that has been accepted by the other oracle. Thus

$$\epsilon_2' = 0.$$

Hence, we conclude that

$$\gamma = n^2 \cdot (\epsilon_{MAC} + 2^{-l}).$$

4.5 Discussion for Practical Implementation

4.5.1 Practical considerations for global counter

In our design, the global counter broadcasts unpredictable timestamps (r, t) at a high frequency. It is assumed that broadcasted timestamps reach all participants simultaneously. This assumption is justified by limiting users in an area of relatively small radius compared to the height of the global counter. The users are assumed to locate within a small area of radius r_{max} meters and the global counter broadcasts from a location that is perpendicular to the ground and has a height of h meters from the the center of the area that user users are

located in (e.g., a satellite or a base station). The minimum distance of the broadcasts travel is $D_{min} = \sqrt{h^2}$ and the maximum distance of the broadcasts travel is $D_{max} = \sqrt{h^2 + r_{max}^2}$. When $h \gg r_{max}$, $D_{max} \simeq D_{min}$ and so users located within this area receive broadcasts quasi simultaneously.

As mentioned above, the granularity of the DB-GC protocol relies on the frequency of the broadcasts and the security of the DB-GC protocol relies on the length of the timestamps. In the following, the parameter choices of the global counter and related performance requirement with respect to data transfer rate, symbol rate, and storage are discussed.

Symbol rate. The symbol rate represents how many symbols are transmitted per second. In the presented case, one timestamp is counted as a symbol and is broadcasted in its entirety. Symbols are separated by short spaces with 0 amplitude and the symbol rate determines the space between individual symbols. Based on our design, the symbol rate of timestamps broadcasted by the global counter is highly related to the granularity of the system. The higher granularity requires a higher symbol rate. In this estimation, it is assumed that an accuracy of 0.3 meter is achieved. Therefore, a symbol rate of $10^9/second$ is required, that is 10^9 symbols to be broadcasted every second.

Symbol size. Symbol size refers to the length of a timestamp. Each timestamp $(r, t,)$ consists of a random string r, a time value t and an authentication code σ. In order to determine the symbol size, the size of each of two components are needed.

- Random string. The length of random string determines the success probability of adversary in guessing a valid timestamp. The longer the random string, the less likely the adversary can succeed in forging a future timestamp. For simplicity, we use a random string of 20 bits, which gives the adversary a success probability of 2^{-20} in forging a valid timestamp.

- Time value. The time value is an increasing count that is used to measure intervals. Assuming the lifetime of a timestamp is 30 minutes[10]. Hence, we need non-duplicate counts that are enough for broadcasting for 30 minutes and the count will start over at every 30 minutes. As mentioned before, in order to achieve a granularity of 0.3 meter, 10^9 symbols need to be broadcasted every second, which requires 10^9 different counts per second. For 30 minutes, $10^9 \times 60 \times 30 = 1.8 * 10^{12}$ indices is needed, which results in approximately 40 bits for time value field.

Hence, each symbol will have length of $40 + 20 = 60$ bits.

Data transfer rate. According to the calculation above, the ideal data transfer rate is $60 * 10^9 = 60$ Gbits/s, in order to achieve the assumed security and granularity. Current advanced wireless technology allows data to be transferred at multi-gigabit speeds. For example, the IEEE 802.11ad protocol enable devices to deliver data transfer rates up to 7 Gbit/s, while maintaining compatibility with existing Wi-Fi devices [87]. But it still cannot accommodate the need. In order to further decrease the data transfer rate, one can decrease the accuracy to 3 meter and the data transfer rate will be decreased to 6 Gbits/s, which is achievable with current technology.

Storage The global counter should have storage that is enough for storing timestamps that are within their lifetime (e.g, 30 minutes). For a 60-bit timestamp, this requires $60 \times 1.8 \times 10^{12} = 1.08 \times 10^{14} bits \sim 100TB$, which is achievable with current technology.

Lookup time In the proposed design, the global counter also maintains a query server for users to validate the correctness of the timestamp. Instead of making the query server searches through the whole storage for each query, we chose to use hashtable to improve the

[10]The time model we proposed is mainly used for measuring distance of few meters to few kilometres

efficiency of the lookup process: All timestamps are stored in the hashtable and therefore the lookup time is constant.

4.5.2 Practical DB-GC protocol

Note that in DB-GC protocol, it is assumed that the processing time, including the time for receiving the timstamp, generating message authentication code (MAC) and sending the message, is negligible compared to the propagation time. This is however not realistic when implementing the protocol in practice and so the measured distance is greater than the actual distance. In order to closely approximate to the real distance, we use another timestamp to isolate the time for generating message authentication code. This more practical DB-GC protocol is shown in Figure 4.5.

In this protocol, the prover is first required to obtain the timestamp $T_i = (t_i, r_i)$ and generate message authentication code $Mac_P = MAC(X_{PV}, P|V|t_i|r_i)$ as normal DB-GC protocol. After generating the MAC, the prover is then required to obtain the other timestamp $T'_i = (t'_i, r'_i)$, attach it to the message and send it out. Note that this timestamp (t'_i, r'_i) is directly sent upon receiving without the message authentication code, so this procedure consumes negligible time if there is a direct channel from the receiving antenna to the sending antenna[11]. However, sending it in an unauthenticated way allows the adversary to substitute it with (t_a, r_a). In collusion fraud, the malicious prover colludes with the helper to shorten the distance. Taking the advantage that the helper locates close to the verifier, the helper can replace this timestamp with a future one to shorten the distance. Hence, we require the the verifier to verify the time between two timestamp T_i and T'_i. By having a threshold for the time to generate message authentication code, we limit the adversary's ability of cheating on the distance in terrorist fraud.

[11]The receiving signal does not require to go through the time-consuming steps, such as ADC/DAC.

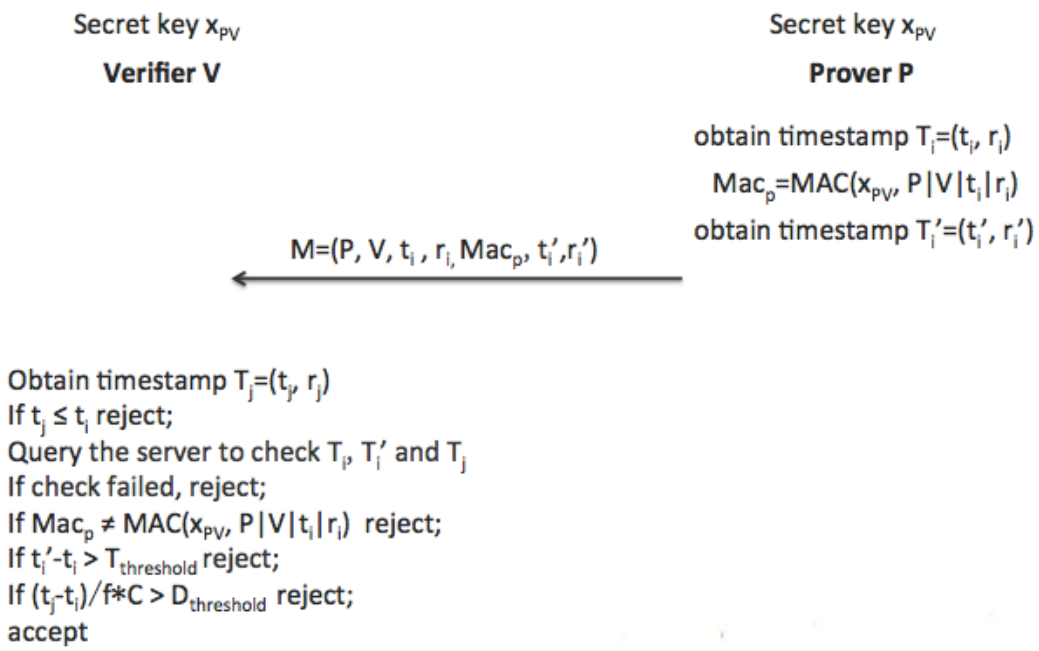

Figure 4.5: Extension of DB-GC protocol

4.5.3 Distance upper bounding in practice

In this work, it is proposed to implement time by having a trusted party broadcast unpredictable timestamps at high frequency. And then we designed our distance upper bounding protocol that uses one-way transmission time to measure the distance based on this time model. In the following, the precision requirement with a real world example is discussed and how well the assumptions can fit for this environment and provide some practical considerations when implementing the global counter is examined.

Distance upper bounding is normally applicable to a scenario in which the prover wants to prove he is within a short distance to the verifier. For example, Passive Keyless Entry and Start (PKES) system is widely used in modern cars, in which the user can open and start the car by simply approaching to the car with the key fob in his pocket. Such a system is very popular because of its convenience. However, PKES system is vulnerable to relay attack and therefore researchers have proposed to incorporate distance bounding protocol to enhance its security [30]. Typically, PKES system requires the key fob within about of 2.5

meters to the car handle [60] for users to have a line of sight to the car. For functionality and ease of implementation consideration, the system requires an accuracy of 2.5 meters and a communication range of tens of meters.

Line of sight propagation. In distance upper bounding scenario, the prover normally wants to prove he is within a short distance to the verifier. In the example of the PKES system, the requirement is 2.5 meters and with such a short distance, it is not hard to guarantee the line of sight propagation. Hence, the assumption of line of sight propagation is reasonable and achievable in this scenario.

Multipath effect. However, multipath may have significant impact on the accuracy of the result, as there could be cars and wall surrounding that reflect signal (can be regarded as an indoor environment). Under multi-path propagation, a receiver generally receives a set of distorted and attenuated copies of the original signal. According to [52], the channel spread of the time-delays is in the order of tens of nanoseconds in a typical indoor environment and this leads to ranging distances that have differences of a few meters. In order to be precise, a receiver needs to identify the first arrived signal. This could be extremely difficult when the first path is not the strongest path. Impulse Radio Ultra-wideband (IR-UWB) is a particularly well suited candidate for this task, as it transmits very narrow pulses and make channel resolution easier[89].

Processing time. Realizing the strict processing time that conventional distance bounding protocols require, a novel distance bounding protocol that utilizes one-way transmission time to measure the distance is proposed. In this design (shown in Figure 4.5), the processing time is minimized since there is no requirement for conversion between digital and analog signal. By assuming there is a direct replay channel between the receiving and sending antenna, the

processing time of our proposed protocol is approximated [12] to the result of analog system [64].

Simply attaching the timestamp without MAC is vulnerable to attacker/helper's manipulation. Hence a threshold in introduced to limit the adversary's ability to cheat on the distance in collusion fraud. The threshold can be obtained by maximum value of all samples. According to [81], the maximum processing time for 64 bytes message using black2s hash function is 22.35ns (using architecture amd64, HW+AES (306c3) and CPU 2013 Intel Xeon E3-1275 V3, 4 x 3500MHz). And according to [76], the fastest digital implementation so far takes 170 ns for receiving and sending the message. Hence, if the malicious prover is able to decrease the time for receiving the message, generating MAC and sending the message to zero, the attacker/helper can cheat on the distance up to 28.8m in collusion fraud. And the adversary's ability to cheat in distance fraud and mafia fraud is limited to 0.15m.

Hence it is concluded that for the environment described in the example, it is possible to achieve the required security and accuracy.

[12]Since there is a direct channel between the receiving antenna and sending antenna, the analog signal that is being detected by the receiving antenna is directly sent to the sending antenna. There is no ADC/DAC conversion and signal modulation/demodulation, which consume the most of processing time in digital system.

Chapter 5

False Rejection attack in Distance Upper Bounding

5.1 Introduction and Motivation

Distance (upper) bounding protocols [39, 64, 66, 10] have been proposed to protect authentication mechanism from man-in-the-middle (MiM) attack. DUB has been widely used for proximity-based application scenario. MiM attack can be a serious threat to various application scenarios like Bluetooth, RFID devices, e-Passports and so on [65, 73, 3], by relaying messages between two honest parties to get authenticated. With DUB protocols, proximity-based authentication allows user(prover) to be authenticated only when coming close to the verifier physically with a pre-shared secret. In this way, one can prevent man-in-the-middle attack by forcing the adversary to come close to the verifier.

All DUB protocols focus on preventing the adversary or the dishonest provers from making the verifier believe he is at a closer location than he actually is. However, the fact that the adversary can easily delay the round-trip time to make the honest prover to appear at a farer place, or tamper the response to fail the challenge-response phase, was

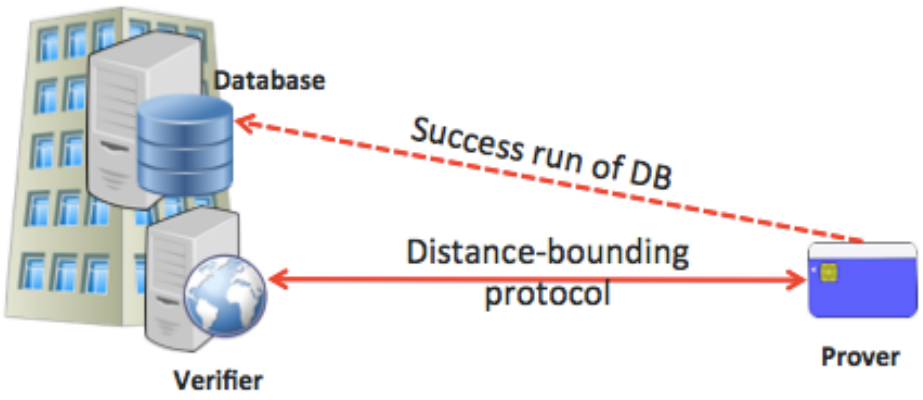

Figure 5.1: Distance-bounding protocol

overlooked. As the result of a failure protocol, the honest prover who locates close to the verifier, will not be able to get authenticated or access to the resources that he should have accessed. In addition, the prover will not aware that the attack is happening, as well as the verifier. With such a denial-of-service like attack, proximity-based authentication will suffer from falsely rejection. We therefore define such attack in distance upper bounding protocol, which is mounted by tampering or delaying the challenge-response communication of distance upper bounding protocol and results in the verifier returning rejection for honest prover with high probability, as false rejection (FJ) attack.

One should notice that tampering or delaying the challenge or the response is not a security threat to standalone distance upper bounding protocol that aims to estimate the distance only. This is because the security goal of such DUB protocol is to obtain the upper-bound of distance between the prover and the verifier. Hence its security guarantee is that there should not be an prover making the verifier believe he is at a closer location than he actually is. Tampering or delaying the challenge or the response only leads to the result that the prover is appeared further than its physical location, which is undesirable. Hence, it does not violate the security guarantee of such DUB protocol. However, latter literature of DUB includes authentication component with distance measurement and therefore false rejection attack becomes a valid attack. Consequently, when DUB protocol is used for proximity-based authentication, false rejection attack leads to a significant threat. Imagine the following scenario shown in Figure 5.1: the database in an office building employs distance bounding protocol for proximity-based authentication to prevent relay attack. Only the employee who is close to the building can get authenticated and gain access to this database. However, the adversary can easily make the protocol fail by tampering the challenge or the response or make the verifier believe that the honest prover is located at a farer place by delaying the challenge or the response. As a result, an legitimate employee who actually locates close to the building is not able to get access to the database, which leads to the falsely rejection. The

other example would be using distance bounding protocol in implantable medical device [65]. Mounting false rejection attack in this scenario may lead to serious health issue of users, even lead to death. In addition, unlike traditional easy-detected denial-of-service (DoS) attack, which is conducted by simply blocking the the whole communication, such attack will not be aware by either the prover nor verifier.

Our contribution of this work can be summarized as follows: First, we identify the overlooked problem when applying distance bounding protocol to proximity-based authentication and define a new class of attack - false rejection attack. The existing distance bounding protocol is vulnerable to message tampering and delay. As a result, the adversary can mount false rejection attack to prevent honest prover from getting authenticated in the proximity-based authentication. This attack type poses a serious threat to many proximity-based authentication applications; Second, we formally define this attack using the same notations in Boureanu's model for distance bounding[8]; Third, we examine the existing distance bounding protocol and propose possible solutions to prevent false rejection attack.

This chapter is organized in the following structure. Section5.2 describes the falsely rejection attack we found towards proximity-based authentication and its formal definition under Boureanu et al. distance bounding model [83]. And in Section5.3, we proposed possible ways to defeat this attack.

5.2 False rejection attack

In this section, we formally define a new class of attack, false rejection attack, that has been overlooked when applying distance bounding protocol to proximity-based authentication. We relate this class of attacks to the three classical attack types in distance bounding protocols and introduce the difference. We find that this attack is a significant threat towards most proximity-based authentication that employs DB-based approach.

Definition 13 (False Rejection Attack) *A False Rejection attack is an attack in which*

an external attacker tampers or delays the challenge-response communication of distance-bounding protocol to make the verifier reject honest prover who is within the bound (close to the verifier). Note that this attack is only valid for those distance bounding protocols that includes authentication or when applying distance bounding protocol to proximity-based authentication scenario.

Proximity-based authentication allows user(prover) to get authenticated by simply coming close to a verifier with a pre-shared secret. Most proximity-based authentication uses distance bounding protocol to measure the distance to determine the proximity and remains resistant to relay attack[1]. In the application scenario of proximity-based authentication, in Figure 5.1, the verifier should be able to validate both the identity of the prover and its distance. If the legitimate prover locates close to the verifier, the verifier should accept and grant the access to the service; otherwise the verifier should reject the prover from accessing the resource. When distance bounding protocol is used for this proximity-based authentication, the verifier uses it to check the identity of the prover, as well as measuring their distance. The authentication only succeeds if the prover has been successfully identified and the measuring distance between them is smaller than the bound. Obviously, the prover is supposed to be rejected by the verifier when authentication is not successful, for either the the prover has not been identified successfully or the measured distance is further than the bound.

However, the vulnerability of distance bounding protocol allows the adversary easily tamper or delay the communication of challenge-response phase. We refer the false rejection attack that is conducted by tampering the communication to tamper-based FR attack and by delaying the communication to delay-based FR attack.

Tamper-based FR attack: the adversary can tamper the challenge-response phase by flipping one bit of either challenge or response (This is practically easy according to [6]).

[1]Most recent distance bounding protocols incorporate the authentication with distance estimation and can be seen as a variant of proximity-based authentication.

As the result, the verification phase would fail. If the challenge bit c_i is flipped to c_i', the honest prover will reply the response r_i' according to the tampered challenge c_i', and so the verifier will reject the prover because the response $r_i' \neq r_i$[2], where r_i is the correct response to c_i. If the response bit r_i is flipped to r_i', the verifier will also reject the prover because the response is not correct. Note that tampering the initialization phase will not succeed, as response table is exchanged in an authenticated way. Therefore, any modifications will make both parties be aware of the undergoing attack.

Delay-based FR attack: the adversary could also delay either direction of the communication in the challenge-response phase with sufficient delay. As a result, the measured distance will exceed the bound, which leads to a failed authentication. Generally, delay is done by jam-and-replay, in which the adversary first jam the communication between the prover and verifier, and then replay the recorded message to introduce additional delay. Note that jamming the initialization phase does not help here, since initialization phase is not time-critical and time measurement is only done during challenge-response phase. Also one should aware that although it's relatively easy to jam the communication [92], many jamming detection techniques [92, 75] are also available.

By tampering or delaying the communication of the challenge-response phase, the adversary can easily make the verifier falsely reject the legitimate prover who is close to the verifier. In this way, the verifier will falsely conclude that the prover is not close or the identity of the prover cannot be authenticated without knowing there's an attack happening. Consequently, the prover will not be able to use the service (e.g., in a payment system scenario), or access to the the resource or service (e.g., in an access control scenario) without being aware of the undergoing attack.

As shown in Figure 5.2, the reason that the verifier outputs reject can come from two different reasons: either the response is incorrect, or the measured distance exceed the bound. Without considering the noise, in the former case, the incorrectness may come from that the

[2]this would not hold when the responses for challenge $c_i = 0$ and $c_i = 1$ are identical

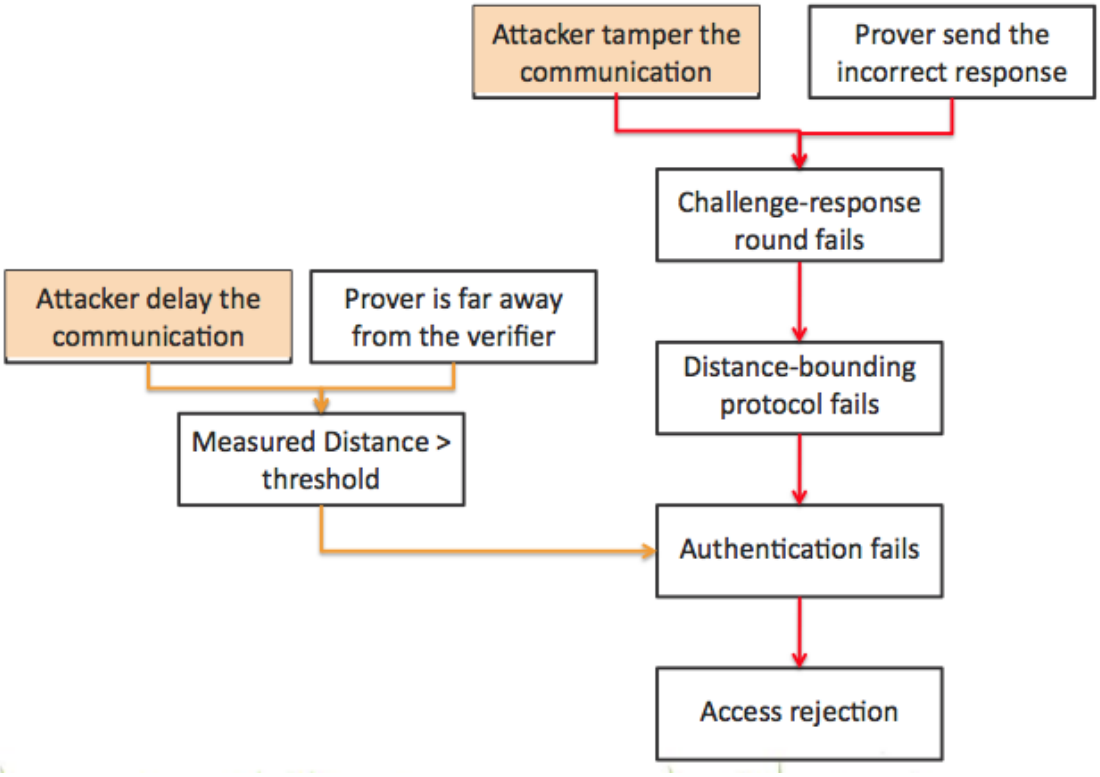

Figure 5.2: Flow chart of access rejection

prover sents the incorrect response or the attacker has tampered the correct response that is sent by the prover to make it invalid; and in the latter case, it can also because that the prover is really far away from the verifier or it is the attacker who intentionally delays the communication, so that the measured distance exceed the bound. This poses challenges to provide countermeasures for false rejection attack.

5.2.1 Formal Definition of FR attack in Baureanu et al. DB model

In this sections, we use standard notations that are used in Baureanu et al. formal distance bounding model [8]. In the false rejection attack, there is an external adversary who does not have the secret-key of the protocol, interacts with honest provers and verifiers, and makes the verifier reject an honest prover who is within the bound.

In Baureanu et al. DB model, the correctness of the distance bounding protocol is defined by p-completeness property, which states that a run of distance bounding protocol between

the verifier and a honest prover who is close to the verifier should succeed with probability greater than p. A general communication of distance bounding protocol is define by an experiment denoted as $exp = (P(x; r_P) \leftrightarrow V(y; r_V))$, where r_P and r_V are random coins, and x and y are secret key of the prover and the verifier, respectively. At the end of a protocol instance, V has an output Out_V, which is 1 or 0, showing acceptance or rejection, respectively. Hence, the formal statement of p-completeness property is as follows:

p-Completeness: $(\forall s)(\forall loc_V; loc_P$ such that $d(loc_V; loc_P) \leq B)$ we have

$$Pr_{r_k;r_P;r_V}\left[Out_v = 1 : \begin{array}{l}(x; y) \leftarrow Gen(1^s; r_k)) \\ P^{[GC]}(x; r_P) \leftrightarrow V^{[GC]}(y; r_V)\end{array}\right] \geq p$$

However, in the false rejection attack, an external attacker involves in the protocol to break p-completeness property. In order to define a distance bounding protocol that is secure against false rejection attack, we extend the experiment to include the adversary: $exp = (P(x; r_P) \leftrightarrow \mathcal{A}(r_A) \leftrightarrow V(y; r_V))$. We define an extended p-completeness property by stating that a run of distance bounding protocol between the verifier and an honest prover who is close to the verifier should succeed with probability greater than p even with the existence of the attacker. And our extended definition for p-completeness property is formally defined as follows:

p-Completeness: $(\forall s)(\forall loc_V; loc_P$ such that $d(loc_V; loc_P) \leq B)$ we have

$$Pr_{r_k;r_P;r_V}\left[Out_v = 1 : \begin{array}{l}(x; y) \leftarrow Gen(1^s; r_k)) \\ P^{[GC]}(x; r_P) \leftrightarrow \mathcal{A}(r_A) \leftrightarrow V^{[GC]}(y; r_V)\end{array}\right] \geq p$$

And accordingly, the false rejection attack is defined as follows:

Definition 14 (FR-resistance) *A DLB protocol* Π *is* η*-resistant to FR attack if* $(\forall s)$*,* $(\forall m, \ell, z)$ *that are polynomially bounded,* $(\forall A_1, A_2)$ *polynomially bounded, for all locations such that* $d(loc_{P_j}, loc_V) < B$*, where* $j \in \{m+1, \cdots, \ell\}$*, we have*

$$Pr_{r_v}\left[\begin{array}{ll} & (x,y) \leftarrow Gen(1^s) \\ Out_v = 0: & P_1(x)...P_m(x) \leftrightarrow A_1 \leftrightarrow V_1(y)...V_z(y) \\ & P_{m+1}(x)...P_\ell(x) \leftrightarrow A_2(View_{A_1}) \leftrightarrow V(y) \end{array}\right] \leq \eta$$

Here probability is over all random coins of the protocol, and $View_{A1}$ is the final view of A_1. The definition effectively allows polynomially bounded number of $P(x')$, $P^*(x')$, and $V(y')$ with independent (x', y'), anywhere.

Similar to MiM attack, we model a strong adversary by allowing him to have learning phase during which it interacts with m honest provers and z verifiers. It then uses its view in the attack phase, and engages in $\ell - m$ protocol instances between honest provers and the target verifier.

5.2.2 Example of false rejection attack in real-world scenario

Consider a hospital scenario shown in Figure 5.3, there is a mainframe keeping sensitive and important documents of patients (e.g., medical history). These documents can be accessed wirelessly by all authorized personnel, in order to facilitate the process. As an added security mechanism, in the case of wireless access, these documents can only be accessed by the authorized personnel inside the hospital building. This is verified every time when an user tries to log in to the system, by running a distance bounding protocol between a station in the building (acting as the verifier) and the user device (acting as the prover).

Assume that an attacker has managed to get into the public area of the hospital building, and aims to assassinate an important governor who is seriously injured in a car accident and needs a surgery immediately. The victim is well-protected in a secure area and the attacker could not get close to him. The doctor is preparing the surgery and needs immediate access to the history record of the patient, which is stored in the mainframe.

If the distance bounding protocol in use is vulnerable to false rejection attack, the attacker can exploit this and either tamper or delay the challenge-response phase to execute false

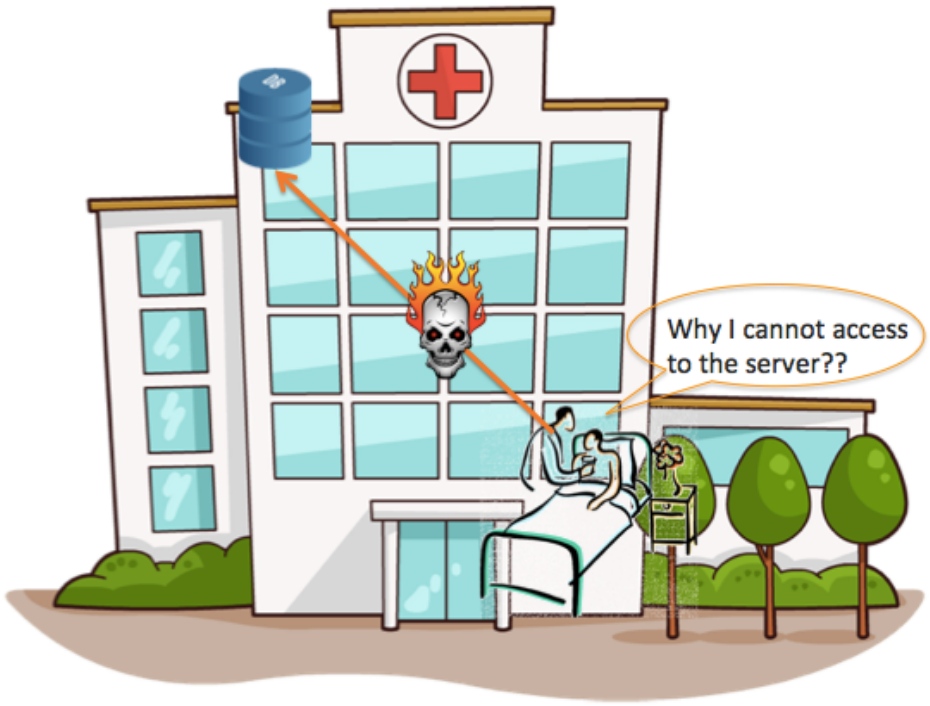

Figure 5.3: Example scenario of false rejection attack

rejection attack. The mainframe security system would keep rejecting the access request of the doctor, because the failure of the distance bounding protocol. And being unable to access to patient's medical record may delay the surgery, and lead to dangerous situation or even death of the patient.

As straightforward as this type of attack, almost every distance bounding protocol is vulnerable to false rejection attack. The strict restriction on processing time makes complicated anti-jamming or authentication channel impossible. Hence, all distance bounding protocols that use time-of-flight (TOF) and have time-critical challenge and response phase are vulnerable to false rejection attack.

5.2.3 Relation to historical attacks

We now relate false rejection attack to the three attack types that are traditionally considered for distance bounding protocols.

First, one should notice that the main difference between false rejection attack and three

other attacks is the goal of the attacker. distance fraud (DF), Main-in-the-middle (MiM) attack and collusion fraud (CF) all aim to shorten the distance between the prover and verifier, in order to make an ineligible prover (does not have correct credentials or is far from the verifier) to be accepted by the verifier. Instead, false rejection attack aims to make the verifier reject an eligible prover by either extending the measured distance or making the response incorrect.

Different from distance fraud or collusion fraud in which the attacker itself is an legitimate user of the system, the attacker in false rejection attack in an outsider who does not have the credential of the system. Compared to conceptually close Man-in-the-middle attack, false rejection attack is easier to be executed. Although false rejection attacker also needs to physically present between the verifier and target prover as MiM attack, he does not need to guess anything and the success probability is much higher than MiM attack.

In addition, unlike traditional easy-detected denial-of-service (DoS) attack, which is conducted by simply blocking the the whole communication, such attack will not be aware by either the prover nor the verifier.

5.3 Protecting against False Rejection Attack

In this section, we investigate how to minimize the vulnerability of distance bounding protocol that the attacker can take advantage to make the verifier falsely rejected the valid user. Without loss of generality, any distance bounding protocol can be divided into three phases: (1) The initialization phase, where nonces and commitments are exchanged; (2) The time-critical fast exchange phase, where the physical distance is measured using rapid bit exchange; and (3) The verification phase, where verification of the response and the time is conducted. Because the initialization phase is not time-critical and involves authentication mechanism in most design, and the verification phase is a non-interactive phase, the only phase that needs to be protected from delay or tamper is the time-critical fast exchange

phase.

The fast exchange phase follows the following schema: the verifier sends out the challenge to the prover and the prover is required to reply the corresponding response immediately after receiving the challenge. This challenge-response process is repeated for a number of rounds. The distance measurement is then derived from the measured round-trip time. Using authentication mechanism could prevent the attacker from modifying messages during the transmission (tamper-based false rejection attack), but it could not protect the protocol from delay-based false rejection attack. In order to protect from both types of attack, we propose to use frequency hopping with the assumption that the adversary could not intercept all communication channels.

Frequency hopping [56, 93, 91] is a well-known technique for transmitting radio signal by rapidly switching the carrier from a number of frequency channels, following a pre-determined sequence that is agreed between two parties. Frequency hopping has been the most popularly considering approach to mitigate jamming attacks. The intuition of the effectiveness of frequency hopping is that it is difficult for the attacker to intercept or identify the target transmission in real time if the pre-determined sequence is unknown. By applying frequency hopping to distance bounding protocol, the prover and the verifier can agree on the pseudorandom sequence in the initialization phase and exchange it in a secure way using shared key. Because the prover is honest in the false rejection attack, this pseudorandom sequence will not be leaked to the attacker. During the time-critical distance bounding phase, frequency hopping is applied to all communications in this phase. As the result, the attacker is hard to intercept the transmission and therefore could not tamper or delay the signal.

However, adding frequency hopping or authentication scheme increases the processing time at the prover's side and so has impact on the accuracy of measured distance. This motivates us to propose a novel distance bounding protocol that uses one-way transmission time (See Chapter 4 for details). In this way, frequency hopping or other cryptographic

techniques can be combined with distance bounding protocol without harming the accuracy of the result.

Chapter 6

Feasibility of Replay Attack in Context-based Proximity Authentication

6.1 Introduction and motivation

The increased popularity of mobile device which equipped positioning capabilities (e.g., GPS) has triggered more attention to location-based application. But generally, existing services choose to trust a users claim about location, without an effective way to verify the location claims. However, simply trusting the users claim is not feasible in some area (e.g. location-based access control), when giving the fake location claim can bring certain advantage. For example, a restaurant is giving out some cash prize to those customers who visited this restaurant. A malicious user can contribute the fake location claim to the restaurant to obtain the advantage without actually being here. Another example can be the university library. Normally, the library online resource is only open to those students who are actually in the library. However, malicious user can also access to these resource by fake claiming he is here, if theres no way to verify the location claims. Hence, its significantly important to research about how to verify the users physical location in wireless network.

Context-based proximity authentication is a novel area that research about validating user's location claim through context information. This approach achieves proximity evaluation by asking users to sense the context information and compare the result. A number of such schemes have been proposed, including utilizing ambient light [35], acoustic environment [80, 35], temperature, humidity, air pressure [70] and Radio Frequency based signal such as WIFI and Bluetooth [80, 44, 82]. In [62] context information is extracted and used for constructing location tag, which is used as the proof-of-presence. And the location tag is

then proposed to applied in private proximity test and proximity authentication [94, 47, 55]. Proximity test is a special form of proximity authentication, where pre-shared secret is not required. In proximity test, the user is able to test if the other user is within its proximity. In these approaches, the prover and verifier sample their context via sensors at the same time and compare the result to see if two parties are in the proximity. If the measurements are similar enough, we say two parties are close.

Using context information to validate user's location claims requires the information contained in the source of context to have the following properties:

- Reproducibility: If two users collect the same source of context information C_1 and C_2 at the same location and time, then C_1 and C_2 matches [1] with high probability.

- Unpredictability: An adversary, who is not at location Loc_u at time T_i, can only produce the context information C that matches the one constructed at location Loc_u at time T_i with negligible probability. This requires the context information should have high entropy in terms of time and location.

- One-wayness: An adversary has negligible probability to figure out the location and time from a given context information, if he is not there at that time. And it should be easy to sense the context information from environmental signal at any location and time.

One recent work [94] gives the detailed design and implementation of proximity test (authentication) using location tag which is generated from 802.11 frames (context information). The location tag is a novel form of user location representation to defeat location cheating attacks in LBS. In this work, authors demonstrate how to use rea-word WiFi and cellular signal to generate unforgeable location tags in practice. By examining the above requirements, we find out that [94] does not evaluate the entropy of 802.11 frames in terms of time.

[1] Matches can be determined using different techniques, such as hamming distance. In this work, we use the percentage of overlap of two sets to determine matching.

This may lead to replay attack, in which the malicious user uses historical data to claim a false location. We further looked into this problem and investigate the feasibility of replay attack when using 802.11 frame as the source of context information for proximity-based authentication. We prove that the effectiveness of replay attack when using the control frame only as the source of the context information with real-world data. We therefore conclude that the control frame is a time-invariant source and advise to combine other time-variant source when using 802.11 control frame as the source of context information.

6.2 Location tag and Replay attack

In [62], location tag was first introduced as a token of proof associated with a point in space and time. Generally, location tag is a collection of context information presented at a certain location and a certain time. An ideal location tag should at least have the reproducibility property: If two measurements of context information at the same space and time produce location tags T_1 and T_2, then T_1 and T_2 match with high probability. On the other hand, from the security point of view, an ideal location tag must have unpredictability property: An adversary not at specific location and time is unable to produce a location tag that matches the tag constructed at that location and time. Location tag has wide applications for LBS. For example, in proximity-based authentication, the prover and the verifier record context measurements simultaneously. And then the prover sends his context measurement to the verifier. The verifier compares the context measurement that sent by the prover with his own and accepts the prover if two measurements are similar enough (as shown in Figure 6.1).

Different kinds of context information has been proposed to generate location tag. [55] proposed to generate location tag using GSM signal and [94] proposed to generate location tag using the combination of WiFi and GSM signal. In this chapter, we mainly focus on the security analysis of location tag that is generated by WiFi signal.

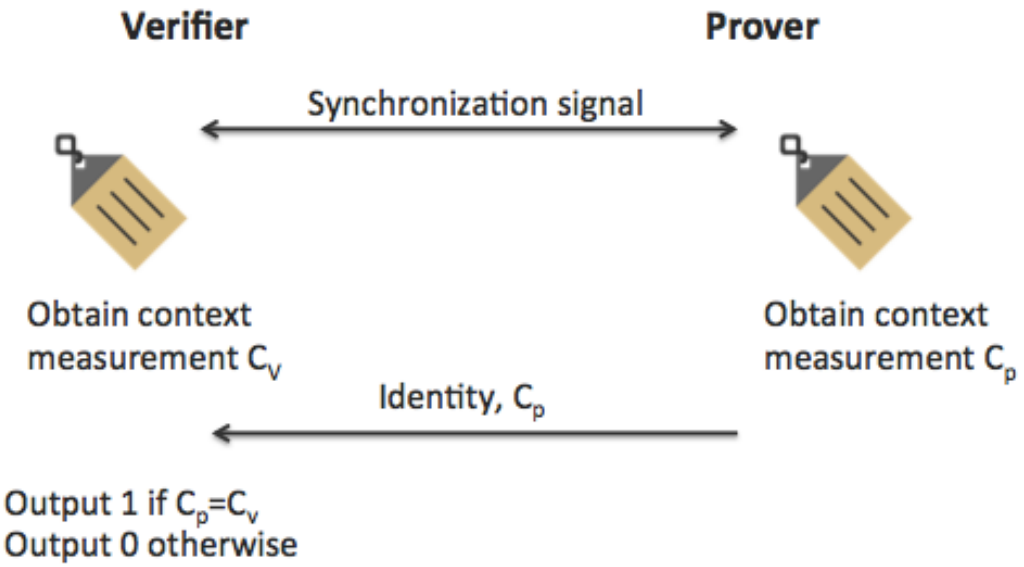

Figure 6.1: Proximity-based authentication using context measurement

6.2.1 Introduction of 802.11 frame

The well-known term WiFi refers to wireless local network (WLAN) that are based on IEEE 802.11 standard. The structure of 802.11 frame is shown in Figure 6.2. The captured 802.11 frames can mainly be classified into three types:

- Management frame, which is used to maintain communication. It includes Beacon frame, authentication frame, association frame and probe frame.
- Control frame, which is used to facilitate data exchange between stations. It includes CTS, RTS and ACK.
- Data frame, which carries upper level packet.

[94] shows that the probing request frames (in the class of management frames) contain the most amount of entropy since algorithm used to scan for access points is not explicitly defined in 802.11 standard. On the other hand, the beacon frames contain the least amount of entropy that are transmitted with consecutive sequence numbers. Although data frames contain high entropy, but it suffers from low reproducibility. Hence the author suggests to

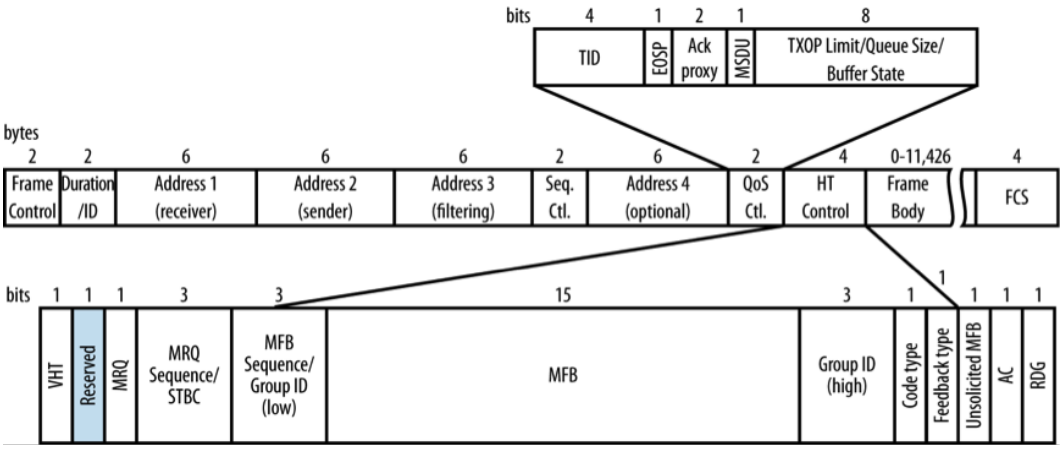

Figure 6.2: 802.11 frame structure, taken from [85]

use the combination of control frames and management frames only, because the content of data frame highly depends on the user, instead of location or time. However, in the following of the chapter, we will show that control frames are reusable within few hours of time and this leads to the vulnerability of replay attack when using control frames as one of the source of context information.

6.2.2 Replay attack

In the replay attack towards context-based proximity authentication, a malicious prover P' presents a context measurement (location tag) that is obtained in the past at the same location to a verifier V. As shown in Figure 6.3, replay attack is done by malicious prover firstly recording a context measurement C_{t-k} at time point $t-k$, an then presenting this context measurement C_{t-k} later at time t when the malicious prover is no longer at that location. If C_{t-k} and C_t are similar enough, the verifier falsely accepts the prover.

Replay attack puts a serious threat to some proximity-based authentication scenario, especially for those the advantage is distributed virtually after passing the authentication. For example, a music store provides a check-in promotion to attract customers: when customers physically come to the store, he/she is offered a free download of a piece of music. Each time when user check-in, the music that offered to be downloaded is different. The proximity

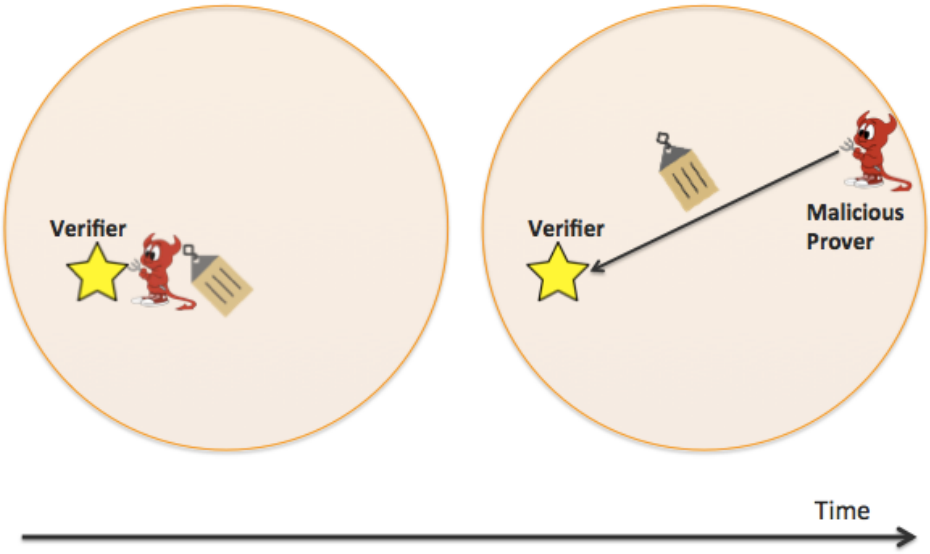

Figure 6.3: Replay attack

test is done by a context-based approach. If the protocol is vulnerable to replay attack, the adversary can record the context measurement at the first time of his visit and download the music later without showing up at the store. Even worse, the adversary can distributed the context information he recorded to help others get this advantage in exchange for other benefits. The other example is electronic election. Consider there's a anonymous electronic election for a neighbourhood. People within certain distance of the neighbourhood is considered as the resident of the neighbourhood and has right to participate the electronic election. The evaluation of proximity is done through context-based approach. If the protocol used in this electronic election is vulnerable to replay attack, the adversary can can share the context measurement he recorded with his colluding parties. In this way, his colluding parties can also participate this election even if they are not any close to the neighbourhood. Hence the adversary might be able to affect the result of the election if the number of his colluding parties is sufficient, which is unfair for such election.

6.3 Feasibility of Replay Attack using 802.11 Frame

In order to evaluate the feasibility of replay attack when using 802.11 frame as the source of context measurement, two experiment are conducted: the first one is to evaluate the reproducibility of 802.11 frame; the other one is to evaluate the unpredictability of 802.11 frame. The result shows that the reproducibility of management frame and control frame is relatively high and data frame is quite low, which is inline with the result shown in [94]. However, as for unpredictability test, management frame is shown to be more resilient to replay attack, while the adversary can succeed in replay attack using control frame that collected within 30 min 4 hours. Our experiment result suggests that using control frame only as the source for context measurement could lead to replay attack. One should also be extremely careful when using control frame as one of the sources for context measurement (e.g., [94] uses the combination of control frame with management frame as the source of context measurement), as this may increase the success probability of replay attack.

6.3.1 Experiment set-up

To test the replay attack feasibility, I carry out the experiment by capturing and analyzing 802.11 WLAN network. we use one Macbook pro with 2.7Ghz CPU, 802.11 a/b/g/n network card and one Macbook air with 1.3Ghz CPU, 802.11 a/b/g/n/ac network card to monitor and record the WLAN traffic at varying time. Using the recorded data, we evaluate the following two aspects as experiment result: (1) measuring the success probability of location tag reproduction among two users at the same location ($< 20cm$) and same time ($< 1s$) using reproducibility data set; (2) measuring the success probability of using historical data to get accepted, which is replay attack. The detailed experiment configuration is as follows.

1. We deployed two client laptops running Wireshark in the promiscuous mode.

2. Each capture last for 30 seconds and capture around $15,000 \sim 30,000$ packets depending on the network. Each capture is output to a respective (.pcap) file for further

processing.

3. For reproducibility dataset, two client laptops start capture at roughly same time ($< 1s$) and same location ($< 20cm$). Capture are repeated for 100 times. In total, 200 samples ($\sim 300,000$ packets) are captured. Every two samples that captured at the same location and time is labelled as co-located-time.

4. For replay attack dataset, each client laptop did one capture for every 30 seconds and last for 10 hours. The capture is repeated for 7 days and last for in total 70 hours and in total 8400 samples ($\sim 168,000,000$ packets) are captured. Every packet is labelled with the time and date.

5. Experiment environment: The experiment is conducted in an apartment. Among all packets, 48.5% are management frames, 32.8% are control frames and 18.7% are data frame. Also, 1213 different MAC address were detected and most of them are 802.11 clients. There are around 23 access points can be detected in this area, which is a comparable size to metropolitan area.

6. Fields used in the experiment: Since most fields of 802.11 frames have fixed or identical value in normal case and some fields are highly depends on the client configuration (e.g. frame number, time stamp), we only use some of the fields in our experiment for efficiency consideration. The fields I used are MAC address of sender (wlan.ta, 6bytes), MAC address of receiver (wlan.ra, 6bytes), frame length (frame.len, 2bytes), frame type (wlan.fc.type, 1byte), frame subtype (wlan.fc.subtype, 1byte), and sequence number (wlan.seq, 2bytes).

6.3.2 Reproducibility result

In reproducibility test, every two samples that are labelled as co-located-time are compared in terms of their similarity of management frame, control frame and data frame. The similarity

here refers to the percentage of common entries among both datasets. As mentioned in [94], two captures are considered reproducible if the similarity is higher than roughly 80%. Therefore, as shown in Figure 6.4, the management frame and control frame both satisfies reproducibility property, while data frame not. And the control frame performs better than management frame in reproducibility test.

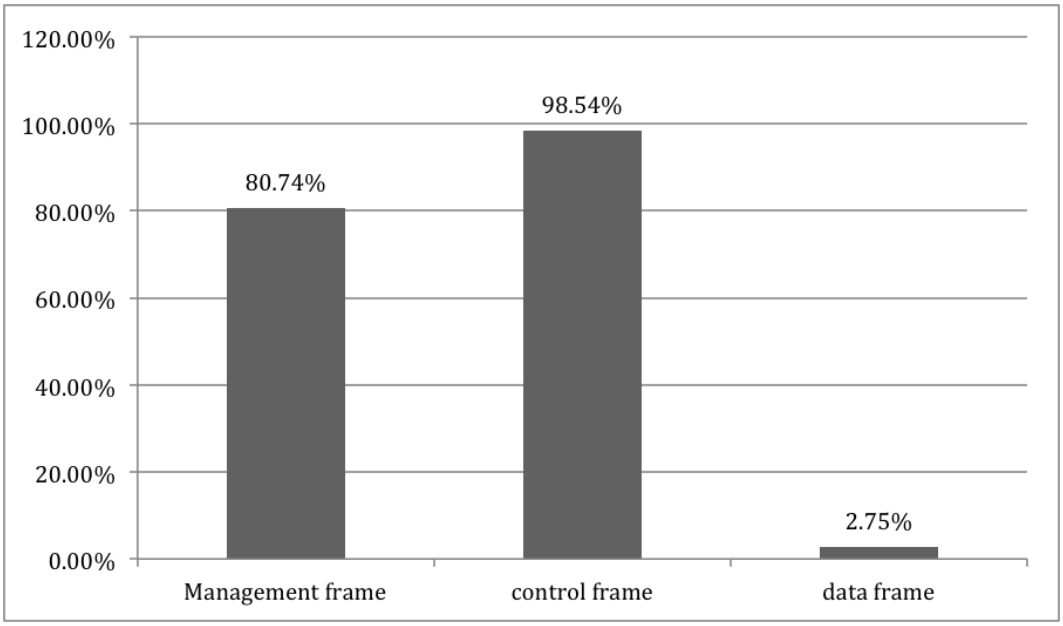

Figure 6.4: Result of reproducibility test

6.3.3 The result of feasibility of replay attack

In order to evaluate the feasibility of replay attack, we captured 802.11 traffic at the same location at 30 second frequency for a consecutive long period of time. We then compare every two captures for their similarity ans show our results in Table 6.1. As we can see, the control frame has relatively high similarity for time interval that is less than 3 hours. And the management frame and data frame has very low similarity constantly. If we define any two datasets whose similarity is higher than 80% as reproducible datasets, and one successful replay attack is counted if two captures are reproducible datasets with different time label. We then conclude that the control frame is most vulnerable to replay attack

and the adversary has high probability in succeeding with replay attack using historical data that is obtained within 3 hours.

Time Interval	<15min	<30min	<1hour	<3hour	<6hour
Management frame	4.18%	3.40%	1.41%	0.71%	0.51%
Control frame	98.06%	93.13%	88.35%	80.13%	2.73%
Data frame	1.90%	0.60%	0.31%	1.03%	0.91%

Table 6.1: The average similarity of different type of frames over different time intervals

Chapter 7

Conclusion and Future Work

7.1 Conclusion

Securely bounding the distance of wireless devices has become a developing issue given the proliferation of wireless devices and location-based services. This thesis explored solutions in the area of secure distance bounding to provide various countermeasures against the sophisticated spoofing attacks. The thesis started with a brief discussion of current threats towards location-based (LBS) applications and introduced secure positioning and localization system to prevent location cheating in LBS. A comprehensive review of distance bounding protocols, along with its attacks and formal model has been provided. This thesis made two main contributions and two smaller contributions to the area. The two main contributions are summarized as follows: (i) Proposing and formalizing solutions to the new problem of Distance Lower Bounding (DLB) that is dual to DUB problem; and (ii) Proposing a novel approach to construct DUB protocols and providing a new formal time model for asynchronous time in distributed system. The two other contributions are: (iii) Discovering a new attack that is called false rejection attack on DUB protocols, which prevents a legitimate user from being authenticated and therefore breaks the correctness guarantee of DUB protocols; and (iv) Investigating the feasibility of replay attack in the context-based proximity authentication using real world data.

Contribution (i) raised new research directions and opportunities in the area of distance bounding as DLB is a new problem with many potential applications, but had not been considered before. This work showed the major underlying difference between DLB problem and DUB problem: it is impossible to construct secure DLB protocol without making physical assumptions on the prover and its ability to access to its device and communication

channel. This difference comes from the fact that delay becomes advantage for the adversary in DLB, while it is not true in DUB. Since it is hard to prevent delay using cryptographic protocols only, this poses challenges when constructing secure DLB protocol. In this work, we formalized the problem of DLB in the setting that the prover is not trusted and provided formal security definitions against three general classes of attacks (DF, MiM and CF). We formally proved that it is impossible to have secure DLB protocol without making physical assumptions on the adversary's ability to access to the prover's device and communication channel. We showed that an adversary can always succeed in DF attack if it has unrestricted access to the prover's device and the adversary can always succeed in MiM attack if it can control the communication channel between the prover and the verifier. Resistance to CF attack requires the resistance to both DF attack and MiM attack. Therefore our protocol considers a malicious prover whose device has bounded memory, and an external attacker that has restricted access to the communication channel between the prover and the verifier, and provides security against $\text{rDF}^{[\mathtt{BM}]}$, $\text{rMiM}^{[\overline{\mathtt{OC}}]}$, and $\text{rCF}^{[\mathtt{BM},\ \overline{\mathtt{OC}}]}$, using reasonable assumptions. Enforcing assumptions in practice however would need special technologies such as targeted jamming [32]. Note that the assumptions that are used in this work can be replaced with other reasonable assumptions and therefore leads to different distance bounding protocols. For example, in this work, we also introduced a secure DLB protocol that utilizes a software-based *externally verifiable code execution (EVCE)* system such as Pioneer [69], to guarantee that the *target executable* code associated with the distance measurement, is executed without modification by a malicious code that may reside on the device. The important point to note is that *one must restrict the adversary's physical access* to the environment to achieve *any level* of DLB security.

Contribution (ii) proposed a new approach to construct distance upper bounding (DUB) protocol, which is completely different from any previous DUB protocols. In this approach, time measurement is relied on a trusted party that broadcasts unpredictable timestamps

at a high frequency. The provers and the verifiers therefore can obtain synchronized time through timestamps without having a local clock. And timestamps can be used to measure the one-way transmission time of a message. The unpredictability of timestamps guarantees that the adversary could not forge a future timestamp to shorten the one-way transmission time. We then designed an efficient one-round secure DUB protocol that relies on the above idea, in which one-way transmission time is measured using unpredictable timestamps that are broadcasted by the trusted party. In order to prove the security of the protocol, we provided a formal model for this time implementation in distributed system and then extended it for modelling security of distance upper bounding. In this model, we defined three general attack (DF, MiM, CF) in concurrent setting and by intuition, these definitions are equivalent to those in well-established DUB model [8]. We then proved the security of our protocol. This work provided important and interesting results for two reasons: (1) Firstly, plenty works that designed DUB protocols have been proposed these years, but there are only a few implementations. The reason is that it is assumed that the processing time is negligible compared to the propagation time in DUB, however, it is hard to achieve this strict requirement with current state-of-the-art hardware. Moreover, this requirement also limits the possible development of DUB applications. For example, we identified the false rejection attack that can disrupt the correctness of DUB protocol and proposed to use frequency hopping to defeat this attack. This solution, however, is not applicable to the distance bounding schemes that use round-trip time for distance measurement as frequency hopping consumes too much time and makes the result inaccurate. However, with our new scheme that uses one-way transmission time for distance measurement, processing time is no longer the limitation of the implementation of DUB protocols and frequency hopping can be applied to DUB protocol as well. (2) The time model we provided can be used for other time-based protocol in distributed system.

Remarks for contribution (i) and (ii). Research into distance bounding protocol currently focuses on an abstract level and its security guarantee is considered in ideal environments without physical details. Hence, it could be challenging to achieve specific security and accuracy requirements when implementing such protocols in practice. In this thesis, we proposed multiple distance bounding protocols and provided security analysis in the ideal environment, in which the signal transmitted between the prover and verifier is line of slight propagation, all parties are located in a free space and the processing time is negligible compared to the propagation time of the signal between the prover and the verifier. However, in practice, because of the restriction of the environment, these assumptions will rarely be true. Hence, the travel time measurement will be biased by multipath propagation, as well as non line of sight propagation. In addition, it is hard to achieve negligible processing time with current technology. Although we consider these phenomena and justify that these assumptions could be reasonably achieved in the typical application scenario, it is not sufficient to guarantee that one can achieve the same security and precision. Hence, when implementing such a system for measuring the distance between transmitters and receivers, one must take these phenomena into account. In addition, depending on the different accuracy and ranging requirements of the application envisaged, extensive experiments should be conducted to evaluate the effects that caused by different environments.

Besides two main contributions described above, two other contributions are summarized as follows: Contribution (iii) identifies the overlooked problem and defined a new class of attack which is named false rejection attack. In false rejection attack, the adversary tampers or delays the challenge-response phase to make the verifier falsely reject honest prover. This attack type poses a serious threat in many proximity-based application scenario. Contribution (iv) investigated how replay attack can impact the security of context-based proximity authentication in a scenario that the prover is not fully trusted. We showed the feasibility of such attack in a scenario that users used 802.11 frame (WIFI) as the source of

context information and advised countermeasures to defeat such attack.

7.2 Future work

Distance lower and upper bounding. This thesis presented the first treatment of the DLB problem in the untrusted prover setting with a number of applications, raising new research directions and opportunities in location based services. An important open question is how to efficiently incorporate DLB in DUB protocols to provide security against distance enlargement attacks and distance reduction attacks. Note that the design of DLB protocol makes it impossible to reply the response before receiving the challenge and therefore the the proposed DLB protocol provides the same security guarantee against attacks that aim to shorten the distance. The proposed DLB protocol results in a upper bound and a lower bound of the distance, which effectively composes a shape of ring. And the prover is guaranteed to be located in this ring area, which gives the possibility to develop more sophisticated access control schemes. However, compared to the conventional DUB protocols, the proposed DLB protocol consumes more energy as additional messages are sent in order to remotely erase the memory of the prover's device. Hence, it will be very interesting to see an alternative design of the distance bounding protocol that is more efficient and can also prevent enlarging and shortening distance at the same time.

Multiple round of distance upper bounding protocol using timestamps. This thesis also presented a new approach to construct distance upper bounding (DUB) protocol, which is completely different from any previous DUB protocols. A formal model for a time implementation in distributed system is provided. For the simplicity consideration, this time model only considers one round protocol and therefore the proposed protocol contains one round as well. However, having multiple rounds will enhance the security of distance bounding protocol. Hence it is worth for further research to extend this time model in order to

include multiple rounds protocols. In addition, it will be an interesting open question to see if this time model can be used for other time-based protocols to improve their security and efficiency.

Realization of distance bounding protocol. Note that since research in distance bounding area currently focuses on an abstract level and its security guarantee is considered in ideal environments without physical details, it could be challenging to achieve specific the security and precision accuracy requirements when implementing such protocols in the real world. In addition, distance bounding protocols normally require fast processing, which does not match current state-of-the-art hardware. In this thesis, we proposed two distance bounding protocols that loosen the requirement of fast processing time and therefore makes it possible to have their implementation in practice. Therefore it will be interesting to see the implementation prototype of the two main protocols that are discussed in this thesis and compare the results with the existing prototype.

In addition, considering the different environments in which distance bounding protocol would be deployed, there are also signal propagation factors affecting result accuracy. A comprehensive guide of how to examine the security and precision requirement of the targeted application scenario and how to conduct extensive experiments to determine the effectiveness of the protocol in the real environments is a big plus when deploying distance bounding protocols for commercial use.

Countermeasures for false rejection attack. Although we have proposed to use frequency hopping to defeat false rejection attack, it is still interesting to see if there is any other countermeasures that can prevent false rejection attack. For example, the adversary can mount false rejection attack by delaying the signal between the prover and the verifier. However, it is hard to delay the signal that is broadcasted in all directions. Utilizing this

fact, it might be possible to defeat false rejection attack by having multiple hidden observers that passively receive the message. By comparing the triangulated locations of the prover using the results from different set of observers, it is possible to tell if there is a false rejection attacker delaying the communication.

Bibliography

[1] G. Avoine, M. A. Bingöl, S. Kardaş, C. Lauradoux, and B. Martin. A framework for analyzing rfid distance bounding protocols. *Journal of Computer Security*, 19(2):289–317, 2011.

[2] G. Avoine, C. Lauradoux, and B. Martin. How secret-sharing can defeat terrorist fraud. In *ACM conference on Wireless network security*, 2011.

[3] G. Avoine and A. Tchamkerten. An efficient distance bounding rfid authentication protocol: balancing false-acceptance rate and memory requirement. In *Information Security*, pages 250–261. Springer, 2009.

[4] M. Barbosa and P. Farshim. Security analysis of standard authentication and key agreement protocols utilising timestamps. In *Progress in Cryptology–AFRICACRYPT 2009*, pages 235–253. Springer, 2009.

[5] M. Bellare. New proofs for nmac and hmac: Security without collision-resistance. In *Advances in Cryptology-CRYPTO 2006*, pages 602–619. Springer, 2006.

[6] N. Borisov, I. Goldberg, and D. Wagner. Intercepting mobile communications: the insecurity of 802.11. In *Proceedings of the 7th annual international conference on Mobile computing and networking*, pages 180–189. ACM, 2001.

[7] I. Boureanu, A. Mitrokotsa, and S. Vaudenay. On the pseudorandom function assumption in (secure) distance-bounding protocols. In *LATINCRYPT*.

[8] I. Boureanu, A. Mitrokotsa, and S. Vaudenay. Practical & provably secure distance-bounding.

[9] I. Boureanu, A. Mitrokotsa, and S. Vaudenay. Towards secure distance bounding. *FSE 2013*.

[10] I. Boureanu, A. Mitrokotsa, and S. Vaudenay. Secure and lightweight distance-bounding. In *Lightweight Cryptography for Security and Privacy*, pages 97–113. Springer, 2013.

[11] S. Brands and D. Chaum. Distance-bounding protocols. In *EUROCRYPT 1993*, pages 344–359.

[12] C. Cachin and U. Maurer. Unconditional security against memory-bounded adversaries. In *CRYPTO'97*.

[13] M. Calle and J. Kabara. Measuring energy consumption in wireless sensor networks using gsp. In *Personal, Indoor and Mobile Radio Communications*.

[14] S. Capkun and J.-P. Hubaux. Secure positioning in wireless networks. *IEEE Journal on Selected Areas in Communications*, 2006.

[15] D. Cash, Y. Z. Ding, Y. Dodis, W. Lee, R. Lipton, and S. Walfish. Intrusion-resilient key exchange in the bounded retrieval model. In *Theory of Cryptography*. 2007.

[16] J. T. Chiang, J. J. Haas, and Y.-C. Hu. Secure and precise location verification using distance bounding and simultaneous multilateration. In *Proceedings of the second ACM conference on Wireless network security*, pages 181–192. ACM, 2009.

[17] C.-Y. Chow, J. Bao, and M. F. Mokbel. Towards location-based social networking services. In *Proceedings of the 2nd ACM SIGSPATIAL International Workshop on Location Based Social Networks*, pages 31–38. ACM, 2010.

[18] C. Cremers, K. B. Rasmussen, B. Schmidt, and S. Capkun. Distance hijacking attacks on distance bounding protocols. In *S&P*, pages 113–127.

[19] R. D. Darnell and C. C. Douglas. Cellular position locating system, Aug. 27 1991. US Patent 5,043,736.

[20] S. Deconinck. Towards secure distance bounding. 2013.

[21] Y. Desmedt. Major security problems with the unforgeable(feige)-fiat-shamir proofs of identity and how to overcome them. In *SecuriCom*, volume 88, pages 15–17, 1988.

[22] G. Di Crescenzo, R. Lipton, and S. Walfish. Perfectly secure password protocols in the bounded retrieval model. In *Theory of Cryptography*.

[23] S. Drimer and S. J. Murdoch. Keep your enemies close: Distance bounding against smartcard relay attacks. In *USENIX Security*, volume 2007, 2007.

[24] U. Dürholz, M. Fischlin, M. Kasper, and C. Onete. A formal approach to distance-bounding rfid protocols. In *Information Security*, pages 47–62. Springer, 2011.

[25] S. Dziembowski. Intrusion-resilience via the bounded-storage model. In *Theory of Cryptography*, pages 207–224. 2006.

[26] E. Elnahrawy, X. Li, and R. P. Martin. The limits of localization using signal strength: A comparative study. In *Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on*, pages 406–414. IEEE, 2004.

[27] K. Finkenzeller and R. Waddington. *RFID handbook: radio-frequency identification fundamentals and applications*. Wiley New York, 1999.

[28] M. Fischlin and C. Onete. Terrorism in distance bounding: modeling terrorist-fraud resistance. In *Applied Cryptography and Network Security*.

[29] M. Fischlin and C. Onete. Subtle kinks in distance-bounding: an analysis of prominent protocols. In *Proceedings of the sixth ACM conference on Security and privacy in wireless and mobile networks*, pages 195–206. ACM, 2013.

[30] A. Francillon, B. Danev, and S. Capkun. Relay attacks on passive keyless entry and start systems in modern cars. In *NDSS*, 2011.

[31] D. Giustiniano and S. Mangold. Caesar: carrier sense-based ranging in off-the-shelf 802.11 wireless lan. In *Proceedings of the Seventh COnference on emerging Networking EXperiments and Technologies*, page 10. ACM, 2011.

[32] S. Gollakota, H. Hassanieh, B. Ransford, D. Katabi, and K. Fu. They can hear your heartbeats: non-invasive security for implantable medical devices. In *ACM SIGCOMM*, pages 2–13, 2011.

[33] S. K. Gupta, T. Mukheriee, K. Venkatasubramanian, and T. Taylor. Proximity based access control in smart-emergency departments. In *Pervasive Computing and Communications Workshops, 2006. PerCom Workshops 2006. Fourth Annual IEEE International Conference on*, pages 5–pp. IEEE, 2006.

[34] S. Haber and W. S. Stornetta. *How to time-stamp a digital document*. Springer, 1991.

[35] T. Halevi, D. Ma, N. Saxena, and T. Xiang. Secure proximity detection for nfc devices based on ambient sensor data. In *Computer Security–ESORICS 2012*, pages 379–396. Springer, 2012.

[36] G. P. Hancke. A practical relay attack on iso 14443 proximity cards. *Technical report, University of Cambridge Computer Laboratory*, pages 1–13, 2005.

[37] G. P. Hancke. Design of a secure distance-bounding channel for rfid. *Journal of Network and Computer Applications*, 34(3):877–887, 2011.

[38] G. P. Hancke. Distance-bounding for rfid: Effectiveness of terrorist fraud in the presence of bit errors. In *RFID-Technologies and Applications*, 2012.

[39] G. P. Hancke and M. G. Kuhn. An rfid distance bounding protocol. In *SecureComm*, pages 67–73, 2005.

[40] W. He, X. Liu, and M. Ren. Location cheating: A security challenge to location-based social network services. In *Distributed Computing Systems (ICDCS), 2011 31st International Conference on*, pages 740–749. IEEE, 2011.

[41] B. Hofmann-Wellenhof, H. Lichtenegger, and J. Collins. Global positioning system. theory and practice. 1993.

[42] T. E. Humphreys, B. M. Ledvina, M. L. Psiaki, B. W. OHanlon, and P. M. Kintner Jr. Assessing the spoofing threat: Development of a portable gps civilian spoofer. In *Proceedings of the ION GNSS international technical meeting of the satellite division*, volume 55, page 56, 2008.

[43] C. T. Inc. Micaz datasheet.

[44] A. Kalamandeen, A. Scannell, E. de Lara, A. Sheth, and A. LaMarca. Ensemble: cooperative proximity-based authentication. In *Proceedings of the 8th international conference on Mobile systems, applications, and services*, pages 331–344. ACM, 2010.

[45] C. H. Kim, G. Avoine, F. Koeune, F.-X. Standaert, and O. Pereira. The swiss-knife rfid distance bounding protocol. In *ICISC 2008*, pages 98–115.

[46] M. Kuhn, H. Luecken, and N. O. Tippenhauer. Uwb impulse radio based distance bounding. In *Positioning Navigation and Communication (WPNC), 2010 7th Workshop on*, pages 28–37. IEEE, 2010.

[47] Z. Lin, D. F. Kune, and N. Hopper. Efficient private proximity testing with gsm location sketches. In *Financial Cryptography and Data Security*, pages 73–88. Springer, 2012.

[48] M. Loh and A. Tam. Wireless smart card and integrated personal area network, near field communication and contactless payment system, Sept. 19 2008. US Patent App. 12/234,499.

[49] H. Massias, X. Serret Avila, and J.-J. Quisquater. Timestamps: Main issues on their use and implementation. In *Enabling Technologies: Infrastructure for Collaborative Enterprises, 1999.(WET ICE'99) Proceedings. IEEE 8th International Workshops on*, pages 178–183. IEEE, 1999.

[50] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone. *Handbook of applied cryptography*. CRC press, 2010.

[51] A. Mitrokotsa, P. Peris-Lopez, C. Dimitrakakis, and S. Vaudenay. On selecting the nonce length in distance-bounding protocols. *The Computer Journal*, page bxt033, 2013.

[52] A. F. Molisch, K. Balakrishnan, D. Cassioli, C.-C. Chong, S. Emami, A. Fort, J. Karedal, J. Kunisch, H. Schantz, U. Schuster, et al. Ieee 802.15. 4a channel model-final report. *IEEE P802*, 15(04):0662, 2004.

[53] T. Moran, R. Shaltiel, and A. Ta-Shma. Non-interactive timestamping in the bounded storage model. In *Advances in Cryptology–CRYPTO 2004*, pages 460–476. Springer, 2004.

[54] J. Munilla and A. Peinado. Distance bounding protocols for rfid enhanced by using void-challenges and analysis in noisy channels. *Wireless communications and mobile computing*, 8(9):1227–1232, 2008.

[55] A. Narayanan, N. Thiagarajan, M. Lakhani, M. Hamburg, and D. Boneh. Location privacy via private proximity testing. In *NDSS*, 2011.

[56] V. Navda, A. Bohra, S. Ganguly, and D. Rubenstein. Using channel hopping to increase 802.11 resilience to jamming attacks. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 2526–2530. IEEE, 2007.

[57] D. Niculescu and B. Nath. Vor base stations for indoor 802.11 positioning. In *Proceedings*

of the 10th annual international conference on Mobile computing and networking, pages 58–69. ACM, 2004.

[58] NIST. Nist randomness beacon, May 2014.

[59] D. Perito and G. Tsudik. Secure code update for embedded devices via proofs of secure erasure. In *Computer Security–ESORICS 2010*. 2010.

[60] M. Poturalski, M. Flury, P. Papadimitratos, J.-P. Hubaux, and J. Boudec. On secure and precise ir-uwb ranging. *Wireless Communications, IEEE Transactions on*, 11(3):1087–1099, 2012.

[61] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. In *Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 32–43. ACM, 2000.

[62] D. Qiu, D. Boneh, S. Lo, and P. Enge. Robust location tag generation from noisy location data for security applications. In *The Institute of Navigation International Technical Meeting*, 2009.

[63] A. Ranganathan, N. O. Tippenhauer, B. Škorić, D. Singelée, and S. Čapkun. Design and implementation of a terrorist fraud resilient distance bounding system. In *ESORICS 2012*, pages 415–432.

[64] K. B. Rasmussen and S. Capkun. Realization of rf distance bounding. In *USENIX Security Symposium*, pages 389–402, 2010.

[65] K. B. Rasmussen, C. Castelluccia, T. S. Heydt-Benjamin, and S. Capkun. Proximity-based access control for implantable medical devices. In *Computer and communications security*, pages 410–419, 2009.

[66] J. Reid, J. M. G. Nieto, T. Tang, and B. Senadji. Detecting relay attacks with timing-based protocols. In *Information, comp. and comm. security*.

[67] Q. Ren and Q. Liang. Throughput and energy-efficiency-aware protocol for ultrawideband communication in wireless sensor networks: a cross-layer approach. *Mobile Computing, IEEE Transactions on*, 7(6):805–816, 2008.

[68] J. Schwenk. Modelling time for authenticated key exchange protocols. In *Computer Security-ESORICS 2014*, pages 277–294. Springer, 2014.

[69] A. Seshadri, M. Luk, A. Perrig, L. v. Doorn, and P. Khosla. Externally verifiable code execution. *Communications of the ACM*, pages 45–49, 2006.

[70] B. Shrestha, N. Saxena, H. T. T. Truong, and N. Asokan. Drone to the rescue: Relay-resilient authentication using ambient multi-sensing. In *Proc. Eighteenth International Conference on Financial Cryptography and Data Security*, 2014.

[71] D. Singelee and B. Preneel. Location verification using secure distance bounding protocols. In *Mobile Adhoc and Sensor Systems Conference*, pages 7–pp, 2005.

[72] D. Singelée and B. Preneel. Distance bounding in noisy environments. In *Security and Privacy in Ad-hoc and Sensor Networks*, pages 101–115. 2007.

[73] B. Song and C. J. Mitchell. Rfid authentication protocol for low-cost tags. In *Wireless network security*, 2008.

[74] M. Strasser, S. Capkun, and M. Cagalj. Jamming-resistant key establishment using uncoordinated frequency hopping. In *Security and Privacy, 2008. SP 2008. IEEE Symposium on*, pages 64–78. IEEE, 2008.

[75] N. Sufyan, N. A. Saqib, and M. Zia. Detection of jamming attacks in 802.11 b wireless networks. *EURASIP Journal on Wireless Communications and Networking*, 2013(1):1–18, 2013.

[76] N. O. Tippenhauer. *Physical-Layer Security Aspects of Wireless Localization*. PhD thesis, Diss., Eidgenössische Technische Hochschule ETH Zürich, Nr. 20344, 2012.

[77] N. O. Tippenhauer and S. Čapkun. Id-based secure distance bounding and localization. In *Computer Security–ESORICS 2009*, pages 621–636. Springer, 2009.

[78] N. O. Tippenhauer, C. Pöpper, K. B. Rasmussen, and S. Capkun. On the requirements for successful gps spoofing attacks. In *Proceedings of the 18th ACM conference on Computer and communications security*, pages 75–86. ACM, 2011.

[79] N. O. Tippenhauer, K. B. Rasmussen, and S. Capkun. Secure ranging with message temporal integrity. *IACR Cryptology ePrint Archive*, 2009.

[80] H. T. T. Truong, X. Gao, B. Shrestha, N. Saxena, N. Asokan, and P. Nurmi. Comparing and fusing different sensor modalities for relay attack resistance in zero-interaction authentication. In *Pervasive Computing and Communications (PerCom), 2014 IEEE International Conference on*, pages 163–171. IEEE, 2014.

[81] VAMPIRE. ebacs: Ecrypt benchmarking of cryptographic systems, Mar. 2009.

[82] A. Varshavsky, A. Scannell, A. LaMarca, and E. De Lara. *Amigo: Proximity-based authentication of mobile devices*. Springer, 2007.

[83] S. Vaudenay, I. Boureanu, A. Mitrokotsa, et al. Practical & provably secure distance-bounding. In *The 16th Information Security Conference*.

[84] J. S. Warner and R. G. Johnston. A simple demonstration that the global positioning system (gps) is vulnerable to spoofing. *Journal of Security Administration*, 25(2):19–27, 2002.

[85] Wikipedia. Ieee 802.11.

[86] Wikipedia. Time.

[87] Wikipedia. Ieee 802.11, Jan. 2015.

[88] B. A. Wilhelm, O. Steely, and A. Gluck. Method and system for using contactless payment cards in a transit system, Nov. 15 2007. US Patent App. 11/940,443.

[89] M. Z. Win and R. A. Scholtz. Impulse radio: How it works. *IEEE Communications letters*, 2(2):36–38, 1998.

[90] R. S.-N. Xifan Zheng and H. Ahmadi. Distance lower bounding. In *Proceedings of 2014 International Conference on Information and Communications Security*. Springer, 2014.

[91] W. Xu, K. Ma, W. Trappe, and Y. Zhang. Jamming sensor networks: attack and defense strategies. *Network, IEEE*, 20(3):41–47, 2006.

[92] W. Xu, W. Trappe, Y. Zhang, and T. Wood. The feasibility of launching and detecting jamming attacks in wireless networks. In *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, pages 46–57. ACM, 2005.

[93] W. Xu, T. Wood, W. Trappe, and Y. Zhang. Channel surfing and spatial retreats: defenses against wireless denial of service. In *Proceedings of the 3rd ACM workshop on Wireless security*, pages 80–89. ACM, 2004.

[94] Y. Zheng, M. Li, W. Lou, and Y. T. Hou. Sharp: Private proximity test and secure handshake with cheat-proof location tags. In *Computer Security–ESORICS 2012*, pages 361–378. Springer, 2012.