A NEW SECURE AUTHENTICATION BASED DISTANCE BOUNDING PROTOCOL

By

Ahmed Raheeq Sultan

Submitted to the Faculty of Department of Information Security Military College of Signals, National University of Sciences and Technology, Islamabad in partial fulfillment of the requirements for the degree of MS in Information Security

July 2020

DECLARATION

I certify that this research work titled "A New Secure Authentication Based Distance Bounding Protocol" is my own work. No portion of this work presented in this dissertation has been submitted in support of another award or qualification either at this institution or elsewhere. The material that has been used from other sources has been properly acknowledged / referred.

_____

Signature of Student

Ahmed Raheeq Sultan

00000203775

ABSTRACT

We have numerous systems being used in daily life where two entities authenticate each other over a range of distance. The distance involved is relatively small, but still attacks were documented. Distance Bounding (DB) Protocol were introduced to cater for the security requirements. The schemes, however are still prone to several threats; mainly the Relay Attack (Terrorist and Mafia Fraud).

In Mafia Fraud, attempts are made to get accepted as the prover either by replaying of messages or by the help a malicious key. This can further be executed by Impersonation Fraud and Man in the Middle attack. In Terrorist fraud, attempt is made to extract the secret from the verifying entity as to get accepted. This is carried out by extracting the key from the message captured or by physically tempering the verifying/ proving entity. Given the nature of the attacks, their mitigation is necessary, and should be achieved as to not put computational overhead on the scheme. There is thus a need for a more secure protocol which can ensure confidentiality, integrity and authentication.

This thesis presents a comprehensive and comparative performance analysis of twelve DB protocols; including the detailed description of the protocol, the means of defence against the attack and explains the means by which the protocol was compromised. Based on the results concluded, a protocol is proposed which incorporates the design elements needed for added security, computationally easy to implement and resistant to most of the threats mentioned. Analysis of the protocol is carried out against the security requirement. We simulated the proposed protocol in Python Language. The analysis yields that the protocol can withstand attacks such as; relay Attack (Terrorist and Mafia Fraud), Replay Attack, Distance Fraud and De – synchronization Attack. The details of how the protocol is secure against these attacks is mentioned in the thesis.

Key Words: *Distance Bounding (DB) Protocols, Attacks, Mitigation, Cryptographic Protocol*

COPYRIGHT STATEMENT

DEDICATION

“The example of those who take allies other than Allah is like that of the spider who takes a home. And indeed, the weakest of homes is the home of the spider; if they only knew.

Indeed, Allah knows whatever thing they call upon; other than Him. And He is the Exalted in Might; the Wise.

And these examples, We present to the people, but none will understand them except those of knowledge.**”**

(Chapter 20: Surah Al-'Ankabut: Ayat 41-43)

Dedicated to my exceptional parents, adored siblings, teachers and friends; whose tremendous support and cooperation led me to this wonderful triumph; and of course, to my beloved country Pakistan.

ACKNOWLEDGEMENT

I am grateful to Allah Almighty for giving me strength to keep going on with this thesis, irrespective of many challenges and troubles. All praises for HIM and HIM alone.

Next, I am grateful to all my family and especially to my parents. Without their consistent support and prayers, this thesis would not have been possible. My sister Aiman Sultan who encouraged me in the early days.

I am very grateful to my Project Supervisor Brig Dr. Imran Rashid who supervised the thesis / research in a very encouraging and helpful manner. I am also grateful to my Co-Supervisor Asst Prof Dr. Fawad Khan who took me under his wing. As supervisor and co-supervisor, their support and supervisions have always been a valuable resource for me.

I am also thankful to committee members who have always guided me with their profound and valuable support that has helped me in achieving my research aims.

I would also thank my ex company commander Maj Bilal Ahmed, for his motivational and inspirational discussions.

Finally, I would like to express my appreciation to all the people who have provided valuable support to my study and whose names I couldn't bring to memory.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF EQUATIONS

ACRONYMS

National Institute of Standards and Technology	NIST
Initialisation Phase	IP
Rapid Bit Exchange Phase	RBEP
Authentication Phase	AP
Mutual Authentication Phase	MAP
Mafia Fraud	MF
Terrorist Fraud	TF
Relay Attack	RA
Denial of Service Attack	DOS
Distance Fraud	DF
De – synchronization Attack	DA
Message Authentication Code	MAC
Public Key Infrastructure	PKI
Basic Public Key Infrastructure	BPKI
One Way Collision Resistant Hash Function	OWCHF
Zero Knowledge Protocol	ZKP
Symmetric Encryption	SE
Pseudo Random Functions	PRF
keyed – Hash Message Authentication Code	HMAC
Hash Function	h
Error Correction Code	ECC
Node Capture Attack	NCA
Physically Unclonable Functions	PUFs
Ultra Wide Band	UWB
Cryptographically Secure Pseudo Random Number Generator	CSPRNG

Chapter 1

INTRODUCTION

With the advancement in technology, new innovations and ideas have been brought into the world. This has brought ease and comfort across the globe but has also increased the chances of threats and theft which effect the overall security of the practice under consideration**.**

Take the example of a scenario where two entities need to communicate over a particular distance. In daily life, there is often a need where one entity needs to verify another before giving access. Take for example the keyless entry system in today's cars. The Electronic Car Unit (abbreviated as ECU) would like to know that the person trying to gain access to the car is no more than a few meters away. For this the ECU needs to determine the maximum limit on the physical distance of the driver.

For better understanding take the example of an E-tag system used in cars. The system need to be sure that the car is near so that it may open the gate. In opening the gate too early, there is a chance of malicious entry. In opening the gate too late and the user would have to wait. The distance bounding protocols introduced in [1] fixes this issue. The first protocol was given in 1993, and was a primitive approach lacking security and other constraint issues.

A basic DB protocol consists of a Tag and a Reader, where the two parties communicate over a range of distance. The whole process is based on exchange of challenge and received bits between the two parties. The time of round trip is calculated which forms the basis of the protocol and enables the verifier to compute an upper-bound on the distance between both the parties.

1.1 Problem Statement

DB Protocols are cryptographic protocols that enable one party; the verifier "V" to verify a second party; the prover "P" [2], which is achieved by the help of the maximum limit applied on the distance between both. This works on the challenge bits sent and then received by the verifier after which the round trip delay time is computed. The prover is

then verified and given access. The process of the bit exchange is prone to security threats and different attacks can be launched against it. This compromises the overall security of the protocol.

Although different researchers have introduced innovations in the already existing schemes, some are still vulnerable to security threats. Most of the protocols cannot be implemented in real time; being computationally heavy in terms of processing, storage, cryptographic primitives applied etc.

According to BBC News, in England and Wales, for the first time in 8 years, 106,000 cars were stolen in 2018 [3]. This thesis aims to design one such protocol incorporating the design elements needed for added security and which is resistant to the attacks presented in the security requirements (Section 2.3).

1.2 Research Objectives

The main objectives of thesis are:

a. The analysis of already made protocols to find out weaknesses and short comings.
b. Study the attacks on well-known protocols and perform the comparative analysis of well-known DB protocols.
c. Design one such protocol incorporating the design elements needed for added security and caters for all the security requirements.
d. Analysis against threats and software simulation and verification of the designed protocol.
e. Analysis of proposed scheme in terms of security and efficiency. Merits and de – merits of the proposed protocol are listed.

1.3 Scope

The scope of the research is:

a. Research of Distance Bounding on RFID, keyless entry and similar devices.
b. The distance involved are very small ranging from 3 – 5 meters.
c. Some attacks are possible even on the most secure protocols. In this research, we will not take them into consideration. We will focus on the security parameters given in Section 2.3.

d. The various generators / functions involved in the protocol is not of our concern. For example; finding a suitable Pseudo Random Function is not the scope of this thesis. Pseudo Random Function, themselves are a separate topic of research.

1.4 Contribution

This thesis will contribute in the following ways:

a. A comprehensive descriptive and comparative performance analysis of twelve DB protocols is presented. This includes the detailed description of the protocol, the means of defence against the attacks, and the means by which the protocol was compromised.
b. A new protocol has been proposed which incorporates the design elements needed for added security.
c. Analysis against threat model and verification based on simulation has been carried out on Python. Results are given in an ideal and attack scenario. Merit and De – merits of the proposed scheme are listed.
d. Research areas for the future are highlighted.

The next chapter will cover point “a”, while the later chapters will cover points “b”, “c”, and “d” in detail.

1.5 Thesis Outline

The basic outline of the thesis is:

Chapter 1: Introduction is given, including the problem statement, research objectives, scope, and contributions.

Chapter 2: This outlines the different functions mentioned in this thesis, along with the explanation of the DB Protocol itself, security requirements, literature review and analysis of existing DB Protocols.

Chapter 3: This presents the proposed protocol and its detailed explanation.

Chapter 4: Threat modelling, security verification and simulation of the proposed protocol is presented.

Chapter 5: Conclusion and the open areas for research are highlighted.

PRELIMINARIES

Before digging into the protocol itself, there are some cryptographic functions that need to be addressed. The literature review is also carried out in this chapter with the comparative analysis. The well – known DB protocols are also shown with each of the phase explained in detail.

2.1 Cryptographic Background

2.1.1 Hash functions "h"

A variable input is given to the hash function, which converts it to a fixed length value output. The values returned by these functions are called hashes.

Given that one knows the hash, it is practically impossible to obtain the input value. These makes the hashes very secure and unbreakable. More of this is given in [4].

Another property which the hash must have, is that by flipping of one bit from the input, the change in the output of the hash should be more than 50%. This is also known avalanche effect [5]. The figure 2.1 depicts the process of generation of hash.

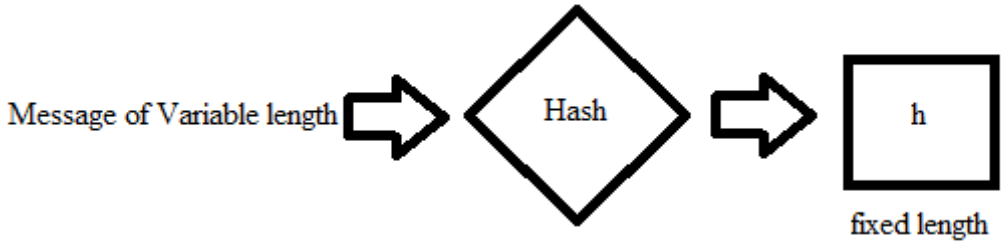

Figure 2.1 Depiction of Hash Function

2.1.2 Message Authentication Code (MAC)

Message Authentication Code (MAC) [6] is a short piece of information, used to determine the authenticity and integrity of the message; i.e., that it came from the actual sender and that it has not been tempered with.

The verifier can detect changes made to the message. “Tag” is also a name given to the Message Authentication Code (MAC). The figure 2.2 shows the generation of MAC on the sender side, and the verification of message on the receiver side, with the help of MAC.

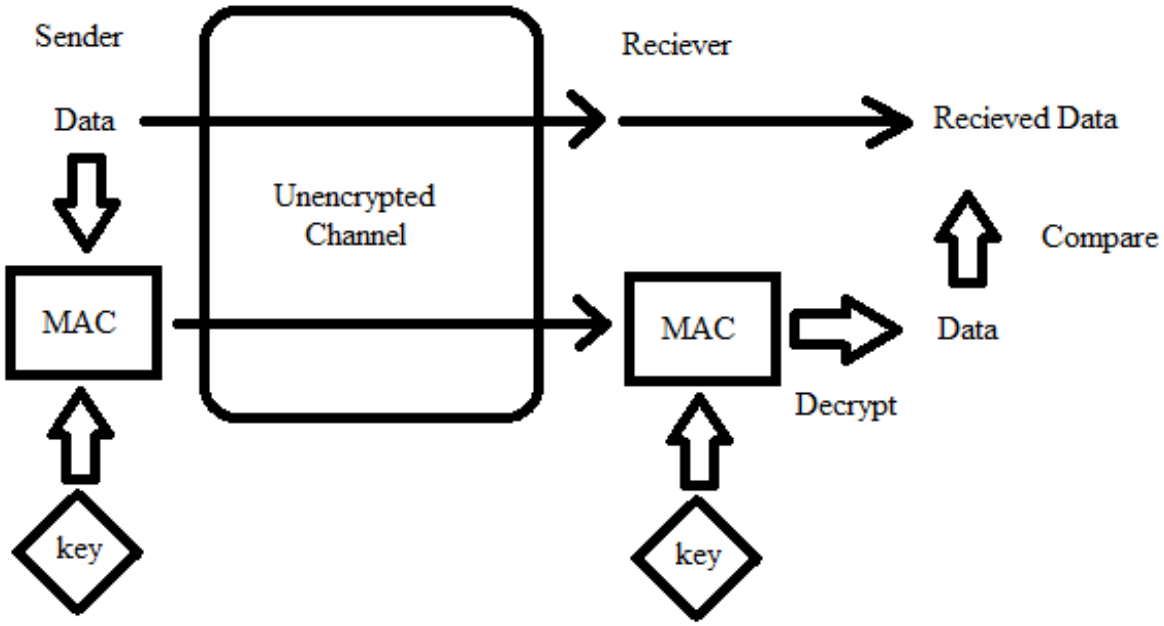

Figure 2.2 Depiction of Message Authentication Code (MAC)

2.1.3 Pseudo Random Function (PRF)

Pseudo Random Function [7] as the name suggests, are functions whose all outputs are random answers (such that they are close to randomness, because absolute randomness is impossible), irrespective of how the inputs are chosen.

Pseudo Random Functions are not be confused with Pseudo Random Generators. The latter generates single random output for random input. The PRF generates random outputs regardless of the input given.

2.1.4 Commitment Schemes (CS)

This scheme allows a party to commit to a certain value, but keeps it hidden from other parties. The value can be revealed at a later stage. Once committed (as the name suggests) the party cannot change the value.

The scheme consists of two algorithms;

- Com = Commit (msg, nonce); function takes a message and a nonce as input and return a commitment.
- Verify (com, msg, nonce); function takes a commitment, message and a nonce as input and return true if values of comm match and false otherwise.

As stated before, two properties should hold;

- Given "com", it is computationally impossible to find the message.
- For an attacker, it is computationally impracticable to find a rogue pair (msg', nonce'), such that;

  comm = comm' where; comm = legitimate commitment
  comm' = rogue commitment

Detailed explanation can be found in [8].

2.1.5 Zero Knowledge Protocol (ZKP)

It is a method in which two parties; the prover proves to the verifier, that he knows a certain value without communicating the value itself. No other information is conveyed. The challenge is for the prover to prove himself without revealing any additional information [9]. Also named as Zero Knowledge Proof.

2.2 Distance Bounding Protocol (DB – Protocol)

As explained before in Section 1.1, these are cryptographic systems, which allows two parties (Verifier and the Prover) to verify each other over a particular distance. The protocols has three phases namely; Initialization Phase, Rapid Bit Exchange Phase and Authentication Phase.

First Step - Initialization Phase: The protocol starts with both the parties sending each other challenge bits (Nonces, Bit String, etc). The next step involves both parties generating their specific bit-sequences using a function (Pseudorandom Function PRF, Hash Function, MAC Algorithm, etc).

Second Step – Rapid Bit Exchange Phase: The verifier send a bit as a challenge to the prover. The prover replies with a response – a bit (based on the bit received from the verifier side). This is iterated "n" times (where n is pre-determined). The verifier then computes the time of the phase.

Third Step – Authentication Phase: The basic protocol in terms of authentication was studied by [10], [11]. Nonce is generated by reader which is used to create Pseudo Random Number, using Pre – shared Pseudo Random Function and key. The nonce and the output of the PRF are concatenated and sent to the tag. The tag (verifying entity) then verifies the string and applies the same process and sends his nonce to the reader. Both in this way authenticate each other, thus given the name "**Mutual Authentication".**

2.3 Security Requirements of Distance Bounding Protocols

The attacks on DB protocols are namely [2]–[11]: Relay attack (Mafia and Terrorist Fraud), Distance Fraud, Impersonation Fraud, Man in the Middle Attack, Collusion Fraud and Node Capture Attack. Their detail is given as follows:

1. **Mafia Fraud (MF)** [21]**,** [22]**:** Adversary between prover and verifier, tries to make the verifier accept him as the prover, taking advantage of the actual prover's position. Man in the middle attack (MIM) is initiated by a malicious rival between actual reader R and tag T. Rogue reader R' interacts with actual tag T, and vice versa. The honest reader R thinks it is communicating with the actual tag T while in real, it is connected with rogue tag T'. However, a tag cannot be impersonated. Only possible when the tag is cooperating with the adversary.
2. **Terrorist Fraud (TF):** It is an extension of Mafia Fraud Attack. With the help of an adversary, the malicious prover gains access via the verifier, but the adversary alone cannot get the access. The Tag T is not legitimate and uses a rogue tag T' to convince reader R of its location. The attack becomes possible when the tag reveals its secret key to the adversary.

   The way to prevent this attack is such that the Rapid Bit Exchange Phase, are amalgamated by means of cryptography. The protocol cannot be split into two discrete segments by the rival. This can be accomplished in two ways:

- Use confidential hardware
- Use well secured private (or symmetric) key during RBEP.

More examples can be found in [23] and [24].

3. **Distance Fraud (DF):** An illegitimate prover; at a certain distance; tries to get access form the verifier.
4. **Impersonation Fraud (IF):** An adversary tries to masquerade as the legitimate prover, and tries to get access from the verifier.
5. **Man in the Middle Attack (MIM):** It is generalized mafia and impersonation fraud [13]. The goal of the attacker is to make the verifier accept the prover with the key *"x"*. *(Key x is known to the attacker).*
6. **Collusion Fraud (CF):** A far away prover with key "x", helps the attacker to make the verifier accept his response. The attacker however cannot launch an attack himself on the protocol later [25], [26].
7. **Mutual Authentication (MA):** Both reader and tag get the conviction that they are communicating with the claimed legitimate entity. (Reader in case if Tag; Tag in case of Reader.
8. **Node Capture Attack (NCA):** Legitimate nodes are physically captured by attacker to extract vital information from them. The attacker can then make his own node clone or use that information as per his will. More examples can be found in [27]–[29].
9. **Relay Attack (RA):** The attacker only relays messages between two parties [30]. The attacker may or may not read or influence messages.
10. **Replay Attack / Playback Attack (PA):** It can be easily described as an inferior version of Man in the Middle Attack (MIM). The attacker re-transmits the legitimate data as per his own choice.
11. **De-synchronization Attack (DSA)** [31]**:** A attack on the RFID system in which the shared key of verifier and prover does not match. This happens because of an attacker jamming the communication. This is usually employed in case of secret key update scheme [32].

Brelurut et al. [33] presented the following theorems:

1. **DF->DH:** A protocol resistant to DF is also resistant to DH.

2. **MIM->MF and IF:** A protocol resistant to MIM attack is also resistant to MF and IF.
3. **CF->TF:** A protocol resistant to CF is also resistant to TF.
4. **DF->TF:** A protocol non-resistant to DF is also non-resistant to TF, with a better success probability.

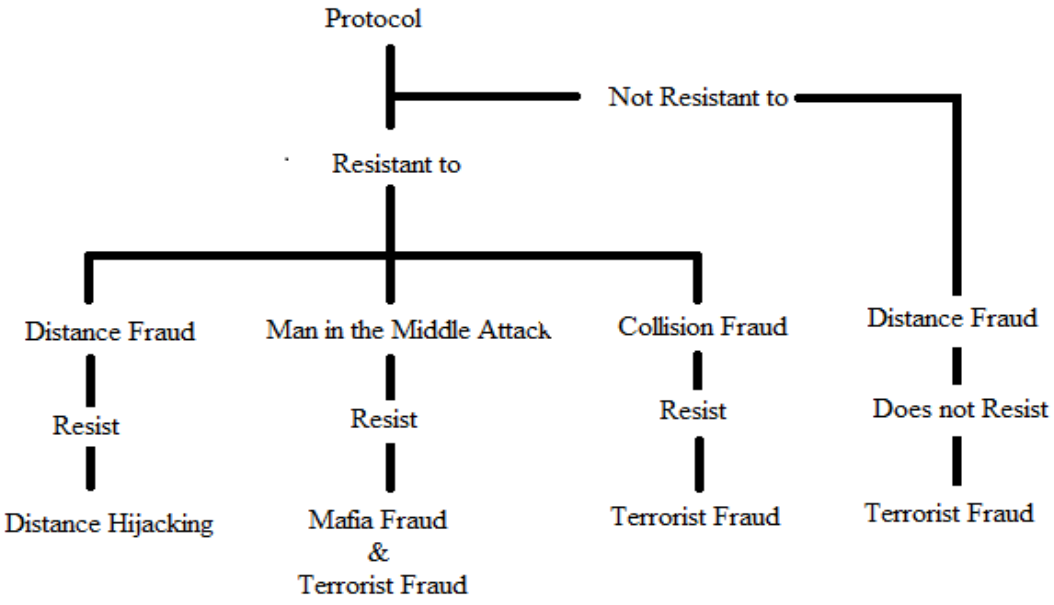

Figure 2.3 Brelurut Theorem

2.4 Literature Review

2.4.1 Prevailing Distance Bounding Protocols

A survey highlighting the security of the DB protocols has been carried out by *Avoine et al.* [34] and *Brelurut et al.* [33], which tells about the different attacks on 23 different DB protocols, countermeasures, and methods of analysis pointing out that cluster based comparison can be modified for better practicality.

A set of mechanisms for secure verification of time encounter between nodes in multi node wireless networks is introduced by *Čapkun et al.* [35]. It was based on one-way hash and Merkle Hash Tree. They were the first to address problem of securing topology and tracking. The introduced mitigation to wormhole attack, securing the routing protocols, cheating detection by topology tracking. Incorporation of challenges for mutual authentication were proposed. The work would have been better if the study was verified by software simulations.

For the first time, integrity and privacy was introduced by *Waters and Felten* [36]. They introduced the concept of location manager; authenticated with Public Key Infrastructure (PKI). They approved exact location of device even when it is held by adversary. In practical application, deployment of the protocol faced issued due to physical factors and ownership issues. The protocol actually traded off security for location proving.

A new protocol was presented by *Nikov and Vauclair* [37]. In their scheme, the resource requirement for the prover are relaxed. They used symmetric techniques, with authenticated nonce (this increased the overall efficiency) and lighter, non – iterative pre – processing phase.

Kardaş et al. [38] introduced Physically Unclonable Function (PUF) which is a digital fingerprint having a unique identity (UI) for a semi-conductor chip. This protocol introduced a very strong adversary, and he can access the tags' volatile memory. It proved that *Sadeghi et al.* [39] is not safe according to this model. The use of PUF enhances the security and privacy of the protocol, making it cost effective. The use of signature provides ideal security against TF. *Tuyls and Batina* [40] presented PUFs to store key; for public-key cryptosystems was used. As all the keys are fabricated at different interval, therefore whole secret key cannot be extracted from the tag. The protocol provides security against TF, MF and DF, which can be further increased by addition of signature in the last stage of the protocol. It is the first paper with $(1/2)^n$ security against all frauds.

The process to avoid relay attacks during authentication was given by *Avoine et al.* [41]. The RFID not only reduces the success probability of the adversary, but also decreases the rounds executed within the protocol.

The problem that replaying of messages will increase the round trip was shown by *Mordohai et al.* [42]. It also shows that extra validation messages after RBEP are not required for false acceptance rate under mafia fraud. Work for future is on multi party authentication in DB protocols.

A modified RIFD DB protocol providing security against mafia and terrorist fraud is evaluated by *Tu and Piramuthu* [43]. The RFID tag reader is vulnerable to MF or TF due to inability of reader to verify location of tag. They proposed the use of multiple readers

and triangulation to minimize relay attack. The protocol is a concatenation of *Brands and Chaum* [1], *Hancke and Kuhn* [44], *Reid et al.* [17].

A unified framework for RFID DB protocol is presented by *Avoine et al.* [12]. Black Box and White Box models are presented and for the test; protocol of *Munilla and Peinado* [45] is used. The framework can be altered to analyse or design DB protocols.

The concept that secret sharing scheme, based on threshold cryptography, can defeat terrorist fraud was presented by *Avoine et al.* [16]. Test protocol of *Hancke and Kuhn* [44] was used to form two types of new protocols; Threshold DB and Thrifty Threshold DB. A protocol of the same kind *Bussard and Bagga* [18] is already there. The same model has also been applied to (Swiss Knife) *Kim et al.* [19].

SKI was introduced, which is first family of lightweight and provably secure DBP was given by *Boureanu et al.* [46]. These are secure even under real time scenarios. Countermeasure against TF and MF are secret sharing with leakage scheme and circular keying with Pseudo Random Functions (PRF). PRF is also used in reuse of keys and to fix common security claims. Further improvements can be to enhance the design to guarantee resistance to TF in presence of noise.

A new protocol with heighted security and lightweight nature was introduced by *Boureanu et al.* [15]. It showed how both the protocols (Swiss Knife) *Kim et al.* [19] and *Avione et al.* [16], are resistant to DF and MF; but susceptible to the new *Hancke - TF attack* [47].

SKI can resist all frauds was given by *Boureanu et al.* [13]. Claims are made that SKI is the first protocol with all accompanying security guarantees.

Distance Hijacking attack on 19 protocols was studied by *Cremers et al.* [48], with countermeasures, modelling and formal analysis. Future works include adding privacy preservations and protecting location privacy.

Different attacks on *Kim et al.* [19], [20] ; showing attacks such as relay attack, terrorist fraud, mafia fraud, dictionary attack in ideal and real life communication channel is presented by *Peris-Lopez et al.* [14]. General design guidelines are given for designing a secure and effective DB protocol.

A new RFID based protocol is introduced by *Hancke and Kuhn* [44] with practical implementation and consideration of noise. The author claims that their protocol is much faster and efficient than *Brands and Chaum* [1].

A new protocol is introduced by *Bussard and Bagga* [18] with implementation and security analysis. Using DB protocols to countermeasure Mafia Fraud and Observer Fraud with implementation of PKI scheme is shown in *Brands and Chaum* [1].

The introduction of void challenges is introduced by *Magagula et al.* [45]. It is a challenge which the reader leaves deliberately to check whether an adversary is trying to get the response from the card in advance. The protocol of *Hancke and Kuhn* [44] was used as basis and then the protocol was modified using the void challenge technique. This decreases the adversary's probability to access the system. Analysis in noisy case, Bit Error Rate (BER) and false alarms is also achieved. The author states that the proposed protocol works better than the original.

Hancke and Kuhn [44] proposed that to achieve DB resolution for RF based devices, ultra wide band (UWB) radio is necessary. But it is of mentioning here that UWB devices have been used to implement DB protocol by *Tippenhauer and C˘apkun* [49] and *Luecken et al.* [50]. *Reid et al.* [17] proposes an alternative solution which detects the relay attack without going for the expensive UWB radio. It is the first symmetric DB protocol. The technique however; is informal, and a formal definition is still an open area of research. The range of the overall system is also reduced by applying this technique which is again an open area for further study.

Singelee' and Preneel [51] presented a low cost DB protocol for noisy environments, which uses binary codes in the rapid exchange phase, to correct bit errors.

C˘apkun and Hubaux [52] dealt with the problem of positioning is wireless systems. Position and Distance Spoofing attacks were executed on positioning techniques to check their resistance. They proposed a mechanism for securing position in wireless devices and sensor networks and verified the same with simulations.

A new type of relay attack was proposed by *Guoheng et al.* [53], which launched the spoofing attack within an effective distance range. The problem was rectified by using time stamping verification; which verify the efficiency and correct flaws in the protocol.

A new distance bound model with three parties (the third being the hardware) has been proposed by *Kilinc and Vaudenay* [54]. The model is called Secure Hardware Model (SHM), in which the prover has the hardware but cannot access it fully. A new protocol is given in sync with the proposed hardware model.

Assuming that the information established from prover can be replayed to launch a terrorist fraud was proposed by Avione et al. [55]. Basic construction for provably secure DB protocols was presented with symmetric key and public key.

Bussard [56] proposed anonymity of the prover by the help of a dedicated scheme, which is an extension of group signatures. Proof of knowledge scheme was applied with cryptographic and distance measuring techniques. A framework for establishing trust based on history was implemented.

In other works, Privacy and information leakage was studied by *Bussard* [57]. The concept of three verifiers was introduced by *C˘apkun and Hubaux* [58], *Shmatikov and Wang* [58], and *Singelee' and Preneel* [59]. Collision attacks were studied by *Chandran et al.* [61], and *Chiang et al.* [62]. The study of DB was studied in RFIDs by *Drimer and Murdoch* [63] and sensor networks by *Meadows et al.* [64], and *C˘apkun and Hubaux* [58].

Electronic equipment was used to execute DB protocols by *Rasmussen and C˘apkun* [65]. A new DB protocol was proposed by Sastry *et al.* [66]. The protocol is based on Ultrasound and wireless radio communication, and can only be used to verify the position of the nodes. Mitigation of wormhole attack was proposed by a new mechanism "Packet Leashes" by *Yih et al*. [67].

A mechanism for securing against spoofing attack has been proposed by *Kuhn* [68]. The reliance on long term shared secret is exempted. Another protocol has been proposed by *Meadows* [69] which uses only a single round in rapid bit exchange phase.

Avoine and Tchamkerten [70] proposed a low complexity protocol without compromising the performance of the protocol. In their protocol, the verifier has the choice to accept or reject msg of identity even in protocol in halted in between.

The next section provide the detailed explanation of 12 existing DB Protocols. The protocols are shown in a tabular manner clearly indicating the numerous functions used in the phases, the key exchanges involved through the RBEP. The overview of each protocol is also given. Another analysis is also presented which shows the computational time taken by the protocol, the use of PKI, attack possibility and resistance, reason for success of the attack. Further study is depicted through tables and figures in the next sections. The charts show all the major phases of the protocol, described in Section 2.2 in detail, with the key exchanges and the nonce communication

2.5 Analysis of Protocols

The descriptive analysis of existing protocols showing the 3 main steps and key exchanges are shown below. Diagrams are made for better understanding.

2.5.1 Cˇapkun and Hubaux [52]

- Pre Shared Key. Length of nonces are pre-determined.

Table 2.1 Cˇapkun et al

Prover	Verifier
Initialization Phase	**Initialization Phase**
• Generate Random Nonce: "reply" • Commit to this Nonce • Send the committed nonce to verifier	• Generate Random Nonce: "start" • Commit to this Nonce • Send the committed nonce to prover
→	
←	
Rapid Bit Exchange Phase	**Rapid Bit Exchange Phase**
• Wait for the start from verifier	• Send start to prover
←	
• Replies with "start ⊕ reply"	
→	• Stops timer
Authentication Phase	**Authentication Phase**
• Compute MAC (key ; start | reply) • Committed value of reply nonce used in initialization phase	
→	• Verify MAC and committed value

2.5.2 **Waters and Felten** [36]

- It is assumed that the verifier has an asymmetric encryption key-pair and that the prover possesses a signature key-pair.
- Length of nonces are pre-determined with security parameter.

Table 2.2 Waters and Felten

Prover	**Verifier**
Initialization Phase	**Initialization Phase**
• Generate two random Nonce: "start" and "reply" • Sign ID with own private key; and after encrypting "start, reply and signature"; with verifier's public key; send it to verifier →	• Verifier decrypt the entire encrypted message. Extract nonces and checks prover's signature • Generate nonce "echo" with length as per security parameter
Rapid Bit Exchange Phase	**Rapid Bit Exchange Phase**
←	• Start timer and send nonce "start and echo" to prover
• Verify that first part of message is "start" and reply with nonce "reply and echo". →	• Stops the timer. Record the round trip latency
Authentication Phase	**Authentication Phase**
	• Check received message. First part is nonce "reply" and second part is nonce "echo"

2.5.3 **Brands and Chaum** [1]

- Pioneer DB Protocol.
- Length of "α, β" are pre-determined with security parameter "k".

Table 2.3 Brands and Chaum

Prover	**Verifier**
Initialization Phase • Generate random $\beta_i \varepsilon \{0, 1\}$ for i = 0,1,…….k	**Initialization Phase** • Generate random $\alpha_i \varepsilon \{0, 1\}$ for i = 0,1,…….k
Rapid Bit Exchange Phase (iterates "k" times)	**Rapid Bit Exchange Phase (iterates "k" times)**
	• Start timer and send α_i prover ←
• Reply with β_i →	
	• Stops the timer.
Authentication Phase	**Authentication Phase**
• Concatenate α_i and β_i; sign with own private key. Send to verifier →	
	• Concatenate α_i and β_i. Verify received signatures by using prover's public key for decryption.

2.5.4 **Hancke and Kuhn** [44]

- Shared secret key "x" is used.
- One way collision resistant hash (OWCRH) function "h" is used.

Table 2.4 Hancke and Kuhn

Prover	**Verifier**
Initialization Phase	**Initialization Phase**
• Generate random nonce r_p and send to verifier →	
	• Generate random nonce r_v and send to prover ←
• Derive two bit strings "m" and "r" from shared key "x" such that: $m \mid r = h(x; r_p \mid r_v)$	• Derive two bit strings "m" and "r" from shared key "x" such that: $m \mid r = h(x; r_p \mid r_v)$
	• Generate random α_i ; i = 0,1,…….k
Rapid Bit Exchange Phase (iterates "k" times)	**Rapid Bit Exchange Phase (iterates "k" times)**
	• Start timer and send α_i to prover ←
• Calculate reply β_i and send; If; $\alpha_i = 0$; $\beta_i = r_i$ If; $\alpha_i = 1$; $\beta_i = m_i$ →	
	• Stops the timer. Verify values of β_i
Authentication Phase	**Authentication Phase**
	• Check received value of β_i. If wrong then stop the session.

2.5.5 **Bussard and Bagga** [18]

- The protocol is public-key based and uses zero-knowledge techniques.
- It is assumed that the prover and verifier have a private key "x" and public key "y" respectively.
- Security parameter "k".

Table 2.5 Bussard and Bagga

Prover	**Verifier**
Initialization Phase • Generate k-bit key "s" • Encrypt private key using symmetric key encryption method (OTP) e = enc (s; x) all have length "k" • Commit to s_i and e_i and send to verifier. → i=1,……..k	**Initialization Phase** • Generate random α_i ; i = 0,1,…….k
Rapid Bit Exchange Phase (iterates "k" times) • Calculate reply β_i and send; If; $\alpha_i = 0$; $\beta_i = e_i$ If; $\alpha_i = 1$; $\beta_i = s_i$ →	**Rapid Bit Exchange Phase (iterates "k" times)** • Start timer and send α_i to prover ← • Stops the timer
Authentication Phase	**Authentication Phase** • Check received value of β_i. If; $\alpha_i = 0$; open respective $\beta_i = e_i$ If; $\alpha_i = 1$; open respective $\beta_i = s_i$ for i = 1,…….k • Zero knowledge protocol executed to obtain private key "x" (of the respective pair "y") and "e" is cipher text of key "s"

2.5.6 **Reid et al** [17]

- Shared secret key "x" is used.
- Security parameter "k".
- Pseudo Random function (PRF) "h" is used.

Table 2.6 Reid et al

Prover	**Verifier**
Initialization Phase	**Initialization Phase**
• Generate random nonce r_p and send to verifier →	• Generate random nonce r_v and send to prover ←
• Derive two bit strings "m" from shared key "x" such that: $m = h(x; r_p \mid r_v)$	• Derive two bit strings "m" and "r" from shared key "x" such that: $m = h(x; r_p \mid r_v)$
• Encrypt shared key using symmetric key method $e = enc(m; x)$ "x" and "e" have length "k"	• Encrypt shared key using symmetric key method $e = enc(m; x)$ x" and "e" have length "k"
	• Generate random α_i ; i = 0,1,…….k
Rapid Bit Exchange Phase (iterates "k" times)	**Rapid Bit Exchange Phase (iterates "k" times)**
	• Start timer and send α_i to prover ←
• Calculate reply β_i and send; If; $\alpha_i = 0$; $\beta_i = e_i$ If; $\alpha_i = 1$; $\beta_i = m_i$ →	
	• Stops the timer. Verify values of β_i
Authentication Phase	**Authentication Phase**
	• Check received value of β_i. If wrong then stop the session.

2.5.7 **Nikov & Vauclair** [37]

- Two Common Shared Secret
  - Distance Authentication Key "K"
  - Seed "R"; both having fixed length k'.
- Security parameter "k".
- Counter "j" for both parties (set at zero when the protocol runs for the first time).
- Pseudo Random function (PRF) "h" is used.

Table 2.7 Nikov & Vauclair

Prover	**Verifier**
Initialization / Pre - Processing Phase • Compute fixed parts of a_i and b_i where i = j+1,……j+k • Compute tags ma_i and mb_i such that $ma_i = h(K; a_i)$ $mb_i = h(K; b_i)$ where i = j+1,……j+k	**Initialization / Pre - Processing Phase** • Computer fixed parts of a_i and b_i where i = j+1,……j+k • Compute tags ma_i and mb_i such that $ma_i = h(K; a_i)$ $mb_i = h(K; b_i)$ where i = j+1,……j+k
Rapid Bit Exchange Phase	**Rapid Bit Exchange Phase** • Choose a i such that i = j+1,……j+k • Send i, ma_i to prover and start timer.
• Compare received values of ma_i with pre-computed one (from the Initialization Phase) • If values are correct, send i, mb_i to verifier	• Stops the timer
Authentication Phase • Increase counter "j" with "k".	**Authentication Phase** • Compare received values of mb_i with pre-computed one (from the Initialization Phase) • If values are correct, compute the round trip time • Increase counter "j" with "k".

2.5.8 **Munilla and Peinado** [45]

- Shared Secret Key "K"
- Security parameter "k".
- Hash Function "h" is used with generates "3n" bits.
- These 3n bits will be split into 3 parts "P", v_0 and v_1. Each "P", v_0 and v_1 will have 4 bits each.

Table 2.8 Munilla and Peinado

Prover	**Verifier**
Initialization Phase • Generate random nonce N_a and send to verifier • Derive values as follows: $\{H\}^{3n} = h(K, N_a, N_b)$ $\{P\} = \mathbf{1011} = H_1 \| H_2 \| H_3 \| H_4$ $\{v_0\} = \mathbf{10x0} = H_5 \| H_6 \| H_7 \| H_8$ $\{v_1\} = \mathbf{01x0} = H_9 \| H_{10} \| H_{11} \| H_{12}$ Values are written for understanding only. Each bit of {P} shows whether void challenge is sent or not. For P = 0, void challenge is being executed. "x" shows that response will not be sent	**Initialization Phase** • Generate random nonce N_b and send to prover • Derive values as follows: $\{H\}^{3n} = h(K, N_a, N_b)$ $\{P\} = \mathbf{1011} = H_1 \| H_2 \| H_3 \| H_4$ $\{v_0\} = \mathbf{10x0} = H_5 \| H_6 \| H_7 \| H_8$ $\{v_1\} = \mathbf{01x0} = H_9 \| H_{10} \| H_{11} \| H_{12}$ Values are written for understanding only. Each bit of {P} shows whether void challenge is sent or not. For P = 0, void challenge is being executed. "x" shows that response will not be sent
Rapid Bit Exchange Phase (iterates "k" times) • Check α_i and; If; $\alpha_i = 0$; send bit corresponding bit of v_0 If; $\alpha_i = 1$; send bit corresponding bit of v_1	**Rapid Bit Exchange Phase (iterates "k" times)** • Start timer and send α_i ; i = {0 or 1} to prover • Stops the timer. Store v_0 and v_1
Authentication Phase • Compute h (K, v_0, v_1) and send to verifier.	**Authentication Phase** • Check received value.

2.5.9 Singel´ee and Preneel [51]

- An Error Correction Code (ECC) (n, k) which can correct "x" bits; during Rapid Bit Exchange Phase is agreed upon by both parties. For this the Hamming Distance d_{min} of the binary code is $\mathbf{x = (d_{min} - 1) / 2}$
- Message Authentication Code (MAC) is used.

Table 2.9 Singel´ee and Preneel

Prover	**Verifier**
Initialization / Pre - Processing Phase • Generate k – bits r_1,……., r_k where r = {0,1} • Apply ECC (n, k) to get; r_1,……., r_k, r_{k+1}, r_n • Commit (r_1 |……..| r_n) and send to verifier	**Initialization / Pre - Processing Phase** • Generate k – bits s_1,……., s_k where r = {0,1} • Apply ECC (n, k) to get; s_1,……., s_k, s_{k+1}, s_n • Commit (s_1 |……..| s_n) and send to prover.
Rapid Bit Exchange Phase • Compute $\alpha_1 = r_1$ and send to verifier • Compute $\alpha_i = r_i \oplus \beta_{i-1}$ and send to verifier • Compute $\alpha_n = r_n \oplus \beta_{n-1}$ and send to verifier	**Rapid Bit Exchange Phase** • Compute $\beta_1 = s_1 \oplus \alpha_1$ and send to prover • Compute $\beta_i = s_i \oplus \alpha_i$ and send to prover • Compute $\beta_n = s_n \oplus \alpha_n$ and send to prover • Stops the timer • Round trip in each round is measured. Maximum round trip is measured and
Authentication Phase $s_i = \alpha_i \oplus \beta_i$ • Use ECC to correct errors. • Concatenate bits such that; $y_A = MAC_k (r_1 \mid s_1 \ldots\ldots \mid r_k \mid s_k)$ and send to verifier • Verify commitment of Initialization phase with received. • Verify y_B	**Authentication Phase** $r_i = \alpha_i \oplus \beta_{i-1}$ • Use ECC to correct errors. • Concatenate bits such that; • $y_B = MAC_k (s_1 \mid r_1 \ldots\ldots s_k \mid r_k)$ and send to prover • Verify commitment of Initialization phase with received. • Verify y_A

2.5.10 Tu and Piramathu [43]

- Shared secret key "x" is used.
- Hash Function "h" is used.

Table 2.10 Tu and Piramathu

Prover	Verifier
Initialization / Pre - Processing Phase • Generate random nonce r_B and send to verifier • $x \oplus r_A$; and send to verifier.	**Initialization / Pre - Processing Phase** • Generate random nonce r_A and send to prover • $x \oplus r_B$; and send to prover.
• Compute; $k = h(r_A, x \mid r_B)$ $c = k \oplus x$ for (u, v) = {(r_A , r_A), (r_B, r_B), (r_B , r_A), ($r_A \oplus r_B$, $r_A \oplus r_A$)} for i = 1 to n / 4	• Compute; $k = h(r_A, x \mid r_B)$ $c = k \oplus x$ for (u, v) = {(r_A , r_A), (r_B, r_B), (r_B , r_A), ($r_A \oplus r_B$, $r_A \oplus r_A$)} for i = 1 to n / 4
Rapid Bit Exchange Phase	**Rapid Bit Exchange Phase**
	• Pick q_i where q = {0,1} and send to prover
• Reply with C_i such that: If; $q_i = 0$; send $C_i = k_i$ If; $q_i = 1$; send $C_i = c_i$	
	• Stop timer. Check C_i and Δt_i. If C_i , Δt_i invalid, abort protocol.
Authentication Phase	**Authentication Phase**
	• Calculate $k_{temp} = h(u, x \| v)$ and send to prover
• Verify $k_{temp} = h(u, x \| v)$. • If invalid, abort the protocol.	

2.5.11 **Kim et al** [19]

- Shared secret key "x" is used. Choosing of random "d" with Hamming Weight "m". Pseudo Random function (PRF) "*f*" is used. CB is system wide constant

Table 2.11 Kim et al

Prover	**Verifier**
Initialization / Pre - Processing Phase	**Initialization / Pre - Processing Phase**
	• Pick random nonce N_A and "d" such that; H(d) = m and send to prover
• Pick random nonce N_B and send to verifier and compute	
$a = f_x (C_B, N_B)$; "a" is temporary key	
$Z_0 = a$ $Z_1 = a \oplus x$	
• Compute m – bit vectors R_i^0 and R_i^1	
For i = 1 to "m" j : index of the next 1 of $d_{(2)}$ $R_i^0 = Z_j^0$, $R_i^1 = Z_j^1$	
Rapid Bit Exchange Phase	**Rapid Bit Exchange Phase**
• Reply with r_i such that: If; $c_i = 0$; send R_i^0 If; $c_i = 1$; send R_i^1	• Pick $c_i = \{0,1\}$ and send to prover
	• Stop timer. Δt_i equal or less than t_{max}.
Authentication Phase	**Authentication Phase**
• Compute $t_B = f_x (c_1 \ldots\ldots c_n, ID, N_A, N_B)$ • Send t_B , $c_1 \ldots\ldots c_n$ to verifier.	• Check ID and compute R^0 and R^1 • Compute error of each round; err_c = count $\{c_i /= c_i'\}$ c_i' = error bit err_r = count $\{c_i = c_i'$ and $r_i /= Zi^{ci}\}$ err_t = count $\{c_i = c_i'$ and $\Delta t_i > t_{max}\}$ • If $err_c + err_r + err_t >= T$ reject and stop the protocol
Mutual Authentication Phase	**Mutual Authentication Phase**
• Compute and compare t_A	• Compute $t_A = f_x (N_B)$ and send to prover

2.5.12 **Avoine and Tchamkerten** [70]

- Shared secret key **"x"**
- Two security parameters **n = α.k** and **m**
- (PRF) **"*f*"** is used; whose output size is at least **m + α (2k+1 − 2) bits**
- Decision Tree used in rapid bit exchange phase.

Table 2.12 Avoine and Tchamkerten

Prover	**Verifier**
Initialization / Pre - Processing Phase	**Initialization / Pre - Processing Phase**
	• Pick random nonce N_A and send to prover
←	
• Pick random nonce N_B and compute • Compute; $a = f_x (N_B, N_A)$; and send to verifier	
→	
$R_i^0 = Z_j^0$, $R_i^1 = Z_j^1$	• Compute; $a = f_x (N_B, N_A)$
Rapid Bit Exchange Phase	**Rapid Bit Exchange Phase**
i = 0,1,……. α j = 0,1,…….k	i = 0,1,……. α j = 0,1,…….k
• Reply with r_j^i from the corresponding value of the tree, such that:	• Pick c_j^i and send to prover
←	
r_j^i = node (c_1^i ,…….., c_j^i)	
→	
	• Stop timer.
Authentication Phase	**Authentication Phase**
	• Concatenate and check all m – bits received. • i should be greater than 1 but greater than α • j should be greater than 1 but greater than k • Δt_j^i equal or less than t_{max}.

2.6 Tabular Analysis and Performance Metrics

The comparative performance and security analysis of the twelve well known protocols was carried out for a better understanding and working of the overall DB protocol. The conclusion drawn can be used to strengthen the protocol that we intend to design in the later stages of this study. The table 2.13 shows the comparative study on twelve DB protocols in a tabular manner.

Another table is the study of attacks a protocol is resistant to, the attacks which are possible and the analysis of how an attack is possible. The analysis also tells whether the issue has been resolved by a further study or not. The table 2.14 shows the analysis in a tabular form.

The comparison is based on several points; security dependence, pre – processing capability, cryptographic primitives used, defence and vulnerability to known attacks, error resistance to channel error, privacy preservation to outsiders, total computational cost of the entire protocol (based on the constrained functions used), and the verification time taken by each scheme.

Table 2.13 Comparison of Existing DB Protocols

S.No.	Author	Security	Pre Processing Capability	Number of Phases	Cryptographic Primitives	Defence Against	Vulnerable to	Resistance to Channel Errors	Privacy Preservation Against Attackers	Mutual Authentication	Total Computation (Both sides)	Verification Time (s)
1.	**CH** [52]	b	h	3	No	MF	TF & NCA	Yes	N/A	Yes	2(Commit) + 2(MAC)	N/A
2.	**WF** [36]	N & X.509	No	3	Yes "S"	N/A	DOS & NCA	N/A	Yes	Yes "IDs"	4(PKI)	N/A
3.	**BC** [1]	n	No	3	No	MF & TF	TF & NCA	No	N/A	No	2(BPKI)	N/A
4.	**HK** [44]	s	h	3	No	MF	TF & NCA	Yes	N/A	No	2(OWCRHF)	2.51
5.	**BB** [18]	k	No	3	Yes	MF, TF & DF	N/A	Yes	Yes "S"	Yes	1(PKI) + 1(ZKP)	N/A
6.	**Reid et al** [17]	x	PRF	3	No	MF & TF	NCA	Yes	No	No	2(PRF) + (SE)	2.74
7.	**NV** [37]	R	Yes	3	No	MF	TF & NCA	No	N/A	No	4(PRF) + 2(HMAC)	N/A
8.	**MP** [45]	x	PRF	3	Yes "S"	RA	NCA	Yes	N/A	Yes	n(h) n = number of bits	3.24
9.	**SP** [51]	(n , k)	No HD	3	No "ECC"	MF	TF & NCA	Yes	N/A	Yes	4(ECC) + 2(MAC)	N/A
10.	**TP** [43]	L	h GSK	3	No	MF & TF	TF (SK) & NCA	No	N/A	Yes K(temp)	4(h)	N/A
11.	**Kim et al** [19] **MA**	s	PRF	4 "MAP"	No	MF, TF & RA	NCA	Yes	Yes	Yes	3(PRF) (1 PC) + (2 O)	N/A
	WMA	s	PRF	3	No	MF, TF & RA	NCA	Yes	Yes	No	2(PRF) (1 PC) + (1 O)	2.92
12.	**ATP** [70]	s	PRF	3	No	MF & TF	NCA	Yes "DT"	N/A	No	2(PRF)	2

List of Abbreviations used in Table 2.13

Waters and Felten	WF	Swiss Knife	SK
C˘apkun & Hubaux	CH	IP Initialisation Phase	IP
Avoine & Tchamkerten	ATP	Rapid Bit Exchange Phase	RBEP
Brands & Chaum	BC	Authentication Phase	AP
Hancke & Kuhn	HK	Mutual Authentication Phase	MAP
Bussard & Bagga	BB	Mafia Fraud	MF
Nikov & Vauclair	NV	Terrorist Fraud	TF
Munilla & Peinado	MP	Relay Attack	RA
Singel´ee & Preneel	SP	Denial of Service Attack	DOS
Tu & Piramathu	TP	Distance Fraud	DF
Length of bits	b	Message Authentication Code	MAC
Length of Nonce	N	Public Key Infrastructure	PKI
Location Manager	X.509 Certificate	Basic Public Key Infrastructure	BPKI
Security Parameter	“n” & “k”	One Way Collision	
Shared Secret	“s”	Resistant Hash Function	OWCRHF
Shared Key	“x” & “m”	Zero Knowledge Protocol	ZKP
Seed	“R” of length “k”	Symmetric Encryption	SE
Error Correction Code	(n , k)	Pseudo Random Functions	PRF
Long Term Key	“L”	keyed – Hash Message	
Hamming Distance	HD	Authentication Code	HMAC
Signatures	S	Hash Function	h
Generate Session Key	GSK	Error Correction Code	ECC
Pre computed	PC	Node Capture Attack	NCA
Online	O	Without Mutual Authentication	WMA
Decision Tress	DT	Mutual Authentication	MA

Table 2.14 Attack Description on DB Protocols

S. No.	Author / Name of Protocol	Description
1.	**Cˇapkun et al** Secure Tracking of Node Encounters in Multi-hop Wireless Networks	Solved the problem of mutual authentication in **Brands and Chaum.** Made **MAD (Mutual Authentication Distance-bounding)**. Non resilient to bit error, with which adversary can collect different entities in the protocol to guess the key.
2.	**Waters et al** Secure Private Proof of Locations	Adversary can block the device from communicating to the outside or place a **buffer** to make it look like that the device is far away from the **Location Manager**. Proposed **Public Key Cryptography** in **DB**, which is not applicable in small devices like **RFID**.
3.	**Brands and Chaum (1993)** Distance bounding Protocol	**Oldest protocol** in the list. Security and privacy parameters were not researched at that time.
4.	**Hancke and Kuhn** RFID Distance Bounding Protocol	Secret Key is **"K"**. Nonce **"Nv"** is sent by reader. The **"h (K, Nv)"** is generated and then split into registers **"R0"** and **"R1"**. These **"R0"** and **"R1"** can be collected by the adversary to calculate the key **"K"**. Solved by **Reid et al.**
5.	**Bussard and Bagga** Distance Bounding Proof of Knowledge to Avoid Real Time Attacks	Proposed **Public Key Cryptography** in **DB**, which is not applicable in small devices like **RFID**.
6.	**Reid et al** Detecting Relay Attacks with Timing-Based Protocols	Solved the problem of mutual authentication faced by **Hancke and Kuhn.** Suggested intermingling of **"R0"** and **"R1"** with key **"K"**; **$R0=Enc_{R1}(K)$**. This lacked privacy and sends identities without protection.
7.	**Nikov & Vauclair** Yet Another Secure Distance Bounding Protocol	**"K"** is key. Sequences **"ai"** and **"bi"** are calculated. Then **"mai = h (K, ai)"** and **"mbi = h (K, bi)"** is calculated. These can be extracted to calculate key **"K".**
8.	**Munilla and Peinado** Distance bounding protocols	Used Void challenges to reduce success probability of adversary without using signing messages.

	for RFID enhanced by using void challenges and analysis in noisy channels	Uses **three physical states (0, 1 and void)** which is difficult to implement.
9.	**Singel´ee and Preneel** Distance Bounding in Noisy Environment	Error Correction Code used to correct errors. Trusted hardware used but TF still possible due to non – use of symmetric key during fast exchanges.
10.	**Tu and Piramathu** RFID Distance Bounding Protocol	Adversary toggles value of challenge bit. If accepted by the reader than the answer transmitted it right, save it. If reader refuses the challenge bit answer is the toggled value of the answer. Calculate key "x" from these iterations. Use of one reader increases the success probability of an adversary to launch mafia fraud. Can be decreased by using multiple readers through triangulation.
11.	**Kim et al** The Swiss-Knife RFID Distance Bounding Protocol	None as per the literature
12.	**Avoine and Tchamkerten** An Efficient Distance Bounding RFID Authentication Protocol: Balancing False-Acceptance Rate and Memory Requirement	Used decision trees to reduce success probability of adversary. The acceptance or rejection a message of identity depends upon the verifier in case the protocol gets halted.

Summary

In this chapter, an overview of the DB protocol is given, explaining each step in detail. Security requirements are presented, and existing DB protocols are evaluated based on their performance against the mode of bit exchange, defence and vulnerability to attacks**.**

THE NEW AUTHENTICATION BASED DB PROTOCOL

The new proposed protocol is presented and explained in this chapter. Two protocols were made, the second being a modified and lighter version of the first protocol. Each step of the final protocol is discussed in detail, with the preliminaries of the protocol, the key exchanges and the tabular form.

3.1 Introduction

The basis of our protocol is formed by the ideas taken from various renowned protocols; MAP1 of *Bellare and Rogaway* [10], MAP1.1 of *Guttman et al.* [11], protocols of *Kim et al.* [19], *Hancke and Kuhn* [44] and *Munilla and Peinado* [45]. The introduction of symmetric schemes, pre – shared main and transient key; as well as a bit string are major amendments. One important thing to note is that the protocol presented is very low on computational cost.

The notion of mutual authentication has also been introduced, but is a constrained step for the prover and can be exempted if needed. The details of these will be discussed later on.

3.2 The Protocol

The protocol like all DB protocols is divided into three primary steps namely; Initialization phase, rapid bit exchange phase, authentication phase. These phases have been discussed in Section 2.2; but for specifically this protocol, each phase will be discussed in detail.

Before discussing the protocol some preliminaries need to be addressed. All DB protocols involving the round trip time; work on the following assumptions [22]:

- The noise delay and cryptographic operation delays should not slow down the protocol.
- Speed of light will be used to calculate tmax.
- 1 bit is sent for the calculation of round trip time.
- During the Rapid Bit Exchange Phase, no other computation is occurring.

All protocols [1], [17]–[20], [37], [43], [44], [51], [70] work on the supposition given above. The exceptions are the Munilla, Ortiz and Peinado [45] [71] which use three states; 0, 1 and void.

Taking the 2nd point of the assumption under consideration, then the value of tmax for a distance of 6 meters (two way distance 12 meters) and speed of light 299,792,458 meters per second, comes out to be:

Equation 3.1 Calculation of time "t" for two way journey

$$time = \frac{distance}{speed} = \frac{d}{c} = \frac{12\ meters}{299792458\ meters\ per\ second} = 4\ x\ 10^{-8}\ seconds$$

This is the time without involving computational time of the whole mechanism of the scheme.

3.2.1 1st Protocol

The first protocol is shown in Table 3.1. The protocol was an unusual approach and carried computational overhead but was secure due to the extensive use of PKI, symmetric encryption and arrays. The summary of the protocol is:

- Pre-Shared private key "x" and transient key "m"
- The protocol is public-key based. It is assumed that the prover owns a private key "PPR" and public key "PPU" and the verifier has the corresponding private key "VPR" and public key "VPU".
- Security parameter "k".

Therefore it was modified and computational strained part was changed with secure, but lighter schemes.

Table 3.1 1st Protocol

Prover	Verifier
Pre – Computation Phase • Pick pre-shared; k – bit string N_A with each bit value stored successively in array such as; $[Y_0]^n = N_A$ $[Y_1]^n = N_A \oplus m$ n = 0,1,…….k • Encrypt private key using symmetric key encryption method p = enc (m; x) Temporary key all have length "k" • sign = enc ($\mathbf{P_{PR}}$ | p) • **E** = enc ($\mathbf{V_{PU}}$ | sign) **Initialization Phase** • Send **"E** = enc ($\mathbf{V_{PU}}$ | sign)**"** to verifier • Decrypt only the message **"E"** to check **"sign"** of the verifier for mutual authentication **Rapid Bit Exchange Phase (iterates "k" times)** **n = position of entry in array initialized at 0** • Calculate reply β_i and send; If; $\alpha_i = 0$; $\beta_i = [Y_1]^i$ If; $\alpha_i = 1$; $\beta_i = [Y_0]^i$ **Authentication Phase (optional)**	**Pre – Computation Phase** • Pick pre-shared; k – bit string N_A with each bit value stored successively in array such as; $[Y_0]^n = N_A$ $[Y_1]^n = N_A \oplus m$ n = 0,1,…….k • Encrypt private key using symmetric key encryption method p = enc (m; x) Temporary key all have length "k" • sign = enc ($\mathbf{V_{PR}}$ | p) • **E** = enc ($\mathbf{P_{PU}}$ | sign) **Initialization Phase** • Decrypt and check for **"p"** • Send **"E** = enc ($\mathbf{P_{PU}}$ | sign)**"** to prover • Generate random α_i ; i = 0,1,…….k **Rapid Bit Exchange Phase (iterates "k" times)** • Start timer and send α_i to prover • Stops the timer, Δt_i equal or less than t_{max}. **Authentication Phase (optional)** • Generate N_A by the same method and store in array. • Check received value of β_i. If; $\alpha_i = 0$; open respective $\beta_i = [Y_1]^i$ If; $\alpha_i = 1$; open respective $\beta_i = [Y_0]^i$ for i = 1,…….k If values are true then allow access.

3.2.2 2nd Protocol

The goal of the protocol is mutual authentication (authentication of both the verifier by the prover and vice versa). The basic assumption for the protocol are stated in Section 3.2 and Section 4. The scope of the protocol is in Section 1.3.

The protocol uses minimal computational power and resources, excluding the need to use Public Key Infrastructure (PKI) and various other heavy encryption standards. The scheme uses Hashlib and Hmac in python.

The protocol used two pre-shared secret keys "x" and "m"; uses Pseudo Random Function (PRF) in the Pre – Computational and Initialization Phase.

A security parameter or bit size is defined in the protocol (we will take this value as 512 for example). The rest of the protocol is explain step wise below.

3.2.2.1 Pre – Computational Phase

Both Verifier and Prover generate random nonces N_v and N_p respectively. These nonces are used to generate k-bit strings "a" and "b" on the prover and verifier side respectively. The prover uses the shared secret "x" while the verifier uses shared secret "m".

3.2.2.2 Initialization Phase

Both parties send their nonces to each other. They use the nonces of each other to generate "b" and "a" at the prover and verifier side respectively. Only this time the prover uses the shared secret "m" while the verifier uses shared secret "x".

The verifier chooses a k-bit random α_i.

3.2.2.3 Rapid Bit Exchange Phase

The verifier starts the timer and sends first bit of the α_i to the prover. The prover replies with his response β_i in the following manner:

If $\alpha_i = 0$; $\beta_i = a_i$ and if $\alpha_i = 1$; $\beta_i = b_i$

Although the value of α_i is random, the prover will be on the lookout for a specific sequence of bits, e.g.; 1011. After completion of this sequences 1011, the immediate next bit of αi will determine the response from the prover. The replies are reversed.

If $\alpha_i = 0$; $\beta_i = b_i$ and if $\alpha_i = 1$; $\beta_i = a_i$

The rest of the protocol will follow as usual. The significance of this sequence is that it allows the prover to verify the verifier without putting computational load on the protocol.

If mutual authentication is not required this part can be replaced with the simple relaying of a challenge bit α_i from the verifier, resulting in a response bit β_i from the prover side.

3.2.2.4 Authentication Phase

The verifier will verify the responses βi. He will compute the error in the transmission, which includes checking the received value of βi with the pre – computed one. As an example for understanding we will assume that β be the bit received after the RBEP and β' be the original (intended) values (pre-computed on the verifier side). Error will be calculated as:

$$err\beta = count \{ \beta i \; != \beta i' \text{ and } \Delta ti > tmax \}$$

where; βi = response bit ; $\beta i'$ = pre - computed bit

If the value error is greater than a pre – computed threshold "T" then stop the protocol.

The tabular representation of the protocol is given in table 3.2.

Summary

This chapter presents the design of proposed protocol explaining how each step of DB is carried out in detail. The assumptions taken into consideration, the preliminaries of the scheme, and the key / bit exchanges in each step are explained in detail. The tabular form of the scheme is also presented for the understanding of the reader.

Table 3.2 2nd Protocol

Prover	Verifier
Pre – Computation Phase • Generate random nonce N_p. Derive k-bit string "a" from shared key "x": $a = f(x; N_p)$	**Pre – Computation Phase** • Generate random nonce Nv. Derive k-bit strings "b" from shared key "x": $b = f(m; N_v)$
Initialization Phase • Send nonce N_p to verifier.	**Initialization Phase** • Send nonce N_v to prover.
• Derive k-bit string "b" from received nonce N_v and shared key "m" such that: $b = f(m; N_v)$	• Derive k-bit string "a" from received nonce N_p and shared key "x" such that: $a = f(x; N_p)$ • Generate random α_i ; i = 0,1,…….k-1.
Rapid Bit Exchange Phase (iterates "k" times)	**Rapid Bit Exchange Phase (iterates "k" times)** • Start timer and send α_i to prover
• Calculate reply β_i and send; If; $\alpha_i = 0$; $\beta_i = a_i$ If; $\alpha_i = 1$; $\beta_i = b_i$	• After completion of this sequences 1011, the immediate next bit of α_i will determine the response from the prover. The rest of the protocol will follow as usual.
• If prover detects the pre shared sequence, for the immediate next bit, calculate reply β_i and send; If; $\alpha_i = 0$; $\beta_i = b_i$ If; $\alpha_i = 1$; $\beta_i = a_i$	
Authentication Phase	**Authentication Phase** • Verify values of βi. Compute error: $err\beta = count \{ \beta i \mathrel{!}= \beta i' \text{ and } \Delta t_i > t_{max} \}$ where βi = response bit ; $\beta i'$ = pre - computed bit • If $err\beta >= T$ (threshold of error) reject and stop protocol.

THREAT MODEL AND SECURITY ANALYSIS

This chapter explain the threat model of the DB Protocols. It explain how the protocol is secure against the security requirements given in Section 2.3. The scheme is checked against all attacks stated before. The simulation the protocol and the code is also presented with the results of the run shown with the efficency of the scheme.

4.1 Assumptions

Before analysing the protocol, the following **assumptions** are made:

1. The legitimate Prover and Verifier are denoted by P and V, while the rogue Prover and Verifier are denoted by P' and V'.
2. Statistical Attacks like brute force attacks are possible even on the most secure encryption standards like DES and AES. We will not explain them in much detail as it falls out of the scope of this research. Also keep in mind that for k = 512, there exists 2^{512} combinations for αi. Brute forcing the RBEP with each combination is impractical and useless for the assailant.
3. The pre-shared secret is only present with the verifier and the prover and there is no way for an attacker to extract them other than by means of a Node Capture Attack (NCA). This means that a collision fraud attack isn't possible.

4.2 Threat Analysis

4.2.1 Mafia Fraud (MF)

Keeping in mind that the only way to achieve full protection against mafia fraud attack is to:

1. Use PKI.
2. Use Zero knowledge protocol.

Both of them being computationally heavy and therefore cannot be applied in this domain. Thus a clever approach is needed.

In a scenario where Actual Prover P and Verifier V are not close to each other, it is impossible for P' and V', to relay messages between them without considerate delay (which would cease the protocol). The use of random αi also prevents replay attack

4.2.1.1 Pre Computation and Initialization Phase

As the function used in Pre Computation and Initialization Phase is pseudo-random, therefore guessing of any bit of "a" and "b" by the attacker is negligible. The probability is further minimized by the use of different keys for "a" and "b". This further achieves randomness..

4.2.1.2 Rapid Bit Exchange Phase

For the Rapid Bit Exchange Phase, the Actual Prover "P" looks for a specific sequence (for example: 1011). When this sequences is completed, in the next consecutive bit only, the reply from the Actual Prover P is reversed (as seen in the protocol).

4.2.1.3 Authentication Phase

If in any case if the attacker is using him own pair of a' and b'; then the probability that he will send the specific sequence of α_i in RBEP is very low (lower than (1/2) n as depicted by Hancke and Kuhn), and becomes even lower due to use of random α_i in each RBEP. Also the probability becomes even lower than he can respond correctly (send correct β_i) with the order reversed. The checking of corresponding bits by the verifier V and calculating error further reduces the attackers' chances.

The same can be applied to Impersonation Fraud (IF) and Man in the Middle Attack (MIM) (more info on this in Section 2.3).

4.2.2 Terrorist Fraud (TF)

Let's remember that to prevent this attack is such that the Rapid Bit Exchange Phase, are mingled by means of cryptography. The protocol cannot be split into two discrete segments by the rival. This can be accomplished in two ways:

1. Use confidential hardware
2. Use well secured private (or symmetric) key during RBEP.

Both of these steps are computationally constrained and slows down the protocol. In simple words, the attacker should not be able to achieve the information that the Actual Verifier

V holds, which is the shared key "x" and transient key "m". Coming over to the protocol, let Actual Prover P be far from the Actual Verifier V and close to the rogue Prover P'. He relays the pair ai and bi as per the protocol to the Rogue Prover P', independent of the value of α_i (as the Actual Verifier V isn't close by). If the verifier is close by, then the attack becomes replay attack (this will be discussed later). Even if P' possess all the values of "a_i" and "b_i", and relays it to the Actual Verifier V, even then guessing the shared secret "x" and "m" is impossible because:

1. The probability that the pair "a_i" and "b_i" are in sync with α_i while communicating with V is very low (very less than $(1/2)^n$). Also for 512 bits, the combination of α_i becomes 2^{512}. This means that guessing the sequences is near to impossible.
2. Even if some of the values do go in sync with the challenges, they will not be able to sustain the error threshold, and will be filtered. This will also only give very less and ineffective information regarding the shared secrets.
3. Both function encrypting the nonces use a separate key "x" and "m".
4. Independently "a" and "b" cannot be used to obtain the information that the attacker seeks.
5. Also as the Verifier V does not reject the value of β_i, therefore the "ith" position remains secret to the attacker.
6. The prover will be on the lookout for the specific sequence which will be used by the prover as a way of authenticating the verifier. Given that this sequence isn't received in the entire Rapid Bit Exchange Phase, it will raise suspicions to the prover. Also if the prover's reply is not reverse, this will alert the verifier.

4.2.3 Distance Fraud (DF)

For the attacker to execute a distance fraud attack, the value of β_i should be responded in advance by the rogue Prover P', for which he needs to choose the respond at random and send it to the Actual Verifier V.

The probability that the β_i chosen by the P' is matching the actual β_i is very low; keeping in mind that not only are these value pseudo-random, but also use different keys for randomness. The reversing of β_i in the Rapid Bit Exchange Phase further minimizes the chances of a match. If the value of α_i bits for the RBEP is 512, then the combinations

possible become 2^{512}. Guessing these many combination is impossible for the rogue Prover P' in the given time without much delay.

The error checking in the Authentication Phase will again make the correctness of the value (the attack), insignificant.

Also if it is resistant to DF; therefore according to *Brelurut et al.* [33], it is secure against Distance Hijacking (DH) as well.

4.2.4 Node Capture Attack (NCA)

The threat of theft is possible in all devices. The tag or reader; if stolen can be can be read by a chip / RFID reader and necessary information can be extracted. Therefore the following measures should be taken:

1. Keep the verifier (reader in case of an RFID) in a secure place. For example, in a key less entry system used in cars, the reader (ECU) is inside the vehicle in the dashboard and is well protected by lock and key. If a malicious entity gains entry into the vehicle, even then accessing reader (ECU) is a difficult task, usually involving the breakage of the dashboard panel at the back of the steering wheel.
2. Keep the prover (tag in case of RFID or key fob in case of keyless entry system) on your person while in the vicinity of the verifier (reader). The tag should be kept secure even when not in use.

4.2.5 Mutual Authentication

The verifier authenticates the prover in the last phase (Authentication Phase) where the error is computed, time of the protocol is checked and access is given.

Older schemes either lacked mutual authentication, or the ones that did; involved the use of signatures which made the protocol computationally heavy or impractical.

In scheme employed, the prover is on the lookout for a specific pre-shared. This specific sequence authenticates the verifier (detail can be found in Section 3.2.2). This provides only Pseudo – Authentication as the sequence can come in any of the combinations, but even then it is better than having no authentication at all. This also is a means of double verifying the prover as the response of this sequences in the later phase authenticates the prover.

4.2.6 Relay Attack

It is impossible for P' and V', to relay messages between themselves and towards the legitimate parties without considerate delay (which would cease the protocol). There is also a chance of error during transmission. Also the time of the RBEP Δt_i is checked in the Authentication Phase i.e., $\Delta t_i =< t_{max}$ (standard time for protocol run).

4.2.7 Replay Attack

Take an example where Actual Prover P and Actual Verifier V are close to each other, and an attacker posing as MIM. He can capture the values of β_i and then replay them at a later time.

The issue is solved in the RBEP. The value of αi is random and therefore the response which the verifier wants is also random (response β_i depends upon the value αi). There is a 1/2 probability that the first value of α_i may match (either 0 or 1). But this probability becomes unimportant with the bit size of 512, the combinations of αi being 2^{512}, and as we move from the second bit to the last (512th bit).

4.2.8 Noise Error

To cater for noise errors, one of the possible solution is to increase the number of round in the Rapid Bit Exchange Phase, keeping the time factor below the required delay threshold T.

Another solution is to divide the number of bits of αi into smaller chunks. If the bit size of the αi is 512 bits then we can make 8 chunks of 64 bit each. The value of error ($err\gamma$) for each individual chunk will be checked and then percentage correctness can be calculated. Acceptable error should be no more than 38%.

4.2.9 De-synchronization Attack

The chances of De-synchronization Attack are very less because:

1. The keys are pre shared, not updated and remain the same.
2. The distance involved is very less (few meters).

4.3 Simulation and Results

The simulation of the protocol was executed using Python language on Python 3.7.5 version. The code of the protocol is given in table 4.1. Comments are added against each line for better understanding. The protocol uses libraries of “hmac” and ‘hashlib”. Keyed hash message authentication code (HMAC) is used to encrypt the nonces with the respective secret key, with a Hash Function. (SHA -512 is used)

The code is already mentioned below. The main body calls the function of time, main encryption function, validation and calculation of error. The efficiency is also calculated in the main body of the code. In the secondary functions all three phases of the DB scheme are executed. Encryption is carried out and random numbers are generated. The challenge bits and response bits communication is shown. A verifier function compares the value of each bit of received βi, with that of the pre computed βi and error count is generated. For the sake of example (and for testing we are giving the value of received βi ourselves). Error percentage is calculated for the code.

The execution of the protocol is shown in Figure below. Each figure shown the time taken and the value of βi.

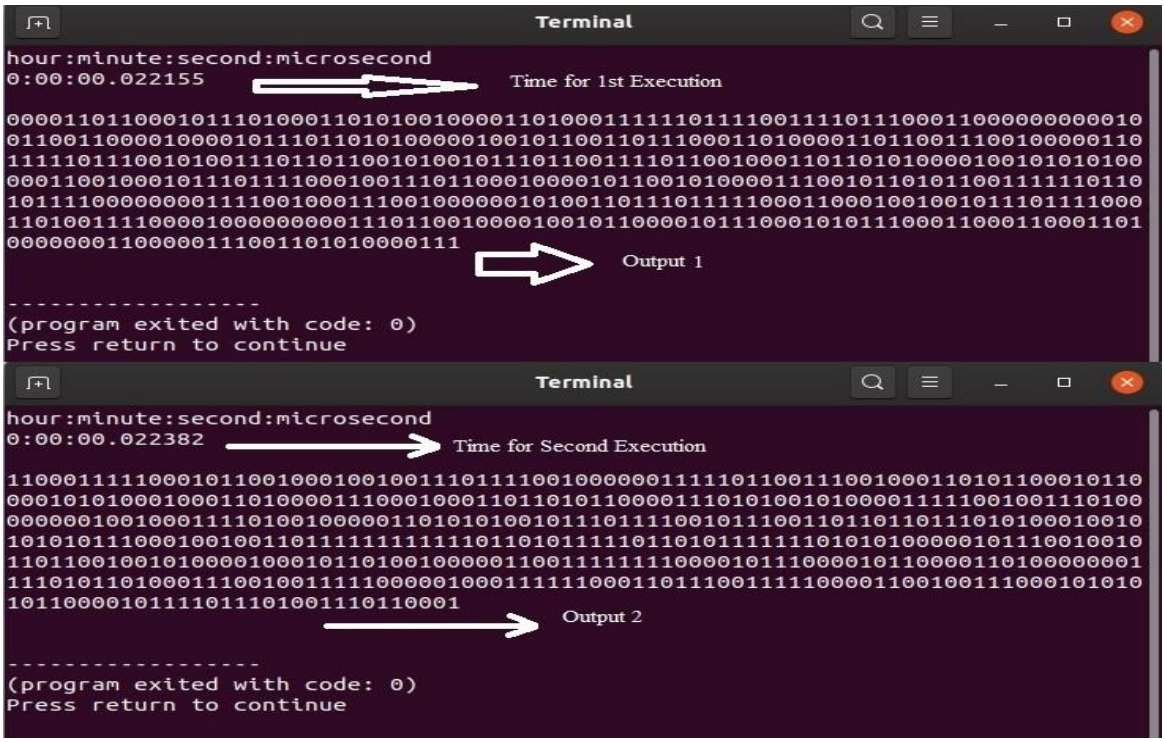

Figure 4.1 Execution of the Protocol under No Attack Scenario

4.3.1 Attack Scenario

Relay attack is initiated on the protocol where bit transmitted are stored and replayed in the next run of the protocol.

The received value of βi, as stated before is self-given, as this determines the quality of the code. To show the attack, we capture the entire value of βi of the last run of the code is saved and used as a received βi in the next run. The percentage error as shown comes out to be 47.85 % with error count at 245 bits out of 512. This means that nearly half of the bits are incorrect.

Similarly for the next run, the error percentage rises to 50.39 % with error count at 258 bits out of 512. This shows that the scheme is resilient to any change in the bits an can detect the error with greater efficiency. This is depicted in Figure 4.3

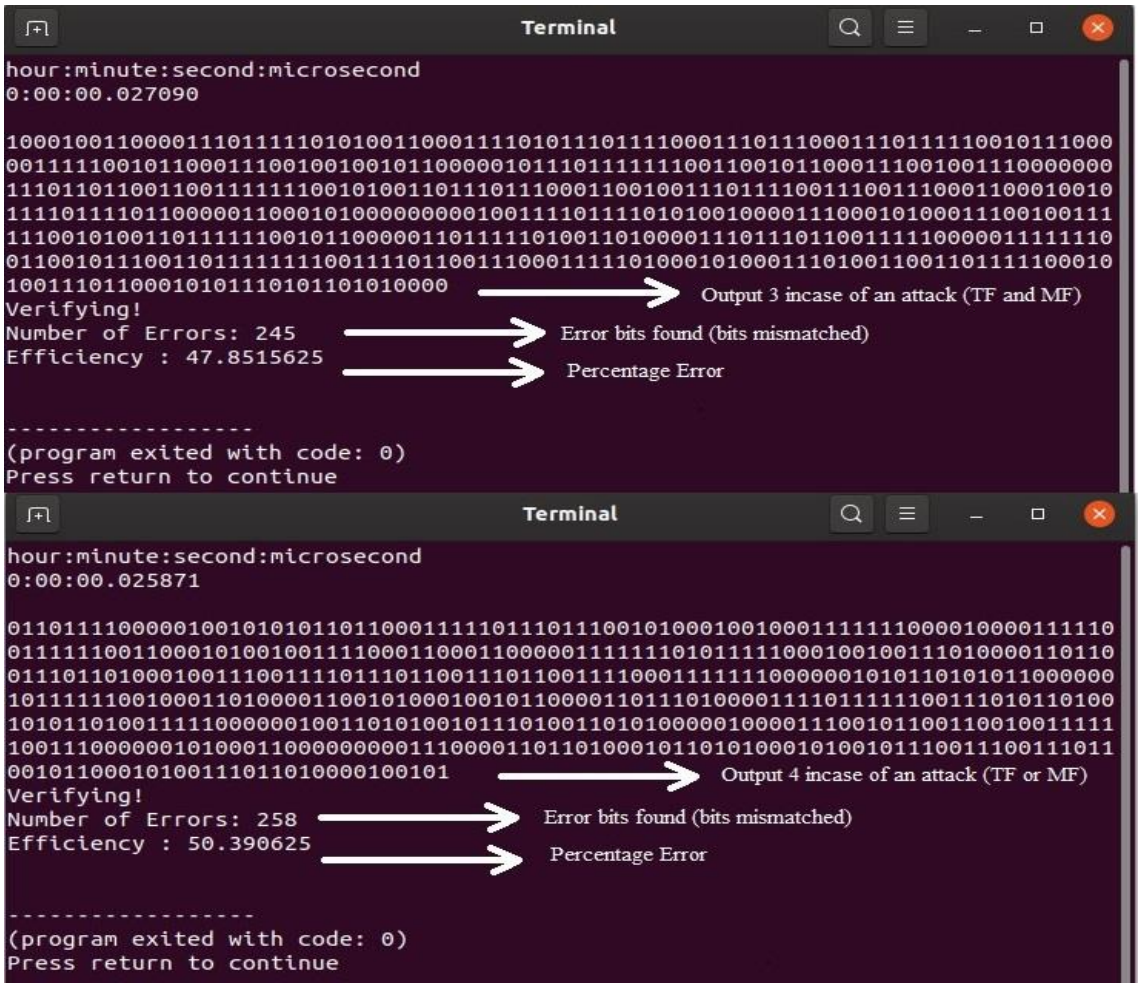

Figure 4.2 Execution of the Protocol under Relay (TF or MF) Attack Scenario

Table 4.1 Python Code for 2nd Proposed Protocol

Code	Explanation
#!/usr/bin/env python3 import hmac, hashlib, random, os, datetime from bitstring import BitArray	Libraries
def enc():	Pre Computation and Initialization Phase
bits_size = 512#bits	Bit size define – 512 bits
byte_size = int(bits_size / 8)#bytes	Bits converted to bytes
x = os.urandom(byte_size)	Generate random key "x"
m = os.urandom(byte_size)	Generate of random key "m"
Np = os.urandom(byte_size)	Generate Nonce "Np"
Nv = os.urandom(byte_size)	Generate Nonce "Nv"
hmac_code = hmac.new(key=x, msg=Np, digestmod=hashlib.sha512) a = hmac_code.digest()	Encrypt nonce "Np" with "x" to get random number "**a**" (sha512 is used)
hmac_code = hmac.new(key=m, msg=Nv, digestmod=hashlib.sha512) b = hmac_code.digest()	Encrypt nonce "Nv" with "m" to get random number "**b**" (sha512 is used)
ai = os.urandom(byte_size)	Generate random number "αi" – 512 bit.
betai = ""	
for i in range(len(BitArray(ai).bin)): if BitArray(ai).bin[i] == '0':	Rapid Bit Exchange Phase Starts
betai = betai + BitArray(a).bin[i]	Sending challenge bit "αi"; getting response bit of "a" or "b".
else: betai = betai + BitArray(b).bin[i] return betai	Values of "a" or "b" stored in "βi".
def verifier(a,b):	Authentication Phase
count = 0 for i in range(len(a)):	Verify each received bit of "βi" against pre-computed bits "βi"
if a[i] != b[i]: count = count + 1	Count the bits not matching pre-computed bits of βi.
return count	Return the number of miss-matched bits.

def main():	Main Function
begin_time = datetime.datetime.now()	Compute time of protocol
e = enc()	Calling of encryption function (PCP, IP and RBEP)
received_e = "11110111000010000000000000101101011001110000010110100101101000001101010011100001010010100010011101010101010111111000011011100111010111011110010101100001101011011010010100100101100110101001001110100101111111100000111001110100110111111111101101001111000100101011111001111011101110100011011000111110010111110100100010011011011010010011110100101110111001100111111111001110101110110001111001100111010100001000111000100111000101010010011011110110100011001010101110111010100110000000100011111010011010011111010101100 0"	Self-given value of received βi (from prover side)
print("hour:minute:second:microsecond")	Time of protocol
print(datetime.datetime.now() - begin_time) print()	
print(e)	Print output bit string of "βi"
print("Verifying!")	Print Verifying
count = verifier(e, received_e)	Call validation function (AP)
print("Number of Errors: {}".format(count))	Print number of miss-matched bits.
efficiency = count/len(e) * 100	Determine percentage error.
if efficiency>70 print("Efficiency : {}".format(efficiency))	Check percentage error against a given threshold.
else: print("Protocol Rejected")	Accept or reject Protocol
return	
if __name__ == "__main__": main()	End

4.4 Merits and demerits of the Scheme

4.4.1 Merits

The proposed protocol has the following qualities:

- It has the capability of pre-processing which means that the initial generation of the nonce and the encrypted numbers “a” and “b” can be carried out beforehand.
- The protocol offers defence against most of the attacks.
- It is resistance to channel errors and preserves the privacy of the protocol.
- The total computation is of the scheme is 2(PRF) + (EC).

4.4.2 Demerits

On the other hand, the shortcomings are:

- It does not include Public Key Infrastructure (Public and Private key)
- Error correction code is not applied due to computational and cost overhead. The protocol can only detect the error. It lacks the capability of correcting them.
- The scheme is still vulnerable to noise errors, node capture and de – synchronization attack.

4.5 Comparison with existing DB Protocols

The table 4.2 shows the comparison of both out schemes with the existing DB protocols. The table is an extension of table 2.13.

The first protocol was not analysed for frauds as it had considerable overhead. It was modified to the second protocol which was analysed for attacks and error resistance. Any errors faced during the RPEP are detected. The protocol has privacy preservation from outsiders and does not reveal secret keys to the attacker. The possibility of a Man in Middle Attack exists where the attacker can only sniff the traffic, but cannot relay it over a long distance because that would cause delay in propagation time. The total computation involves the use of two PRFs, one on each side in the Initialization Phase and Error Check towards the verifier side in the Authentication Phase.

Table 4.2 Comparison of Proposed Schemes

S.No.	Protocol	Security	Pre Processing Capability	Number of Phases	Cryptographic Primitives	Defence Against	Vulnerable to	Error resistance	Privacy Preservation Against Attackers	Mutual Authentication	Total Computation (Both sides)	Verification Time (s)
1.	**1st Proposed Scheme**	k	PRF and arrays	3	PKI & S	N/A	N/A	Yes	N/A	No	2(E) + 3(PKI)	N/A
2.	**2nd Proposed Scheme**	k	PRF	3	PRF & h	MF, TF, DH, DF and PA	NCA	Yes	Yes	Optional	2(PRF) + (EC).	2

Key

Length of Security Parameter	k	Pseudo Random Function	PRF
Signatures	S	Mafia Fraud	MF
Terrorist Fraud	TF	Distance Fraud	DF
Playback Attack	PA	Man In Middle Attack	MIM
Node Capture Attack	NCA	De – synchronization Attack	DA
Error Check	EC	Public Key Infrastructure	PKI

Summary

The chapter carries out the security analysis of the proposed scheme. The assumptions while carrying out the analysis are explained at the very start of this chapter. The protocol is validated not only by threat modelling but also by software simulation on python. The code is explained step by step with the protocol in ideal and attack scenario. Finally the merits and demerits of the scheme are listed.

Chapter 5

CONCLUSION AND FUTURE WORK

5.1 Conclusion

DB protocol enable to parties validation over a distance. This offer ease and technological superiority but also arises security problems with it. This thesis has listed some of the security requirements for an efficient and attack resilient protocol. The literature review of several protocol in practice has been carried out, highlighting the bit exchange, attack possible and mitigation achieved (if any), on the DB scheme. Furthermore a protocol is proposed which offer low computation and is resistant to the security requirements. The claim is validated; by threat modelling and by software analysis. Results for protocol run is different time scenarios are shown. The merits and de-merits of the protocols are listed. The comparison of the proposed protocol is carried out with the existing DB protocol in literature.

5.2 Future Work

Open areas for research in future (but not limited to) are the following:

- Exploration of more metrics to analyse DB protocols.
- Software analysis of the DB schemes presented in literature review, with threat modelling and formal analysis.
- Verification of the proposed protocol on tools other than used in this research.
- Implementation of Error Correction Code (ECC) on the proposed protocol.
- Hardware implementation of the proposed protocol.

BIBLIOGRPAHY

[1] S. Brands and D. Chaum, "Advances in Cryptology — EUROCRYPT '99," *Adv. Cryptol. — EUROCRYPT '93*, vol. 1592, pp. 344–359, 1999.

[2] S. Capkun, K. El Defrawy, and G. Tsudik, "Group distance bounding protocols," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 6740 LNCS, pp. 302–312, 2011.

[3] "New cars 'can be broken into in 10 seconds.'" [Online]. Available: https://www.bbc.com/news/business-49273028.

[4] R. Sobti and G. Geetha, "Cryptographic Hash functions - a review," *Int. J. Comput. Sci. Issues*, no. December, 2012.

[5] Y. M. Motara and B. Irwin, "SHA-1 and the Strict Avalanche Criterion," *2016 Inf. Secur. South Africa - Proc. 2016 ISSA Conf.*, pp. 35–40, 2016.

[6] D. J. Bernstein, "The Poly1305-AES Message-Authentication Code," pp. 32–49, 2005.

[7] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby, *Pseudorandom generator from any one-way function*, vol. 28, no. 4. 1999.

[8] A. a. Al-saggaf and H. S. Acharya, "A Fuzzy Commitment Scheme," p. 4, 2008.

[9] L. C. Guillou and J. J. Quisquater, "A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 330 LNCS, pp. 123–128, 1988.

[10] M. Bellare and P. Rogaway, "Entity authentication and key distribution," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 773 LNCS, pp. 232–249, 1994.

[11] J. D. Guttman, F. J. Thayer, and L. D. Zuck, "The faithfulness of abstract protocol analysis: Message authentication," *J. Comput. Secur.*, vol. 12, no. 6, pp. 865–891, 2004.

[12] G. Avoine, M. A. Bingöl, S. Kardaş, C. Lauradoux, and B. Martin, "A framework for analyzing RFID distance bounding protocols," *J. Comput. Secur.*, vol. 19, no. 2, pp. 289–317, 2011.

[13] I. Boureanu, A. Mitrokotsa, and S. Vaudenay, "Towards secure distance bounding," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 8424 LNCS, pp. 55–67, 2014.

[14] P. Peris-Lopez, J. C. Hernandez-Castro, C. Dimitrakakis, A. Mitrokotsa, and J. M. E. Tapiador, "Shedding Light on RFID Distance Bounding Protocols and Terrorist Fraud Attacks," pp. 1–21, 2009.

[15] I. Boureanu, A. Mitrokotsa, and S. Vaudenay, "Secure & Lightweight Distance-Bounding," pp. 1–17.

[16] G. Avoine, C. Lauradoux, and B. Martin, "How secret-sharing can defeat terrorist fraud," *WiSec'11 - Proc. 4th ACM Conf. Wirel. Netw. Secur.*, pp. 145–155, 2011.

[17] J. Reid, J. M. G. Nieto, T. Tang, and B. Senadji, "Detecting relay attacks with timing-based protocols," *Proc. 2nd ACM Symp. Information, Comput. Commun. Secur. ASIACCS '07*, pp. 204–213, 2007.

[18] L. Bussard and W. Bagga, "Distance-bounding proof of knowledge to avoid real-time attacks," *IFIP Adv. Inf. Commun. Technol.*, vol. 181, pp. 223–238, 2005.

[19] C. H. Kim, G. Avoine, F. Koeune, F. X. Standaert, and O. Pereira, "The Swiss-Knife RFID distance bounding protocol," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5461, pp. 98–115, 2009.

[20] C. H. Kim and G. Avoine, "RFID distance bounding protocols with mixed challenges," *IEEE Trans. Wirel. Commun.*, vol. 10, no. 5, pp. 1618–1626, 2011.

[21] Y. Desmedt, "NoMajor security problems with the 'Unforgeable' (Feige)-Fiat-Shamir proofs of identiy and how to overcome them," *Proc. Secur.*, pp. 15–17, 1988.

[22] Y. Desmedt, C. Goutier, and S. Bengio, "Special uses and abuses of the fiat-shamir passport protocol," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif.*

Intell. Lect. Notes Bioinformatics), vol. 293 LNCS, pp. 21–39, 1988.

[23] A. Ranganathan *et al.*, "Terrorism in distance bounding: Modeling terrorist-fraud resistance," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7954 LNCS, pp. 414–431, 2013.

[24] M. Fischlin and C. Onete, "Terrorism in distance bounding: Modeling terrorist-fraud resistance," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7954 LNCS, pp. 414–431, 2013.

[25] T. Xie, F. Liu, and D. Feng, "Fast collision attack on MD5," *IACR ePrint Arch. Rep.*, vol. 104, p. 17, 2006.

[26] K. Schramm, G. Leander, P. Felke, and C. Paar, "A collision-attack on AES combining side channel- and differential-attack," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 3156, pp. 163–175, 2004.

[27] C. Lin and G. Wu, "Enhancing the attacking efficiency of the node capture attack in WSN: A matrix approach," *J. Supercomput.*, vol. 66, no. 2, pp. 989–1007, 2013.

[28] P. Tague and R. Poovendran, "Modeling Node Capture Attacks in Wireless Sensor Networks Invited Paper," *Electr. Eng.*, pp. 1221–1224, 2008.

[29] M. Conti, R. Di Pietro, L. V. Mancini, and A. Mei, "Emergent properties: Detection of the node-capture attack in mobile wireless sensor networks," *WiSec'08 Proc. 1st ACM Conf. Wirel. Netw. Secur.*, pp. 214–219, 2008.

[30] R. Silberschneider, T. Korak, and M. Hutter, "Access Without Permission : A Practical RFID Relay Attack."

[31] N. W. Lo and K. H. Yeh, "De-synchronization attack on RFID authentication protocols," *ISITA/ISSSTA 2010 - 2010 Int. Symp. Inf. Theory Its Appl.*, pp. 566–570, 2010.

[32] J. Park, S. Jung, N. W. Lo, and K. H. Yeh, "Shared secret key update scheme between RADIUS server and access point using PUFs," *Proc. 2017 4th Int. Conf.*

Comput. Appl. Inf. Process. Technol. CAIPT 2017, vol. 2018-Janua, pp. 1–5, 2010.

[33] P. L. Agn`es Brelurut, David Gerault, "Survey of Distance Bounding Protocols and Threats," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9482, pp. 268–276, 2016.

[34] G. Avoine *et al.*, "Security of Distance-Bounding : A Survey," *ACM Comput. Surv.*, vol. 51, no. 5, 2018.

[35] S. Čapkun, L. Buttyán, and J. P. Hubaux, "SECTOR: Secure tracking of node encounters in multi-hop wireless networks," *Proc. 1st ACM Work. Secur. Ad Hoc Secur. Ad Hoc Sens. Networks (in Assoc. with 10th ACM Conf. Comput. Commun. Secur.*, pp. 21–32, 2003.

[36] B. R. Waters and E. W. Felten, "Secure, Private Proofs of Location," p. 11, 2003.

[37] V. Nikov and M. Vauclair, "Yet another secure distance-bounding protocol," *SECRYPT 2008 - Int. Conf. Secur. Cryptogr. Proc.*, pp. 218–221, 2008.

[38] S. Kardaş, M. S. Kiraz, M. A. Bingöl, and H. Demirci, "A novel RFID distance bounding protocol based on physically unclonable functions," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7055 LNCS, pp. 78–93, 2012.

[39] S. Kardaş, S. Çelik, M. Yildiz, and A. Levi, "PUF-enhanced offline RFID security and privacy," *J. Netw. Comput. Appl.*, vol. 35, no. 6, pp. 2059–2067, 2012.

[40] P. Tuyls and L. Batina, "RFID-Tags for Anti-counterfeiting," pp. 115–116, 2006.

[41] G. Avoine, C. Floerkemeier, and B. Martin, "RFID distance bounding multistate enhancement," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5922 LNCS, pp. 290–307, 2009.

[42] P. Mordohai *et al.*, "Mutual Authentication," *Encycl. Biometrics*, vol. 10, no. 2, pp. 999–999, 2009.

[43] Y. Tu and S. Piramuthu, "RFID Distance Bounding Protocols," *1st Int. EURASIP*

Work. RFID Technol., 2007.

[44] G. P. Hancke and M. G. Kuhn, "An RFID distance bounding protocol," *Proc. - First Int. Conf. Secur. Priv. Emerg. Areas Commun. Networks, Secur. 2005*, vol. 2005, pp. 67–73, 2005.

[45] P. A. Jorge Munilla, "Distance bounding protocols for RFID enhanced by using void-challenges and analysis in noisy channels," no. January 2011, pp. 1414–1428, 2012.

[46] I. Boureanu, A. Mitrokotsa, and S. Vaudenay, "Practical and provably secure distance-bounding," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7807, pp. 248–258, 2015.

[47] G. P. Hancke, "Distance-bounding for RFID: Effectiveness of 'terrorist fraud' in the presence of bit errors," *2012 IEEE Int. Conf. RFID-Technologies Appl. RFID-TA 2012*, pp. 91–96, 2012.

[48] C. Cremers, K. B. Rasmussen, B. Schmidt, and S. Capkun, "Distance hijacking attacks on distance bounding protocols," *Proc. - IEEE Symp. Secur. Priv.*, pp. 113–127, 2012.

[49] N. O. Tippenhauer and S. Čapkun, "ID-based secure distance bounding and localization," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5789 LNCS, pp. 621–636, 2009.

[50] M. Kuhn, H. Luecken, and N. O. Tippenhauer, "UWB impulse radio based distance bounding," *Proc. 2010 7th Work. Positioning, Navig. Commun. WPNC'10*, pp. 28–37, 2010.

[51] D. Singelée and B. Preneel, "Distance bounding in noisy environments," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 4572 LNCS, pp. 101–115, 2007.

[52] S. Čapkun, "Secure positioning in wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 2, pp. 221–232, 2006.

[53] G. Wei, H. Zhang, and Y. Wang, “A new relay attack on distance bounding protocols and its solution with time-stamped authentication for RFID,” *Wuhan Univ. J. Nat. Sci.*, vol. 21, no. 1, pp. 37–46, 2016.

[54] H. Kılınç and S. Vaudenay, “Formal analysis of distance bounding with secure hardware,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 10892 LNCS, no. MiM, pp. 579–597, 2018.

[55] G. Avoine *et al.*, “A terrorist-fraud resistant and extractor-free anonymous distance-bounding protocol,” *ASIA CCS 2017 - Proc. 2017 ACM Asia Conf. Comput. Commun. Secur.*, pp. 800–814, 2017.

[56] L. Bussard, “Trust establishment protocols for communicating devices,” 2004.

[57] K. B. Rasmussen and S. Čapkun, “Location privacy of distance bounding protocols,” *Proc. ACM Conf. Comput. Commun. Secur.*, no. i, pp. 149–159, 2008.

[58] S. Čapkun and J. P. Hubaux, “Secure positioning of wireless devices with application to sensor networks,” *Proc. - IEEE INFOCOM*, vol. 3, pp. 1917–1928, 2005.

[59] V. Shmatikov and M. H. Wang, “Secure verification of location claims with simultaneous distance modification,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 4846 LNCS, no. 1, pp. 181–195, 2007.

[60] D. Singelee and B. Preneel, “Location verification using secure distance bounding protocols,” *2nd IEEE Int. Conf. Mob. Ad-hoc Sens. Syst. MASS 2005*, vol. 2005, pp. 834–840, 2005.

[61] N. Chandran, V. Goyal, R. Moriarty, and R. Ostrovsky, “Position-based cryptography,” *SIAM J. Comput.*, vol. 43, no. 4, pp. 1291–1341, 2014.

[62] J. T. Chiang, J. J. Haas, and Y. C. Hu, “Secure and precise location verification using distance bounding and simultaneous multilateration,” *Proc. 2nd ACM Conf. Wirel. Netw. Secur. WiSec’09*, pp. 181–191, 2009.

[63] S. Drimer and S. J. Murdoch, “Keep your enemies close: Distance bounding against smartcard relay attacks,” *16th USENIX Secur. Symp.*, pp. 87–102, 2007.

[64] C. Meadows, P. Syverson, and L. W. Chang, “Towards more efficient distance bounding protocols for use in sensor networks,” *2006 Secur. Work.*, 2006.

[65] K. B. Rasmussen and S. Čapkun, “Realization of RF distance bounding,” *Proc. 19th USENIX Secur. Symp.*, pp. 389–401, 2010.

[66] N. Sastry, U. Shankar, and D. Wagner, “Secure Verification of Location Claims,” *Proc. Work. Wirel. Secur.*, no. Section 2, pp. 1–10, 2003.

[67] Y. C. Hu, A. Perrig, and D. B. Johnson, “Packet leashes: A defense against wormhole attacks in wireless networks,” *Proc. - IEEE INFOCOM*, vol. 3, pp. 1976–1986, 2003.

[68] M. G. Kuhn, “An Asymmetric Security Mechanism for Navigation Signals,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 3200, pp. 239–252, 2004.

[69] C. Meadows, R. Poovendran, D. Pavlovic, L. W. Chang, and P. Syverson, “Distance bounding protocols: Authentication logic analysis and collusion attacks,” *Adv. Inf. Secur.*, vol. 30, pp. 279–298, 2007.

[70] G. Avoine and A. Tchamkerten, “An efficient distance bounding rfid authentication protocol: Balancing false-acceptance rate and memory requirement,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5735 LNCS, pp. 250–261, 2009.

[71] Munilla, J., Ortiz, A., Peinado, “A.: Distance Bounding Protocols with Void-Challenges for RFID,” *Work. RFID Secur. - RFIDSec*, 2006.