An algebraic time-advantage-based key establishment protocol

by

Sanchita Barman

A thesis submitted to the graduate faculty
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Computer Engineering

Program of Study Committee:
George T Amariucai, Co-major Professor
Yong Guan, Co-major Professor
Simanta Mitra

Iowa State University
Ames, Iowa
2017

DEDICATION

This work is dedicated to all the problems that I ran into, both academic and non-academic during my stay at ISU, without whom I could have finished my work much earlier. They taught me that it's still possible to get some work done against all kind of adversaries.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Page

ACKNOWLEDGMENTS

I would like to like to take this opportunity to express my gratitude to those who helped me with the various aspects of conducting this research and writing the thesis. First and foremost my guides Dr George T Amariucai and Dr Yong Guan for their guidance, patience and incredible support throughout this time. Their insights and encouragement have always inspired me to keep on trying and finally be able to complete this work. I would also like to thank them for the confidence they showed in me when I first started working with them.

I would additionally like to thank my husband, parents and brother for their unconditional mental and financial support.

ABSTRACT

In this thesis we have built a key-establishment protocol which takes advantage of a resource : time. When two devices spends a pre-determined, mostly uninterrupted time interval with each other they would be able to establish a key. However it is not just the quantity of time but also the quality which matters. The information gained about the key with time by the legitimate party can is flexible and can be chosen by the user. We have analyzed our protocol thoroughly and discussed the circumstances an adversary can gain access to information about the key.

CHAPTER 1. OVERVIEW

Security mainly involves being able to differentiate between a legitimate party and an adversary. Most of the time security provided by any system is due to a secret key shared by the legitimate parties. Hence it is very important to have a very secure way of communicating the key between legitimate parties. In my thesis I along with my guides have built a secure way to establish a key between two parties.

At present there exists a lot of state-of-the-art key establishment protocols. They mostly rely on advantages which already exists or a more unconventional ones. Some examples or recent approaches are public key infrastructure, physical layer security or privacy amplification. We have used a resource which has not been used much : time. A lot of times the parties involved in a secure key exchange spend a lot of time in close proximity. They are in the same environment and which is mostly secure. To give some examples : RFID tags embedded in clothing and grocery packages spend most of their time in an inventory for most of the time; artillery and weapons are mostly supposed to stay in a military weapon storage and so on. In our work we have explored how to exploit this situation to build a secure key-establishment protocol. That means we have used the long and mostly uninterrupted interval of time as an advantage to establish a key.

It might seem that the time resources could be used similar to using physical-channel where the secure environment could be used to share common randomness over a wireless channel. However the problem with this approach is that the legitimate parties have no way to know whether the location is secure. Hence often they use a trusted third party. We have found a way to not use a third party for our work and hence making the protocol more practical and efficient.

We have targeted our protocol to work in low-cost devices which are typical of the smart home environment. These devices usually have specific challenges which include that they might have neither the notion of time or the computational power to engage in sophisticated cryptographic algorithms. However our work will be applicable to more than just low-cost devices like that used in smart homes.

1.1 Introduction

To be precise, the following are the goals we want our protocol to achieve :-

- It will introduce a new paradigm in the context of key establishment : using time as a resource.
- It will investigate the possible implementation of puzzle-based paradigm.
- It will provide a theoretical framework for formal evaluation of general time based key-establishment protocols.
- It will produce novel designs for time-based key-establishment protocols, which will out-perform physical-channel-based protocols, in that they :
  - will not rely on the security of the authentication protocols, and
  - will not require human intervention.
- It will create a building block for the engineering of feasible solutions to low-cost wireless device security and privacy.

1.2 Problem statement

Our problem can be reduced to a much simpler problem mentioned below. We will utilize the solution protocol to meet our goal.

There is a main device which has to be paired with a dependent device. The main device say A will broadcast clues at regular time intervals. The dependent device say B_i ; $(i \in \mathbf{I})$ will try to use the clues to calculate a key. B_i has to spend at least a pre-defined amount of quality time with A to be able to get the key.

1.3 Motivation

Our work does not only consider long intervals of time but also high-quality time intervals. The time-based key-establishment protocols are subjected to two constraints :

1. They should stand alone.

2. They should be automatic.

The first constraint ensures that we avoid generating fresh keys that are subjected to the scrutiny of the old keys - for example if a fresh key is encrypted with the old session key. The second constraint is imposed so that the protocols are not subjected to human interactions. We have kept human interactions to a minimum since they might cause a major security risk. Due to these two constraints, we are assuming that the legitimate parties are completely unaware of their location.They will not understand the difference between a physically secure location of an adversary-prone environment. So they do not know when to initiate a key-establishment protocol. Most importantly, the legitimate parties are unaware of the real identity of their partners involved in the protocol.

The main device can broadcast a series of random numbers at regular intervals of time over a continuous duration of time. The legitimate receiver can listen to all the numbers and then compute the key using the numbers. This approach solves gives advantage to the user which can spend all the required duration with the main device. However the problem is that using this approach would mean that the legitimate user cannot miss any clues. Hence this does not satisfy one of the main conditions of the requirements. There is no authorization done here as well.

We can simply use Shamir's secret sharing scheme [26]. The shares will be broadcast in regular intervals of time over a required duration of time. This approach will enable the legitimate user to miss some clues. However we will not be able to control the rate of information gain.

Since we want authorization, we will have the main device broadcast the hash values. And then to allow the legitimate user to miss clues, we can add a sequence number to the clues. That means, now the clue which is broadcast comprises of a random number, a hash value and a sequence number. The legitimate receiver listens to the clues for a required amount of time. The hash values enables it to do authorization. And the sequence number which is sent back to the main device enables it to know which clues have been missed and determine whether the receiver was listening to the main device contiguously for a required amount of time. However there is a major flaw in this design. It is not immune to injection attacks. That is if an adversary listens to some of the clues and decides to broadcast them with changed sequence number. The legitimate receiver then tries to send back the real clue. Now since the sequence number might not be the same for the legitimate user and the main device since their clocks might not be synchronized. However if an adversary injects a false clue with a non-consecutive sequence number, a key cannot be established. So the legitimate parties cannot establish a key even though they fulfill all the conditions.

In order to remove this drawback we can use a codebook. But this requires all the legitimate parties to have prior knowledge about it. And that defeats the purpose.

We have taken care of all these situations in our protocol.

1.4 Application

Recent technological advances, combined with an increasing demand for smart appliances, provide a glimpse of a near future in which the smart home becomes a major and ubiquitous presence in the internet of things. The smart home environment is bound to include a number of low-cost devices, such as wireless sensors, wireless switches and RFID transponders. The low cost constraints these devices to have very low capability to do computations.

At present numerous controversies surround the wide scale deployment of RFID.The main concern arises due to consumer's fear of privacy invasion. RFID tags are indeed often subjects of privacy attacks which might enable an adversary to track the movement of a person. Similar attacks can be used to gain information about patients in clinics or senior citizens living in assisted living facilities. These attacks can also be used to inventory their personal possessions. There might arise other ways to breach privacy with rise in usage of such smart devices. Hence there is a increasing demand for secure authentication which would offer strong security guarantees.

However these authentication process for low-cost wireless devices have to start with a secure secret-key sharing. This is where our protocol will have immediate application.

Apart from low-cost devices our protocol can also be used for regular devices very easily. It will still not be possible for an adversary to gain much information about the secret key. We can simply vary some of the parameters in order to achieve this goal.

CHAPTER 2. RELATED WORK

Shannon proved that the perfect secrecy of a transmitted message is achieved when the entropy of the shared secret key is as large as the original message [27]. This resulted in usage of more random, longer and fast changing secret keys for communication security. Then the public key cryptography [9] was introduced. Security in this case was based on a known intractable problem. This required more advancement in the techniques of generation and distribution of secret keys. The RSA public key encryption scheme [24] was introduced shortly after and is still used extensively. Public key cryptography has been a starting point of many security notions like and non-malleability [4] and chosen plaintext attack. However since they are too expensive compared to symmetric key cryptography they are mainly used for key establishment [21]

Wyner's wiretap channel [8] was a pioneer for communication security. This paper first introduced the idea of using the physical characteristics of the channel as an advantage for the legitimate receiver over the adversary instead of a pre-shared secret key. However this method required that the receiver's channel to be better in some sense than that of the eavesdropper [7] which is a limitation. Hence it could never replace the usage of classical cryptography. A lot of work has been done to artificially create a less noisy channel [8, 7]. A significant breakthrough in this direction is Maurer's idea of common randomness [19, 2, 6]. In this method the legitimate parties are required to tune to the same external weak radio signal. Then in order to extract the key, they need to engage in a three phase dialog - advantage distillation, information reconciliation and privacy amplification. What is advantage distillation ? It is the process where the legitimate party generates a sequence W. The adversary has much less information about this sequence compared to the legitimate parties. The information reconciliation phase [19] uses W to produce a sequence Q which is entirely known to the legitimate user but partially known to the adversary. Privacy amplification [19] is the process which is used to extract the shared secret with maximum entropy from Q. The adversary has no knowledge about the secret. If there is no external signal, random signal can be generated and broadcast my one of the legitimate parties. Maurer's work created the path for more research in this field. This lead to papers which proposed different forms of common randomness, varying from the coefficients of a fast-fading reciprocal communication channel [29, 28, 31, 18], to the output of built-in accelerometer when the two legitimate parties are shaken together [20]. Another interesting concept was introduced in [30], in which the superior quality of the legitimate receiver's channel will be ensured by a transport layer "stop-and-wait" re-transmission protocol. That means if the legitimate receiver loses a packet, the transmitter sees no acknowledgment message and hence re-transmits the packet. However when an adversary loses a packet, it is mostly not re-transmitted. This serves to be an advantage the legitimate user has over the adversary. However, privacy-amplification-based schemes are heavily dependent on certain bounds of the attacker's ability to obtain a correct copy of the common randomness. The attacker's information about the source of common randomness is usually characterized by either second-order conditional Renyi entropy –if the secrecy metric [5] is mutual information; or the smooth Renyi entropy – if the secrecy metric is expressed in terms of variational distance [23, 10, 15]. But essentially, such bounds are still unavoidably related to physical advantages, just as the early wire-tap framework of [8].

"Key wrapping" is another way of distributing keys. It is the process of transmission of keys with another key called the "key-wrapping key". This concept has been analyzed in [25] and [12]. In [25] it has been expressed it as a special case of deterministic authenticated encryption. And in [12], two weaker security notions has been introduced which is in turn inspired by formal security models of symmetric key cryptography. A key hierarchy is established in the process of key-wrapping. It extends the information leakage process which is inherent in any imperfect cipher to key wrapping key. In this case every level of the hierarchy has an expiration date. Although this method seems effective, however it is difficult to implement it by lightweight devices.

2.1 Existing approaches based on time

Any work with time as a resource can resemble works on privacy amplification by [19, 1, 6], since it is after all a physical resource. But if you want to depend just on time and no other entity, the protocol has to be designed differently. One such technique is a puzzle based approach. This however is different from the Merkle puzzles [22].

One of the most notable contribution in this field is Ari Juel's work in [14]. He uses Shamir's threshold secret sharing scheme to share a secret tag key over a large number of RFID tags. The secret key has been distributed in such a way that only the vendor of the associated product would have access to it in the storage room. Our present work is can be said to be influenced by this work, however it has a very different focus and also has a lot more security considerations and advantages.

In this work [16], secret sharing has been used to prevent random readers from accessing private tags. However this work only considers just the quantity and not the quality of time the legitimate parties are supposed to spend with each other.

A more concrete time-base key-establishment protocol has been proposed in [3] in their Adopted Pet protocol. In this work Amariucai et al based their algorithm on breakable ciphers. It depends on LFSR-based linear leakage system. It mainly assumes that two legitimate parties, RFID tags and reader in their case would be able to spend more high-quality online time with each other than an adversary which might be located further. It also assumes the presence of a comparatively more sophisticated device which tries to gather information about the password for other devices (RFID tags) in close proximity. The RFID tags has an internal LFSR of length L. The characteristic polynomial of it is of degree L, which consists of it's secret. If the tag trust the reader then it responds to it's queries with a valid UPC. However if that is not the case then the tag generates ans broadcast a single bit using the LFSR. The reader has to spend enough time with the tag in order to be able to recover $2L$ contiguous bits and hence solve the LFSR's characteristic polynomial. The reader can use advanced algorithms like Berlekamp-Massey algorithm [17] to solve the system of linear equation which contains L equations and L unknowns. However for an adversary, they cannot spend enough time with the tag and hence does not obtain consecutive bits. So if we assume that it does get $2L$ bits but not consecutive ones. It has k unknown bits spread among them. So now it has $L + k$ equations and $L + k$ unknowns. However the system is no longer linear. There is no known efficient method to solve quadratic equations on a finite field. This kind of problem is NP-complete [11] and known as ”MQ problem”.

Even though this approach is efficient, it has it's drawbacks. Breakable ciphers used in this paper might be hard to implement. It does not leave any flexibility on the rate of information gain with time.

Ari Juels have addressed some of these issues in [13]. In this patent one device generates a set of keys over time. Then it generates a plaintext information : an erasure code using of at least one of the keys in the set. Then it broadcasts the code over time. The second device has to listen for enough time to be able to recover that part of the key. The second device has to obtain a certain number of parts of the key to be able to recover the secret value. This approach however does not allow the listening device the flexibility to start listening at any time of the broadcast and still recover the key. It also does not allow us to chose the rate of information leakage, i.e the information leakage function. It also does not state any concrete security measure or calibration parameter.

CHAPTER 3. TOOLS NEEDED TO BUILD OUR PROTOCOL

Let us revisit our problem. Our main goal is to design a key-establishment protocol which will essentially be able to differentiate between a legitimate user from an adversary.The adversary can be passive and be present near the legitimate parties for a short amount of time. Or they might be an active adversary which introduces fake clues. The adversary In order to do that it uses the following conditions :-

1. Time : A legitimate user which will be able to spend a pre-defined quality amount of time with the broadcasting device.
2. The legitimate user should be in direct contact with the broadcasting device. The devices can be mobile and hence should be able to support any kind of change in noise level in the environment throughout the duration of broadcast of clues.
3. It is an autonomous protocol.
4. Users can control the rate of information gain with time during the required time interval. This mean the user can design the information leakage function according to the anticipated surrounding of the devices during broadcast of the clues. We will elaborate this point in the next section when we discuss information leakage function.
5. User can choose the key. The protocol does not require to stick to any kind of key.
6. A legitimate user can start listening to the clues at any time when they are being broadcast provided they spend enough quality time with it.
7. There is no need for a built-in clock.
8. The protocol should support any arbitrary authorization policies based on the amount of time spent with the broadcasting device.

So essentially a legitimate user will be able to spend a pre-defined interval of time with the main device. The main device is going to broadcast clues at equally spaced time instances for a certain time interval. In order to recover a secret key one will get n of clues. Since there might be some disturbances in the environment where the clues are being broadcast, a legitimate party is allowed to miss at most k of clues. However the legitimate user does not miss all k clues all together. There will be a frequency to which they can miss clues. We will elaborate on this in the next chapter. An adversary is one who will not be able to spend the same amount of quality time as a legitimate receiver. They can at most hear m of clues and then have to miss at least p of clues.

The adversary can however keep listening after the required clues has been broadcast. The adversary can keep listening for the clues. However obviously it will not be of any significance.

In this chapter we will discuss the definitions we have used for our protocol. Also we will state and prove the theorems associated to our work. We will further give a brief overview of *Shamir's secret sharing scheme* which we have used to build our protocol.

3.1 Definitions and theorems

***Robustness** : A protocol is said to be (n, k)-robust if the legitimate user who listens to at least n consecutive clues, out of which he can miss at most k clues, can recover the secret key with probability 1.*

***Security**: A protocol is called (m, p)-secure if the attacker who listens to at most m consecutive clues, after she misses at least p consecutive clues, never recovers the secret key.*

Theorem 1 *If $\exists$ an a where $a \geq 1$ and $a \in I$ such that the following conditions are satisfied*

1. $n - k \leq a \times m$
2. $k \geq (a - 1) \times p,$

then there does not exist a protocol which is (n, k)-robust and (m, p)-secure.

Proof :

We can construct a protocol which is robust, which we will show in the next chapter.

The maximum number of clues an adversary can obtain is $a \times m$. Since they will miss p clues after every m clues two cases will arise :

Case 1 : $a \times m \leq n - (a-1) \times p$. This implies $a \times m + (a-1) \times p \leq n$. This means that the duration of time when the adversary can listen to the clues, is less than the duration in which the clues are being broadcast. However since $n - k \leq a \times m$ the adversary is getting more than enough clues required to obtain the key. This case also implies the second condition in the theorem.

Case 2 : $a \times m > n - (a-1) \times p$. This implies $a \times m + (a-1) \times p > n$. This means that the adversary can listen for more time than when the clues are being broadcast. In this case only the first n clues will matter for the adversary. According to the second condition $k \geq (a-1) \times p$. And we can say that the adversary misses less than or equal to k clues. Hence they will not miss more than the allowed number of clues. Hence the adversary will be able to get enough number of clues to recover the secret key.

Theorem 2 *If $\forall a$ condition 1 and 2 do not hold simultaneously then $\exists$ a protocol which is (n,k)-robust and (m,p)-secure.*

Proof:

If we have an a such that $n - k < a \times m$, then the adversary never get a sufficient number of clues.

If we have an a such that $k > (a-1) \times p$, then the adversary misses more than enough clues and hence cannot get a sufficient number of clues.

In both the above cases the adversary will not be able to recover the required number of clues to get the secret key.

We will look at a special case : we take a such that $a \times m \geq (n-k)$ but $(a-1) \times m < n - k$. So now in order for the theorem to hold, we should have $k > (a-1) \times p$. We have two conditions :

Case 1 : $(a-1) \times m + (a-1) \times p \geq n$: we can say that the adversary can at most obtain $(a-1) \times m$ number of clues. Which we know is less than $n-k$ and hence not sufficient to recover the key.

Case 2 : $(a-1) \times m + (a-1) \times p < n$: in this case, out of n clues the adversary misses $(a-1) \times p$ clues which is more than k. Hence they cannot recover the secret.

3.2 Shamir's secret sharing scheme

The concept of Shamir's secret sharing scheme is very simple : you have a secret x; you have n participants among which the secret is shared. You want that any $n-k$ participants among the n participants can use their shares together to recover the secret. We are going to use this scheme in our protocol. Hence I am going to discuss the keys parts in it over here. It is not possible for less than $n-k$ participants to recover the secret.

We choose a large prime number p and let $\mathbb{F} = \mathbb{Z}/p\mathbb{Z}$, i.e choose a finite field. Let the secret be $x = (x_1, x_2, x_3, \ldots x_n)$. We take a polynomial f of degree $n-k-1$. We choose the coefficients : $f_1, f_2, f_3 \ldots f_{n-k-1} \in \mathbb{Z}/p\mathbb{Z}$. So we have $f(z) = f_0 + f_1 * x + f_2 * z^2 + f_3 * z^3 \ldots f_{n-k-1} * z^{n-k-1}$ and $f(0) = x$. We choose n unique points $\in \mathbb{F}$: $(a_1, a_2 \ldots a_n)$. The shares are the tuples : $(a_1, f(a_1) = b_1), (a_2, f(a_2) = b_2), \ldots (a_n, f(a_n) = b_n)$. Any $n-k$ tuples among these tuples are enough to recover the secret x. This means we have n points on a curve of degree $n-k-1$ polynomial.

In order to recover the secret using Lagrange interpolation for finite fields. We have at least $n-k$ unique values of a $n-k-1$ degree polynomial.

$$f(0) = \sum_{i=1}^{n-k-1} f(a_i) \prod_{m=0, m \neq j}^{n-k-1} \frac{a_m}{a_m - a_j} \tag{3.1}$$

Hence it is a linear system in k unknowns. We have at least $n-k$ equations. So there will exist a unique solution for this system of equations using Gaussian elimination. Hence we can find f_0 which is the secret.

3.3 Demonstration

Both legitimate user and the adversary are aiming for the same key. They start almost at the same time. I will give some examples of the cases which might arise in our protocol. I have discussed some cases in the following tables. The Xrepresents the redundant clues, i.e the clues which can be missed. And the •represent the clues which are not redundant. For the sake of simplicity, I have grouped the redundant clues together, but that might not necessarily happen. However the cases we would be discussing will not be affected. There are k redundant clues.

The clues are broadcast in regular time intervals. That is at each time instants one clues is broadcast. Hence we can also consider one time instant as one clue. Length of time interval $[M_1, P_2]$ is m; $[P_1, P_2]$ is p; $[N_1, N_2]$ is n; $[K_1, K_2]$ is k.

- **Case 1a** : $n < m$ and $k < p$.

  There can be number of scenarios for this condition :

  Table 3.1 Scenario 1 : $n < m$ and $k < p$.

$M, N_1 \rightarrow$	•
	•
	•
	$\dots$
	$\dots$
$K_1 \rightarrow$	X
	$\dots$
	$\dots$
$K_2 \rightarrow$	X
	•
$N_2 \rightarrow$	•
$P_1 \rightarrow$	
$P_2 \rightarrow$	

  The adversary starts listening from the same position as the legitimate user. The clues are being broadcast till K_2.

Hence in this case in table 3.1, the adversary is able to get the clues. The adversary just have to obtain $n-k$ clues from the first n clues which are broadcast. In this case $m-p$ is greater than n.

Table 3.2 Scenario 2 : $n < m$ and $k < p$

$M, N_1 \rightarrow$	•
	•
	•
	…
	…
	•
	•
$P_1, K_1 \rightarrow$	X
	…
	…
$N_2, K_2 \rightarrow$	X
$P_2 \rightarrow$	

In this case in table 3.2 the adversary is able to retrieve the key. The situation is somewhat similar to the previous case. In this case $m-p$ is equal to $n-k$. Hence the adversary is able to obtain all the necessary number of clues.

Table 3.3 Scenario 3 : $n < m$ and $k < p$

$M, N_1 \rightarrow$	•
	•
	•
	…
	…
	•
	•
$K_1 \rightarrow$	X
	X
	…
$P_1 \rightarrow$	X
$N_2, K_2 \rightarrow$	X
$P_2 \rightarrow$	

In this case in table 3.3 the adversary is able to retrieve the key. This situation again is similar to the first case. In this case, $m - p$ is greater than $n - k$. Hence the adversary is able to obtain all the necessary number of clues.

Table 3.4 Scenario4 :$n < m$ and $k < p$

$M, N_1 \rightarrow$	•
	•
	•
	…
	…
$P_1 \rightarrow$	•
	•
$K_1 \rightarrow$	X
	X
	…
	X
$N_2, K_2 \rightarrow$	X
$P_2 \rightarrow$	

In this case in table 3.4 the adversary cannot retrieve the key. $m - p$ is lesser than $n - k$. Hence the adversary is not able to obtain all the necessary number of clues.

- **Case 1b** : $n < m$ and $k > p$

  The adversary can listen to more consecutive clues than the total number of clues. Also it misses less clues consecutive clues than the total clues a legitimate party can miss.

Table 3.5 $n < m$ and $k > p$

$M, N_1 \rightarrow$	•
	•
	•
	…
	…
	•
	•
$K_1 \rightarrow$	X
	X
	…
	X
$N_2, K_2 \rightarrow$	X
$P_2 \rightarrow$	
$P_2 \rightarrow$	

In this case in table 3.5 the adversary is able to retrieve the key. This situation again is similar to the first case. In this case, $m - p$ is greater than n. Hence the adversary is able to obtain all the necessary number of clues.

- **Case 2a** : $n > m$ and $k < p$

Table 3.6 $n > m$ and $k < p$

$M, N_1 \rightarrow$	•
	•
$P_1 \rightarrow$	•
	…
	…
	•
	•
$P_2, K_1 \rightarrow$	X
	X
	…
	X
$N_2, K_2 \rightarrow$	X
	…
	•

In this case in table 3.6, $n - k$ is greater than $m - p$. However, here we might consider some sub-cases for analysis. (I will not make separate tables for each sub-case, but rather write describe what might happen.)

- Suppose we have $2m = n$. Since $k < p$, we will have $k < 2p$. So the adversary does not get enough clues for retrieving the key.
- Suppose we have $2m > n$. Then again since $k < p$, it will not be feasible for the adversary to get enough clues for retrieving the key.

- **Case 2b** : $n > m$ and $p = 2k$.

Table 3.7 $n > m$ and $p = 2k$

$N_1 \rightarrow$	•
	•
	•
	…
	…
$M_1 \rightarrow$	•
	•
$K_1, P_1 \rightarrow$	X
	…
$N_2, K_2 \rightarrow$	X
$P_2 \rightarrow$	

In this casein table 3.7, the new cycle for the adversary starts a little later. So if he has not missed any clues in the earlier cycle. he can retrieve enough clues to get the keys.

- **Case 2c** : $n > m$ and $p > k$.

Table 3.8 $n > m$ and $p > k$.

$M_1, \rightarrow$	
$N_1, P_1 \rightarrow$	•
	•
	•
	•
$K_1 \rightarrow$	X
	X
	…
	X
$P_2, K_2 \rightarrow$	X
	…
$N_2 \rightarrow$	•

In this case in the table 3.8 the adversary starts the cycle before the legitimate user. However since $p > k$, the adversary cannot get enough clues to recover the key.

- **Case 2d** : $n > m$ and $p < k$.

Table 3.9 $n > m$ and $p < k$.

$M_1, \rightarrow$	
$N_1, K_1, P_1 \rightarrow$	•
	…
$P_2, \rightarrow$	•
	•
	•
$K_1, P_1 \rightarrow$	•
	…
$P_2, \rightarrow$	•
	…
$K_2, \rightarrow$	•
	•
	•
$P_1 \rightarrow$	•
	…
$P_2, \rightarrow$	•

In this case in table 3.9, it actually depends on how small p is than k and also how small m is than n. The given figure shows a specific case, where the adversary is not able to recover the key. However suppose, p was even smaller, say $3p = k$. In that case in the given situation, where the adversary's cycle start before the legitimate user, he is able to gather enough clues for the key.

CHAPTER 4. PROTOCOL

We have listed the requirements of the protocol in detail in the previous chapter. We have also found and analyzed the situations in which such a protocol will exist. In this chapter we will build the protocol which we claim to fulfill all the conditions.

So let's try to see what might work for our case to build intuition. The main device can broadcast a series of random numbers at regular intervals of time over a continuous duration of time. The legitimate receiver can listen to all the numbers and then compute the key using the numbers. This approach solves gives advantage to the user which can spend all the required duration with the main device. However the problem is that using this approach would mean that the legitimate user cannot miss any clues. Hence this does not satisfy one of the main conditions of the requirements. There is no authorization done here as well.

So now let us try another approach: We can simply use Shamir's secret sharing scheme [26]. The shares will be broadcast in regular intervals of time over a required duration of time. This approach will enable the legitimate user to miss some clues. However we will not be able to control the rate of information gain.

We can use another approach using the random numbers. Since we want authorization, we will have the main device broadcast the hash values. And then to allow the legitimate user to miss clues, we can add a sequence number to the clues. That means, now the clue which is broadcast comprises of a random number, a hash value and a sequence number. The legitimate receiver listens to the clues for a required amount of time. The hash values enables it to do authorization. And the sequence number which is sent back to the main device enables it to know which clues have been missed and determine whether the receiver was listening to the main device contiguously for a required amount of time. However there is a major flaw in this design. It is not immune to injection attacks. That is if an adversary listens to some of the clues and decides to broadcast them with changed sequence number. The legitimate receiver then tries to send back the real clue. Now since the sequence number might not be the same for the legitimate user and the main device since their clocks might not be synchronized. However if an adversary injects a false clue with a non-consecutive sequence number, a key cannot be established. So the legitimate parties cannot establish a key even though they fulfill all the conditions.

In order to remove this drawback we can use a codebook. However authorization can longer be ensured in this case

We have seen other related work which is not going to work given our situation and fulfill all the required conditions. Hence we propose our solution which we claim to solve our problem.

We will first discuss the construction. Then we will describe the steps of the protocol. In the next chapter we have proved our claim. We have also analyzed the properties of the protocol.

4.1 Construction

Without loss of generality we will assume that there is one legitimate user and one adversary. There exists one broadcasting device which gives out clues at regular intervals of time. We will call this device the main device or A. The legitimate user is B and the adversary is C. In this chapter we will describe the construction of our solution gradually explaining the significance of each step. We will show how A will establish a key with B while C will not be able to get it. We are going to use the tools we discussed in the previous chapter. Before going into the steps we will discuss a few properties which are essential to build up the protocol.

We will call our secret key K. K is a function of a multidimensional vector $\bar{X}$, i.e $K = f(\bar{X})$. The main device A will broadcast clues over a pre-defined period of time. The number of clues will depend on a pre-defined function which we will discuss shortly. Here $\bar{X}$ is the secret. $\bar{X} = (\bar{x_1}, \bar{x_2}, \ldots \bar{x_n})$. The elements of the secret $\bar{X}$ are again vectors. Each $\bar{x_i}$'s are divided into shares. The shares are designed like that in Shamir's threshold secret sharing scheme. The number of vectors, shares and the threshold can be decided by the required amount of robustness of the protocol. However as we have seen earlier, a simple construction of shares for the key will not suffice. Hence we designed our protocol differently to meet our needs. The shares for each vector is going to be broadcast simultaneously.We will see how the shares are constructed and how the clues look like in the later sections.

4.1.1 Evolving secret

One of the requirements of our protocol is that an user would be able to start at any time when the clues are being broadcast and would be able to get enough clues to calculate a unique key. To elaborate, suppose A will be broadcasting clues from time instant $t = 0$ to $t = t_A$. A legitimate party needs q number of clues in total. So any user which listens to at least q number of consecutive clues within the time A is broadcasting the clues, they should be able to recover a key. However the key obviously cannot be same for everyone. The key will be different for users starting from different times. Thus the secret key is going to evolve with time. We have assumed that there is only one legitimate user for the sake of simplicity. However if there are more than one, then this property will be useful in establishing unique keys with different users.

The key generated by the user will depend on the number of clues he has been listening to prior to the final clue. At each time instant the A will broadcast a set of clues for users starting to listen from different time.

Table 4.1 Distribution of shares for evolving secret

	•					
	•	•				
	•	•	•			
	•	•	•	•		
	X	•	•	•	•	
	X	X	•	•	•	•
	X	X	X	•	•	•
$t_0 \longrightarrow (S_0)$:	X	X	X	X	•	•
$t_1 \longrightarrow (S_1)$:		X	X	X	X	•
$t_2 \longrightarrow (S_2)$:			X	X	X	X
$t_3 \longrightarrow (S_3)$:				X	X	X
$t_4 \longrightarrow (S_4)$:					X	X
$t_5 \longrightarrow (S_5)$:						X

Table 4.1 is an example of a very simple distribution of shares. Suppose we have a system where the main device broadcasts eight shares where the threshold is 4. A legitimate user who has been listening and been able to get 4 clues before t_0 will be able to enough clues to compute the secret S_0. Similarly another legitimate user who can get enough information at t_1 to create S_1. All the secrets will be unique and can be used to be recover the key to be paired with the main device. In this particular example X can be a redundant or extra clue and • can be a deterministic clue or the other way round. We will discuss the significance of the two types of clues when we discuss the size of clues in the next chapter.

4.1.2 Information leakage function

In real world it might not always be possible for the two devices to spend an absolute uninterrupted interval of time with each other where no other conditions can impact the broadcast. There might be other factors, for example other processes running in the device, amount of disturbance in the broadcasting environment, interrupted intervals of time when the devices may actually be in proximity and so on.

We want our legitimate users to be able to pair with the main device if they spend a pre-defined amount of time. Also it's not just the amount of time which will matter, it will also the quality. That means that the user would have to do some computations in order to obtain the key.

The information leakage function effects the security of the system in an unconventional way. It does not provide any security mathematically. However it is used to identify the noise level in the wireless network the devices are subjected to. Also by using this function, the protocol becomes flexible enough to be used in a broad range of device for a number of situations.

I will explain the situation with some examples. I have two phones, an old one which is secure and a new one which I want to pair with the old one. (The possibility of such a situation arising is pretty low. However one might think this situation using any other two devices.) Suppose I leave for work in the morning, use busy public transportation service and constantly both my phones. In this case it is better if there is not much information to be gained when the devices are in mistrusted environment.

Another example : Suppose I want my new iPad to pair with my macbook. But I leave my macbook at home and only take my iPad to work. So the devices can only be near each other when I come back from work in the evening till when I leave in the morning. Sometimes I might go out for dinner with friends after I get back from work. Hence I would want my macbook to give out more information when my iPad is more likely to be near it, which is say after 10 pm. So I will use an appropriate mathematical function as the Information leakage function. My rate at which my macbook broadcasts clues for the keys and the amount of information my iPad can get if it listens most of the intended time, it can establish a key.The two devices can use that key in order to communicate with each other and hence get paired. Similarly I can use the same technique to pair other devices with my macbook, which might start listening from a different time. The keys, as we have discussed before would be unique for different devices.

We will call this function the *Information leakage function. Information leakage function* is the ratio of the information gained by a legitimate user over the number of clues obtained over time. I will explain this with a simple example.

In both the above examples I have not used lightweight devices. However whether we would be able to implement our protocol for the said devices will depend on the parameters we will use. We will discuss them in the next chapter.

This is one of the major improvements we have been able to achieve in this protocol over other works so far. Of course this means that the number of clues the legitimate user is allowed to miss is going to vary with time. So missing h number of consecutive clues during the beginning of the broadcast might not have the same effect if the same number of clues are lost later on in the broadcasting duration. However we are making sure that the allowance of loss of clues depends on the gain of information rate which is required by the user.

Information leakage function should be a monotonically increasing function.And we would only consider the positive quadrant of the field used as the range. This is because the information keeps increasing. In practice we will not want to lose information when the legitimate user spends time with the main device. This is also the reason why we will only consider the positive quadrant.

We will call the information leakage function I.We have given an example in section 4.2.1 in figure 4.2. After plotting the function on a plane, we'll divide the Y-axis into equal intervals. We will then project the intersection with the function onto the X-axis. Let the length of the intervals be l_i. Then we multiply l_i's with 10^d, where we chose d such that the minimum $l_i \times 10^d$ is greater than 1. We then take the ceiling of each of the result and map it to a natural number, say n_i. We will then construct the clues according to this function.

4.1.3 Redundancies

Redundancy means the number of clues an user is allowed to miss in a given interval of time. This number would vary with the number of clues requires for the element in the secret. The redundancies can be determined by the robustness of the protocol. This will depend on the user to decide the k. The robustness can depend on many factors like disturbance in the broadcasting environment, the processing capacity of the listener and so on.

Apart from allowing the legitimate user to miss clues, we have another condition to fulfill in our protocol. We need to fulfill the information growth rate given by the user. This means that when the curve is steep, i.e the growth s faster, fewer clues will be required to get an element of the secret. That also means that the legitimate user will be allowed to miss fewer clues. This holds true for duration in the time interval where information growth is slower. A legitimate user would need more clues to obtain an element in the secret key. Hence a proportionate number of clues would be allowed to be missed by the legitimate user.

The number of redundant clues is also significant since it helps in error correction of the code ; in our case clues. Suppose an adversary injects fake clues. The redundancies used in the protocol can help disregard any such clues.

Considering all the factors for the redundancies, the user can decide on the value of k which might be some percentage of the total number of clues which will be broadcast. There can be two ways to distribute the redundancies. This redundancy will be divided among the set of clues which are broadcast : $\{k_1, k_2 \ldots\}$ and $k_1 + k_2 + \ldots k_n = k$. According to the number of clues which is actually required for the vectors in the key. That means $n_1 : n_2 : \ldots n_m = k_1 : k_2 : \ldots k_n$. The keys will be divided accordingly.Clues for the first element will have k_1 redundancies, the second set will have $k_1 + k_2$ redundancies and so on. We will further see with an example in the following section. This means the frequency of the clue which can be missed i,e the redundant clues is evenly distributed throughout the time interval. We will see in the next section how the clues will be constructed.

Another way for distributing the clues can be done when the user does not want an even frequency of missed clues. That means the user can allow a legitimate user to miss too many clues regardless of the number of clues required to recover the key during a certain section of the time interval. In other sections of the time interval they expect the wireless network environment to be better and hence allow less clues to be missed. Such a situation can arise depending on the noise expected in wireless network environment. The protocol and it's analysis will not be be different from the

earlier distribution of k_i's. So in order to keep matters simple, we will discuss only about the earlier distribution.
We will further analyze the significance of this part in the next chapter.

4.1.4 Clues

The clues for the keys will be broadcast by the main device A. The number of clues i.e the number of shares and the threshold, as we have seen in the earlier section will be decided according to the information leakage function I. In this section we will discuss how the clues are being constructed.
We have seen the construction of l_i and n_i in the previous section. The first element of the key would have n_1 shares with k_1 redundancies. The second element can have $n_1 + n_2$ shares and $k_1 + k_2$ redundancies. In this case there are k_1 redundant clues within the first n_1 clues and k_2 redundancies among the n_2 number of clues which is being broadcast later on. Shares of the third and so on elements will be designed similarly. We are distributing the clues in this way due to a number of reasons : We are broadcasting the clues for different elements of the key simultaneously; and we want the construction of the clues to follow the information leakage function chosen by the user. We claim that this type of construction gives the user more flexibility while implementing the protocol while satisfying the security conditions we want the protocol to follow.
So formally this is what we do : We already know how many clues would be there for each element in the secret. We also know the redundancies. We will use Shamir's secret sharing scheme to construct the shares. We will call these shares, the clues. We do this for every element in the $\bar{X}$. The first element $\bar{x_1}$, we will have n_1 shares. The threshold is $n_1 - k_1$. So chose a finite field $\mathbb{F}_1$ where $|\mathbb{F}_1| \geq n_1 - k_1$. Now we choose $n_1 - k_1$ random points from $\mathbb{F}_1$, say $r_1, r_2, \ldots r_{n_1}$. We choose the coefficients from $\mathbb{F}_{\not\Bbbk}$ and hence construct a polynomial of degree $n_1 - k_1 - 1$, $p_1$such that $p_1(0) = \bar{x_1}$. We will calculate $p_1(r_i)$ where $1 \leq i \leq n_1$. Now, the clues for the first element are $\{(r_i, p_1(r_i)|r_1 \in \mathbb{F}_1, 1 \leq i \leq n_1\}$. Each of the clues are broadcast in each time instant.
Now for the second element we construct the clues similarly. However we need to make a few mod-

ifications.We need to do this so that the information gain is according to the information leakage function I. We have discussed the information gain function in an earlier section. The number of clues in this case is $n_1 + n_2$. The redundancies is $k_1 + k_2$. So we chose another finite field $\mathbb{F}_2$ and chose another polynomial p_2 and construct the shares according to Shamir's secret sharing scheme. The construction is similar to the first n_1 clues for the first element. So the clues for the second element will look like $\{r_i, p_2(r_i)|r_1 \in \mathbb{F}_2, 1 \leq i \leq n_1 + n_2\}$We will construct the clues for the rest of the elements in a similar way.

The clues for all the elements of the secret are broadcast simultaneously.Hence if there are m elements in the secret of the key, at each time instant the clue will look like $(r_i, p_1(r_i), p_2(r_i), \ldots p_m(r_i))$. However we learned about the evolving secrets. So at the first time instant only one such tuple will be sent by A. Second time two such tuples will be sent. The second tuple is for users which starts listening from there and missed the first clue. So suppose the user B starts listening to the clues. At time instant t_i it has to consider the ith tuple from the last one as it's clue.

4.2 Protocol

We will describe the protocol with an example.

4.2.1 Step 1 : Choosing the information leakage function :

We have chosen a specific function $I = \frac{1}{10} \times 2^x$. We will only consider the positive range. This function in the positive range fulfills all the conditions for a information leakage function. This function is decided depending on the situation where the protocol is to be used. As we have seen in an earlier section, it depends on a various factors.

The function in this example can be used where it is required that information gain is slower earlier and after some time it increases exponentially. That means it will take a longer time to gain information in the beginning :i.e in this case from time instant 0 to roughly 3.2 (we will get the actual size of this interval in the next section) a legitimate user will be able to obtain one element of the clue. However they would need to listen for a much shorter interval of time in order to gain the next element of the clue. We have plotted the function in figure 4.2. We are only going to consider the positive range. Also we will limit ourselves to just till the total time the main device intends to broadcast clues. Even though it is not shown in the plot, the function will reach a point beyond which it is a flat line. This is the point where the main device has broadcast enough clues for a legitimate receiver to gather enough clues to obtain all the elements of the key.

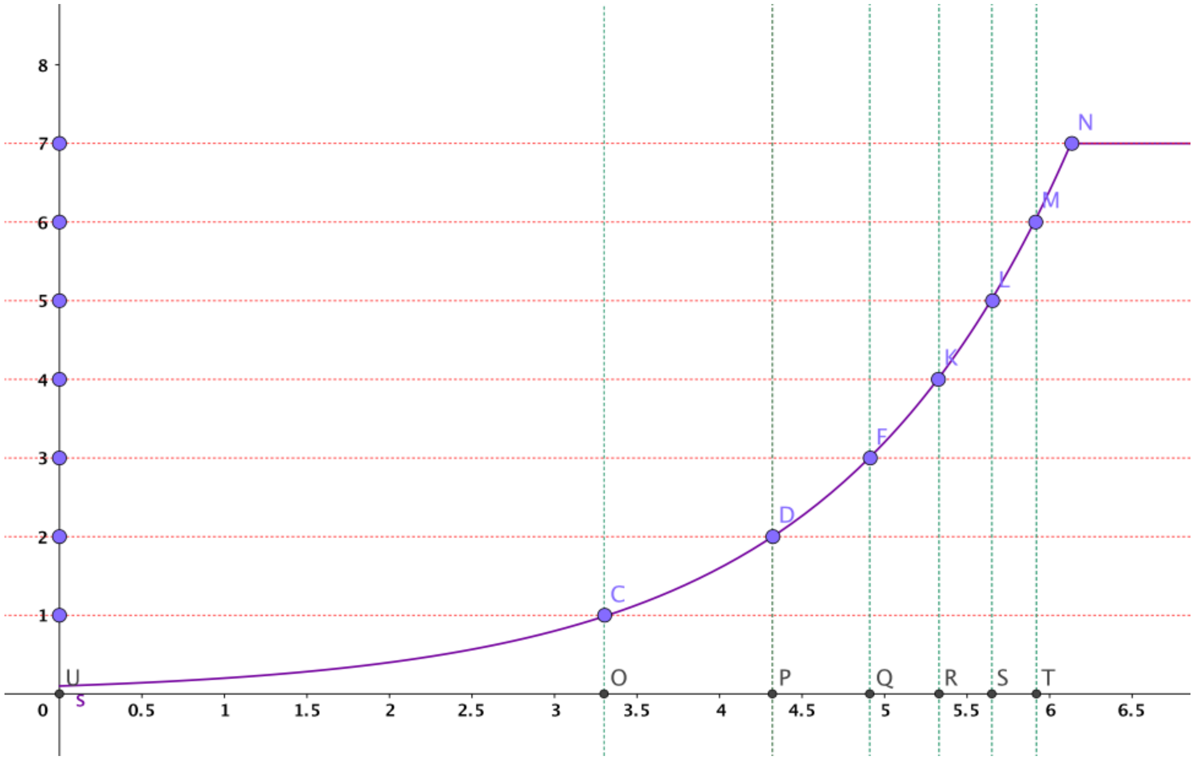

Figure 4.1 Function

4.2.2 Step 2 : Getting the thresholds :

In this case, as you can see from the above plot, we have divided the Y-axis in equal intervals. To make matters simple, we will just assume that legitimate user is supposed to obtain enough clues for the key after the eighth interval of the Y-axis

Here we have : $l_1 = 3.32$, $l_2 = 1$, $l_3 = 0.59$, $l_4 = 0.41$, $l_5 = 0.32$, $l_6 = 0.27$ and $l_7 = 0.22$. We can see that d is 1 here. So now we have : $n_1 = 34$, $n_2 = 10$, $n_3 = 6$, $n_4 = 5$, $n_5 = 5$, $n_6 = 3$ and $n_7 = 3$. The number of redundancies will be distributed according to the number of clues i.e the threshold for each vector. Hence $n_1 : n_2 : n_3 \ldots = k_1 : k_2 : \ldots$.

So now the clues to be broadcast will be : n_1, $n_2 + n_1$ and so on.

4.2.3 Step 3 : Broadcasting the clues :

The main device now starts broadcasting the clues. It will broadcast clues for each element in the key simultaneously. Table 4.2 represents how the clues are being broadcast. While broadcasting the clues, there are a few things we need to consider. The protocol must allow a legitimate user to start listening from anytime while the clues are being broadcast. It has to support evolution of the secret as we have already discussed in a previous section.If the legitimate user can spend enough quality time with the main device and hence gather enough clues, it must be able to recover all the elements of the secret. And the protocol should also enable information gain according to the information leakage function chosen by the user. We claim that the way the main device broadcast the clues enables the protocol to fulfill these requirements.

We will use table 4.2 to explain how do the clues look like. Each row represents the clues which will be broadcast in that time instant. Notice that clues for all the elements in the secret are being broadcast simultaneously. Each column represents the clues which is essential for a device which starts listening from a similar time instant. In the next chapter we have a table 5.1. In that table we can see that all the X's represent the clues for a legitimate party which starts listening from time instant 3. And all the $\boxed{\text{X}}$'s are the clues which that user uses from at one time instant.

For the sake of simplicity we have shown a limited window of when a legitimate user is allowed to start listening in order to obtain all the required clues. However in practice, this window can be made larger or smaller according to the requirement of the user.

Now for the example we have chosen : In the first set of clues the user has to obtain 34 out of $34 + k_1$ clues, in the second set they have to obtain 44 out of $34 + 10$ clues. The rest of the set of clues will be broadcast according the way discussed earlier.

Table 4.2 Clues broadcast over a pre-determined time interval

Clues for 1st element					Clues for 2nd element					Clues for 3rd element					...	Clues for mth element				
•					•					•					...	•				
•	•				•	•				•	•				...	•	•			
•	•	•			•	•	•			•	•	•			...	•	•	•		
•	•	•	•		•	•	•	•		•	•	•	•		...	•	•	•	•	
•	•	•	•	...	•	•	•	•	...	•	•	•	•	...	...	•	•	•	•	...
•	•	•	...	...	•	•	•	...	...	•	•	•	...	...	...	•	•	•	...	...
...	...	...	...	...	•	•	•	...	...	•	•	•	...	•	...	•	•	•	...	•
...	...	...	...	...	•	•	•	...	...	•	•	•	...	•	...	•	•	•	...	•
•	•	•	...	•	•	•	•	...	...	•	•	•	...	•	...	•	•	•	...	•
	•	•	...	•	•	•	•	...	...	•	•	•	...	•	...	•	•	•	...	•
		•	...	•	...	...	...	...	...	•	•	•	...	•	...	•	•	•	...	•
			•	•	...	...	...	...	...	•	•	•	...	•	...	•	•	•	...	•
				•	•	•	•	...	•	•	•	•	...	•	...	•	•	•	...	•
					•	•	•	...	•	•	•	•	...	•	...	•	•	•	...	•
					•	•	•	...	•	•	•	•	...	•	...	•	•	•	...	•
					•	•	•	...	•	•	•	•	...	•	...	•	•	•	...	•
						•	•	...	•	•	•	•	...	•	...	•	•	•	...	•
							•	...	•	•	•	•	...	•	...	•	•	•	...	•
								•	•	•	•	•	...	•	...	•	•	•	...	•
									•	•	•	•	...	•	...	•	•	•	...	•
										•	•	•	...	•	...	•	•	•	...	•
										•	•	•	...	•	...	•	•	•	...	•
										...	...	...	...	...	...	•	•	•	...	•
										...	...	...	...	...	...	•	•	•	...	•
										•	•	•	...	•	...	•	•	•	...	•
										•	•	•	...	•	...	•	•	•	...	•
											•	•	...	•	...	•	•	•	...	•
												•	...	•	...	•	•	•	...	•
													•	•	...	•	•	•	...	•
														•	...	•	•	•	...	•
															...	•	•	•	...	•
															...	•	•	•	...	•
															...	•	•	•	...	•
															...	...	...	...	...	...
															...	•	•	•	...	•
															...	•	•	•	...	•
															...		•	•	...	•
															...			•	...	•
															...				•	•
															...					•

4.2.4 Step 4 : Computing the key :

To re-iterate who qualifies as a legitimate user,a party which is able to spend a contiguous interval of time with the main device listening to the clues. A legitimate user should be able to get all the required clues.Using the technique used in Shamir's secret sharing, the legitimate user is able to compute all the elements of the key. Hence they have the set $\{\bar{x_1}, \bar{x_2}, \ldots \bar{x_t}\}$ and hence can compute $f(\bar{X})$ which is the required key.

The following flowchart describes our protocol briefly :-

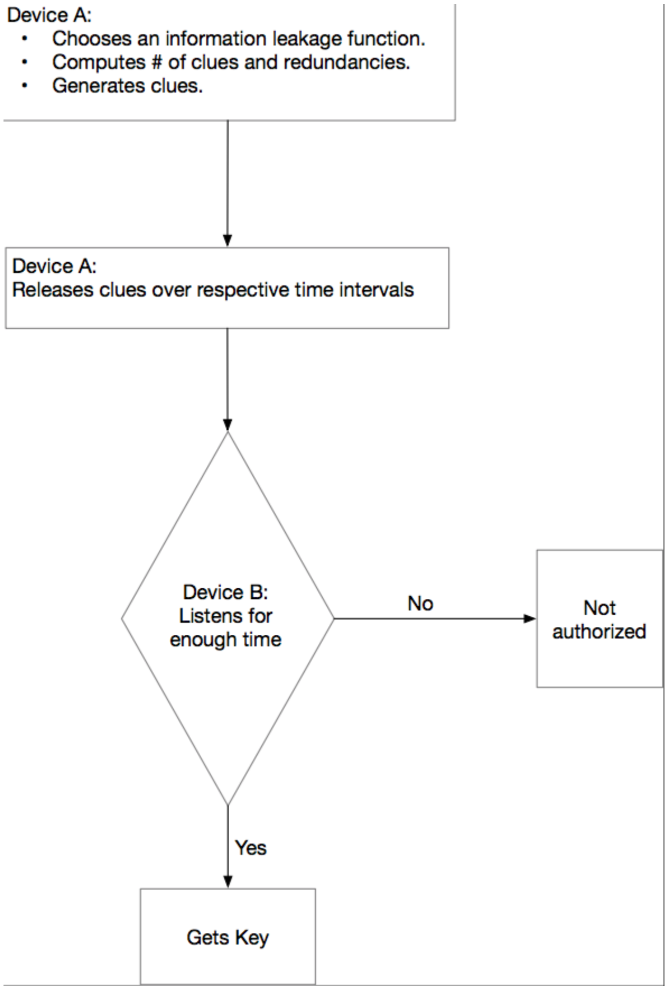
Device A:
• Chooses an information leakage function.
• Computes # of clues and redundancies.
• Generates clues.
Device A:
Releases clues over respective time intervals
Device B:
Listens for
enough time
No
Not
authorized
Yes
Gets Key

Figure 4.2 Function

CHAPTER 5. ANALYSIS

In this chapter we will discuss the correctness of our protocol along with some analysis of the security features. We will look at the protocol from an adversary point of view. This is going to help us understand the security of our protocol and under what conditions it might stop being useful. We will also discuss about the ways the we can make the protocol more efficient in terms of size of the clues.

5.1 Correctness of the protocol

We claim that our protocol will meet all the conditions. We have already proved the conditions under which a robust and secure protocol will exist in the two theorems in chapter 3. In chapter 4 we have shown the construction of a (n, k)-robust protocol. We will discuss whether our protocol indeed satisfies our claim.

We have used Shamir's secret sharing scheme for construction of the clues. These clues are being broadcast by A, i.e the main device. We know that k is the total number of clues which is allowed to be missed. From the construction of the clues which we discussed in the previous chapter, we can see that the following :-

The threshold for the number of clues for the first element in the secret is n_1. The number of clues which can be missed is k_1. For the second clue the threshold is $n_1 + n_2$. The number of clues which can be missed is $k_1 + k_2$. The corresponding threshold and total number of clues to be missed has been calculated similarly. So the total number of clues which can be missed is $k_1 + k_2 +_3 + \ldots + k_t = k$. And the total number of clues is $n_1 + n_2 + n_3 + \ldots + n_t = n$. So a legitimate user should be able to recover all the elements of the secret if they listen to n clues and miss at most k clues among them. Hence our protocol is (n, k)-robust.

An user will not be able to obtain all the elements of the secret even if it misses less than

k clues. We have distributed the redundant clue through all over the broadcasting duration. According to our construction, an user will not be able to recover the ith element if they miss more than $k_1 + k_2 + \ldots + k_i$ clues. However this is less than k. We will see why we have done this in our next point.

We have also claimed that the clues will be broadcast such that it fulfills the information leakage function. So again we can observe the following from our construction :-

From table 5.1 we can get a good idea of how the clues are broadcast. The clues for all the element of the secret are broadcast simultaneously. So any listening device can miss k_1 clues from the first $n_1 + k_1$ clues and k_2 more clues from the next $n_2 + k_2$ clues. There is no other way for a listening device to miss clues and still recover the element in the secret. Because if they miss more than k_1 clues from the first $n_1 + k_1$ clues they will not be able to recover the first element. This is also the case for all the other elements of the secret. So for the first element the device indeed has to listen to n_1 clues and then n_2 more for the second one and so on. So the information gain happens accordingly. Hence it does follow information gain function the user has chosen to construct the cues.

For example, let us consider a legitimate user which has not been listening to the clues being broadcast from the beginning. However from the table 5.1 we can see that the clues are broadcast so that evolving secrets is supported. As we have discussed before, each column represents the clues which for a user which starts listening at a unique time. The secrets obtained from each column will be unique. In table 5.1, X's are the clues for a user which starts listening from time instant 3. When that user listens till $C_{3,1}$ from wen it has started, they should be able to gather enough clues for recovering. They are allowed to miss at most k_1 clues in this interval so in practice, a legitimate user who has been able to listen to $n_1 - k_1$ by now should be able to construct the first element. Now in order to obtain enough clues for the next element, the user has to listen till $C_{3,2}$.This means that if the aforesaid legitimate user obtains enough clues by this interval, they will be able to compute two elements of the secret key. After $C_{3,1}$, the legitimate user can listen to $n_2$more clues till $C_{3,2}$. Among these

Table 5.1 Clues for a user which starts at time instant 3

	Clues for 1st element					Clues for 2nd element					Clues for 3rd element					…	Clues for mth element				
	•					•					•					…	•				
	•	•				•	•				•	•				…	•	•			
	•	•	X			•	•	X			•	•	X			…	•	•	X		
	•	•	X	•		•	•	X	•		•	•	X	•		…	•	•	X	•	
	•	•	X	•	…	•	•	X	•	…	•	•	X	•	…	…	•	•	X	•	…
	•	•	X	…	…	•	•	X	…	…	•	•	X	…	…	…	•	•	X	…	…
	…	…	…	…	…	•	•	X	…	…	•	•	X	…	•	…	•	•	X	…	•
	…	…	…	…	…	•	•	X	…	…	•	•	X	…	•	…	•	•	X	…	•
	•	•	$\boxed{\text{X}}$	…	•	•	•	$\boxed{\text{X}}$	…	…	•	•	$\boxed{\text{X}}$	…	•	…	•	•	$\boxed{\text{X}}$	…	•
		•	X	…	•	•	•	X	…	…	•	•	X	…	•	…	•	•	X	…	•
$C_{3,1} \rightarrow$			X	…	•	…	…	…	…	…	•	•	X	…	•	…	•	•	X	…	•
				•	•	…	…	…	…	…	•	•	X	…	•	…	•	•	X	…	•
					•	•	•	X	…	•	•	•	X	…	•	…	•	•	X	…	•
						•	•	X	…	•	•	•	X	…	•	…	•	•	X	…	•
						•	•	X	…	•	•	•	X	…	•	…	•	•	X	…	•
						•	•	X	…	•	•	•	X	…	•	…	•	•	X	…	•
							•	X	…	•	•	•	X	…	•	…	•	•	X	…	•
$C_{3,2} \rightarrow$								X	…	•	•	•	X	…	•	…	•	•	X	…	•
									•	•	•	•	X	…	•	…	•	•	X	…	•
										•	•	•	X	…	•	…	•	•	X	…	•
											•	•	X	…	•	…	•	•	X	…	•
											•	•	X	…	•	…	•	•	X	…	•
											…	…	…	…	…	…	•	•	X	…	•
											…	…	…	…	…	…	•	•	X	…	•
											•	•	X	…	•	…	•	•	X	…	•
											•	•	X	…	•	…	•	•	X	…	•
												•	X	…	•	…	•	•	X	…	•
$C_{3,3} \rightarrow$													X	…	•	…	•	•	X	…	•
														•	•	…	•	•	X	…	•
															•	…	•	•	X	…	•
																…	•	•	X	…	•
																…	•	•	X	…	•
																…	•	•	X	…	•
																…	…	…	…	…	…
																…	•	•	X	…	•
																…	•	•	X	…	•
																…		•	X	…	•
																…			X	…	•
																…				•	•
$C_{3,m} \rightarrow$																…					•

new clues they can miss k_2 clues. Hence the total number of clues which they are allowed to miss by now is $k_1 + k_2$.

Since our protocol satisfies the information leakage function, loosing k_1 consecutive clues in the time interval $[0\ C_{3,1}]$ will not have to same effect as loosing it in the interval $[C_{3,2}\ C_{33}]$. However since one not restricted by the ways to decide the redundancies, they can design the k_i's in such a way that it fulfills their condition.

Our protocol is autonomous :

Our protocol as we have sen from the construction does not depend on any third party. It is enables the legitimate parties to establish a key among themselves provided all of them follow some pre-defined rules.

There is no need for a built-in clock :

The clues are broadcast at regular intervals of time. There is no need for a clock to exist for this to happen. So the main device controls the length of the time interval in which it would broadcast the clues. The listening parties do not have to have any clock which is synchronized with that of the main device. Even if they do not have any sense of time, they are supposed to listen to the clues which is being broadcast at regular intervals. So listening to enough number of clues automatically means spending time with the main device.

The clues broadcast have error correcting properties :

The redundant clues can serve as error correcting clues as well. While implementation the user can chose the redundant number in such a way that it takes allows the legitimate user to miss clues due to possible noise in the wireless network environment. Some more redundant clues can be added, so that if an adversary tampers with a clue which is being broadcast, the legitimate user can still recover the secret.

The secrecy of the key does not depend on the design of the protocol. The parameters used in the implementation would be public knowledge.

Considering all the above arguments we can say that our protocol is correct and so is our claims.

5.2 From adversary's point of view

An adversary is one who will not be able to spend the same amount of quality time as a legitimate user. As we discussed earlier, an adversary will be able to listen to at most m number of clues and then will miss at least p number of clues. We will find out the feasible values for m and p for our protocol. It is possible from the value of m and p to determine whether our protocol could be used, given a certain environment.

5.2.1 Calculating m and p

Let us go back to the lemma we discussed earlier. We can see the conditions when we cannot get a secure and robust protocol. So we need to determine the condition when there will exist a protocol :

1. $n - k > a \times m$

2. $k < (a - 1) \times p$, $\forall a$ where $a \geq 1$ and $a \in I$.

The optimal value of m denotes the maximum number of consecutive clues which is possible for an adversary to obtain. This should be followed by missing at least p number of clues.

According to our construction we have different number of clues to obtain different element of the secret key : n_1, $n_1 + n_2$, $n_1 + n_2 + n_3$, $n_1 + n_2 + n_3 + n_4$ and so on. The redundant or extra clues are also different : k_1, k_2, k_3, k_4 and so on.

The value of m can be at most $n_1 - k_1 - 1$. This is because the adversary should not be able to spend as much time as a legitimate user to obtain clues.

For $a > 1$, we can find the maximum value of m for (n_1, k_1). And we will find the corresponding feasible values of p for : $(n_1 + n_2 + \ldots + n_i, k_1 + k_2 + \ldots + k_i | 1 < i \leq t)$. Among all the values of p, we will take the least feasible value which satisfies the condition in theorem 2 for all the (n, k) pairs. This is the optimal values of p.

Feasible values of m_i and p_i would be those which fulfill all the conditions for existence of a robust and secure protocol. We can elaborate the required conditions to make it simpler :-

1. $a_i > 1$
2. $n_i - k_i > a_i \times m_i$
3. $p_i > \frac{k_i}{a_i - 1}$

We would prefer our m_i to be as big a possible. Hence we will find the highest value of m_i for which conditions 1 and 2 are satisfied. That means we will find the largest m_i such that $\frac{n_i - k_i}{m_i} > 1$. Then we find the exact value of a_i for which conditions 1 and 2 are satisfied. For values of a_i larger than this, we find the value of p_i for which condition 3 is satisfied.

In this way we find the feasible values of m_i and p_i for all n_i's and k_i's used in the protocol. Then we use the least value of m_i,and the corresponding value of p_i and use them as m and p.

So in this way we can find out what kind of adversary the protocol can withstand. We have plotted the feasible regions for the values of m and p for three different pairs of (n, k). Figure 5.1 shows the feasible region when $n = 19$ and $k = 8$. Figure 5.2 shows the feasible region when $n = 14$ and $k = 5$.Figure 5.3 shows the feasible region when $n = 8$ and $k = 5$.

5.2.2 Information loss function

Information gain is the ratio of the obtained clues to the total number of required clues. The information loss function shows the loss in information gain by the adversary when they miss clues against time.

The total information required by a legitimate user to obtain all the clues is $n_1 + n_2 + \ldots n_t$. Now for the first clue there is a redundancy of k_1 clues, so the number of clues broadcast will be n_1. Similarly for the second clue it should be $n_1 + n_2$ and so on. Hence when we consider from the adversary's point of view – there will be no effect when they miss the first $k_1$clues. But after that, since we are using Shamir's secret sharing scheme, it will be impossible for the listening device to recover that element. Similar thing will happen for the second element. So now we will obtain a decreasing step function as shown in the following graph. This function is decreasing and is actually the inverse of the approximation of the information gain function.

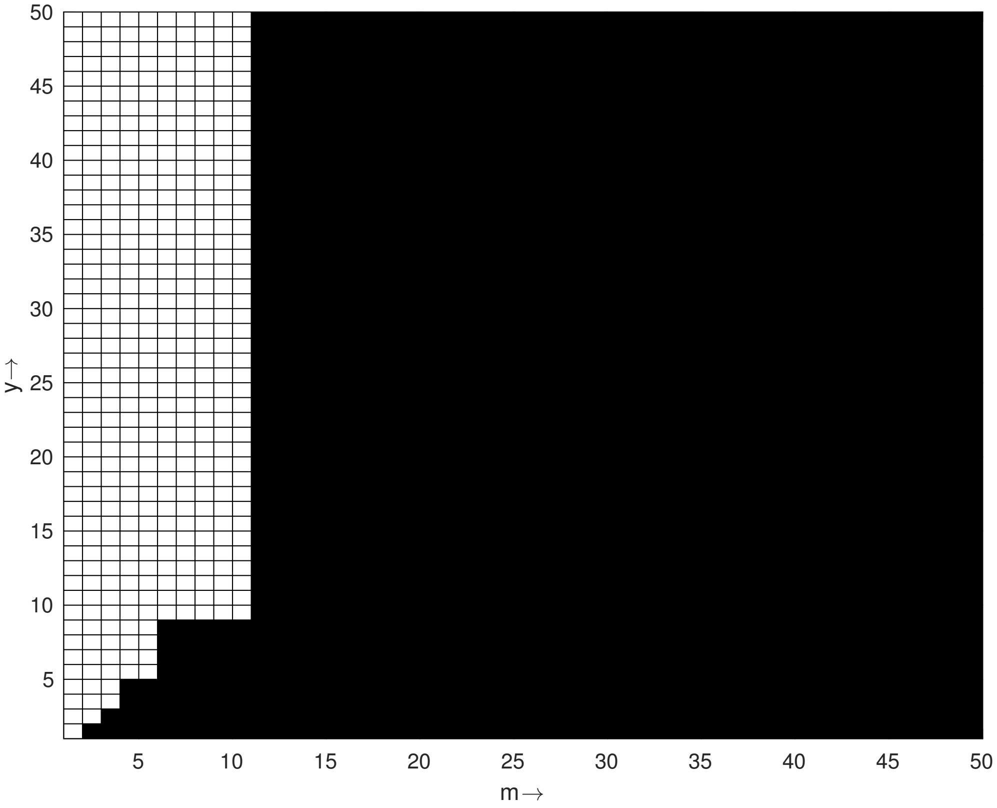

Figure 5.1 Feasible region for $n = 19$ and $k = 8$

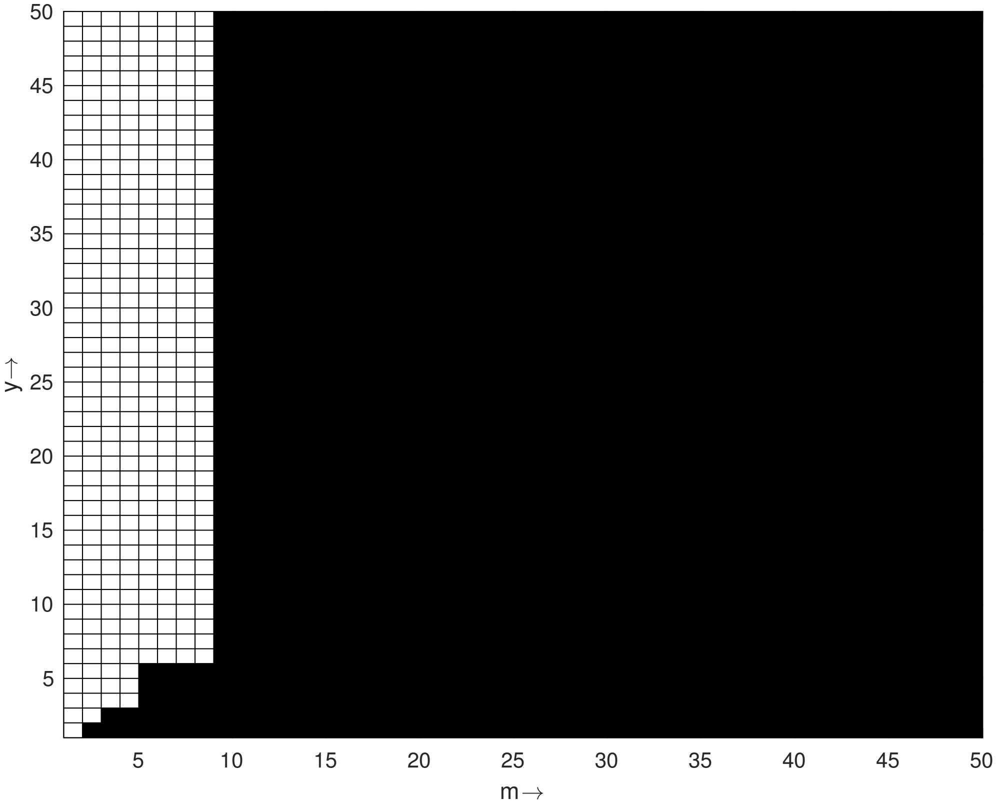

Figure 5.2 Feasible region for $n = 14$ and $k = 5$

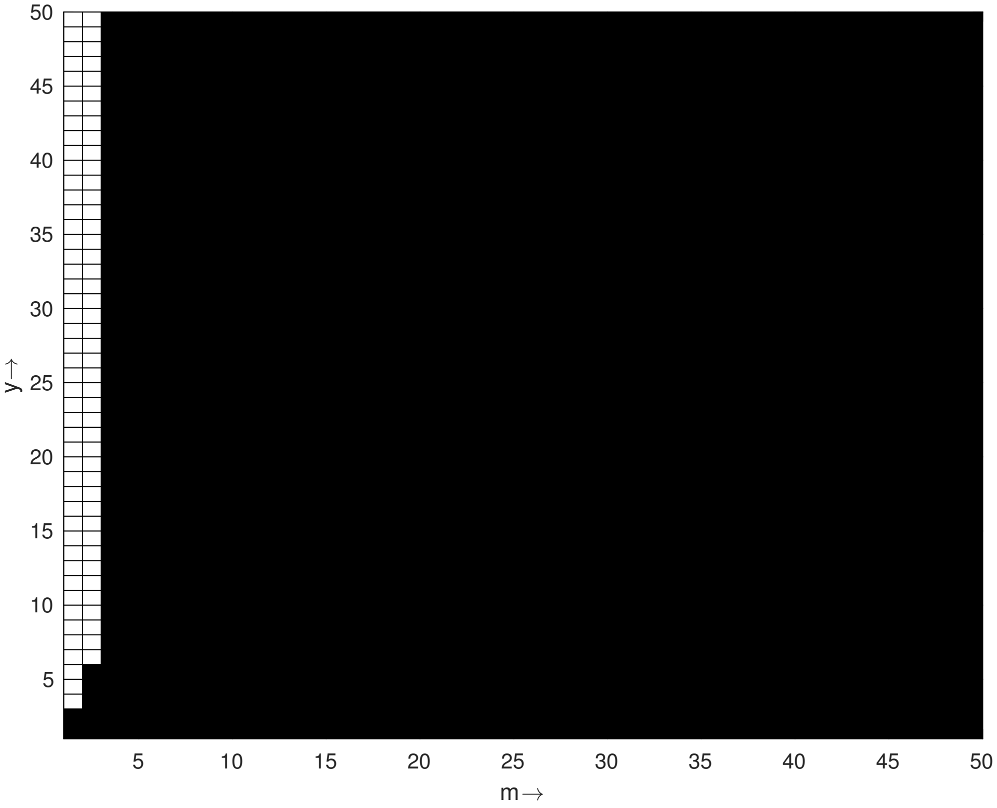

Figure 5.3 Feasible region for $n = 8$ and $k = 5$

5.3 Size of clues

We have seen how the clues have been constructed in the previous chapter. We know that at any given instant t, one element of the clue will look like this :$C_{1,1} = \{r_i, p_1(r_i), p_2(r_i, \ldots p_t(r_i))|r_i \in \mathbb{F}_2, 1 \leq i \leq n_1 + n_2 + \ldots + n_t\}$. Now since we have an evolving secret, the main device has to send similar clues for users which will start from other time instances. So at a particular time multiple such clues will be broadcast. Hence the size of each clue to be broadcast can be an issue. However we can reduce this size considerably.

We are using polynomials over a finite field for construction of the elements in the clues. We can take care of the size of clues while we choose our polynomials.There are points on the curve which makes will finally determine the curve. So while constructing the polynomials we can choose a few points which will overlap with some other curve in the field. There can be two ways of doing this :

Let us take an example of one element of the clue in one time instant $C_{1,1}$. We have seen how they have been constructed. While choosing p_2 we can make sure that $p_2(r_i) = p_1(r_i)$ for the first n_1 clues. It will change for the clues beyond that.

From the table 5.2 we can see the symbols in red. The broadcasting device can only broadcast these clues. This can reduce the size of the clues considerably.

Similarly we can reuse the clues which are meant of devices listening at different time instances. While sending the non-deterministic points we will just send one set of clues.

Table 5.2 Reduced clues for a user

	Clues for 1st element : n_1					Clues for 2nd element : n_2					Clues for 3rd element : n_3					...	Clues for mth element : n_m				
	●					●					●					...	●				
	●	●				●	●				●	●				...	●	●			
	●	●	µ			●	●	µ			●	●	µ			...	●	●	µ		
	●	●	µ	●		●	●	µ	●		●	●	µ	●		...	●	●	µ	●	
	●	●	µ	●	...	●	●	µ	●	...	●	●	µ	●	...	...	●	●	µ	●	...
	●	●	µ	...	...	●	●	µ	...	...	●	●	µ	...	...	...	●	●	µ	...	...
	...	...	...	...	...	●	●	µ	...	...	●	●	µ	...	●	...	●	●	µ	...	●
	...	...	...	...	...	●	●	µ	...	...	●	●	µ	...	●	...	●	●	µ	...	●
	●	●	µ	...	●	●	●	µ	...	...	●	●	µ	...	●	...	●	●	µ	...	●
		●	µ	...	●	●	●	µ	...	...	●	●	µ	...	●	...	●	●	µ	...	●
$C_{3,1}$ →			µ	...	●	...	...	...	...	...	●	●	µ	...	●	...	●	●	µ	...	●
				●	●	...	...	...	...	...	●	●	µ	...	●	...	●	●	µ	...	●
					●	●	●	µ	...	●	●	●	µ	...	●	...	●	●	µ	...	●
						●	●	µ	...	●	●	●	µ	...	●	...	●	●	µ	...	●
						●	●	µ	...	●	●	●	µ	...	●	...	●	●	µ	...	●
						●	●	µ	...	●	●	●	µ	...	●	...	●	●	µ	...	●
							●	µ	...	●	●	●	µ	...	●	...	●	●	µ	...	●
$C_{3,2}$ →								µ	...	●	●	●	µ	...	●	...	●	●	µ	...	●
									●	●	●	●	µ	...	●	...	●	●	µ	...	●
										●	●	●	µ	...	●	...	●	●	µ	...	●
											●	●	µ	...	●	...	●	●	µ	...	●
											●	●	µ	...	●	...	●	●	µ	...	●
											...	...	...	...	...	...	●	●	X	...	●
											...	...	...	...	...	...	●	●	µ	...	●
											●	●	µ	...	●	...	●	●	µ	...	●
											●	●	µ	...	●	...	●	●	µ	...	●
												●	µ	...	●	...	●	●	µ	...	●
28 $C_{3,3}$ →													µ	...	●	...	●	●	µ	...	●
														●	●	...	●	●	µ	...	●
															●	...	●	●	µ	...	●
																...	●	●	µ	...	●
																...	●	●	µ	...	●
																...	●	●	µ	...	●
																...	...	...	...	...	...
																...	●	●	µ	...	●
																...	●	●	µ	...	●
																...		●	µ	...	●
																...			µ	...	●
																...				●	●
$C_{3,m}$ →																...					●

Table 5.3 Reduced distribution of shares for evolving secret

	●					
	●	●				
	●	●	●			
	●	●	●	●		
	X	●	●	●	●	
	X	X	●	●	●	●
	X	X	X	●	●	●
$t_0 \longrightarrow (S_0)$:	X	X	X	X	●	●
$t_1 \longrightarrow (S_1)$:		X	X	X	X	●
$t_2 \longrightarrow (S_2)$:			X	X	X	X
$t_3 \longrightarrow (S_3)$:				X	X	X
$t_4 \longrightarrow (S_4)$:					X	X
$t_5 \longrightarrow (S_5)$:						X

The elements in table 5.3, one row of the box can be same. Hence we can send one element instead of the whole row.

However while these methods reduce the size of the clues considerably, it will have an adverse effect on the security of the protocol. So now, the polynomials will intersect with each other. Hence in case of a brute force attack, the search space of the adversary reduces as well.

BIBLIOGRAPHY

[1] R. Ahlswede and I. Csiszar. Common randomness in information theory and cryptography. ii. cr capacity. *IEEE Trans. Inf. Theor.*, 44(1):225–240, September 2006.

[2] Rudolf Ahlswede and Imre Csiszár. Common randomness in information theory and cryptography - part II: CR capacity. *IEEE Trans. Information Theory*, 44(1):225–240, 1998.

[3] George T. Amariucai, Clifford Bergman, and Yong Guan. An automatic, time-based, secure pairing protocol for passive RFID. In *RFID. Security and Privacy - 7th International Workshop, RFIDSec 2011, Amherst, USA, June 26-28, 2011, Revised Selected Papers*, pages 108–126, 2011.

[4] Mihir Bellare, Anand Desai, David Pointcheval, and Phillip Rogaway. *Relations among notions of security for public-key encryption schemes*, pages 26–45. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998.

[5] Charles H. Bennett, Gilles Brassard, Claude Crépeau, and Ueli M. Maurer. Generalized privacy amplification. *IEEE Trans. Information Theory*, 41(6):1915–1923, 1995.

[6] Imre Csiszár and Prakash Narayan. Secrecy capacities for multiterminal channel models. *IEEE Trans. Information Theory*, 54(6):2437–2452, 2008.

[7] Imre Csiszr and Jnos Krner. Broadcast channels with confidential messages. *IEEE Transactions on Information Theory*, 24(3):339–348, 1978.

[8] Wyner A. D. The wire-tap channel. *The Bell System Technical Journal*, 54(8):1355 – 1387, October 1975.

[9] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Trans. Inf. Theor.*, 22(6):644–654, September 2006.

[10] Yevgeniy Dodis, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In *Computing Research Repository - CORR*, volume 38, pages 523–540, 01 2004.

[11] A.S. Fraenkel and Y. Yesha. Complexity of problems in games, graphs and algebraic equations. *Discrete Applied Mathematics*, 1(1):15 – 30, 1979.

[12] Rosario Gennaro and Shai Halevi. More on key wrapping. In *Selected Areas in Cryptography, 16th Annual International Workshop, SAC 2009, Calgary, Alberta, Canada, August 13-14, 2009, Revised Selected Papers*, pages 53–70, 2009.

[13] A. Juels. Secret sharing in cryptographic devices via controlled release of plaintext information. July 8 2014. US Patent 8,774,410.

[14] Ari Juels, Ravikanth Pappu, and Bryan Parno. Unidirectional key distribution across time and space with applications to rfid security. In *Proceedings of the 17th Conference on Security Symposium*, SS'08, pages 75–90, Berkeley, CA, USA, 2008. USENIX Association.

[15] Bhavana Kanukurthi and Leonid Reyzin. Key agreement from close secrets over unsecured channels. In *Advances in Cryptology - EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. Proceedings*, pages 206–223, 2009.

[16] Marc Langheinrich and Remo Marti. Practical minimalist cryptography for RFID privacy. *IEEE Systems Journal*, 1(2):115–128, 2007.

[17] James L. Massey. Shift-register synthesis and BCH decoding. *IEEE Trans. Information Theory*, 15(1):122–127, 1969.

[18] Suhas Mathur, Narayan M, Chunxuan Ye, and Alex Reznik. Radio-telepathy: extracting a secret key from an unauthenticated wireless channel. In *In MobiCom ?08*, pages 128–139, 2008.

[19] U. M. Maurer. Secret key agreement by public discussion from common information. *IEEE Trans. Inf. Theor.*, 39(3):733–742, May 1993.

[20] Rene Mayrhofer and Hans Gellersen. Shake well before use: Authentication based on accelerometer data. In *Pervasive Computing, 5th International Conference, PERVASIVE 2007, Toronto, Canada, May 13-16, 2007, Proceedings*, pages 144–161, 2007.

[21] Alfred J. Menezes, Scott A. Vanstone, and Paul C. Van Oorschot. *Handbook of Applied Cryptography.* CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 1996.

[22] Ralph C. Merkle. Secure communications over insecure channels. *Commun. ACM*, 21(4):294–299, April 1978.

[23] Renato Renner and Stefan Wolf. Simple and tight bounds for information reconciliation and privacy amplification. In *Advances in Cryptology - ASIACRYPT 2005, 11th International Conference on the Theory and Application of Cryptology and Information Security, Chennai, India, December 4-8, 2005, Proceedings*, pages 199–216, 2005.

[24] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[25] Phillip Rogaway and Thomas Shrimpton. Deterministic authenticated-encryption: A provable-security treatment of the key-wrap problem. *IACR Cryptology ePrint Archive*, 2006:221, 2006.

[26] Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, November 1979.

[27] C. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal, Vol 28, pp. 656?715*, Oktober 1949.

[28] Jon W. Wallace and Rajesh K. Sharma. Automatic secret keys from reciprocal MIMO wireless channels: measurement and analysis. *IEEE Trans. Information Forensics and Security*, 5(3):381–392, 2010.

[29] Robert D. Wilson, David Tse, and Robert A. Scholtz. Channel identification: Secret sharing using reciprocity in ultrawideband channels. *IEEE Trans. Information Forensics and Security*, 2(3-1):364–375, 2007.

[30] Sheng Xiao, Weibo Gong, and Donald F. Towsley. Secure wireless communication with dynamic secrets. In *INFOCOM 2010. 29th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies, 15-19 March 2010, San Diego, CA, USA*, pages 1568–1576, 2010.

[31] Chunxuan Ye, Suhas Mathur, Alex Reznik, Yogendra Shah, Wade Trappe, and Narayan B. Mandayam. Information-theoretically secret key generation for fading wireless channels. *IEEE Trans. Information Forensics and Security*, 5(2):240–254, 2010.