Decentralization and Proactive Secret-sharing Techniques for Resource-constraint Devices

A thesis submitted during 2019 to the University of Hyderabad in partial fulfillment of the award of a Ph.D. degree in the School of Computer and Information Sciences

by

N Chaitanya Kumar

Reg.No. 14MCPC09

School of Computer and Information Sciences

University of Hyderabad

P.O. Central University

Hyderabad – 500046, India

April - 2019

CERTIFICATE

This is to certify that the thesis entitled **Decentralization and Proactive Secret-sharing Techniques for Resource-constraint Devices** submitted by **N Chaitanya Kumar** bearing **Reg. No. 14MCPC09** for fulfillment of the requirements for the award of **Doctor of Philosophy in Computer Science** is a bonafide work carried out by him under my supervision and guidance.

The thesis is free from plagiarism and has not been submitted previously in part or in full to this or any other University or Institution for the award of any degree or diploma. The student has the following publications before submission of the thesis for adjudication and has produced the evidence for the same.

1. N CHAITanya KUMAR, ABDUL BASIT, PriyADARSHI SINGH, V Ch VENKAIAH, Y. V. SUBBA RAO . **Node Authentication using BLS Signature in Distributed PKI based MANETs**. *International Journal of Network Security & Its Applications* , volume 9, pages 11-19, 2017.
2. N CHAITanya KUMAR, ABDUL BASIT, PriyADARSHI SINGH, V. Ch. VENKAIAh. **Proactive Secret Sharing For Long Lived MANETs Using Elliptic Curve Cryptography**. *International Conference on Inventive Computing and Informatics (ICICI 2017)*, November 23-24, 2017.
3. N CHAITanya KUMAR, ABDUL BASIT, PriyADARSHI SINGH, V. Ch. VENKAIAh. **Lightweight Cryptography for Distributed PKI Based MANETs**. *International Journal of Computer Networks & Communications* , volume 10, pages 63-69, 2018.
4. N CHAITanya KUMAR, ABDUL BASIT, PriyADARSHI SINGH, V. Ch. VENKAIAh. **Subgroup Operations In Identity Based Encryption (IBE) Using Weil Pairing In Decentralized Networks**. *International Journal of Network Security*, volume 21, 2019.

Further, the student has passed the following courses towards fulfillment of coursework requirement for Ph.D.

Course Code	Name	Credits	Pass/Fail
CS 801	Data Structures and Algorithms	4	Pass
CS 802	Operating Systems and Programming	4	Pass
CS 811	High Performance Computing	4	Pass
IT 811	Secure Computing	4	Pass

Dr. V. Ch. Venkaiah
Supervisor
School of Computer and Information Sciences
University of Hyderabad

Prof. K.Narayana Murthy
Dean
School of Computer and Information Sciences
University of Hyderabad

DECLARATION

I, **N. Chaitanya Kumar**, hereby declare that this thesis entitled **Decentralization and Proactive Secret-sharing Techniques for Resource-constraint Devices** submitted by me under the guidance and supervision of **Dr. V. Ch. Venkaiah** is a bonafide research work. I also declare that it has not been submitted previously in part or in full to this University or any other University or Institution for the award of any degree or diploma.

Date :

Name: N Chaitanya Kumar

Signature of the Student

Reg. No.: 14MCPC09

Decentralization and Proactive Secret-sharing Techniques for Resource-constrained Devices

1 Introduction

A computer network connects computing devices and support communication between these devices. Networks use different schemes to create these connections. A centralized network is one such scheme commonly used to create and support a network. A centralized network implies a central focus of control i.e through effective management it controls speed, flexibility and reorganization within the network. While a centralization network does have benefits, this type of network also poses several significant challenges. The possibility of a complete failure is a major concern for networks that use a centralized scheme. Another disadvantage of the centralized scheme is related to access. In many applications, individuals connecting to the network have different needs. Centralized systems require users to connect to network uniformly using the same process. This type of network may not support the flexibility required by multiple users with varied needs. Security is a critical component of any networked system. Centralized systems create centralized targets. Regardless of the security measures taken to protect the network, a centralized system is easier to attack than the decentralized system. As a solution to these problems decentralized network schemes have emerged. In decentralized networks no single system controls the network access or security or other issues like data management. While adopting decentralized schemes to any network, two main challenges, namely, to provide user privacy and to implement equal usability and functionality without centralized components, have to be met. With the advent of the Internet of Things(IoT), many applications are developed for resource constrained

devices but they all are lacking security. In this thesis, we come up with a set of security protocols that can be adopted to resource constraint devices. The focus of designing our protocols is to make them suitable for decentralized networks that are formed of resource constrained devices. So the proposed protocols establish a secure communication among the resource-constrained devices using lightweight protocols. Also, these protocols make the network secure for longer period of time using the concept of proactive secret sharing.

2 Thesis Contribution and Chapter Organization

The computer networks that are formed of computing devices are divided into two categories - centralized and decentralized networks. In centralized networks the entire network is managed by a central authority. The problem with the centralized network is the connection between devices will have to exclusively go through the centralized authority, even if they happen to be a few feet away and the centralized authority will not be able to respond to the growing needs of the network. In decentralized networks the entire network is managed by all the nodes of the network without the help of any centralized authority. In decentralized networks security is a major concern. Since nodes will be passing data between each other, it is difficult to guarantee the integrity of each packet because any malicious node could easily replace one packet with another and the receiver would never be able to tell the difference. So these networks require a different set of protocols than the traditional security protocols. To secure a decentralized network, each node should be the part of the security system and should adopt the security protocols from time to time. In this thesis we present a comprehensive set of protocols that can be used for secure communication in decentralized networks that are formed of resource-constrained devices. The whole

network is managed by only its members without the need of external or centralized authority. The concept of secret sharing is used to distribute the shares of the secret key among the network members and the secret can only be reconstructed when a threshold number of users collaborate. We also used the concept of proactive secret sharing to change the shares of the users frequently thus making it difficult for the attacker to find the secret. The network users can communicate among themselves securely using either Public Key Infrastructure (PKI) or Identity Based Encryption (IBE) policies. Firstly, in PKI, all the users have their own public and private keys and a certificate authority maintains the public keys of the users by issuing a certificate in order to avoid the abuse of the public keys which are openly available. In our protocol, since we don't have any Centralized Authority(CA), the job of the CA is distributed among the users. We used lightweight signature i.e BLS Signature for signing the certificates. Once all the users have their own public and private keys then they need a cryptographic algorithm to communicate securely among themselves. We used a lightweight Tiny Encryption Algorithm (TEA) for this purpose. It is a very simple symmetric algorithm and works much faster thus making it suitable for the network of resource-constrained devices. Secondly, in case of IBE, the public identity of user is taken as the public key and the private key can be obtained by a trusted third party called Private Key Generator(PKG). In our protocol, we decentralized the job of PKG and the users themselves maintain the PKG. Finally, we have also proposed a protocol that supports the operations only within the subgroup of users.

The thesis is divided into five chapters. Chapter 1 titled Introduction and Background introduces the problems addressed in this thesis and gives the background necessary for the proposed protocols. Background consists of the basics of elliptic curve cryptography and bilinear pairings. It explains in detail about how bilinear pairings are useful in cryptography and the detailed mathematics behind it. Chapter 2 explains the use of BLS sig-

nature to sign a certificate. Here all the users of the distributed network are assumed to have their own public key and private key. To authenticate their public key each user prepares a certificate linking their public key to their identity and asks the other users to sign on it. This certificate is later used to avoid impersonation attacks. In chapter 3, the concept of proactive secret sharing is adopted to increase the lifetime of the security of the network. Over the time an attacker may try to manage the shares of the participants of a network. So the share of an user must be frequently changed without modifying the secret. In chapter 4, the concept of lightweight cryptography is introduced. Since the proposed protocols are for the resource constrained devices, a lightweight crypto algorithm is suitable to communicate between the network devices. Here we used Tiny Encryption Algorithm (TEA) which is very fast and relatively secure compared to other algorithms. TEA is a symmetric algorithm i.e both encryption and decryption routines use the same key. Since all the users are having their own shares of the network secret key, they use Diffie-Hellman key exchange algorithm to generate the secret key for the TEA algorithm. In chapter 5, the secure communication among a subgroup of users is discussed. By considering IBE scenario, a sub group of users can form their own network and can communicate securely. A subgroup contains a public key and a private key. The users of a subgroup holds a share of the private key. To send a message to this subgroup, it is encrypted with the public key of the subgroup. To decrypt a message threshold number of users collaborate to get the original message.

3 Publications of Thesis

1. **N Chaitanya Kumar**, Abdul Basit, Priyadarshi Singh, V Ch Venkaiah, Y. V. Subba Rao . **Node Authentication using BLS Signature in Distributed PKI based MANETs**. *International Journal of Network Security & Its Applications* , volume 9, pages 11-

19, 2017.

2. **N Chaitanya Kumar**, Abdul Basit, Priyadarshi Singh, V. Ch. Venkaiah. **Proactive Secret Sharing For Long Lived MANETs Using Elliptic Curve Cryptography**. *International Conference on Inventive Computing and Informatics (ICICI 2017)*, November 23-24, 2017.

3. **N Chaitanya Kumar**, Abdul Basit, Priyadarshi Singh, V. Ch. Venkaiah. **Lightweight Cryptography for Distributed PKI Based MANETs**. *International Journal of Computer Networks & Communications* , volume 10, pages 63-69, 2018.

4. **N Chaitanya Kumar**, Abdul Basit, Priyadarshi Singh, V. Ch. Venkaiah. **Subgroup Operations In Identity Based Encryption (IBE) Using Weil Pairing In Decentralized Networks**. *International Journal of Network Security*, volume 21, 2019.

Acknowledgements

The accomplishment of doctoral thesis and degree is a result of collective support and blessings of various people during my journey of Ph.D. I feel overwhelm and pleasure to acknowledge each one for the kind of contribution provided to me throughout my life.

The foremost is my advisor, Dr. V. Ch. Venkaiah who has not only taught me science but the facts behind and beyond science. Sir has endowed me with a sacred vision to pursue productive full time research in computer science. The importance of bearing an optimistic attitude and crystal clear concept of the subject is a boon conferred by sir toward shaping my research questions during my years of Ph.D. I am extremely obliged to sir for making himself always available for rigorous scientific discussions in spite of his busy schedule. Those insightful inputs by sir are the substantial asset to build the carrier further. I wish to acknowledge my sir for being the guiding light in my dark days of research. The periodical scientific counseling and supervision when I was at the initial phase of my Ph.D carrier is accountably assisted by sir. I genuinely acknowledge sir for taking out his valuable time for undertaking detailed analysis of the given research problem and giving me precise comments.

I would like to thank my DRC memebrs, Prof. K. Narayana Murthy and Dr. Salman Abdul Moiz, for their valuable suggestions. I extend admiration to my colleagues, Mr. Abdul Basith and Mr. Priyadarshi for building an enthusiastic environment and engaging in critical discussions related to research in both inside and outside the lab. I thank all my Ph.D. classmates for showering the fun times in hostel days stay.

I extend deepest salutation to my whole family members for availing me every needful facilities and emotional assistance. Their support imprints me the property to be flexible as water and rigid as rock to balance my

twisted research life. I wish to express that I am indebted and owe so much to my mother, Mrs.Sathyavathi and father, Mr. Ramanaiah such that my depth of expression acknowledgement will always be small when compare to their contributions of parenting. The unconditional sacrifices and determined motivation is what I have learnt from them at every stage of my life and this has enforced me to become a responsible citizen at first place. With an immense pride this doctoral dissertation is dedicated to my parents for providing me the strong roots to standalone and sustain in such long journey of life.

Finally, I bow to The Almighty for being a source of unknown power and strength in my life.

N Chaitanya Kumar

Abstract

The computer networks that are formed of computing devices are divided into two categories - centralized and decentralized networks. In centralized networks the entire network is managed by a central authority. The problem with the centralized network is the connection between devices will have to exclusively go through the centralized authority, even if they happen to be a few feet away and the centralized authority will not be able to respond to the growing needs of the network. In decentralized networks the entire network is managed by all the nodes of the network without the help of any centralized authority. In decentralized networks security is a major concern. Since nodes will be passing data between each other, it is difficult to guarantee the integrity of each packet because any malicious node could easily replace one packet with another and the receiver would never be able to tell the difference. So these networks require a different set of protocols than the traditional security protocol. To secure a decentralized network, each node should be the part of the security system and should adopt the security protocols from time to time. In this thesis we present a comprehensive set of protocols that can be used for secure communication in decentralized networks that are formed of resource constrained devices. The whole network is managed by only its members without the need of external or centralized authority. The concept of secret sharing is used to distribute the shares of the secret key among the network members and the secret can only be reconstructed when a threshold number of users collaborate. We also used the concept of proactive secret sharing to change the shares of the users frequently thus making it difficult for the attacker to find the secret. The network users can communicate among them selves securely using either Public Key Infrastructure (PKI) or Identity Based Encryption (IBE) policies. Firstly, in PKI, all the users have their own public and

private keys and a certificate authority maintains the public keys of the users by issuing a certificate in order to avoid the abuse of the public keys which are openly available. In our protocol, since we don't have any Centralized Authority(CA), the job of the CA is distributed among the users. We used lightweight signature i.e BLS Signature for signing the certificates. Once all the users have their own public and private keys then they need a cryptographic algorithm to communicate securely among themselves. We used a lightweight Tiny Encryption Algorithm (TEA) for this purpose. It is a very simple symmetric algorithm and works much faster thus making it suitable for the network of resource constrained devices. Secondly, in case of IBE, the public identity of user is taken as the public key and the private key can be obtained by a trusted third party called Private Key Generator(PKG). In our protocol, we decentralized the job of PKG and the users themselves maintain the PKG. Finally, we have also proposed a protocol that supports the operations only within the subgroup of users.

Dedicated to my parents and family members, for everything what I am today and above all, devoted to The Almighty God!

Contents

CONTENTS

CONTENTS

CONTENTS

List of Figures

List of Tables

Chapter 1

Introduction

A computer network is the network formed of connected computing devices and establishes communication among these connected devices. These networks are created using different schemes. One of commonly used schemes is centralized network. A centralized network has a central authority which controls and effectively manages the networks. It also controls the speed of the network, flexibility and reorganization of the network. In addition to its advantages, centralization network also has several disadvantages. One of the major concerns while using the centralized scheme is the possibility of total failure of the network if the central system is failed. The centralized scheme also has a disadvantage related to accessing of the network. Different users will connect to the network with different needs depending on the application they want to use. Centralized systems provides a uniform way to connect to network using the same process. If the users have different needs then this this type of network is not flexible to handle their request. Another concern about the centralized network is security which is a critical in protecting any network system. Centralized networks are vulnerable because of their centralized systems which create centralized targets. Even with all the security measures provided to the centralized network, it is still more vulnerable then the decentralized system. As a solution to these problems a decentralized network scheme emerged. In the decentralized networks no single system controls the network access or security or other issues like data management and flexibility. While adopting the decentralized schemes to any network, two main challenges have to be met, to provide user privacy and to implement equal usability and functionality without centralized components. With the advent of the Internet of Things(IoT), many applications are

developed for resource constrained devices but they all are lacking security. In this thesis, we come up with a set of security protocols that can be adopted to resource constraint devices. The focus of designing our protocols is to make them suitable for decentralized networks that are formed of resource constrained devices and to establish a secure communication among them using lightweight protocols. These protocols also make the network secure for longer period of time using the concept of proactive secret sharing. The thesis is organized into five chapters. The first chapter involves the basics of elliptic curve cryptography and bilinear pairings. It explains in detail about how bilinear pairings are useful in cryptography and the detailed mathematics behind it. In chapter 2 we explained the use of BLS signature to sign the certificate. All the users of the distributed network have their own public and private keys. To authenticate their public key each user prepares a certificate linking their public key to their identity and asks the other users to sign on it. This certificate is later used to avoid impersonation attacks. In chapter 3, the concept of proactive secret sharing is adopted to enhance the network lifetime. Over the time the attacker may try to manage the shares of the network. So the share of an user must be frequently changed without modifying the secret. In chapter 4, the concept of lightweight cryptography is introduced. Since the proposed protocols are for the resource constrained devices, a lightweight crypto algorithm is suitable to communicate between the network devices. Here we used Tiny Encryption Algorithm (TEA) which is very fast and relatively secure compared to other algorithms. TEA is a symmetric algorithm i.e encryption and decryption are done with the same key. Since all the users are having their own network secret key shares, they generate the symmetric key for TEA algorithm using Diffie-Hellman key exchange. In chapter 5, we discussed how to perform the security operations with in a subgroup of users. By considering IBE scenario, a subgroup of users can form their own network and can communicate securely. The subgroup contains both public and private keys. The users of the subgroup have a share of network secret key. An encrypted message is send to this subgroup using the public key of the subgroup. To decrypt the message threshold number of users collaborate to get the original message.

1.1 TYPES OF CRYPTOGRAPHY:

The technique of applying mathematical algorithms to encrypt and decrypt the message is called cryptography. Cryptography can be used to send confidential information over the insecure channels (e.g the Internet), without the information being read by unauthenticated user and can only be given to the authenticated user. The security of any cryptographic algorithm depends on the underlying mathematical function used for encrypting and decrypting the data. A cryptographic algorithm encrypts a plain text message using a key. The same plain text message can be encrypted to different cipher texts by using different keys. The security of the encrypted data is purely based on the privacy of the key used and the cryptographic algorithm strength. There are several types of keys in cryptography which are intended to serve specific functionality and their meta-data will define the property of the key. Crytpographic systems are divided into two types - Asymmetric systems and Symmetric systems.

1.1.1 Symmetric Cryptography:

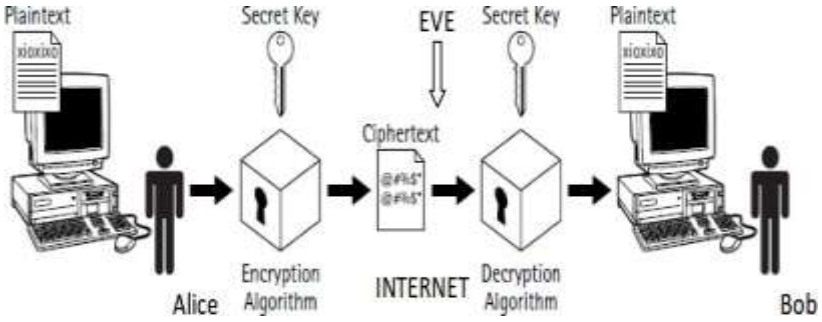
In this type of cryptography, the two parties sharing the secret initially agree on a key. Authentication is resolved only if the two parties tend to hold the same key. The sender encodes the message and sends the key to the receiver before encoding the plain text and the receiver also uses the same key to decrypt the cipher text.
But this system has some drawbacks:

- The keys must be exchanged securely prior to the exchange of actual messages.
- Each pair of users in the network require their own private keys resulting in need of a storage for the keys. Suppose the network has n users, there will be a need of $\frac{n.(n-1)}{2}$ keys in total.
- The problem with the symmetric cryptography is, if some one gets to know the key being used, All the messages that are encrypted with that key can be decrypted.
- Blowfish, RC4, AES, DES are some of the examples of the symmetric key cryptographic algorithms.

  Example for security vulnerability in symmetric cryptography: Assume that 2 persons namely Alice and Bob are communicating over an insecure channel

and there is Eve(enemy). Let us consider that Alice is sending an email to Bob. Eve is either reading the email that is sent to Bob or else she impersonates Alice to send an email to Bob.

Figure 1.1: Symmetric Cryptography



1.1.2 Asymmetric Cryptography:("Public key Cryptography")

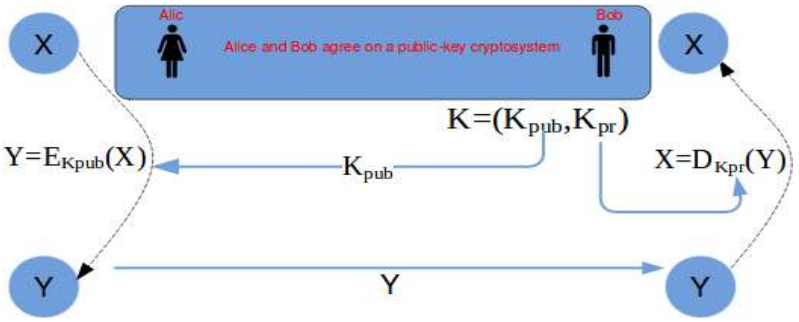
Asymmetric cryptography, popularly known as "public key cryptography" which was first introduced in the year 1976 by Diffie and Hellman. The first and foremost step here is to define two keys: Public and Private. The main intention of this is to use one key for encrypting the plain text and a different key to decrypt the cipher text. The encryption is done using the receiver's public key and the receiver's private key is used to decrypt the messages.

Example:

Let us consider that Bob and Alice wants to communicate. To encrypt a message Alice uses Bob's public key and generates the cipher text. This cipher text can only be decrypted using the private key of Bob. Despite being very efficient, this scheme has the following disadvantages:

- The mathematics used here is way too complicated when compared to symmetric schemes.
- It is slow due to large number of computations involved.
- RSA, ECC, DSA are examples for public key cryptography.

Figure 1.2: Asymmetric Cryptography



- The security of symmetric scheme increases exponentially with the increase in number of rounds which is not the case for asymmetric scheme. Therefore, the key length depends on various factors such as: algorithm used, amount of data being processed with the key, strength of security required.

One of the most efficient scheme, in asymmetric cryptography is the usage of elliptic curves.[36]

1.2 ELLIPTIC CURVES

"An elliptic curve is a curve consisting of a group of finite points of the form (x, y)". The group law is constructed geometrically. Elliptic curve is represented in **Weierstrass equation** form in the following way

$$y^2 = x^3 + cx + d$$

where c and d are real numbers.

1.2.1 Elliptic curve cryptography:(ECC)

"Elliptic curves" were first introduced to cryptography in the year 1986. It was a new cryptosystem proposed separately by Miller and Koblitz. This cryptosystem depends on the hardness of the Discrete Logarithmic Problem, in short DLP. These crypto systems are implemented by a group structure that is given by "Elliptic curve" represented on finite field. The group structure contains a set of all rational points that are defined

over an elliptic curve and an infinity point which is represented by O. The arithmetic operations performed on the underlying finite field are defined using the group operations like addition and multiplication. Let P be an elliptic curve point which is one of the basic building blocks of elliptic curve cryptography and k is an integer then the scalar point multiplication is the operation $k.P$. The basic idea is to self add P again and again $k - 1$ times which gives resultant point Q. The process of recovering the value k from Q and P is the basic idea of "the Elliptic Curve Discrete Logarithmic Problem(ECDLP)".

Why only elliptic curves?

- Elliptic curves are economic as they use fewer bits in comparison to classical methods like RSA to provide equivalent security.
- Implementation of elliptic curve cryptosystems requires less power consumption, smaller chip size and increase in speed.
- Other curves, for example hyper elliptic curves, are not used due to the limiting nature to support other operations. When operations like hashing are done on curves other than elliptic curves, forging can be done very easily and the security of data is at danger.
- These other curves also have a finite range of values and the key size is very less.[40]

Identity Point: The identity point is O. All the points apart from O are called finite points.

Discriminant: "The definition of elliptic curve also requires that the curve be non-singular. Geometrically, this means that the graph has no cusps, self-intersections, or isolated points". Mathematically, the discriminant is calculated as

"

$$\Delta = -16(4a^3 + 27b^2)$$

" **Example:**

Let us consider $y^2 = x^3 + 1$, an elliptic curve and its discriminant is calculated in the

following way

$a = 0$, $b = 1$ and therefore the discriminant is as follows:

$=\Delta = -16(4 * 0^3 + 27 * 1^2)$

$= -16(0 + 27)$

$= -16(27)$

$= -432.$

Based on the discriminant value, the curves are divided in to singular and non singular. All non zero discriminant curves are non-singular. so the curve mentioned above is non-singular.

Order of points: "If Q is a point lies on an elliptic curve, then its order is a positive integer m such that $[m]P = O$.

$[m]P = \underbrace{Q \oplus Q \oplus Q \oplus \oplus Q}_{\text{m times}} = O$

The order of Q is infinite, If there exist no such integer".

"The set of all points with finite order is a subgroup of the group of points. It is called the **torsion subgroup of E** and the group of points with order ∞, together with P_∞ (P_∞ is a point at infinity) is called the **non-torsion subgroup of E**." [40]

Example: $E: y^2 = x^3 + 1$ and $P = (0, 1)$. Here, a is 0 and b is 1. Here $y = 1$ is the tangent line and it intersects with E with a triple point. So $R = P$ (Here, R is tangent line). In particular, $2P = -R = -P$, and so $3P = O$.

Let the point be P(2,3) and now let us find the order. We know that "the gradient of the tangent (λ) at the point P is":

left. $\lambda = \frac{y_0 - y_1}{x_0 - x_1}$ *if* $P \neq Q$ OR

$\lambda = \frac{3x_1^2 + a}{2y_1}$ if P = Q

Where, λ is slope of straight line.

Point Addition:

"$y_3 = \lambda(x_1 - x_3) - y_1$ and $x_3 = \lambda^2 - x_1 - x_2$".

P+P=2P

$\lambda = \frac{3*2^2+0}{2*3} = 2$

$x_3 = 4 - 2 - 2 = 0$

$y_3 = 2(2 - 0) - 3 = -3 + 4 = 1$

Therefore, 2P=(0,1)

Similarly 3P is calculated as

3P=2P+P=(0,1)+(2,3)

$\lambda = \frac{3-1}{2-0} = 1$

$x_3 = 1 - 0 - 2 = -1$

$y_3 = 1(0 + 1) - 1 = -1 + 1 = 0$

Therefore, 3P=(-1,0)

4P=3P+P=(-1,0)+(2,3)

$\lambda = \frac{3-0}{2+1} = 1$

$x_3 = 1 + 1 - 2 = 0$

$y_3 = 1(-1 - 0) - 0 = -1$

Therefore, 4P=(0,-1)

5P=4P+P=(0,-1)+(2,3)

$\lambda = \frac{3+1}{2-0} = 2$

$x_3 = 4 - 0 - 2 = 2$

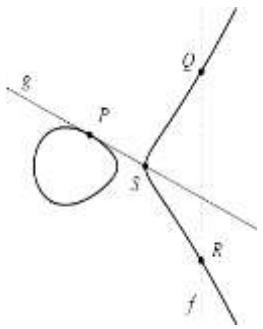
$y_3 = 2(0 - 2) + 1 = -3$

Therefore, 5P=(2,-3)=-P

Let us consider an elliptic curve point P(x,y) and its negative -P is represented as (x,-y). The value of P-P is equal to *O*. Therefore, 6P=0. The order is 6.

Let us show the elliptic curve point representation in figure Figure 1.3 with the points used in above example.

Figure 1.3: Elliptic Curve Point Representation



Groups: The set of numbers on which we perform operations is called a group. There are two types of groups.

The first is F_p^*, which is of a prime order p and represents multiplicative group.
The second one F_p, which is also of the same order and represents additive group.

The group F_p: This group is formed based on prime number p and has numbers starting with 0 to $p - 1$. The addition modulo p operation is used to generate this group elements.
Assume a group F_{13} which contains numbers from 0 to 12:

$$(4 + 10)\ mod\ 13 = 1$$

$$(8 + 9)\ mod\ 13 = 4$$

We can notice that all the answers lie inside the group.

Similarly the additive inverse of -4 is 9 or the other way round.
Since, $(4 + 9)\ modulo\ 13 = 0$.
In this group each element contains exactly one additive inverse.
The subtraction operation can be carried out in terms of the additive inverse as given below.

$$7 - 8 = 7 + (-8) = 7 + 5 = 12$$

The group F_p^*: The Diffie Hellman Key Exchange protocol uses this group. The multiplicative group F_p^* contains the numbers starting from 0 to $p - 1$. Here p is the prime chosen to form the group.

Group Law: The two points on the elliptic curve are added using a chord and tangent composition method and the resulting third point also lies on the same curve.

Let us consider two points on the elliptic curve P , Q and a third point $-R$ can be obtained by adding these two points. The reflection of the point $-R$ over x-axis gives the actual point R which is the addition of the points P and Q. This is denoted as $P + Q = R$. In case, if Q = -P then the line which is passing through P and Q is a

vertical line which intersects the given elliptic curve at the point of infinity. This is represented as $P + (-P) = \infty$ and to add two same points we just have to draw a tangent at that point, which intersects at a point $-R$ and is reflected again to get R. i.e, $P + P = R$, or $P + P = \infty$ if the intersection point is infinity. The figure 1.4 shows the procedure to add two elliptic curve which are formed over set of all real numbers R. This process can also be represented mathematically. The inversion of point "p(x,y), $-P = (x, -y)$" can be obtained just changing sign of y coordinate. Two distinct elliptic curve points can be added as given below.

Let "$P = (x_1, y_1)$ and $Q = (x_2, y_2)$" be two distinct points s.t $P = -Q$. If a line is drawn through P and Q then its slope is given by $\lambda = \frac{y_2 - y_1}{x_2 - x_1}$. The coordinates of $R = P + Q$ which is represented as (x_3, y_3) can be calculated by "reflecting third point of intersection" as:

$$y_3 = -y_1 + \lambda(x_1 - x_3)$$

$$x_3 = \lambda^2 - x_1 - x_2$$

If P1=P2 , Then,

$$\lambda = \frac{3x_1^2 + A}{2y_1}$$

Here, A is coefficient of x.

Example: Let $E = y^2 = x^3 + 1$ with $P = (-1, 0)$ and $Q = (0, 1)$.

$$\lambda = \frac{1 - 0}{0 + 1} = 1$$

$$x_3 = 1^2 + 1 - 0 = 2$$

$$y_3 = 0 + 1(-1 - 2) = -3$$

Thus,

$$P + Q = (2, -3)$$

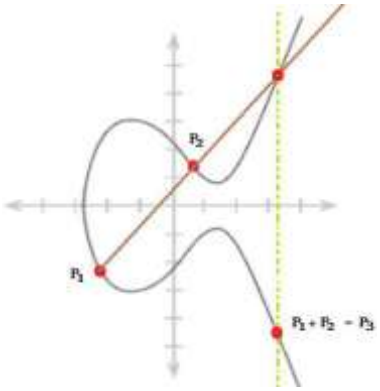
Properties of Addition: Let E is an elliptic curve and P,Q and R are elliptic curve points then the addition law properties of this group are:

$Q + (-Q) = O$ (Point at infinity) $\quad \forall Q \, s \, E$

" $Q + O = O + Q = Q \quad \forall Q \, s \, E$" "$R + (P + Q) = (R + P) + Q \quad \forall R, P \text{ and } Q \, s \, E$"

"$Q + P = P + Q \quad \forall Q, P \, s \, E$"

Figure 1.4: Elliptic Curve Points Addition



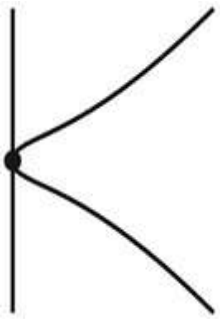
1.3 Singular Curve and Non-Singular Curve

The tangent of a given curve can be calculated using differential calculus. Let $f(x, y) = 0$ is a curve equation. Then the following relation is obtained on differentiating the function f:

$\frac{\partial f}{\partial x} + \frac{\partial f}{\partial y} * \frac{\partial y}{\partial x} = 0$. This is elementary calculus.

Let us consider a curve $f(x, y) = y^2 - x^3$ whose graph looks as follows: Now let us

Figure 1.5: Ellipitc Curve $f(x, y) = y^2 - x^3$



do the partial differentiation of the above curve with y and x. When partially differentiated with respect to y, we get $2y$ and with respect to x we get $-3x^2$. A tangent is drawn to the above curve at origin where y and x are equal to 0 and the equation would look like $0x + 0y = 0$ at the origin, but in true sense it doesn't represent the line equation. For several points that are on the given elliptic curve we can not define

tangent $f_x = f_y = 0$ at those points. Here, f_y and f_x are the f's partial derivatives to y and x respectively. Therefore, such points are called as singular points. The curve having singular points is called as singular curve. The curve which doesn't have singular points is called non-singular curve. It is quite often to have a non-singular curve. Consider the curve $f(x, y) = 0$ and a singular point P with co-ordinates (a, b) where $f(m, n) = 0$, $f_x(m, n) = 0$, $f_y(m, n) = 0$. A curve is non-singular if it's partial derivative doesn't vanish anywhere. The above are three equations with two unknowns and in order to have simultaneous solution something special has to happen. Elliptic Curves are non-singular cubic curves.

Example:

- Let us consider two elliptic curves C1: $y^2 = x$ and C2: $y^4 = x^3 + x^2 + x$. We have a problem with these curves as partial derivative at point $P(0, 0)$ is 0. At point P the first curve crosses over itself and it contains two tangent directions which are distinct. At P the second curve becomes a sharp point. Therefore, we say that these curves are singular as they are possessing singular points at P.

- Now if $P = (r, s)$ is a point defined over any elliptic curve f and the tangent to the elliptic curve at P is

"$\frac{\partial f}{\partial x}(r, s) * (x - r) + \frac{\partial f}{\partial y}(r, s) * (y - s) = 0$". It will be non-singular curve if the partial derivatives exists at all the points on the curve.

1.4 Field

The Set of elements which belong to both the additive abelian group and multiplicative abelian group is called as a field. In true sense we say that for all non zero elements there exists multiplicative inverse and additive inverse. A finite field is formed of a field which contains the finite number of elements.

Let $(F, *)$ is an abelian group which is non empty, where $*$ is a binary operation and F is a set such that $FXF \rightarrow F$ satisfing the associativity rule, closure property and other properties are also satisfied.

The rule is given by: "$a * (b * c) = (a * b) * c$ for all $a, b, c \in F$".
In additive group, the identity is denoted by 0 and the inverse of the element d is

denoted by $-d$. In multiplicative group the identity element is 1 and the inverse of an element d is denoted by d^{-1}. A finite group tt contains a finite number of elements. This finite number is called as order of the group.

1.4.1 Discriminant and i-invariant

Let us consider non homogeneous weierstrass equation denoted by E and is given as follows:

"$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$"

Now let us define the following equations to compute $i(E)$

$d_2 = a_1^2 + 4a_2$

$d_4 = 2a_4 + a_1a_3$

$d_6 = a_3^2 + 4a_6$

$d_8 = a_1^2 + a_6 + 4a_2a_6 - a_1a_3a_4 + a_2a_3^2 - a_4^2$

$c_4 = d_2^2 - 24d_4$

$\Delta = -d_2^2d_8 - 8d_4^3 - 27d_6^2 + 9d_2d_4d_6$

$i(E) = \frac{c_4^3}{\Delta}$

Here, Δ is the weierstrass equation's discriminant and the i-variant of E is $i(E)$, if the obtained Δ is not equal to 0.

1.4.2 Finite Fields

F_q or $ttF(q)$ is a field having finite elements. Here, ttF represents the Galois field and q represents the count of the elements. The order is the count of the number of elements in the given field F_q.

1.4.2.1 Prime Field GF(p)

In $ttF(p)$, p is prime number and F_p represents the elements of the field modulo prime number (p). The following are the mathematical operations that can be performed on a prime field.

- Addition: "$\forall\, a, b \in ttF(p), \exists\, r \in ttF(p)$ such that $r = (a + b) \bmod p$."

- Multiplication:"$\forall\, a, b \in ttF(p), \exists s \in ttF(p)$ such that $s = (a * b) \bmod p$."
- Inversion: "a^{-1} is the inverse of a non zero element a in $ttF(p)$". It is a unique integer which should belong to the $ttF(p)$, let us represent it as c and $a.c = 1$.

Example: Let us take a field with prime number 23. Therefore, it is represented as $ttF(23)$. The elements of the field are $0, 1, 2, 3 \ldots\ldots 21, 22$. Now, let us perform the above mathematical operations on the elements of this field.

- Addition: Let the elements be $15, 20 \in ttF(23)$. Now $15 + 20 \ mod\ 23 = 35 \ mod\ 23 = 12$
- Multiplication: Let the elements be $4,12 \in$ GF(23).Now $4 * 12 \ mod\ 23 = 48 \ mod\ 23 = 2$.
- Inversion: Let the element be 8. $8 \in ttF(23)$. The inverse of 8 is 3. Since, $8 * 3 \bmod 23 = 1$.

1.4.3 Binary Field GF(2^n) polynomial representation

"Let $f(x) = x^n + f_{n-1}x^{n-1} + \ldots\ldots + f_2x^2 + f_x + f_0$, where $f_i \in 0, 1$ and $0 \leq i < n$". Let $f(x)$ be a polynomial that is irreducible and it is having degree m over $ttF(2)$. Then $ttF(2^m)$ will have all the elements whose degree is less than m. The finite field representation is given as:

$(f_{n-1}, f_{n-2}\, f_2, f_1, f_0)$.

The $ttF(2^n)$'s elements will constitute a string of binary numbers having length n. Now let us perform the mathematical operations:

- Addition: " If $a = (a_{m-1}a_{m-2}\ldots\ldots a_2a_1a_0)$ and $b = (b_{m-1}b_{m-2}\ldots\ldots b_2b_1b_0)$ are the elements in $ttF(2^m)$ then $c = a + b = (c_{m-1}c_{m-2}\ c_2c_1c_0)$.
  Here, $c_i = (a_i + b_i) mod 2 = a_i \oplus b_i$".
- Multiplication: "If $a = (a_{m-1}a_{m-2}\ldots\ldots a_2a_1a_0)$ and $b = (b_{m-1}b_{m-2}\ldots\ldots b_2b_1b_0)$ are the elements in $ttF(2^m)$ then $c = a * b = (c_{m-1}c_{m-2}\ldots\ldots c_2c_1c_0)$ where the remainder will be $(c_{m-1}x^{m-1}c_{m-2}x^{m-2}\ c_2x^2c_1xc_0)$ by dividing $a * b$ by $f(x)$ over $ttF(2^m)$".

- Inversion: a^{-1} is the inverse of the element a in $ttF(2^m)$ where $a * a^{-1} = 1$.
  Let, $a = g^i$ for some i, where g is a generator of the multiplicative group. Then
  " $a^{-1} = g^{(-i) mod (2^m-1)}$ and $(-i) mod (2^m - 1) = ((2^m - 1) - i) mod (2^m - 1)$".

Example: Given $f(x_1) = x_1^4 + x_1 + 1$ defined over $ttF(2)$. Then the corresponding field consists of 16 elements because 2^4 is 16. Therefore, the elements are:

"0-0000 1-0001 x_1−0010 x_1+1-0011
x_1^2-0100 x_1^2 + 1-0101 x_1^2 + x_1-0110
x_1^2 + x_1 + 1-0111 x_1^3-1000 x_1^3 + 1-1001
x_1^3 + x_1-1010 x_1^3 + x_1 + 1-1011 x_1^3 + x_1^2-1100
x_1^3 + x_1^2 + 1-1101 x_1^3 + x_1^2+ x_1-1110 x_1^3 + x_1^2 + x_1 + 1-1111"

- Addition: Let us add 0101 and 1100 over GF(2^4).We get 0101 ⊕ 1100= 1001.

- Multiplication:Let us multiply 1010 and 0101 over GF(2^4).
  Here, 1010=$x_1^3 + x_1$ and 0101=$x_1^2 + 1$.
  Therefore,$x_1^3 + x_1 * x_1^2 + 1 \bmod f(x_1)$
  $= x_1^5 + 2x_1^3 + x_1 \bmod f(x_1) = x_1^5 + x_1 \bmod f(x_1) = x_1^2 = 0100$.

- Multiplicative Inversion:The element g=0010 be the field generator.The powers corresponding to g are given by:
  "g^0-0001 g^1-0010 g^2-0100 g^3-1000
  g^4-0011 g^5-0110 g^6-1100 g^7-1011
  g^8-0101 g^9-1010 g^{10}-0111 g^{11}-1110
  g^{12}-1111 g^{13}-1101 g^{14}-1001 g^{15}-0001"

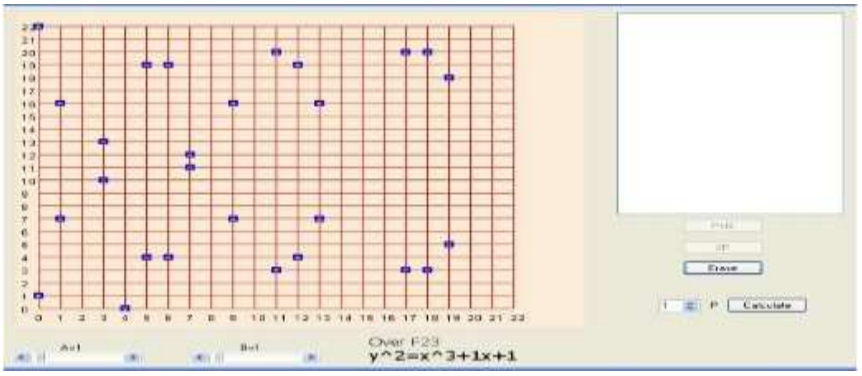
  The mathematical identity is g^0-0001. It's inverse a^{-1} is computed as:
  "$a^{-1} = g^{(-i) mod (2^n-1)}$"

  The inverse of $g^6 = (1100)_2$ is $g^{-6} \bmod 15 = g^9 \bmod 15 = (1010)_2$. Now let us prove that the multiplicative inverse of g^6 is g^9 that is $g^6 * g^9$ should be 1.
  (1100)*(1010)
  $(x_1^3 + x_1^2)*(x_1^3 + x_1) \bmod f(x_1)$
  $(x_1^6 + x_1^5 + x_1^4 + x_1^3) \bmod f(x_1) = 1$
  Therefore,the multiplicative inverse of g^6 is g^9.

1.5 Elliptic Curve over Finite Fields

In the above sections we have seen the prime field and galois field with their associated mathematical operations. The ECC operations are purely dependent on the finite fields. The basic ECC operations are represented in Figure 1.6. In this we have chosen the prime field 23 and the domain parameters are a=1,b=1.

Figure 1.6: Elliptic Curve Points in Finite Field



1.5.1 Elliptic Curve Parameters

There are three levels on which a ECC system is to be implemented. The three levels are:finite field,ECC level and protocol level.

- Field Level:The fundamental field is selected and their respective algorithms are used in this level.
- Elliptic Curve Level:The points are represented and the algorithms are selected for point doubling and addition.
- Protocol Level: Scalar multiplication Algorithms are chosen.

1.5.2 Elliptic Curve Over GF(p) operations in affine coordinates

Let us consider an elliptic curve E on a finite field F_p. Here, p is prime number and is greater than 3 and we chose a,b as arbitary constants belonging to the finite field F_p. The elliptic curve is chosen as:

"$y^2 = ^3 + ax + b$"

so that "$4a^3 + 27b^2 \neq 0$ (mod p)"
The group laws for Elliptic curves over affine coordinates are defined as:

- Identity:"O + Q = Q + O = Q for all Q $\in E(F_q)$. Here, Q is a point lying on the elliptic curve".
- Negation:Suppose Q(x,y) is a point on E(F_q) then we have a point P which is represented as (x,-y), this should belong to the E(F_q) and is denoted as -Q, where Q+P=O
- Point Doubling and Addition : Assume Q,P are two distinct points, where Q=(x_0,y_0), P=(x_1,y_1) $\in$ E(F_q) and P $\neq$ -Q. Q+P=(x_2,y_2) where $x_2=\lambda^2 - x_0 - x_1$ and $y_2=\lambda(x_1 - x_2) - y_1$.
  Here, $\lambda = \frac{y_0 - y_1}{x_0 - x_1} if P \quad Q$ OR
  $\lambda = \frac{3x_1^2 + a}{2y_1}$ if P = Q
  The inversion operation takes a lot of computational time.
  Let the value of p be 23 and the curve be "$y^2 = x^3 + x + 1$",the order of this curve is 28 including the infinity O.

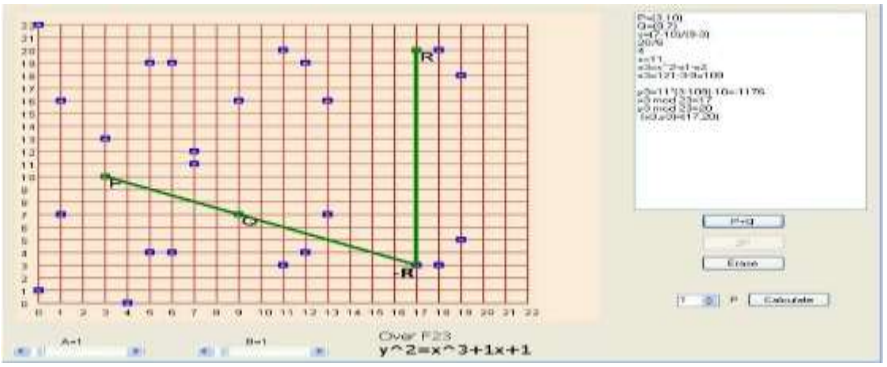
(0,1)	(1,16)	(4,0)	(6,4)	(7,12)	(11,3)
(12,9)	(17,3)	(18,20)	(0,22)	(3,10)	(5,4)
(6,19)	(9,7)	(11,20)	(13,7)	(17,20)	(19,5)
(1,7)	(3,13)	(5,19)	(7,11)	(9,16)	(12,4)
(13,16)	(18,3)	(19,18)	O		

- P+Q=(x_3, y_3)
  Let P=(6,19) and Q=(7,11).Then
  $\lambda = \frac{y_0 - y_1}{x_0 - x_1} = \frac{19-11}{6-7} = 8 \in F_{23}$
  $x_3 = \lambda^2 - x_0 - x_1$, $y_3 = \lambda(x_0 - x_3) - y_1$
  $x_3 = 8^2 - 6 - 7 = 64 - 13 = 51 mod 23 = 5$
  $y_3 = 8(6 - 5) - 11 = 3 mod 23 = 3$.
  As a result, P+Q=(5,3).

- 2P=P+P=(x_3, y_3). Let P=(0,1).
  $\lambda = \frac{3x^2+a}{\frac{1}{2y_i}} = \frac{3*0^2+1}{2*1} = \frac{1}{2} = 1 * 2^{-1}$
  Here 2^{-1}=12 mod 23 since $2 * 12 = 1 \bmod 23$
  As a result, $\lambda = 1 * 12 = 12 \bmod 23$
  $x_3 = \lambda^2 - 2x_0, y_3 = \lambda(x_0 - x_3) - y_1$
  $x_3 = 12^2 - 2 * 0 = 144 - 0 = 144 \bmod 23 = 6, y_3 = 12(0 - 6) - 1 = -73 \bmod 23 = 4$
  As a result,2P=(6,4)

The Figure 1.7 depicts the point addition

Figure 1.7: Elliptic Curve Points Addition in Finite Field



The Figure 1.8 illustrates the point doubling. Here the curve used is E: $y^2 = x^3 + ax + b$ and the point chosen for point doubling is P(3,10). A tangent is drawn over curve at P which cuts the curve exactly with one point (-R). The obtained point R(7,12) represents the reflection of -R.

1.6 Homogenous Polynomial

Homogenous polynomial is a polynomial in three variables with coefficients in F_p, where p is a prime.

"Let $f(X, Y)$ be an affine polynomial of degree n. Then $f(X, Y, Z) = Z^n f(X/Z, Y/Z)$ is a **homogeneous polynomial** in three variables".

Example of a homogenous polynomial:
$y^2 = x^3 + Axz^2 + Bz^3$

Polynomial and Rational Functions: A polynomial on a curve E is formed of the elements of finite field F_p. Sometimes the ring of polynomials are denoted by $F_p[E]$. Let an elliptic curve be $E(F_p)$ represented with $f(X, Y) = 0$. Take a polynomial $g(X, Y)$ and consider its behaviour on the points of $E(F_p)$ only. Then, for example, if g=f, from our point of view, g is the same as the zero function because g(P) for any point P on the curve is zero. This leads us to define the ring of regular functions of E to be

$$F_p[E] = F_p[X, Y]/f$$

Its field of fractions $F_p(E)$ is called rational functions field of E.

"For field F_p, the set $P^2(F_p)$, **the projective plane**, is the set of equivalence classes on $(F_p)^3 - (0, 0, 0)$ defined by the relation: $(x_1, y_1, z_1) = (x_2, y_2, z_2)$ if and only if $(x_1, y_1, z_1) = \lambda(x_2, y_2, z_2)$ for some λ 0. The representative classes are denoted $[x, y, z]$. Then $(F_p)^2$, **a standard plane**, is a subset of $F_p(P^2)$ (as $(x, y) \rightarrow [x, y, 1]$) and is referred to as the **affine plane**". [13]

Examples:

Polynomial Function: $X^3 - (\frac{25}{4})X^2 + 13X - 9 = 0$
Rational Functions: $\frac{x^4 - 2Ax^2 - 8Bx + A^2}{4(x^3 + Ax + B)}$

1.7 Elliptic Curves Domain Parameters:

There are several domains that define the parameters of an elliptic curve. An elliptic curve points also define its characteristics. Apart from a,b the communicating parties should also decide on other parameters. These are called the domain parameters. The fields are defined by p, p is the chosen prime. The domain parameters are shown below:
"

$$D = (p, a, b, g, n, h)$$

" where,

- p represents field size

- a and b are curve defining parameters
- g represents the base point, which is also a generator
- n is the smallest positive integer such that $ng = 0$ and is of order g.
- "Since n is the size of the subgroup of $E(F_p)$, it can be stated that $h = \frac{1}{n}|(E(F_p))|$ is an integer. In cryptographic applications h must be small ($h \leq 4$) and $h = 1$ since G is generator".

1.8 Elliptic Curve Diffie Hellman Problem

Before looking into "the elliptic curve Diffie Hellman problem". Let us know about Discrete Logarithm Problem and Diffie-Hellman key exchange.

1.8.1 Discrete Logarithm Problem

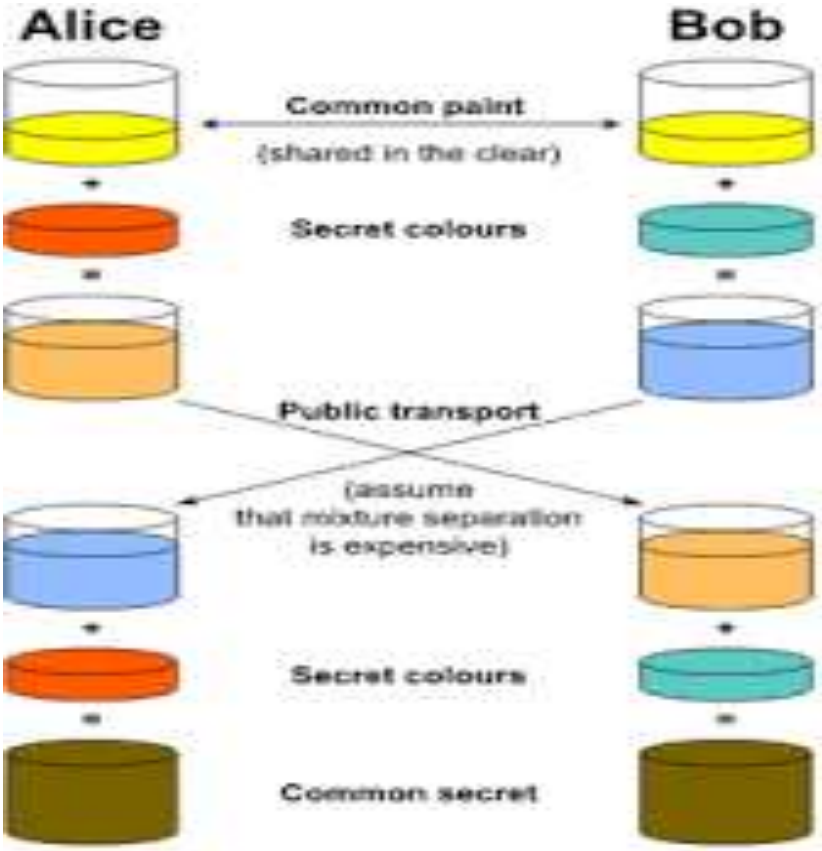
"The Security to the Diffie Hellman key exchange is provided by the Discrete Logarithm Problem". Let us consider a prime number p, two non-zero integers a and b ($mod\ p$), an unknown integer k. The discrete logarithm problem involves finding the value of k using the following formula: $a^k = b(mod\ p)$. Let $E(F_p)$ represents elliptic curve, which has a group law, let a and b be two points of $E(F_p)$, then we are trying to find k such that $ka = b$[2].

1.8.2 DIFFIE-HELLMAN Key Exchange

Diffie-Hellman problem enables secret and safe information exchange over an insecure network. This can be initially explained in an easier way through the usage of colors. The diagram below illustrates the process. Initially, Alice and Bob have a common color which are mixed with each of their private colors to get a distinguishing color each. They exchange the resultant colors and mix them again with each of their private colors to get the actual color they wanted to share. [2]

"The simplest and the original implementation of the protocol uses the multiplicative group of integers modulo p, where p is prime, and g is a primitive root modulo p."

Figure 1.8: Diffie-Hellman Key Exchange



1. Bob and Alice selects common parameters, prime number p and base g.
2. "Alice chooses a secret integer a, then sends to Bob $A = g^a \ mod \ p$"
3. "Bob chooses a secret integer b, then sends to Alice $B = g^b \ mod \ p$"
4. "Alice computes $K_1 = B^a \ mod \ p$"
5. "Bob computes $K_2 = A^b \ mod \ p$"
6. Now Alice and Bob have a common secret.

Example:

"Alice and Bob agree to use a modulus $p = 23$ and base $g = 5$ (which is a primitive root modulo 23).

Alice chooses a secret integer $a = 4$, then sends Bob $A = g^a mod p$.

$A = 5^4 \ mod\ 23 = 4$

Bob chooses a secret integer $b = 3$, then sends Alice $B = g^b \ mod\ p$.

$B = 5^3 \ mod\ 23 = 10$

Alice computes $s = B^a \ mod\ p$.

$s = 10^4 \ mod\ 23 = 18$

Bob computes $s = A^b \ mod\ p$.

$s = 4^3 \ mod\ 23 = 18$"

Alice and Bob now share the secret (the number 18).

1.8.3 Elliptic Curve Diffie Hellman Key Exchange

Let Alice and Bob be two parties communicating with each other. They will be agreeing upon a key which can be later used for encryption along with private keys. They first decide upon elliptic curve E that is defined over F_p and a generator $P \in E$ which must be of a high order. They choose a randomly selected integer which is less than n as private key d. The public key is , $Q = d * P$. Suppose, (d_a, Q_a) , (d_b, Q_b) are the private and public key pairs of Alice and Bob respectively. Alice computes,

$$K_a = (X_a\ ,\ Y_a) = d_a * Q_b$$

whereas Bob computes,

$$K_b = (X_b\ ,\ Y_b) = d_b * Q_a$$

Since, "

$$d_a * Q_b = d_a\, d_b\, g = d_b\, d_a\, g = d_b * Q_a$$

" Therefore,

$$K_a = K_b$$

and which is why

$$X_a = X_b$$

The secret which was shared by the two parties is K_a.

Example: Let the elliptic curve be, $E : y^2 = x^3 + 1$
Let the field be, $ttF(4019)$
and the base point be $P = E(1753, 3787)$.

The private key chosen by Alice is 34 and Bob is 16. The public key of alice is $34 * P$ and that of Bob is $16 * P$. The key pair of Alice is $A = E(2763, 3885)$ and Bob is $B = E(3221, 945)$. Alice computes $34 * B$ and Bob $16 * A$ which are the secrets they shared with each other. Finally, the secret they shared is $E(3461, 3296)$.

1.9 Pairings

"Let tt_1 be a cyclic additive group generated by P, whose prime order is q, and tt_2 be a cyclic multiplicative group with the same order q. In some instances the pairing is symmetric: $tt_1 = tt_2$. When ($tt_1 \neq tt_2$) the pairing is said to be asymmetric
G1 is called *source group* and G2 is called *target group.*"
" Let $e : tt1 \times tt1 \rightarrow tt2$ be a map with the following properties: [10]

- **Bilinearity**: e(aR, bS) = $e(R, S)^{ab}$ for all $R, S \in tt_1$ and $a, b \in Z_q$
- **Non-degeneracy**: There exists R, S $\in tt_1$ such that $e(R, S) \neq 1$. In other words, the map does not send all pairs in $tt_1 \times tt_1$ to the identity in tt_2;
- **Computability**: There is an efficient algorithm to compute $e(R, S)$ for all $R, S \in tt_1$

  In our setting of prime order groups, the Non-degeneracy is equivalent to $e(R, S) \neq 1$ for all $R, S \in tt_1$. So, when P is a generator of tt_1, e(P,P) is a generator of tt_2 .
  Such a bilinear map is called a bilinear pairing (more exactly, called an admissible bilinear pairing)".

Generator:
In the above q is order (q is prime number) there are certain base values(P) which generate distinct reminders for different exponents (x=1,2,..,q-1),here P is called generator.
Example: $P^x mod 7 = reminder$ where x=1,2,..,6
so if P=3 or 5 we get distinct reminder, P is called primitive root of 7.

problems in the additive group (G1;+).

- **Discrete Logarithm Problem (DLP)**: "Given two group elements P and Q, find an integer $n \in F_p^*$, such that Q = nP whenever such an integer exists."
  Example:
  Let $2^2(mod 7) = y$. Find the reminder? y=4. That is when the exponent is known we can compute y.
  However, if exponent is not known and reminder is given determining the exponent will be a computationally difficult problem.
  $2^x(mod 7) = 4$ (This is what Discrete Logarithm problem)

- **Decision Diffie-Hellman Problem (DDHP)**: The DDHP is based on DDH assumption, consider a (additive) group G of cyclic order q, and generator P. The DDH assumption states that, given aP and bP for uniformly and independently chosen $a, b \in F_p$, the value $(a + b)P$ "looks like" a random element in G.
  "This intuitive notion is formally stated by saying that the following two probability distributions are computationally indistinguishable (in the security parameter p):

  – $(aP, bP, (a + b)P)$, where a and b are randomly and independently chosen from F_p.
  – (aP, bP, cP), where a,b,c are randomly and independently chosen from F_p".

  The DDHP states that
  "For $a, b, c \in F_p^*$, given P, aP, bP, cP decide whether $c \equiv ab \bmod p$.
  If it were possible to efficiently compute a from aP in the cyclic group G, then the DDH assumption would not hold in G".
  Given (aP, bP, Q), it is easy to decide whether $Q = (a + b) * P$ by first computing the a and b from aP and bP and then computing (a+b)P and comparing Q.
  Note: "Importantly, the DDH assumption does not hold in the multiplicative group F_p^*, where p is prime. This is because given g^a and g^b, one can efficiently compute the Legendre symbol of g^{ab} , giving a successful method to distinguish

g^{ab} from a random group element".

- "**Computational Diffie-Hellman Problem (CDHP)**: For $a, b \in F_p^*$, given P, aP, bP, compute abP".

  ***Example*:**

  Let P is the generator of the elliptic curve $E(F_p)$ that is given by the elements of F_p.

  Let $a, b \in Z_q$ and $A = aP$ and $B = bP$. Given A and B, we can not compute a,b because of DLP on elliptic curves. without a,b we can not compute abP.

1.9.1 Group used in Pairing

"A group is a very general algebraic object and most cryptographic schemes use groups in some way. In particular Diffie Hellman key exchange uses finite cyclic groups".

In Pairings what group G to use?

- "Diffie-Hellman originally when they wrote paper, they instantiated their protocol using the group $tt = F_p^*$. This works fine in practice but the problem is, the discrete logarithm problem is not as hard as needed. There are sub-exponential algorithms that actually break discrete logarithms in F_p^*. Because of these sub exponential algorithms, we relatively has to use large primes to get required security. Today in practice, the primes over 2000 bits are used and the recommendations are even to use primes or order 3000 bits. so these relatively large primes cause the protocol to run slow".

- The alternative group that is actually better than F_p^* is $tt = E(F_p)$, group of points of elliptical curve over F_p with p as prime number. A much smaller prime can be used to achieve the complexity of F_p^*. This group also has efficient group operations. In practice, a 256-bit prime is used which is ten times faster than comparable to F_p^* security.

- The Elliptical curves also has an additional structure called PAIRINGS. The group of points $E(F_p)$ for a prime power $p > 3$ and $a, b \in F_p$ and a,b chosen such that " $(4a^3 + 27b^2 \neq 0)$"

  " $E(F_p) = (x, y) \in F_p \times F_p$, such that $y^2 = x^3 + ax + b$ "

1.9.2 Applications of Pairing

- Encryption:
  - IBE (Identity-based encryption) - first introduced by Boneh-Franklin and Sakai-Ohgishi-Kasahara
  - IBKE (Identity-based key exchange) - first introduced by Smart and Sakai-Ohgishi-Kasahara
- BLS signature proposed by Boneh-Lynn-Shacham
  "A contemporary example of using bilinear pairings is exemplified in the Boneh-Lynn-Shacham signature scheme".
- Ring signatures
- NIZKs, SNARGS, Accumulators.

1.9.3 Pairings in cryptography

- $e(g^x, h^y) = e(g^y, h^x)$.
  $e(g^x, h^y)$. is encryption i.e x,y comes out and goes in reverse order in decryption $e(g^y, h^x$.)
  where g and h are 2 different generators.

1.10 Mathematical Foundations Of Bilinear Pairings

1.10.1 Projective Space

"In mathematics, a projective space can be thought of as the set of lines through the origin of a vector space V [33]. When $V = R^2$ and $V = R^3$ the projective spaces can be lines and lines and planes respectively, where R denotes the field of real numbers, R^2 denotes ordered pairs of real numbers, and R^3 denotes ordered triplets of real numbers.

The idea of a projective space relates to perspective, more precisely to the way an eye or a camera projects a 3D scene to a 2D image".

1.10.2 Vector Space

"A vector space (also called a linear space) is a collection of objects called vectors, which may be added together and multiplied ("scaled") by numbers, called scalars. Scalars are often taken to be real numbers, but there are also vector spaces with scalar multiplication by complex numbers, rational numbers, or generally any field." Vector space satisfies Assosiative, Commutative, Identity, Inverse upon addition. Compatibility of scalar multiplication with field multipilcation.

Example

"Vector addition and scalar multiplication: a vector p is added to another vector n. n is stretched by a factor of 2, yielding the sum p+2n."

1.10.3 Divisors of Rational functions

"How a rational function on elliptic curve E relates to its zeros and poles. To begin, we look at the simpler example of a rational function of one variable [1]. A rational function is simply a ratio of two polynomial functions, thus we may state that a general rational function of one variable takes the form:"

$$F(X) = \frac{a_0 + a_1 x + \ldots\ldots\ldots + a_m x^m}{b_0 + b_1 x + \ldots\ldots\ldots + b_n x^n}$$

if we allow factorization over a complex number, we can find $a_1, a_2, \ldots\ldots, a_r$ .and $\beta_1, \beta_2, \ldots\ldots, \beta_s$ so that

"

$$F(X) = \frac{q(x - a_1)^{d1}(x - a_2)^{d2\ldots\ldots\ldots\ldots}(X - a_r)^{dr}}{b(x - \beta_1)^{e_1}(X - \beta_2)^{e_2}\ldots(X - \beta_s)^{e_s}}$$

"

a_i, i=1,2....r are known as zeros, whereas, β_j, j=1,2,...s are known as poles.

Then F has zeros at $a_1, a_2, \ldots, a_r$. and poles at $\beta_1, \beta_2, \ldots, \beta_s$. , "where each zero a_i has multiplicity d_i , and each pole β_j has multiplicity e_j, we define the divisor of F, div(F), as the formal sum

div(F) = $d_1(a_1) + d_2(a_2)+.....d_r(a_r) - e_1(\beta_1) - e_2(\beta_2)\ e_s(a_s)$".

Note "The zeros have positive coefficients, the poles have negative cofficients. We can consider rational functions of two variables, $f(X, Y) : E \rightarrow C$ and look at their divisors. (Here, E is an elliptic curve)"

Example Consider the elliptic curve " $E: Y^2 = X^3 + aX + b$ "

Let a_1, a_2, and a_3 be such that " $Y^2 = (X - a_1)(X - a_2)(X - a_3)$", and distinct.

"If we consider the function f(X,Y)=Y, as a rational function, we have that f vanishes at three points, P1 = (a_1,0), P2 =(a_2,0), and P3=(a_3,0),giving us three zeros, each with mutiplicity 1.

To find the poles, we must remember that as a rational function, $Y = Y/1$ . . Homogenizing, we have $f(X, Y, Z) = Y/Z$. Then to find the poles of f , we must analyze Z as a polynomial. For all affine points on elliptic curve E, Z=1 and Z=0 is only at ∞. As a result of our projective transformation, it is easy to show that Z = 0 is the tangent line at that point. This gives us that the pole O has multiplicity 3. Thus, div(Y)=(P1)+(P2)+(P3)-3(O). div(Y)=1+1+1-3(i.e The zeros have positive coefficients, the poles have negative cofficients).

div(Y)=0".

1.10.4 Embedding Degree

"Embedding degree is the smallest integer k that transforms an instance of the elliptic curve discrete logarithm problem (ECDLP) over an elliptic curve $E(F_p)$ into an instance of the discrete logarithm problem(DLP) over the field F_{p^k}.

Let $E(F_p)$ be the elliptic curve defined over F_p and one point in $E(F_p)$ has order q where $q \neq p$ is a prime, then the embedding degree with respect to q is the smallest value k such that $q | p^k - 1$, or $p^k \equiv 1 \bmod q$, assuming $p \neq 1 \bmod q$.

Proposition

Let E be an Elliptical Curve over F_p and let $l \neq p$ be a prime.

Assume that $E(F_p)$ has a point of order l. Then one of the following is true:

- The embedding degree of E with respect to l is 1 (this cannot happen if $l > \sqrt{p} + 1$)

.

- if $p \equiv 1 \bmod l$ then the embedding degree is l.

- if $p \not\equiv 1 \bmod l$ then the embedding degree is the smallest $k \leq 2$. such that $p^K \equiv 1 \bmod l$, i.e. the order of p modulo l."
  However, random curves have embedding degree much larger than $(\log p)^2$.

1.11 How Pairings are formed

"A pairing abstractly is something that operates on two groups. There is a source group, which we call tt_1, and there's a target group called tt_2. Now, normally the source group will be the points of an elliptic curve and the target group tt_2 would be just a finite field F_p. The pairing takes two points in the source group and maps them to the target group in such a way that the exponents multiply. The pairing is bilinear, and bilinear here means multiplication of exponents.

Formally, a pairing is a map with bilinearity property because it allows us to multiply exponents in the exponent, but it moves us to a different group, so we can only do the pairing once. And the requirement is that it is a polynomial-time (poly-time) computable and it better not be degenerate. And it's not degenerate, in the sense that it maps generators to generators (P generates $tt_1 \rightarrow e(P, P)$ generates tt_2)".

1.11.1 Symmetric Pairings

"Symmetric, pairings can be used to reduce a hard problem in one group to a different, usually easier problem in another group. For example, in groups equipped with a bilinear mapping such as the Weil pairing or Tate pairing, generalizations of the computational DiffieHellman problem are believed to be infeasible while the simpler decisional DiffieHellman problem can be easily solved using the pairing function. The first group is sometimes referred to as a Gap Group because of the assumed difference in difficulty between these two problems in the group [10].

Supersingular Curves Let F_q be a finite field of characteristic p [41]. E[p] is trivial".

1.12 prerequisite of pairing

To compute a pairing,we need the following elements:

- E, an elliptic curve that is defined on F_q and represented with the equating "$E : y^2 = x^3 + ax + b$,where $a, b \in F_q$"

- The embedding degree k minimal integer such that $r \| (q - 1)$.
  $if gcd(r, q) = 1$,then $E[r] = F_q \times F_q$. . if $K \geq 2$ then $E[r] = E(F^k)[r]$.

1.13 WEIL PAIRING

"For $P, Q \in E[m]$, let f_P , f_Q be rational functions on E such that

$$div(f_P) = m(P) - m(O)$$

and

$$div(f_Q) = m(Q) - m(O)$$

.

The Weil pairing of P and Q is defined as

$$e(P, Q) = \frac{\frac{f_P(Q+S)}{f_P(S)}}{\frac{f_Q(P-S)}{f_Q(S)}}$$

.

where S $\notin$.{ O, P, -Q, P-Q } to ensure the pairing is defined and nonzero.

Theorem The Weil pairing has the following qualities: [1]

- $e_m(P, Q)$ is independent of choice of the functions and the point S.
- The value of the Weil pairing is an m-th root of unity, that is: $e_m(P, Q)^m = 1$.

- The Weil pairing is bilinear in a multiplicative manner:for all P1, P2, $Q \in E[m]$

$$e_m(P1 + P2,\ Q) = e_m(P1,\ Q)e_m(P2,\ Q)$$

and

$$e_m(Q,\ P1 + P2) = e_m(Q,\ P1)e_m(Q,\ P2).$$

- The Weil pairing is alternating, that is, $e_m(P,\ P) = 1$ for all $P \in E[m]$.
This implies

$$e_m(P,\ Q) = e_m(Q,\ P)^{-1} for all P,\ Q \in E[m]$$

.

- The Weil pairing is non-degenerate:
if

$$e_m(P,\ Q) = 1 for all Q \in E[m],\ then P = O$$

Example: Looking at $Y^2 = X^3 + 2X^2 - 3X$, four easy-to-work-with points of order 4 are P1=(1,2),P2=(1,2),P3=(3,6),and P4=(3,6).We will also use Q=(1,0), $S = (2, \sqrt{6})$. , and of course O. To calculate Weil pairings among these four points,we will need the correct functions.

Taking P1 as an example, we need a function f_{P1} . with $div(f_{P1}) . = 4(P1) - 4(O)$. Without any better algorithm, we would be forced to look at lines drawn through our points of interest and use the properties of divisors to create a function. Listed below are the functions used to find f_{Pi} . for i = 1,2,3,4.

Now, we use the divisors above to create a set of coefficients c_k such that

$$4(P_1) - 4(O) = \sum_{k=1}^{6} c_k\ div(g_k)$$

.

These coefficients and functions will give

$$f_{P1} = \prod_{k=1}^{6} c_k\ g^k$$

Solving for the coefficients, $c_1 = 2,\ c_2 = -1,\ c_3 = 0,\ c_4 = 2,\ c_5 = 0,\ c_6 = -2$ and thus, in homogenized form, we end up with:

$$f_{P\,1}(X,\ Y,\ Z) = \frac{(X\ \_\ 3Z)^2\ (Y + X\ \_\ Z)^2}{(Y + 3X - 3Z)\ (Y - 3X + 3Z)\ Z^2}$$

$$f_{P\,2}(X,\ Y,\ Z) = \frac{(X\ \_\ 3Z)^2\ (Y\ \_\ X + Z)^2}{(Y + 3X - 3Z)\ (Y - 3X + 3Z)\ Z^2}$$

$$f_{P3}(X,\ Y,\ Z) = \frac{(X - 3Z)^2\ (Y - 3X + 3Z)}{(Y + 3X - 3Z)\ Z^2}$$

$$f_{P4}(X,\ Y,\ Z) = \frac{(X - 3Z)^2\ (Y + 3X - 3Z)}{(Y - 3X + 3Z)\ Z^2}$$

Then to calculate the pairing of P_1 and P_3 ,

$$e_4(P\,1,\ P\,3) = \frac{f_{P\,1}(P_3 + S)}{f_{P\,1}(S)} \Big/ \frac{f_{P\,3}(P\,3 - S)}{f_{P3}(-S)}$$

after using the addition algorithm to find $P_3 + S = (-2.496, -2.047), P_1 - S = (20.798, -98.990)$ and using $f_{P\,1},\ f_{P3}$ as above.

Weil pairing values for four nice points in E[4] on $Y^2 = X^3 + 2X^2 - 3X$
Note that

$$e_4(P_3,\ P_1) = e_4(P_1,\ P_3)^{-1} = -1$$

. To avoid computing all of these, we use the bilinearity and alternating qualities of e_4

: Note that $[4]P_1 = O$ and $-P_1 = P_2$ implies $P_2 = [3]P_1$.
Thus, $e_4(P_1,\ P_2) = e_4(P_1,\ [3]P_1) = e_4(P_1,\ P_1)^3 = 1^3 = 1$ and by the alternating quality, $e_4(P_2,\ P_1) = e_4(P_1,\ P_2)^{-1} = 1$"

Chapter 2

User Authentication Using BLS Signature In Distributed PKI Based Networks

Authenticating a user in the decentralized network of resource constrained devices is a challenging task due to their dynamic and resource constraint infrastructure. These decentralized networks adapt two different types of approaches: one is identity based cryptography(IBE) and the other is public key cryptography(PKC) in which centralized authority named Certificate Authority (CA) takes responsible for key management. In order to adopt it to a decentralized network, the job of the CA must be distributed. The master secret key is shared among the users of the network, to self-organize the network without a central authority. The key is shared based on a bi-variate implementation of Shamir secret sharing scheme to make the network fully self-managed by its users. In this chapter, we considered PKI based scenario and proposed a new scheme to authenticate a user using BLS signature scheme, that is light weight and easy to implement compared with the existing schemes thus making it suitable for our network.

2.1 Introduction

PKI - Public Key Infrastructure [24] establishes secure communication by providing authentication using digital certificates and encryption using public key cryptography.

2. USER AUTHENTICATION USING BLS SIGNATURE IN DISTRIBUTED PKI BASED NETWORKS

In our protocol, we adopted the distributed PKI procedure to form completely decentralized network. In PKI scenario, the user public keys are issued and managed by the centralized authority(CA). The CA uses the master Secret key(S_k) to sign while issuing the certificage. Since the convenstional PKI system is based on the centralized authority, we can not adopt it in our decentralized network. In decentralized network, the network topology changes frequently and if we select any user as CA and if that user moves out of the network then the total functionality of CA may break-down causing the total network failure. In order to overcome this disadvantage we adopt shamir secret sharing scheme with a (t,n) threshold[47][8][38], which helps in distributing CA power to all the users of the network. This canbe achieved by dividing the master secret key S_k into shares and distributing these shares to all the users of the decentralized network[44]. In this protocol, we used distributed the shares among the users using shamir secret sharing scheme and we used BLS signature scheme[11] to sign the certificates of the users.

2.1.1 Attacks on the decentralized networks

There are two types of attacks possible on the decentralized networks - Passive and Active. A valuable data is captured in transit using passive attacks and the total network can be damaged or disrupted from executing its regular operations using active attacks. An unauthorized malicious user is the one, who does not authenticate itself to other honest users and misbehaves in the network.

The honest and authenticated user may also misbehave if the malicious used takes control of his credentials. Any computer network is formed of different layers and the attacks are carried out specific to each layer. So each layer should have its own security protocols that are specific to the attacks possible in that layer. In most of the decentralized networks, a wireless medium is shared by the user and the attacker can eavesdrop on the transmitted messages or he can send fake messages by compromising the physical layer. Since in most of the wireless networks, a one hop connectivity is used among its user for the communication, the malicious user can easily do traffic monitoring and traffic analysis attacks. In network layer, the attacker exploits the routing algorithms to create routing hops and cause network congestion[20]. The attacker uses a compromised user to perform SYN flooding and denial of service attacks at transport layer. Most of the attacks that take place in application layer are repudiation attacks,

worm attacks and mobile viruses. Some attacks like man-in-the-middle and denial of service can be launched from several layers.

This chapter proposes user authentication using BLS signature, so that many of the attacks can be avoided.

2.1.2 Distributed PKI

Many security services that are required to make the network secure like message authentication, intergrity, confidentiality, non-repudiation, digital signatures and encryption are provided by Public key cryptography(PKC)[43]. Any public key cryptography can be effectively deployed by managing digital certificates using Public key infrastructure(PKI)[24]. In PKI environment, the certificates of the network users are issued and managed by Certificate authority(CA). The user identity and his public key is placed on the certificate and is signed by the CA using master secret key s_k and the verification of this certificate is done using master public key PK. In our case, the network is infrastructure-less and decentralized. so, the same PKI cannot be adopted in our proposed protocol. We distribute the role of certificate authority among the users by sharing the secret key S_k to the network users. The secret key S_k can only be reconstructed when the given threshold number of users cooperate with their shares.

2.1.3 Threshold Cryptography

Since we cannot directly use PKI in the decentralized network, we use threshold cryptography concept to share the secret key S_k among the network users. One of the most widely used and popular secret sharing technique is the one first proposed by Shamir [38]. Shamir secret sharing contains a dealer and a set of users. The job of the dealer is to distribute the network secret S_k to all the users of the network. The dealer sends a share of the secret privately to each user. The scheme is generalized as a threshold (t, n)access structure, where n denotes the total network users and t represents trust level of the network. To generate the secret key out of n users, t users have to participate. We adopt this shamir's(t,n) secret sharing technique in our protocol to distribute the secret among the users. The dealer's role is also distributed by using a bi-variate polynomial, thus making this scheme fully distributed and suitable for decentralized networks. The bi-variate polynomial retains the same security as tri-variate polynomial because the third variable in the tri-variate polynomial is only used for subgroup operations. The

computation of user share at the dealer in the centralized network is distributed among the users of the decentralized network.

2.1.4 Related work

While adopting the PKI to decentralized networks, the major challenge is to distribute the Certificate Authority(CA) role. Many techniques are proposed based on Shamir's secret sharing to distribute CA's secret key s_k to make the decentralized network secure. The first distributed CA is proposed by Zhou and Haas[47]. They distributed the role of CA using threshold cryptography to maintain a selected set of servers connected in PKI scenario. This proposal does not support if the selected users are not always available in pure ad-hoc networks. Later a similar idea is adapted by Kong et al.[28] to distribute the master secret key to all the users. However, their threshold shceme proposal is very specific to RSA and it proved to be insecure[32][22].

In our proposal we used Shamir secret sharing [38], which is implemented by using bi-variate polynomials to distribute the Certificate Authority's master secret key to the decentralized network users. Bivariate polynomials are widely used in protocols that allow the new users to be part of the network without the presence of any centralized authority. Some of the works based on bi-variate polynomial are[9] Anzai et al.[5] and Herranz et al. Their original work inspired us to adapt this technique to our proposed protocol. Daza et al. [15] constructed flexible, decentralized, and distributed gropu key dynamically using bivariate polynomials. The main focus of their work is to generate secret keys than can be used in common group. Later Saxena et al.[37] established pair-wise keys for mobile ad-hoc networks in non-interacitve way using similar techniques. Recently Daxing et al. [42] proposed aggregate signature algorithm for the network using bilinear pairing and Hanaoka et al. [19] constructed multi user setting signature with tight security based on BLS signature.

Our work is more related to the decentralized network cryptographic techniques proposed by Herranz et al. [15]. They proposed a fully self managed network and the ways to authenticate communication among the users. Our protocol proposes the user authentication in their set up using BLS signature proposed by Boneh et al.[11]. This reduces the size of keys used as it uses the bilinear pairing. This scenario is much suitable for our network because its users are mostly resource constraint devices and they can not afford the heavy computational overhead required by larger keys. This

trade off is necessary because the protocol has to provide adequate security and at the same time considering the limitations of the resource constraint devices.

2.2 preliminaries

2.2.1 Self-Organized PKI and Secret Sharing Technique

Self-organized PKI Networks use secret sharing scheme[38] to distribute the PKI role among the network users. The secret sharing concept is first introduced by Blakley [8] and Shamir [38]. The secret sharing scheme consists of a set $V = \{v_1, v_2, v_3, , v_m\}$ of m users and a dealer d . The secret s is kept with the dealer and he sends privately s_i which is the share of the secret. All the users of the network recieve their corresponding shares from the dealer. The secret can be reconstructed when atleast t number of valid users contribute their shares. In (t, m) access structure t is the trust level and m donotes all network users. [38].

In our proposed protocol, "Shamir's secret sharing with (t, n) threshold access structure" is used[38]. Shamir secret sharing uses polynomial interpolation to implement (t, m) access structure. Let us consider the finite field F_p with $p > m$ and let the secret S_k of the network in F_p. To implement (t,m) access structure "a polynomial $P(x)$ with at most $t - 1$ degree" is chosen by the dealer. In the selected polynomial the constant term represents the secret S_k and all other coefficients are independently and randomly chosen from the field F_p. From this $P(x)$ is given by $\sum_{i=1}^{t-1} a_i * x^i + S_k$. A distinct field element a_i is publicly associated with each user v_i. By substituting the a_i in the polynomial, the dealer generates $[s]_i = P(a_i), \forall i = 1, \ldots, m$. If the set of t users u_1, u_2 , u_t are willing to generate the secret, they it can be obtained by computing $\sum_{i=1}^{t} l_i * [s]_i$, here the Lagrange coefficients are given by $l_i = \Pi_{j \neq i} \frac{a_j}{a_j - a_i}$. It is not possible to generate the secrete with less than t number of users.

2.2.2 Bilinear Pairing and Related Assumptions

"Let P be a point on elliptic curve and generates cyclic additive group tt of the prime order p. With the same prime order let tt_1 be another cyclic multiplicative group. Let $a, b \in F_p^*$. In both the groups Discrete Logarithm Problem (DLP) is assumed to be hard. Using the groups tt and tt_1, a bilienear pairing $e : tt \times tt \rightarrow tt_1$ is formed with the following properties:"

- " Bilinear: For all $R, S \in tt_1$, $e(aR, bS) = e(R, S)^{ab}$; "
- " Non-degenerate: There exists R and $S \in tt_1$ such that $e(R, S) \neq 1$; "
- " Computable: There is an efficient algorithm to compute $e(R, S)$ for all $R, S \in tt_1$ ".

The following assumptions assumptions are taken into the consideration while working with the bilinear pairings:

- "In both the groups tt_1 and tt_2, the Discrete logarithm problem(DLP) must be hard. In tt_1 DDHP which stands for The Decisional Diffie-Hellman problem should be easy. In tt_2, the DDHP and also CDHP which statnds for computational Diffie-Hellman problem should be hard".
- The bilinear pairing inversion computation should be hard, that is the BPIP (the bilinear pairing inversion problem) states that:
  - " BPIP : Given $S \in tt_1$ and $e(S, T) \in tt_2$, find $T \in tt_1$."

2.2.3 BLS Signature

The BLS signature technique was first proposed by B. Lynn, H. Schacham, D. Boneh. This scheme is based on CDHP problem defined over certain type of elliptic curves. This scheme is suitable for Gap Diffie-Hellman Group which is discussed below.

2.2.3.1 GDH Group(" Gap Diffie-Hellman Groups ")

"Let G be cyclic multiplicative group generated by some element g whose order is a prime p". There exists three problems in the group G:

- " Group Action: Given u, v ∈ G, find uv ".
- " Decision Diffie-Hellman: For a, b, c ∈ F_p^*, given (g, g^a, g^b, g^c) decide whether c = ab ".
- " Computational Diffie-Hellman: For a, b ∈ F_p^*, given (g, g^a, g^b), compute g^{ab} ".

The definition of the GDH group is given below:

- " G is a τ -decision group for Diffie-Hellman if the group action can be computed in one time unit, and Decision Diffie-Hellman can be computed on G in time at most τ ".
- " The advantage of an algorithm A in solving the Computational Diffie-Hellman problem in a group G is

  $AdvCDH_A = Pr\{[A(g, g^a, g^b)] = g^{ab} : a, b \xleftarrow{R} F_p^*\}$, where the probability is over the choice of a and b. We say that an algorithm A (t, s)-breaks Computational Diffie-Hellman in G if A runs in time at most t, and $AdvCDH_A \geq s$ ".
- " A prime order group G is a (τ, t, s)-GDH group if it is a τ-decision group for Diffie-Hellman and no algorithm (t, s)-breaks Computational Diffie-Hellman on it ".

2.2.3.2 BLS Signature Protocol

- The BLS signature protocol setup is given as follows:
  Consider two cyclic groups tt and tt_1 as defined in Section 2.2.2

  Public information: The public parameters that are available to all the users of the network are:

  - - a hash function which is cryptographically secure $H_1 : \{0,1\}^* \rightarrow tt$,

  a secure bilinear pairing $e : tt \times tt \rightarrow tt_1$

  Public key of Signer: If a Point P is the generator tt, then the public key is $P_{pub} = S_k P$, where S_k denotes signer's secret key.
- Sign:
  The signature is computed for the given message $M \in \{0,1\}^*$ as

  $$Sg = S_k H(M) - wheresgisasignature$$
- Verify:
  The following equation verifies the Signature for its validity.

  "$e(P, sg) = e(P_{pub}, H(M))$"
- Proof:

  " $e(P, sg) = e(P, S_k H(M)) = e(S_k P, H(m)) = e(P_{pub}, H(m))$ "

2.3 Proposed BLS signature in User Authentication

Most of the distributed PKI based networks use the traditional signature algorithms like RSA, which requires huge computational power. since most of the distributed networks are light weighted, they use elliptic curve based signature schemes. One such signature scheme is ECDSA (“ Elliptic Curve Digital Signature Algorithm ”), which is used in Bitcoin. This algorithm requires the inversion and multiplication operations which are computationally heavy. BLS signature is simple compare to other light weight algorithms and it does not require random number at all.

The Table 2.1 shows that BLS signature is light weight compared to other signature schemes[4].

Our protocol is divided into four sections namely Setup - where the users have the

Table 2.1: Comparison of Light Weight Signature Schemes

Signature Scheme	**Signature Size**	**Veriftcation Time**
BLS	160 bits	47.6 ms
CHP	160 bits	73.6 ms
ChCh	320 bits	49.1 ms
Waters	480 bits	91.2 ms
Hess	1120 bits	49.1 ms

common parameters, Key Generation- each used generates their keys, Signature Generation Protocol - the users create their certificates and finally their certificates can be validated using Signature Verification Protocol. The BLS scheme requires, a group G, which is additive cyclic group of order q and its elements are generated by the point on elliptic curve P. Another group tt_1 which is cyclic multiplicative of same order is also required. In addition to the above two groups it requires a bilinear pairing e, and two cryptographically secure hash functions “$H_1 : \{0,1^*\} \rightarrow Z_q$”, another hash function “ $H_2 : \{0,1^*\} \rightarrow tt_1$”, and a public key PK $= P_{pub}$ as discussed in section 2.2.3.2. The hash function H_1(Str,n,hashfcn) takes the input parameters - string Str, integer n, secure hash function hashfcn and gives a value with in the range of “ 0 to n-1 ”. $n \leq 2^{hashlen}$ where hashlen denotes “ the number of octets comprising the output of the hash function hashfcn ”. The Merkle’s method [30] is used to create the H_1 hash

function and it is also provably secure hash function because of underlying hashfcn hash function.

The hash function H_2(E,p,q,id,hashfcn). The input parameters to the function H_2 are : E - elliptic curve ,two primes - p and q, id - a string and hashfcn - hash function. The return value of hash function is $Q_{id} = (x, y)$ which is a elliptic curve point and has prime order q. The two hash functions are given below as defined in RFC 5091[12].

Algorithm 2.3.1 H_1: Hash function to convert the given string into integer

Input of the function H_1:

o A $|s|$ octets length string s, the string s is divided into $|s|$ octets

o n - positive integer

o hashfcn - A secure hash function

Output of the function H_1:

o return positive integer v, $0 \leq v \leq n - 1$

Method:

1. Let the output hashfcn consists of the hashlen number of octets
2. The initial value of a variable $v_0 = 0$
3. The initial value of $var_0 = 0x00\ldots00$
4. " For i = 1 to 2, do:

   (a) Let $t_i = var_{i-1}||s$, which is the ($|s| + hashlen$) - octet string concatenation of the strings var_{i-1} and s

   (b) Let var_i = hashfcn(t_i), which is a hashlen-octet string resulting from the hash algorithm hashfcn on the input t_i

   (c) Let a_i = Value(var_i) be the integer in the range 0 to $256^{hashlen}$ - 1 denoted by the raw octet string var_i interpreted in the unsigned big-endian convention

   (d) Let $v_i = 256^{hashlen} * v_{i-1} + a_i$ "
5. Let $v = v_i(mod n)$

Algorithm 2.3.2 H_2: Translates the given string into elliptic curve point

Input of the hash function H_2:

o p - A choosen prime number for group tt

o q - A choosen prime number for group tt_1

o id - given string

o hashfcn - secure hash function

Output of the hash fucntion H_2:

o An elliptic curve point Q_{id}

Method:

1. Take y = H_1(id, p, hashfcn), H_1 is the function discussed in Algorithm 2.3.1
2. Compute x = $(y^2 - 1)^{\frac{(2*p-1)}{3}}$ mod p, $p \in F_p$
3. a non- zero point $Q' = (x, y) \in (F_p)$
4. A q order point $Q_{id} = [\frac{(p+1)}{q}]Q' \in E(F_p)$

2.3.1 Setup Phase

In this phase all network users receive their share s_i in the master secret key S_k. The following protocol is used to achieve this.

- Let the threshold value is t, k is the number of founding users and n is the total number of users supported by the decentralized network.
- The founding users are the users who will be present during the initial setup of the network. The number of founding users are k, the k value must be greater than t and less than n.
- Bivariate polynomial $f_i(x_1, x_2)$ of maximum degree k-1 and has symmetric in x_1, x_2 is chosen by all the users.
- Every user v_i computes $f_{ij}(H_1(v_j), x_2)$ for itself and other founding users, $1 \leq j \leq k$.
- Every user computes $f_{v_{ij}}(H_1(v_j), x_2)$ and secretly sends to other users corresponding v_j. In addition to that each user v_i also includes the chosen elliptic curve point $w_i = f_i(0, 0) * P$ in their messages.

- Once all the messages are exchanged then each user has his own value $f_{ii}(H_1(\upsilon_i), x_2)$ with it and also the messages received in the previous step from other users. Finally each user υ_i calculates $f_i(x_2) = f(H_1(\upsilon_i), x_2) = \Sigma_{j \in k} f_{ji}(H_1(\upsilon_i), x_2)$
- Once the above steps are completed then each user υ_i has their share $s_i = f_i(0)$ of the network secret and a uni-variate secret polynomial $f(H_1(\upsilon_i), x_2)$.

The network polynomial $f(x_1, x_2) = \Sigma_{i \in \upsilon} f_i(x_1, x_2)$ and $s_k = f(0, 0)$, the network secret key are hidden and unknown to the users. To reconstruct the network secret key S_k, at least t users has to contribute their shares.

If a new user υ_w wants to join, then he has to identify himself to least t users of the network and request for the values $f_{iw}(H_1(\upsilon_w), H_1(\upsilon_i))$. Once the user υ_w has t such values then he uses Lagrange's interpolation to generate his uni-variate secret polynomial. $f_w(x_2)$ can be calculated using Lagrange's interpolation as given below:

- $f_w(x_2) = f(H_1(\upsilon_w), x_2) = \Sigma_{\upsilon_j \in \upsilon} \Pi_{\upsilon_i \in \upsilon, \upsilon_i \neq \upsilon_j} \frac{(x_2 - H_1(\upsilon_i))}{(H_1(\upsilon_j) - H_1(\upsilon_i))} * f(H_1(\upsilon_w), H_1(\upsilon_j))$
- After applying the Lagranges's interpolaion, the user u_w has his share of the secret as $f_w(0)$ and the corresponding uni-variate polynomial is υ_w is $f_w(x_2)$ *i.e.*, $f(H_1(\upsilon_w), x_2)$

2.3.2 Key Generation

Once each user υ_i has their shares of the network secret, they have to get their public and private keys to communicate among them secretly. Any public key cryptographic algorithm can be used for the secure communication among the users. Here we took the RSA to generate user keys. The key generation algorithm generates SK_i and PK_i as private key used to decrypt and public key to encrypt for each user υ_i. The public key PK_i is distributed to all the users and the private key SK_i which is used for signing the certificates and decrypting the messages is kept confidential.

2.3.3 Signature Generation Protocol

Each user υ_i has a share of the network secret key s_i, another secret key SK_i to decrypt the messages and a private key PK_i. To authenticate himself to other network users, each user has to get a certificate linking his identity with his public key. To get this certificate, each user υ_i contacts other t users to sign his certificate which as the message

$v_i||PK_i$. The other users of the network use their partial shares to sign the certificate thus generation partial signature. After obtaining t of such partial signatures, any user can compute his fully valid certificate by using Lagrange's interpolation.

- $w_i = H_2(m) * s_i$, where $H_2(m)$ gives the hash values of message and s_i is the users share in network secret.
- The certificate(chm) is generated as
  $chm = \Sigma_{i \in t} w_i * L_i$, L_i represents Lagranges Coefficient.
  $L_i = \Pi_{w_j \in t, j \neq i} \frac{(0 - H_1(v_j))}{(H_1(v_i) - H_1(v_j))}$

All the users of the network computes their fully valid certificates by using the above protocol and use them to authenticate themselves to the network user for secure communication.

2.3.4 Signature Verification Protocol

If a user v_j wants to validate the user u_i certificate, then he runs the signature verification protocol to check its validity. The user v_j has access to the following information about user v_i:

- chm - which is the certificate of user v_i.
- the generator P and PK, which is the network's public key.
- The identify of user v_i and his public key PK_i

The user v_j runs the BLS signature protocol to validate the v_i's certificate:

- check whether $e(chm, P) = e(H_2(m), PK)$
  The certificate is valid it the above condition is true, otherwise it is invalid.

2.3.5 Example

The two Hash functions H_1, H_2 are defined as follows based on algorithms discussed in Section 2.3

def H_1(uid,p)

t = str(uid).hexdigest(); thash = int(hashlib.sha224(t,16)

var = modulo(thash,p)

return var

NOTE: The length extension attack and pseudo preimage attack on sha224 are not possible here because we are adopting proactive secret sharing technique to avoid these attacks.

def H_2(E,p,q,uid,h)

$y_1 = H_1$(uid,p)

x_1 = pow(($y_2^2 - 1$),((2*p - 1)/3),p)

$Q_1 = E(x, y)$

a_1 = int($\frac{p+1}{q}$)

Q = $a_1 * Q_1$

return Q

- **Setup**
- Let $U_M = \{U_1, U_2, U_3, U_4\}$ be the initial set of users
  Number of users = 4
- The Public Parmeters available to all the users are:
  " An additive group G of prime order q = 4019.
  - The curve used is $E(F_{4019}) : y^2 = x^3 + 1$
  - The Generator is P = E(3198,578)
  - Let t = 2 (degree of polynomials) and q = 67 (q is the order of subgroup) "
- Here we use Weil Pairing as bilinear pairing
- The two explicit hash functions which are collision resistant are - H_2(Hash the given string to Point) : $\{0,1\}^* \rightarrow tt_1$ and another hash functions H_1(Hash the given string to the Range) : $\{0,1\}^* \rightarrow tt$.
- All the users randomly selects a bi-variate polynomial that has symmetry in $x_1$andx_2 in GF(67)
  " $U_1 = 3(x_1)^2(x_2) + 3(x_2)^2(x_1) + 8(x_1)(x_2) + 5(x_2) + 5(x_1) + 5$
  $U_2 = 5(x_1)^2(x_2) + 5(x_2)^2(x_1) + 3(x_1)(x_2) + 8(x_2) + 8(x_1) + 9$
  $U_3 = 8(x_1)^2(x_2) + 8(x_2)^2(x_1) + 5(x_1)(x_2) + 3(x_2) + 3(x_1) + 6$
  $U_4 = 2(x_1)^2(x_2) + 2(x_2)^2(x_1) + 4(x_1)(x_2) + 8(x_2) + 8(x_1) + 4$"

2. USER AUTHENTICATION USING BLS SIGNATURE IN DISTRIBUTED PKI BASED NETWORKS

- Implicitly each user has a polynomial defined in $x_1$and x_2 is given by
  $F((x_1), (x_2)) = U_1 + U_2 + U_3 + U_4$
  = "$18(x_1)^2(x_2) + 18(x_1)(x_2)^2 + 20(x_1)(x_2) + 24(x_1) + 24(x_2) + 24$"

- The network secret S_k = F(0,0) = 24.

- Now every user privately communicates other users the univariate polynomial in x_1 as $F_{ij} = F_i((x_1), H_1(U_j)), 1 \leq j \leq 4$.

- The hash values of the users are
  $h_{u1} = (H_1)(\text{"}User1\text{"}, q) = 37$
  $h_{u2} = (H_1)(\text{"}User2\text{"}, q) = 54$
  $h_{u3} = (H_1)(\text{"}User3\text{"}, q) = 25$
  $h_{u4} = (H_1)(\text{"}User4\text{"}, q) = 17$

- Now all the users substitute the hash values of other users identity in their polynomial and sends the messages Y_1, Y_2, Y_3 and Y_4 by users u_1, u_2, u_3 and u_4 respectively.

- Y_1 = P * 5 is also included by u_1 that value is(152,1437), a point on curve
  $u_{11} = 53(x_1) + 44(x_1)^2 + 56$
  $u_{12} = 6(x_1) + 28(x_1)^2 + 7$
  $u_{13} = 3(x_1) + 8(x_1)^2 + 63$
  $u_{14} = 3(x_1) + 51(x_1)^2 + 23$

- Y_2 = P * 9 is also included by u_2 that value is(409,2266), a point on curve
  $u_{21} = 63(x_1) + 51(x_1)^2 + 37$
  $u_{22} = 10(x_1) + 2(x_1)^2 + 39$
  $u_{23} = 59(x_1) + 58(x_1)^2 + 8$
  $u_{24} = 30(x_1) + 18(x_1)^2 + 11$

- Y_3 = P * 6 is also included by u_3 that value is(3063,3143), a point on curve
  $u_{31} = 18(x_1) + 28(x_1)^2 + 50$
  $u_{32} = 17(x_1) + 30(x_1)^2 + 34$
  $u_{33} = 36(x_1) + (x_1)^2 + 14$
  $u_{34} = 55(x_1) + 2(x_1)^2 + 57$

- Y_4 = P * 4 is also included by u_4 that value is(3863,2497), a point on curve u_4 also includes Y_4 = 4 * P = (3863,2497)

  $u_{41} = 13(x_1) + 7(x_1)^2 + 32$

  $u_{42} = 26(x_1) + 41(x_1)^2 + 34$

  $u_{43} = 18(x_1) + 50(x_1)^2 + 3$

  $u_{44} = 51(x_1) + 34(x_1)^2 + 6$

- Each user adds the values received from other users with his own value and computes their secret polynomial in x_1.

- $S_(U1)((x_1)) = 13(x_1) + 63(x_1)^2 + 41$

- $S_(U2)((x_1)) = 59(x_1) + 34(x_1)^2 + 47$

- $S_(U3)((x_1)) = 49(x_1) + 48(x_1)^2 + 21$

- $S_(U4)((x_1)) = 5(x_1) + 38(x_1)^2 + 30$

- " The Networks public key, PK = s * P "
  = " 24 * E(3198,578) = E(2651, 2267) "

- " PK should also equal to $Y_1 + Y_2 + Y_3 + Y_4$ "
  " =E(152,1437)+E(409,2266)+E(3063,3143)+E(3863,2497)
  = E(2651, 2267) "

- All the users compute their share as $S_(Ui)(0)$.
  From the above polynomials, the users shares are
  $S_3 = 21, S_2 = 47, S_1 = 41, S_4 = 30$

- once all the users have their shares, they can be validated with the polynomial of the network
  $f(x_2) = F(0, (x_2))$
  " $= 24 * (x_2) + 24$ "

- Let U_5 be new user who wants to be part of the network, then he identifies himself to 3 other users and ask them to take him in. $\{U_2, U_3, U_4\}$
  $h_{u5} = (H_1)(\text{"}User5\text{"}, k) = 27$

2. USER AUTHENTICATION USING BLS SIGNATURE IN DISTRIBUTED PKI BASED NETWORKS

- U_5 gets the following messages from users 2,3 and 4
  $S_{U25} = S_(U2)(27)\ modulo\ 67 = 28$
  $S_{U35} = S_(U3)(27)\ modulo\ 67 = 22$
  $S_{U45} = S_(U4)(27)\ modulo\ 67 = 62$

- " U_5 computes its secret univariate polynomial by using Lagrange interpolation $S_5((x_1)) = 17 * (x_1)^2 + 18 * (x_1) + 2$ "

- **Key Generation for secure communication**

- Every user selects their public key and private key for secure communication :
  "U_1 = [public key is (89,649) and corresponding private key is (189,649)]
  U_2 = [public key is (17,321) and corresponding private key is (25,321)]
  U_3 = [public key is (63,115)and corresponding private key is (7,115)]
  U_4 = [public key is (91,202) and corresponding private key is (11,202)]"

- **BLS Signature Generation using partial shares**

- The users shares in the network secret are:
  " $S_1 = 41, S_2 = 47, S_3 = 21, S_4 = 30$ "

- Each user creates their own certificate, which contains their Id and public key
  " m_1='User1'+'89'+'649' "
  " m_2='User2'+'17'+'321' "
  " m_3='User3'+'63'+'115' "
  " m_4='User4'+'91'+'202' "

- Each users requests other users for their partial signatures and computes theirs certificate.

- To get the certificate that is fully signed, User 1 creates his certificate as " (m_1='User1'+'89'+'649') rq, and requests User 2, User 3 and User 4 to partially sign on that certificate".

- Here s = 24, P represents a point E(3198,578) and mpub value is E(2651,2267) and s4 = 30, s3 = 21,s2 = 47.

- " $hm_1 = (H_2)(m_1) = E(163, 1362)$ "

- User 2, User 3 and User 4 partially signs as
  " $p_2 = (hm_1) * s2$ "
  " $p_3 = (hm_1) * s3$ "
  " $p_4 = (hm_1) * s4$ "
- The signature " shm1=E(2350,3239)" is obtained after substituting the partial signatures in Lagranges interpolation".
- **BLS Signature Veriftcation**
- " Calculate e(hm1,mpub)=1365*a + 2045 "
- " Calculate e(shm1,P)=1365*a + 2045 this is equal to e(hm1,mpub) "
- " Hence Verified "
- Once all the users have the certificates, then they can securely communicate among them using any public key cryptography algorithm.

2.3.6 Security Analysis of Proposed BLS Scheme

" The security of our scheme depends on the secure key distribution and the security of BLS signature. In [9], Blundo showed the use of symmetric t variable polynomial of degree k in distributed networks. The theorem is as follows: In a scheme based on symmetric polynomial, if all coefficients of the symmetric polynomial in t variables of degree k are uniformly chosen in GF(q), then the t-conference key distribution scheme is k-secure, and optimal[9]. The proposed system is proven to be secure and also non-authorized entities had no way to obtain information of the secret key. The theorem is as follows:If **G** is a (τ, t', s')-GDH group, then the Gap Signature Scheme on **G** is (t, q_H, q_S, s)-secure against existential forgery on adaptive chosen-message attacks, where $t \leq t' - 2\tau c_A(q_H + q_S)$ and $s \geq 2e.q_S s'$, and c_A is a small constant (in practice, at most 2). The various parameters $\tau, t', s'..,$ *etc* are discussed in section 2.3 and the proof of the theorem can be found in [11]. As our scheme is also based on the same principles our proposed scheme is also secure. A secure and authenticated channel for communication between the users have to be established in the first step in order to ensure the security of our scheme. "

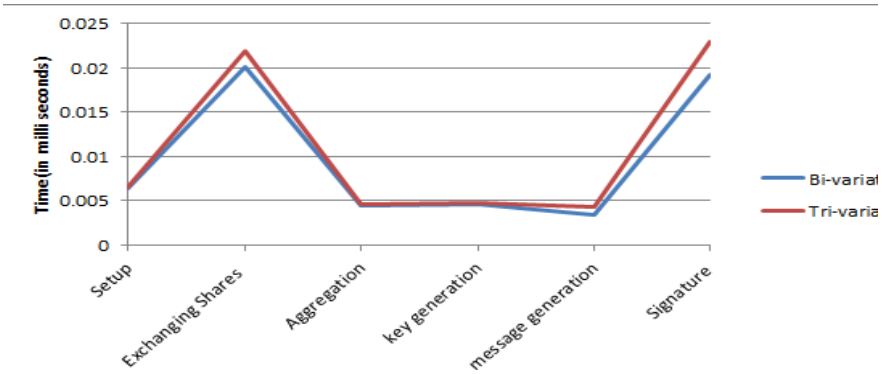
2.4 Comparison between Tri-variate and Bi-variate Polynomials

Our proposed BLS signature protocol is implemented using Bi-variate polynomial and is compared with the existing Tri-variate polynomial implementation and found to be efficient.

Table 2.2: BLS Signature Comparision of bi-variate and Tri-variate

TIME	BI-VARIATE	TRI-VARIATE
Set Up	0.00631594657898	0.00651788711548
Exchanging Shares	0.0201599597931	0.0219240188599
Aggregation	0.00441002845764	0.00461021175385
key generation	0.00465202331543	0.00478997116089
Message generation	0.00334787368774	0.0042599697113
Signature	0.0192819671631	0.0229259147644

Figure 2.1: BLS Signature Comparision of Bi-variate and Tri-variate



2.5 Conclusion

In this chapter, we proposed a protocol based on BLS signature to verify the certificate in the decentralized networks where each users holds a share of the network secret.

The share is distributed to the users among them selves after the few exchanges of the messages in the initial phase. Once the users have their shares, then they prepare a certificate and make them fully signed. They use BLS Signature to generate the certificate. Our scheme uses a bivariate polynomial to reduce the communication overhead. The same technique can be used in performing other functionality of the network like implementing threshold operations in sub group user communication and share verification etc.

Chapter 3

Proactive Secret Sharing For Long Lived Decentralized Networks Using Elliptic Curve Cryptography

In this chapter, we proposed a security protocol for infrastructure-less networks that contains resource-constraint devices. These kind of networks generally use secret sharing schemes to decentralize and distribute the trusted third party's role, where the network secret S_k is distributed among the authenticated users using a (t, n) access structure threshold secret sharing technique. To make the network long live, the shares of the secret are regularly updated without modifying the network secret. This scheme uses the proactive secret sharing concept and Elliptic Curve Cryptography(ECC). The attacker trying to get hold of the secret, should obtain at-least t shares from the users in the given time period. If the threshold value and the time period are selected properly, then the proactive verifiable secret sharing can maintain the overall security of the information in long lived decentralized networks. The traditional security algorithms are heavy weight and requires very much computation power thus utilizing lot of resources. In our proposal, the Elliptic Curve Cryptography is used to verify the commitments because it needs smaller keys compared to the existing proactive secret sharing schemes and thus make it useful for our decentralized networks, which are formed of resource constraint devices.

3.1 Introduction

In threshold cryptography, the private key S_k is distributed among n users of the network using a (t,n) threshold access structure with the help of a secret sharing technique, with each participant u_i having a partial share s_i[48]. For example, in public key cryptography(PKC), let the public key be pk and the corresponding private key be S_k. If a user has encrypted a message using the public key pk then to decrypt the message, atleast t out of n users are required to decrypt the message.

3.2 Background

3.2.1 Shamir Secret Sharing Scheme

The initial proposals on secret sharing techniques are first introduced by Shamir[38] and Blakley[8] in the 1970s. The secret sharing mechanism shares the secret S_k among a group of participants $\{u_1, u_2, \ldots, u_n\}$ of n parties by using a special entity called dealer. The dealer sends privately the share of a secret to each party. Reconstruction process is adopted by the authorized subsets to extract the network secret S_k by pooling their shares. The group of such authorized subsets are called as access structure. The most popular (t,n) access structure is "Shamir secret sharing scheme" [38] that uses the Lagrange's interpolation polynomial, here n denotes the number of users in the network and t is the threshold value. For example let us consider n participants, s is the secret, t is the threshold and the finite field is denoted by F_p. Shamir secret sharing technique uses two phases namely: Share Distribution and Secret reconstruction[6].

Share Distribution

- Choose $\{a_1, a_2, \ldots . a_{t-1}\}$ randomly from the given finite field F_p.
- Now, construct a polynomial of order t-1 and the polynomial is given by: f(x)=$a_{t-1}x^{t-1}$ + $\ldots\ldots a_1x^1$ + a_0. where, a_0 is secret.
- Shares are delivered as (i,y_i). where, "y_i=f(i) mod p and $1 \leq i \leq n$".

Reconstruction

- Lagrange's interpolation is used to obtain the coefficient of the polynomial function f(x) as follows:

"f(x)=$\sum_{i=0}^{t} y_i * l(x)$. Where, $l(x) = \prod_{j=0, j \neq i} \frac{x-j}{i-j}$".

- Now,the secret s is equal to a_0.

3.2.2 Feldman's VSS

"In any secret sharing technique, the dealer sends the share of the secret s_i to user u_i. There may be a scenario, where the dealer may send a wrong partial share or misbehave, so it is necessary for the user u_i to verify the partial secret it received.

A common scheme that helps the user u_i to verify the partial secret s_i is Feldman's Verifiable Secret Sharing scheme. The Feldman's VSS protocol [17] is as follows:

- The dealer choses a cyclic group G of prime order q, and also a generator g of G, as public parameters of the system.
- Then the dealer uses secret sharing technique *i.e.,* The dealer secretly computes a random polynomial $P(x)$ of degree $t-1$ with coefficients in Z_q, such that $P(0) = s$, where s is the secret. Each of the n share holders will receive a value from the set of values $P(1), \ldots, P(n)$ modulo q. Any t share holders can recover the secret s by using polynomial interpolation modulo q, but any set of at most $t-1$ share holders cannot. This is same as Shamir secret sharing scheme.
- Now the dealer distributes commitments to the coefficients of P. If $P(x) = s + a_1x + a_2x^2 + \ldots + a_{t-1}x^{t-1}$, then the commitments are :
  - $c_0 = g^s, c_1 = g^{a1}, \ldots\ldots, c_{t-1} = g^{at-1}$
  - once the commitments are known, every user u_i can verify the share $s_i = v = P(i)$ modulo q, by computing out
    * "$g^{v} = c_0 c_1^{i} c_2^{i_2} \ldots . c_{t-1}^{it-1} = \prod_{j=0}^{t-1}(c_j^{i}) = \prod_{j=0}^{t-1}((g^{a_j})^{i^j}) = g^{\sum_{j=O}^{t-1} a_j * i^j} = g^{p(i)}$".

3.2.3 Bi-variate Polynomial

"In our network the role of the dealer has to be played by the users themselves. To achieve this we use bi-variate polynomials for secret sharing. Similar work is discussed, where Bivariate polynomials have been used to dynamically allow new users to join the network without the need of any external trusted party. Our work is the result of inspiration from the original work of Blundo et al.[9], Nidhi et al Anzai et al.[5],

Abdul et al.[6] and Herranz et al.[15], wherein they constructed decentralized flexible dynamic group key distribution schemes by using polynomials in two variables. The goal is to generate common group secret keys. Saxena et al.[37] used similar technique to establish pairwise keys in a non-interactive way for a mobile ad-hoc scenario.

We choose a bi-variate polynomial that is symmetric in x_1, x_2 with the degree of x_1 *and* x_2 being $t - 1$ where t is the threshold number. An example of bi-variate polynomial symmetric in x_1 and x_2 for $t = 3$ is $p(x_1, x_2) = 5x_1^2 + 4x_1x_2 + 5x_2^2$".

3.3 Our Proposal

In this section, we discuss how the long lived networks are secured using the techniques of proactive secret sharing. The proposed protocol consists of five phases - the first one is *Initial Setup* in which all the users have public parameters and have their own bi-variate polynomial, the second phase is *Share Distribution* in which the users exchange messages to get their share in the network secret, the third one is *Share Verification* in which all the users verify whether they got the correct shares or not, the next phase is *Share Recovery* in which the users can recover their share if it is lost and the final phase is *Share Renewal* in which the users frequently renew their shares to make the network long live.

3.3.1 Initial Setup

There are some parameters which are public: a group G, which is cyclic additive group of prime order q and is generated by the point on elliptic curve P. Another group tt_1 which is cyclic multiplicative of same order is also required. In addition to the above two groups it requires a bilinear pairing e, and two cryptographically secure hash functions $H_1 : \{0,1^*\} \rightarrow Z_q$, another hash function $H_2 : \{0,1^*\} \rightarrow tt_1$, and a public key PK $= P_{pub}$ as discussed in section 2.2.3.2. The hash function H_1(Str,n,hashfcn) takes the input parameters - string Str, integer n, secure hash function hashfcn and gives a value with in the range of " 0 to n-1 ". $n \leq 2^{hashlen}$ where hashlen is " the number of octets comprising the output of the hash function hashfcn ". The Merkle's method [30] is used to create the H_1 hash function and it is also provably secure hash function because of underlying hashfcn hash function.

The hash function H_2(E,p,q,id,hashfcn). The input parameters to the function H_2 are

: E - elliptic curve ,p and q are two primes, id - a string and hashfcn - hash function. The return value of the function is $Q_{id} = (x, y)$ is a point on elliptic curve and has prime order q. The two hash functions are given below as defined in RFC 5091[12].

3.3.2 Share Distribution

In the share distribution phase all the users of the network receive their share s_i of the secret key S_k. The following protocol is used to achieve this.

- Let the threshold value is t, k is the number of founding users and n is the total number of users supported by the decentralized network.
- The founding users are the users who will be present during the initial setup of the network. The number of founding users are k, the k valus must be greater than t and less than n.
- Bivariate polynomial $f_i(x_1, x_2)$ of maximum degree k-1 and has symmetric in x_1, x_2 is chosen by all the users.
- Every user v_i computes $f_{ij}(H_1(v_j), x_2)$ for itself and other founding users, $1 \leq j \leq k$.
- Every user computes $f_{v_{ij}}(H_1(v_j), x_2)$ and secretly sends to other users corresponding v_j. In addition to that each user v_i also includes the chosen elliptic curve point $w_i = f_i(0, 0) * P$ in their messages.
- Once all the messages are exchanged then each user has his own value $f_{ii}(H_1(v_i), x_2)$ with it and also the messages received in the previous step from other users. Finally each user v_i calculates $f_i(x_2) = f(H_1(v_i), x_2) = \Sigma_{j \in k} f_{ji}(H_1(v_i), x_2)$
- Once the above steps are completed then each user v_i has their share $s_i = f_i(0)$ of the network secret and a uni-variate secret polynomial $f(H_1(v_i), x_2)$.

The network polynomial $f(x_1, x_2) = \Sigma_{i \in v} f_i(x_1, x_2)$ and $s_k = f(0, 0)$, the network secret key are hidden and unknown to the users. To reconstruct the network secret key S_k, at least t users has to contribute their shares.

3.3.2.1 Joining of New user(User Aggregation)

If a new user v_w wants to join, then he has to identify himself to least t users of the network and request for the values $f_{iw}(H_1(v_w), H_1(v_i))$. Once the user v_w has t such values then he uses Lagrange's interpolation to generate his uni-variate secret polynomial. $f_w(x_2)$ can be calculated using Lagrange's interpolation as given below:

- $f_w(x_2) = f(H_1(v_w), x_2) = \Sigma_{v \in v} \Pi_{v \in v, v} \frac{(x_2 - H_1(v_j))}{(H_1(v_j) - H_1(v_i))} * f(H_1(v_w), H_1(v_j))$
- After applying the Lagranges's interpolaion, the user v_w has his share of the secret as $f_w(0)$ and the corresponding uni-variate polynomial is v_w is $f_w(x_2)$ *i.e.*, $f(H_1(v_w), x_2)$

3.3.3 Share Verification

Once a user receives the messages from the other users, it has to be verified by running the following protocol. It ensures that the data is not lost in transmission or the other user is not malicious user.

1. Each user $v_j \in v$ gets the commitment value $(c_1)^{ij}$ corresponding to the message receive from $v_i \in v, f_i(x_1, H_1(v_j))$. The received commitment values is $c_1{}^i{}_{ad} = b_{ad}{}^i * Q$ where b^i_{ad} is the co-efficient of $x_1{}^a x_2{}^d$ in the polynomial equation $f_i(x_1, x_2)$ and $c_1{}^{ij} = c_1{}^i{}_{ad} * H_1(v_j)^d$
2. Each user $v_j \in v$ computes $b^i_d * Q$, where b^i_d is a coefficient of $x_1{}^d$ in $f_i(x_1, H_1(v_j))$.
3. Each user verifies his share s_{ij} (the values received by v_i from v_j by comparing it with $c_1{}^{ij}_d = b^i_d * Q$. If all s_{ij} are correct then s_j value is correct.

3.3.4 Share Recovery

In the actual proactive secret sharing scheme, the share recovery phase is used to recover the users secret it it is lost. In our protocol, in section 3.3.2.1 we discussed a procedure to be followed by new user who is willing to join the network. If any on the network users lost their shares then they can follow the same procedure to recover their shares. If any user $v_i \in v$ want to recover his share then he identifies himself to t of the users of the networks and runs the user aggregation protocol which gives him his share of the network and he again becomes the part of the network.

3.3.5 Share Renewal

To make the decentralized networks long live, all of its users $v_i \in v$ have to frequently update their shares without affecting the secret of the network sk. This can be done with the following protocol:

1. Each user $v_i \in v$ selects new random polynomial $f'(x_1, x_2)$ such that $f'_i(0, 0) = 0$.
2. After selecting the new polynomial each user $v_i \in v$ privately communicates the value $f'_{ij}(x_1, x_2) = f'_i(x_1, H_1(v_j))$ to other network users $v_j \in v$.
3. Once the previous step is completed by all the users $v_i \in v$. Each user $v_j \in v$ calculates their new secret polynomial $f'_j(x_1) = \Sigma_{v_i \in v} f'_{ij}(x_1, x_2) = \Sigma_{v_i \in v} f'_i(x_1, H_1(v_j)) = f'(x_1, H_1(v_j))$.
4. After completing the above steps each user $v_i \in v$ has secret polynomial $s'_i(x_1) = s_i(x_1) + f'(x_1, H(v_i))$ and $s'_i(0)$ is the final share of the secret.

3.3.6 Example

The two Hash functions H_1, $H_2$are defined as follows based on algorithms discussed in Section 2.3

def H_1(uid,p)
t = str(uid).hexdigest();
t_hash = int(hashlib.sha224(t,16))
var = modulo(t_hash,p)
return var

def H_2(E,p,q,uid,h)
y_1 = H_1(uid,p)
x_1 = pow(($y_1^2 - 1$),((2*p - 1)/3))
Q_1 = $E(x_1, y_1)$
a_1 = int($\frac{p+1}{q}$)
Q = $a_1 * Q_1$
return Q

- **Setup**

- Let $U_M = \{U_1, U_2, U_3, U_4\}$ be the initial set of users
  Number of users = 4

- The Public Parmeters available to all the users are:
  " An additive group G of prime order q = 4019.
  - The curve used is $E(F_{4019}) : y^2 = x^3 + 1$
  - The Generator is P = E(3198,578)
  - Let t = 2 (degree of polynomials) and q = 67 (q is the order of subgroup) "

- Here we use Weil Pairing as bilinear pairing

- The two explicit hash functions which are collision resistant are - H_2(Hash the given string to Point) : $\{0,1\}^* \rightarrow tt_1$ and another hash functions H_1(Hash the given string to the Range) : $\{0,1\}^* \rightarrow tt$.

- All the users randomly selects a bivariate polynomial that has symmetry in $x_1$andx_2 in GF(67)
  "$U_1 = 3(x_1)^2(x_2) + 3(x_2)^2(x_1) + 8(x_1)(x_2) + 5(x_2) + 5(x_1) + 5$
  $U_2 = 5(x_1)^2(x_2) + 5(x_2)^2(x_1) + 3(x_1)(x_2) + 8(x_2) + 8(x_1) + 9$
  $U_3 = 8(x_1)^2(x_2) + 8(x_2)^2(x_1) + 5(x_1)(x_2) + 3(x_2) + 3(x_1) + 6$
  $U_4 = 2(x_1)^2(x_2) + 2(x_2)^2(x_1) + 4(x_1)(x_2) + 8(x_2) + 8(x_1) + 4$"

- Implicitly each user has a polynomial defined in $x_1$and x_2 is given by
  $F((x_1), (x_2)) = U_1 + U_2 + U_3 + U_4$
  = "$18(x_1)^2(x_2) + 18(x_1)(x_2)^2 + 20(x_1)(x_2) + 24(x_1) + 24(x_2) + 24$"

- The network secret S_k = F(0,0) = 24.

- Now every user privately communicates other users the univariate polynomial in x_1 as $F_{ij} = F_i((x_1), H_1(U_j))$, $1 \leq j \leq 4$.

- The hash values of the users are
  $h_{u1} = (H_1)(\text{"}User1\text{"}, q) = 37$
  $h_{u2} = (H_1)(\text{"}User2\text{"}, q) = 54$
  $h_{u3} = (H_1)(\text{"}User3\text{"}, q) = 25$
  $h_{u4} = (H_1)(\text{"}User4\text{"}, q) = 17$

- Now all the users substitues the hash values of other users identity in their polynomial and sends the following message:

- Y_1 = P * 5 is also included by u_1 that value is(152,1437), a point on curve
  $u_{11} = 53(x_1) + 44(x_1)^2 + 56$
  $u_{12} = 6(x_1) + 28(x_1)^2 + 7$
  $u_{13} = 3(x_1) + 8(x_1)^2 + 63$
  $u_{14} = 3(x_1) + 51(x_1)^2 + 23$

- Y_2 = P * 9 is also included by u_2 that value is(409,2266), a point on curve
  $u_{21} = 63(x_1) + 51(x_1)^2 + 37$
  $u_{22} = 10(x_1) + 2(x_1)^2 + 39$
  $u_{23} = 59(x_1) + 58(x_1)^2 + 8$
  $u_{24} = 30(x_1) + 18(x_1)^2 + 11$

- Y_3 = P * 6 is also included by u_3 that value is(3063,3143), a point on curve
  $u_{31} = 18(x_1) + 28(x_1)^2 + 50$
  $u_{32} = 17(x_1) + 30(x_1)^2 + 34$
  $u_{33} = 36(x_1) + (x_1)^2 + 14$
  $u_{34} = 55(x_1) + 2(x_1)^2 + 57$

- Y_4 = P * 4 is also included by u_4 that value is(3863,2497), a point on curve u_4 also includes Y_4 = 4 * P = (3863,2497)
  $u_{41} = 13(x_1) + 7(x_1)^2 + 32$
  $u_{42} = 26(x_1) + 41(x_1)^2 + 34$
  $u_{43} = 18(x_1) + 50(x_1)^2 + 3$
  $u_{44} = 51(x_1) + 34(x_1)^2 + 6$

- Each users adds the values receive form other users with his own value and computes their secret polynomial in x_1.

- $S_(U1)((x_1)) = 13(x_1) + 63(x_1)^2 + 41$

- $S_(U2)((x_1)) = 59(x_1) + 34(x_1)^2 + 47$

- $S_(U3)((x_1)) = 49(x_1) + 48(x_1)^2 + 21$

- $S_(U4)((x_1)) = 5(x_1) + 38(x_1)^2 + 30$

- " The Networks public key, PK = s * P "
  = " 24 * E(3198,578) = E(2651, 2267) "

- " PK should also equal to $Y_1 + Y_2 + Y_3 + Y_4$ "
  " =E(152,1437)+E(409,2266)+E(3063,3143)+E(3863,2497)
  = E(2651, 2267) "

- All the users compute their share as $S_(Ui)(0)$.
  From the above polynomials, the users shares are
  $S_3 = 21, S_2 = 47, S_1 = 41, S_4 = 30$

- once all the users have their shares, they can be validated with the polynomial of the network
  $f(x_2) = F(0, (x_2))$
  " $= 24 * (x_2) + 24$ "

- Let u_5 be new user who wants to be part of the network, then he identifies himself to 3 other users and ask them to take him in. $\{U_2, U_3, U_4\}$
  $h_{U5} = (H_1)(^J User5^J, k) = 27$

- U_5 gets the following messages from users 2,3 and 4
  $S_{U25} = S_(U2)(27)\ modulo\ 67 = 28$
  $S_{U35} = S_(U3)(27)\ modulo\ 67 = 22$
  $S_{U45} = S_(U4)(27)\ modulo\ 67 = 62$

- " U_5 computes its secret univariate polynomial by using Lagrange interpolation "
  " $S_5((x_1)) = 17 * (x_1)^2 + 18 * (x_1) + 2$ "

- **Share Veriftcation**

- user 1 is having his polynomial as $U_1 = 5+5*x+5*z+8*x*z+3*z^2*x+3*x^2*z$

- while sending $s_{12} = 5 * x^2 - x + 41$ to the second user, he also includes the Commitment $c_1^{12} = \Sigma c_{ad}^1 * H_1(U_j)^d$.
  $c_1{}^i_{ad} = b^i_{ad} * Q$, where Q belongs to G and b^i_{ad} is co-efficient of $x_1{}^a x_2{}^d$ in $f_i(x_1, x_2)$.

3. PROACTIVE SECRET SHARING FOR LONG LIVED DECENTRALIZED NETWORKS USING ELLIPTIC CURVE CRYPTOGRAPHY

- From the previous step, the commitment C^{12} is given by $H_1(U_2)*3*Q+H_1(U_2)^2*3*Q+H(U_2)*8*Q+Q*5+H_1(U_2)*5*Q$
- The value of C^{12} is given by the point(58,35).
- User 2 receives $s_{12} = 41 + 5 * x^2 - x$ and the commitment value C^{12}
- User 2 computes $\Sigma b^j_a * Q$, where Q belongs to G and b^j_a is co-efficient of x in $s_{12} = f_i(x, h(u_2))$ that is, $Q * 5 + Q * 82 + Q * 41$ which gives the point (58,35) and is equal to C^{12}.
- since both are giving the same point the algorithm is verified.
- **Share Renewal Example**
- If four users (u1,u2,u3,u4) wants do share renewal and each of them have a polynomial.

- $s_1(x_1) = -x + 46x_1^2 + 30$
- $s_2(x_1) = 30x_1^2 + 52x_1 + 64$
- $s_3(x_1) = -28 * x_1^2 + 18 * x_1 - 41$
- $s_4(x_1) = 21 - 23 * x_1^2 + 14 * x_1$
  The network secret is 24.
- To renew their shares all the users randomly selects bivariate polynomial without constant term.
  $u_{11} = f^1(x_1, x_2) = x^2x_2 + x_1x^2 + 5x_1x_2 + 2x_1 + 2x_2$
  $u_{21} = f^1_2(x_1, x_2) = 2x^2_1x_2 + 2x_1x^2_2 + 20x_1x_2 + 4x_1 + 4x_2$
  $u_{31} = f^2(x_1, x_2) = 3x^2_1x_2 + 3x_1x^2_2 + 12x_1x_2 + x_1 + x_2$
  $u_{41} = f^3_4(x_1, x_2) = 4x^2_1x_2 + 4x_1x^2_2 + 3x_2 + 3x_1 + 8x_1x_2$
- All the users $u_i \in U$ privately exchanges $f^j_{ij}(x_1, x_2) = f^j_i(x_1, H_1(u_j))$ to other users $u_j \in U$
- All the users run the share distributing algorithm again to exchange their polynomials. The network polynomial is implicitly given by $f^j(x_1, x_2) = \Sigma_{n_i \in N} f^j_i(x_1, x_2)$.

- The newly computed secret polynomials of all the users $u_i \in U$ are
  $s'_i(x_1) = s_i(x_1) + f'(x_1, H_1(u_i))$.
  $s'_1(x_1) = 7 * x_1^2 - 9 - 31 * x_1$
  $s'_2(x_1) = 19 * x_1^2 - 30 + 8 * x_1$
  $s'_3(x_1) = 21 * x_1^2 - 8 + 26 * x_1$
  $s'_4(x_1) = 38 * x_1^2 - 10 * x_1 + 66 - 1x_1$

- "The updated partial secret share $s'_i(0) = s_i(0) + f'(0, h(n_i))$.
  $s'_1 = 74$ $s'_2 = 53$ $s'_3 = 8$ $s'_4 = 82$ "

3.3.7 Security Analysis of Proactive Secret Sharing

The correctness and security of the proposed scheme is discussed in this section. We assume that during the networks life time the maximum number of malicious users in the network are at most *t-1* and the maximum value of $t = n/2 - 1$ because of the *(t,n)* access structure of the secret sharing.

Theorem 1 : "The network secret s is always secure even if there are at most *t-1* corrupted users."

Proof : "To construct the network secret s, at least t users have to co-operate. Since we employed a *(t,n)* threshold scheme, where in at least t partial shares of users are required to reconstruct the network secret using Lagrange's interpolation. If $t-1$ corrupted users try to reconstruct the secret, they can not generate the original polynomial because at least t points are required to generate actual polynomial."

Theorem 2 : "If a corrupted user is able to control up to t-2 users in the share renewal phase, the honest users still can renew their shares."

Proof : "We assume that there are at least t honest users at any time in the lifetime of the network. When the users want to renew their shares, both honest and corrupted users participate in share renewal phase. But the values contributed by the corrupted users are discarded using the share verification protocol discussed in Section 3.3.3. Thus the *t*-honest users can still renew their shares though the malicious user controls $t-2$ shares."

Theorem 3 : "The various public parameters (*tt, q, Q, PK*) doesn't leak any information regarding the the network secret s."

Proof : "The security of the scheme is dependent on the hardness in solving the elliptic curve discrete logarithm problem(ECDLP), i.e. Let E be an elliptic curve over finite field Z. Suppose there are points $Q, R \in E(K)$ given such that $R \in < Q >$. The network public key PK is $s * Q$ and the generator is Q. Though both Q and PK are public, it is not possible to calculate s, because of ECDLP."

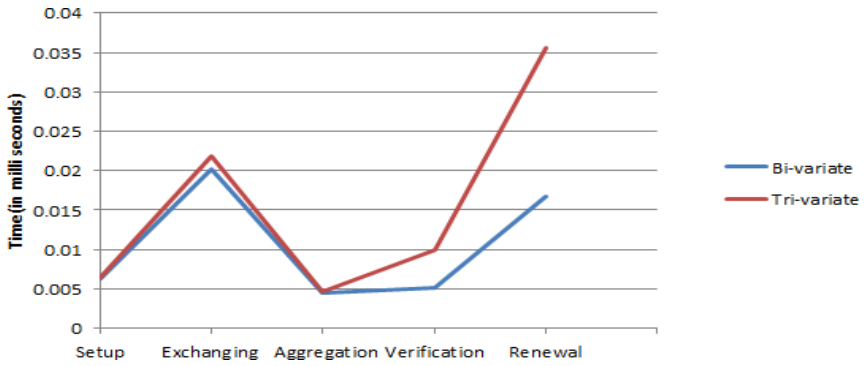
3.4 Comparison between Bi-variate and Tri-varaiate polynomials

Our proposed Proactive secret sharing protocol is implemented using Bi-variate polynomial and is compared with the existing Tri-variate polynomial implementation and found to be efficient.

Table 3.1: Proactive Secret Sharing Comparision of Bi-variate and Tri-variate

TIME	BI-VARIATE	TRI-VARIATE
Set Up	0.00631594657898	0.00651788711548
Exchanging	0.0201599597931	0.0219240188599
Aggregation	0.00441002845764	0.00461021175385
Verification	0.00506997108459	0.00992798805237
Renewal	0.016654968261	0.0356330871582

Figure 3.1: Proactive Secret Sharing Comparision of Bi-variate and Tri-variate



3.4.1 Conclusion

In decentralized networks because of the absence of the centralized authority the role of the CA is distributed among the network users. This is achieved by using a bi-variate polynomial that implement shamir's secret sharing technique with (t,n) access structure. Once all the network users have their share in the network secret, protecting their share for the entire network's life time is difficult. An attacker may collect enough number of information to gain the access of the share. To avoid this, the share of the users have to change frequently without affecting the actual secret of the network. We achieved this by using the concept of proactive secret sharing and implemented with bi-variate polynomials. We also used the concepts are Elliptic curve cryptography to verify the user shares.

Chapter 4

Lightweight Cryptography for Distributed PKI Based Networks

Secure communication is very challenging in distributed networks because of the lack of Central Authority(CA). Here we discussed a suitable protocol that is suitable to the decentralized networks that are formed of resource constraint device and are dynamic in nature. The users themselves manage the entire network without any fixed infrastructure or centralized authority and the power of CA is distributed among the users of the network. We used secret sharing mechanism to create a distributed PKI where all the users have a share in network private key. The traditional PKI protocols are not suitable as they need heavy computing power and a centralized authority to manage private and public keys. In this chapter, we proposed the use of a lightweight crypto algorithm for the secure communication of the users.

4.1 Introduction

The secret sharing concept is first introduced by Blakley [8] and Shamir [38]. The secret sharing scheme consists of a set $V = \{v_1, v_2, v_3, ..., v_m\}$ of m users and a dealer d . The secret s is kept with the dealer and he sends privately s_i which is the share of the secret. All the users of the network recieve their corresponding shares from the dealer. The secret can be reconstructed when atleast t number of valid users contribute their shares. In (t, m) access structure t is the trust level and m denotes all network users. [38].

In our proposed protocol, "Shamir's secret sharing with (t, n) threshold access structure" is used[38]. Shamir secret sharing uses polynomial interpolation to implement (t, m) access structure. Let us consider the finite field F_p with $p > m$ and let the secret S_k of the network in F_p. To implement (t,m) access structure "a polynomial $P(x)$ with at most $t-1$ degree" is chosen by the dealer. In the selected polynomial the constant term represents the secret S_k and all other coefficients are independently and randomly chosen from the field F_p. From this $P(x)$ is given by $\sum_{i=1}^{t-1} a_i * x^i + S_k$. A distinct field element a_i is publicly associated with each user v_i. By substituting the a_i in the polynomial, the dealer generates $[s]_i = P(a_i), \forall i = 1, \ldots, m$. If the set of t users $u_1, u_2 \ldots, u_t$ are willing to generate the secret, they it can be obtained by computing $\sum_{i=1}^{t} l_i * [s]_i$, here the Lagrange coefficients are given by $l_i = \Pi_j \; i\frac{a_j}{a_j - a_i}$. It is not possible to generate the secrete with less than t number of users.

4.1.1 Elliptic Curve Cryptography (ECC)

Victor Miller and Neil Koblitz were independently proposed the concept of ECC systems in 1985 [26]. ECC is defined based on public key encryption algorithms, which use algebraic structure over the elliptic curves that are implemented in finite fields. Compared to other public key algorithms(non-ECC), Ecc requires private keys of smaller size and will provide the security equivalent to non-ECC algorithms. The ECC security depends on the hardness of discrete logarithm problem that is projected over elliptic curves form the finite fiels. The Ellicptic curve discrete logorithm (ECDLP) problem is equally hard as DLP problem in finite fields[31].

4.1.1.1 Elliptic Curve Discrete Logarithm Problem (ECDLP)

"Given Q, point on elliptic curve and s*Q, another point on the elliptic curve then finding the integer s is called ECDLP. All the ECC systems are defined based on the difficulty in solving the ECDLP. The attacker can break the ECC system if he can solve the ECDLP, but ECDLP is much harder than Discrete Logarithm Problem(DLP) defined over the finite fields". The known efficient schemes that are generally used to solve DLP in finite fields are like the index calculus method, baby step and giant step technique proposed by Shank, the Pohlig - Hellmann method and the Pollard's ρ method don't work in solving ECDLP problem. Most of them work only if a large prime can divide the group order. In 1993, Okamoto, Menezes and Vanstone reduced

the problem of ECDLP into DLP over $F^*_{(q^k)}$ but this technique works only for super singular curves that is the curves represented in the form of "$y^2 = x^3 + ax$" and having characteristic "p of $F_q \equiv -1(mod 4)$" and also curves represented by "$y^2 = x^3 + b$ when $p \equiv -1(mod 3)$" that is the values of k is small for these types of curves. Most of the elliptic curves are non singular and the above MOV method works on only very few classes of the curves.

4.1.1.2 The Diffie Hellman Key Exchange using ECC

The most challenging task of any Symmetric Encryption algorithm is to convery the secret key between sender and reciever without the attacker getting it.[3]. The Diffie-Hellman algorithm allows us to do this transfer of key securely over the insecure channel. This algorithm generates the same symmetric key at both sender and receiver sides secretively without anyone intercepting it. This algorithm was first developed by Martin Hellman and Whitfield Diffie in 1976. The basic goal of this algorithm is not data encryption rather it generates a secure symmetric private key at sender side and receiver side, so that the actual key does not have to transmit over the network[26]. Despite of being slow in generating the symmetric key, the algorithm is popular because of its sheer power in secure private key generation. The following procedure generates a common secret key between any two users A and B of the network using ECC.

- The two users A and B agrees on F_q, a finite field and E, an elliptic curve defined over (F_q).
- They also choose public random generator point P that belongs to the curve E.
- The user A randomly selects an integer a, and computes and sends the value $a.P \in E$ to user B.
- The user B randomly selects an inter b and computes and sends the value $b.P \in E$ to user A.
- After exchanging the messages user A calculates the symmetric key $s = a(b.P)$ and B calculates symmetric key as $s = b(a.P)$. so they both have the common secret point s.

- Even if the attacker get hold of the information - base point P, message transmitted by A $a.P$, and the message transmitted by B $b.P$, there is no easy way to calculate abP because of ECDLP.

4.1.2 Lightweight Cryptography

Lightweight cryptography is the collection of cryptographic primitives, techniques and ciphers that can be implemented in highly resource constraint devices. Lightweight cryptography is not the alternative to the traditional cryptographic algorithms. so while using these algorithms, we have to trade off between security and lightweightness. In order to get high security levels just by using small computing power, several lightweight stream ciphers, block ciphers, hash functions and one pass authenticated encryption are proposed by many cryptographers. To design lightweight algorithms, the computational complexity and the hardware requirements and other restrictions related to the target device are to be analyzed. These algorithms provide hardware implementation in low resource devices, which are basis for designing ubiquitous computing devices like a sensor in Radio Frequency Identification (RFID) tag. “In the lightweight context, designer has to analyze the computational complexity of the algorithm, with respect to the demands on the hardware and other limitations of the device”. Roughly, the weight of ”lightweight” primitive is the amount of resources necessary both in terms of time and space for it to run. This weight can be measured in two distinct contexts: in software and in hardware. Lightweights in software does not imply lightweights in hardware and vice-versa. Finally, a measure which is relevant in both contexts is the power consumption. In this chapter we considered TEA - Tiny Encryption Algorithm, which is most efficient and one of the popular and fastest lightweight algorithm.

4.1.2.1 Tiny Encryption Algorithm(TEA)

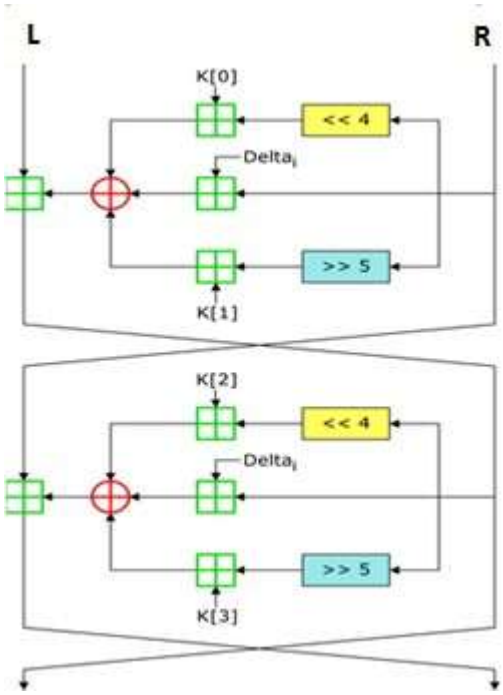
Roger Needham and David Wheeler of Cambridge University created TEA algorithm as a symmetric Feistel type of ciphers. These kind of Feistel ciphers are a special class of iterated block ciphers. The TEA algorithm is very simple which contains simple arithmetic and logical operations, faster to compute and most efficient symmetric key algorithm. TEA is created to encrypt the data with the symmetric key with maximum speed and smaller memory foot print. Tiny Encryption Algorithm user the simple operations like additions, shifts and XORs for 32 rounds. This Algorithm has the key

length of 128-bit keys and the data block size of 64-bits. It performs operations on 32 bit words. In addition to the 128-bit key, TEA also uses a magic constant which is given by the golden ratio. When represented in the integer format, the golden ration has the value 2654435769. In each round the multiples of this golden ration is used to prevent the attacker from not taking the advantage of the similarity in the feistel rounds. The TEA Algorithm is a simple block cipher, which contains very few lines of code, requires very less storage space and very secure.

So far there are no known attacks on TEA algorithms that are successful. This algorithm is also secure as IDEA algorithm, which is developed by Xuejia Lai and Massey. Similar to IDEA algorithm, TEA also uses the algebraic group technique in much simpler form thus making it much faster. In TEA, the same round function or transformation is applied again and again on plain text to generate the cipher text. Since it is a Feistel cipher, first it splits the plain text into two halves then the transformation F is applied using a sub key on one half of the plain text and the output of this transformation F is XORed on plain text in the other half. These two parts are then exchanged. The above mentioned procedure is followed on each round except the final round where the swapping is not carried out.

ENCRYPTION OF TEA In tea encryption[21] process, initially 64 bit plain text will be divided into two halves of 32-bit each, namely L and R. TEA uses a key of lenght 128 bits that is divided into each 32 bits of four blocks namely k[3], k[2], k[1], k[0] and each key is added at different rounds of encryption respectively. Initially R will under go left shift of four bits and added to key k[0] and stored(for eg: m1). Then again R is taken and added to delta (0x9E3779B9) value, which is a key schedule constant and it is stored(eg: m2). Then R is right shifted to 5 bits and added to key k[1] and stored (eg:m3). Then all the above stored values will be XORed (I.e. ((m1 XOR m2)XOR m3)). This value will be finally added to L value and this value will be stored. Then both the values will be swapped thus completing one round of encryption and newly stored value will now become the initial input after swapping. This type of encryption will continue till 64 rounds with adding (k[3],k[2]),(k[1],k[0]) in consecutive rounds. Refer figure 4.1.

Figure 4.1: TEA Encryption

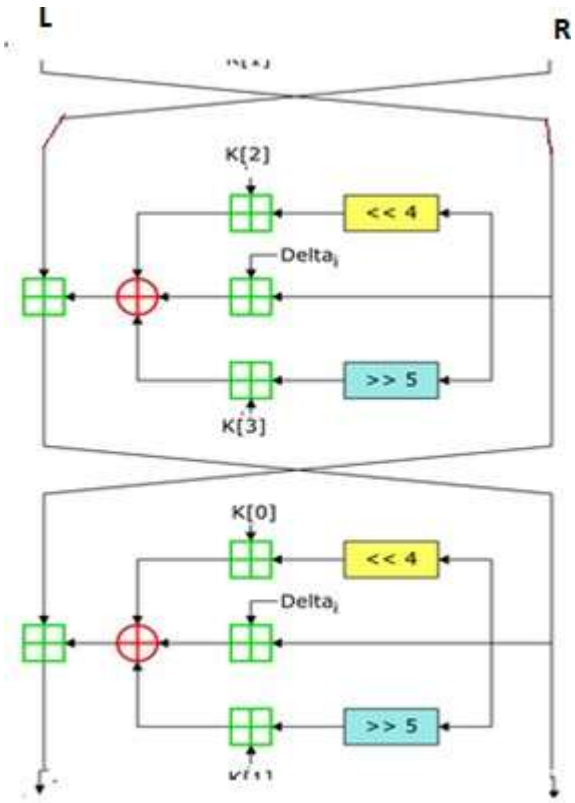


DECRYPTION OF TEA The Decryption is similar to encryption but in reverse process. Suppose the output obtained after all 64 rounds of encryption are L* and R*, then these two are treated as new inputs. Initially R* is right shifted to four bits and added to key k[2] and stored (eg n1) then R* is added to delta sum value and stored in (n2), again R* will be right shifted and added to key k[3] and stored in (n3). These stored values are XORed (n1 XOR n2)XOR n3 and the result is added with L* and L* will be updated to its new value. These values are swapped to get one round of decryption. In this way we need to decrypt all 64 rounds to get our result by using k[3],k[2] and k[1].k[0] keys in simultaneous rounds. Refer figure 4.2

4.2 Proposed System

In our Proposal, TEA algorithm is used to establish a secure communication among the users of the decentralized network. There are four phases in this protocol, the first phase is *Initial Setup* - here the users of the network agrees on public parameters that are needed to run the protocol, in the second phase *Share Distribution* - all the

Figure 4.2: TEA Decryption



users exchanges messages and get a share of the network secret. The third phase is *Computing SecretKey*, here the users uses Diffie-Hellman key exchange protocol to obtain their symmetric secret key and the last phase is *Secure Communication*, here the users use TEA algorithm for secure communication.

4.2.1 Initial Setup

"There are some parameters which are public:" a group G, which is cyclic additive group of prime order q and is generated by the point on elliptic curve P. Another group tt_1 which is cyclic multiplicative of same order is also required. In addition to the above two groups it requires a bilinear pairing e, and two cryptographically secure hash functions "$H_1 : \{0, 1^*\} \rightarrow Z_q^*$", another hash function "$H_2 : \{0, 1^*\} \rightarrow tt_1$", and a public key PK = P_{pub} as discussed in section 2.2.3.2. The hash function H_1(Str,n,hashfcn) takes the input parameters - string Str, integer n, secure hash function hashfcn and gives a value with in the range of " 0 to n-1 ". $n \leq 2^{hashlen}$ where hashlen denotes " the number of octets comprising the output of the hash function hashfcn ". The Merkle's method [30] is used to create the H_1 hash function and it is also provably secure hash function because of underlying hashfcn hash function.
The hash function H_2(E,p,q,id,hashfcn). The input parameters to the function H_2 are : E - elliptic curve ,p and q are two primes, id - a string and hashfcn - hash function. The return value of the function is $Q_{id} = (x, y)$ is a point on elliptic curve and has prime order q.

4.2.2 Share Distribution

In the share distribution phase all the users of the network receive their share s_i of the secret key S_k. The following protocol is used to achieve this.

- Let the threshold value is t, k is the number of founding users and n is the total number of users supported by the decentralized network.

- The founding users are the users who will be present during the initial setup of the network. The number of founding users are k, the k valus must be greater than t and less than n.

- Bivariate polynomial $f_i(x_1, x_2)$ of maximum degree k-1 and has symmetric in x_1, x_2 is chosen by all the users.
- Every user v_i computes $f_{ij}(H_1(v_j), x_2)$ for itself and other founding users, $1 \leq j \leq k$.
- Every user computes $f_{v_{ij}}(H_1(v_j), x_2)$ and secretly sends to other users corresponding v_j. In addition to that each user v_i also includes the chosen elliptic curve point $w_i = f_i(0, 0) * P$ in their messages.
- Once all the messages are exchanged then each user has his own value $f_{ii}(H_1(v_i), x_2)$ with it and also the messages received in the previous step from other users. Finally each user v_i calculates $f_i(x_2) = f(H_1(v_i), x_2) = \Sigma_{j \in k} f_{ji}(H_1(v_i), x_2)$
- Once the above steps are completed then each user v_i has their share $s_i = f_i(0)$ of the network secret and a uni-variate secret polynomial $f(H_1(v_i), x_2)$.

The network polynomial $f(x_1, x_2) = \Sigma_{i \in v} f_i(x_1, x_2)$ and $s_k = f(0, 0)$, the network secret key are hidden and unknown to the users. To reconstruct the network secret key S_k, at least t users has to contribute their shares.

4.2.3 Computing a Secret Key

If the users $u_i, u_j \in U$ want to communicate securely then they need a common secret symmetric key, which is used to encrypt the messages and the same key is used for decryption also. The users follow the "Diffie Hellman Key exchange algorithm" discussed in the section 4.1.1.2 to compute the secret key. They use the following protocol. The General Algorithm for Diffie Hellman is as follows:

1. u_i and u_j chooses random numbers $a, b \in F_p$.
2. u_i sends a point $A = a * P$ and u_j sends $B = b * P$ to u_i (For Group G, P is the generator)
3. u_i computes a point "$R = a * B = a * b * P$" and u_j computes "$R = b * A = a * b * P$".
4. Both u_i and u_j have same secret point R, to use this point as a secret key sk, add x and y coordinates of R i.e $sk = R_x + R_y$.

4.2.4 Secure Communication

Once the users $u_i, u_j \in U$ have a common secret key sk then they communicate securely using TEA algorithm discussed in section 2.3.1.

4.2.5 Example

The network setup is similar to Chaitanya et al. discussed in [27].

The two Hash functions H_1, $H_2$are defined as follows based on algorithms discussed in Section 2.3

def H_1(uid,p)

t = str(uid).hexdigest(); thash = int(hashlib.sha224(t,16)

var = modulo(thash,p)

return var

def H_2(E,p,q,uid,h)

y_1 = H_1(uid,p)

x_1 = pow(($y_2^2 - 1$),((2*p - 1)/3),p)

$Q_1 = E(x, y)$

a_1 = int($\frac{p+1}{q}$)

Q = $a_1 * Q_1$

return Q

- **Setup**

- Let $U_M = \{U_1, U_2, U_3, U_4\}$ be the initial set of users
  Number of users = 4

- The Public Parmeters available to all the users are:
  " An additive group G of prime order q = 4019.
  - The curve used is $E(F_{4019}) : y^2 = x^3 + 1$
  - The Generator is P = E(3198,578)
  - Let t = 2 (degree of polynomials) and q = 67 (q is the order of subgroup) "

- Here we use Weil Pairing as bilinear pairing

4. LIGHTWEIGHT CRYPTOGRAPHY FOR DISTRIBUTED PKI BASED NETWORKS

- The two explicit hash functions which are collision resistant are - H_2(Hash the given string to Point) : $\{0,1\}^* \rightarrow tt_1$ and another hash functions H_1(Hash the given string to the Range) : $\{0,1\}^* \rightarrow tt$.

- All the users randomly selects a bivariate polynomial that has symmetry in $x_1$andx_2 in GF(67)
  " $U_1 = 3(x_1)^2(x_2) + 3(x_2)^2(x_1) + 8(x_1)(x_2) + 5(x_2) + 5(x_1) + 5$
  $U_2 = 5(x_1)^2(x_2) + 5(x_2)^2(x_1) + 3(x_1)(x_2) + 8(x_2) + 8(x_1) + 9$
  $U_3 = 8(x_1)^2(x_2) + 8(x_2)^2(x_1) + 5(x_1)(x_2) + 3(x_2) + 3(x_1) + 6$
  $U_4 = 2(x_1)^2(x_2) + 2(x_2)^2(x_1) + 4(x_1)(x_2) + 8(x_2) + 8(x_1) + 4$"

- Implicitly each user has a polynomial defined in $x_1$and x_2 is given by
  $F((x_1), (x_2)) = U_1 + U_2 + U_3 + U_4$
  = "$18(x_1)^2(x_2) + 18(x_1)(x_2)^2 + 20(x_1)(x_2) + 24(x_1) + 24(x_2) + 24$ "

- The network secret S_k = F(0,0) = 24.

- Now every user privately communicates other users the univariate polynomial in x_1 as $F_{ij} = F_i((x_1), H_1(U_j))$, $1 \leq j \leq 4$.

- The hash values of the users are
  $h_{u1} = (H_1)(\text{"}User1\text{"}, q) = 37$
  $h_{u2} = (H_1)(\text{"}User2\text{"}, q) = 54$
  $h_{u3} = (H_1)(\text{"}User3\text{"}, q) = 25$
  $h_{u4} = (H_1)(\text{"}User4\text{"}, q) = 17$

- Now all the users substitues the hash values of other users identity in their polynomial and sends the following message:

- Y_1 = P * 5 is also included by u_1 that value is(152,1437), a point on curve
  $u_{11} = 53(x_1) + 44(x_1)^2 + 56$
  $u_{12} = 6(x_1) + 28(x_1)^2 + 7$
  $u_{13} = 3(x_1) + 8(x_1)^2 + 63$
  $u_{14} = 3(x_1) + 51(x_1)^2 + 23$

- Y_2 = P * 9 is also included by u_2 that value is(409,2266), a point on curve
  $u_{21} = 63(x_1) + 51(x_1)^2 + 37$
  $u_{22} = 10(x_1) + 2(x_1)^2 + 39$
  $u_{23} = 59(x_1) + 58(x_1)^2 + 8$
  $u_{24} = 30(x_1) + 18(x_1)^2 + 11$

- Y_3 = P * 6 is also included by u_3 that value is(3063,3143), a point on curve
  $u_{31} = 18(x_1) + 28(x_1)^2 + 50$
  $u_{32} = 17(x_1) + 30(x_1)^2 + 34$
  $u_{33} = 36(x_1) + (x_1)^2 + 14$
  $u_{34} = 55(x_1) + 2(x_1)^2 + 57$

- Y_4 = P * 4 is also included by u_4 that value is(3863,2497), a point on curve u_4 also includes Y_4 = 4 * P = (3863,2497)
  $u_{41} = 13(x_1) + 7(x_1)^2 + 32$
  $u_{42} = 26(x_1) + 41(x_1)^2 + 34$
  $u_{43} = 18(x_1) + 50(x_1)^2 + 3$
  $u_{44} = 51(x_1) + 34(x_1)^2 + 6$

- Each users adds the values receive form other users with his own value and computes their secret polynomial in x_1.

- $S(U1)((x_1)) = 13(x_1) + 63(x_1)^2 + 41$

- $S(U2)((x_1)) = 59(x_1) + 34(x_1)^2 + 47$

- $S(U3)((x_1)) = 49(x_1) + 48(x_1)^2 + 21$

- $S(U4)((x_1)) = 5(x_1) + 38(x_1)^2 + 30$

- " The Networks public key, PK = s * P "
  = " 24 * E(3198,578) = E(2651, 2267) "

- " PK should also equal to $Y_1 + Y_2 + Y_3 + Y_4$ "
  " =E(152,1437)+E(409,2266)+E(3063,3143)+E(3863,2497)
  = E(2651, 2267) "

- All the users compute their share as $S_(Ui)(0)$.

  The shares of the users are
  $s_1 = 30,\ s_2 = 64,\ s_3 = 42,\ s_4 = 21$

- If u_1 and u_2 wants to communicate securely then they compute a common secret key using their shares.

- u_1 sends $s_1 * P = 30 * E(38, 50) = E(35, 31)$ to u_2.

- u_2 sends $s_2 * P = 64 * E(38, 50) = E(50, 70)$ to u_1.

- Both u_1 and u_2 computes the same secret point. At u_1, $30 * E(50, 70) = E(6, 47)$ at u_2,$64 * E(35, 31) = E(6, 47)$.

- Now both u_1 and u_2 have a secret key 53, which they use for encryption and decryption.

- If u_1 wants to send a message ”hello” to u_2 using TEA, then it sends the cipher text ’0xe4a6ae978bd5335’.

- Then u_2 decrypts the cipher text to message ”hello” using the key 53.

4.2.6 Security Analysis

The TEA algorithm is prone to key equivalence attack as discussed below:
Key equivalence Attack:This attack is done by using known plain/cipher text pairs of any unknown key (K_1) and trying to find its equivalent key (K_1*). The motivation behind this attack is, the TEA algorithm key K_1 is split into four different sub keys each of size 32 bits (key=K_1[3], K_1[2], K_1[1], K_1[0]). The keys 4K_1[1], K_1[0] are user for the first half and the second half uses K_1[3], K_1[2]. Finding the values of K_1[1] is easy if we can find the values of K_1[0] (same with other half) because “R[i+1] =L[i]+(((R[i] 4)+K[0]) xor((R[i] 5)+K[1])xor(R[i]+Delta)))”. As all values are known we need to guess the value of K[0] from which we can generate k[1] value. To find value of k[1], we have to brute force the value upto 2^{32} possible values of k[0] and then check K[1] for each value and compare the generated k[1] values of two first and second plain/cipher text. If they don’t match then increase the value of K[0] and check again till they match. If they

match check for few more texts for conformation then the value found is the equivalent key of the real key. But we can avoid this attack in our networks by adopting proactive secret sharing techniques, where the users periodically update their shares thus not giving enough time to the attacker to find the equivalent keys.

4.2.7 Conclusion

The use of lightweight cryptography algorithm in decentralized networks is proposed in this chapter. As the users of this network have less computation power, they can not use traditional cryptography algorithms because of heavy computations. In our network setup, first all the users have a share of the network's private key and they use this share to generate symmetric key using "Diffie Hellman key exchange". Then they use the secret key with the TEA algorithm to encrypt/decrypt messages. We also discussed that the general key equivalence attack of TEA can be avoided by adopting proactive secret sharing.

Chapter 5

Subgroup Operations In Identity Based Encryption(IBE) using Weil Pairing In Decentralized Networks

The major drawback of the conventional public key cryptography systems is that the receiver's public key must be known to the sender in advance for the purpose of key setup and key retrieval. This problem can be solved in Identity Based Encryption (IBE) by taking some identity value (such as phone number or an e-mail etc.) as the public key of the receiver. This identity value can be used by any one who is willing to send him a message. The receiver requests the private key for the decryption from "Trusted Third Party called PKG(Private Key Generator)". The Shamir secret sharing technique can be used to decentralize the job of the PKG. The Weil Pairing on elliptic curve is suitable to implement IBE, as it is based on bilinear maps between groups. We propose a scheme that allows threshold decryption involving a subgroup of participants of the network.

5.1 Introduction

Identity Based Encryption(IBE) will allow the sender to use the reciever's identity in order to encrypt the message instead of using his public key. Using the identity

of the user in place of public key has wide range of applications. The identity based encryption system uses an arbitrary string as an identity. The identity based encryption system is first developed in 1984 by shamir[39] to easily manage the certificates in an e-mail system. If a user A wants to send a message to another user B to his e-mail B123@company.com, A encrypts the message simply by using B123@company.com. This process completely eliminates the use of public key certificates. When the user B gets the message then he contacts a third party organization called PKG - Private Key Generator to obtain private key by authenticating himself. Finally, B can read the mail which was sent by A. Weil pairing is a mapping of two computational Diffie-Hellman groups where one group is hard. Initially Weil pairing was used to attack elliptic curve systems[29] [18]. Later, Joux [23] designed a protocol using one round of "Diffie-hellman key exchange" among three parties and proved that Weil pairing are best suitable for this purpose. Sakai et al.[33] also used Weil pairing for the exchange of keys. Operations performed among the subgroup of users belonging to a network and how they deal user aggregation in the network is discussed as Subgroup operations. Our proposed scheme demonstrates a protocol for subgroup operations and also decentralizes the job of PKG. The advantage of PKG being decentralized is that the communication becomes secure, more reliable when compared to existing systems. It also allows the new users to have the same abilities as that of the initial users and each user has their share for the remaining life of the network.

5.2 Preliminaries

5.2.1 Shamir Secret Sharing

The initial proposals on secret sharing techniques are first introduced by Shamir[38] and Blakley[8] in the 1970s. The secret sharing mechanism shares the secret s among a group of participants $\{u_1, u_2, \ldots, u_n\}$ of n parties by using a special figure called dealer. The dealer sends privately the share of a secret to each party. Reconstruction process is adopted by the authorized subsets to extract the network secret S_k by pooling their shares. The group of such authorized subsets are called as access structure. The most popular (t,n) access structure is "Shamir secret sharing scheme" [38] that uses the Lagrange's interpolation polynomial, here n denotes the number of users in the network and t is the threshold value. For example let us consider n participants, s is

the secret, t is the threshold and the finite field is denoted by F_p. Shamir secret sharing technique uses two phases namely: Share Distribution and Secret reconstruction[6].

Share Distribution

- Choose $\{a_1, a_2, \ldots . a_{t-1}\}$ randomly from the given finite field F_p.
- Now, construct a polynomial of order t-1 and the polynomial is given by:

  f(x)=$a_{t-1}x^{t-1}$ + $\ldots\ldots a_1x^1$ + a_0. where, a_0 is secret.
- Shares are delivered as (i,y_i). where, "y_i=f(i) mod p and $1 \leq i \leq n$".

Reconstruction

- Lagrange's interpolation is used to obtain the coefficient of the polynomial function f(x) as follows:

  "f(x)=$\sum_{i=0}^{t} y_i * l(x)$. Where, $l(x) = \prod_{j=0, j \neq i}^{t} \frac{x-j}{i-j}$".
- Now,the secret s is equal to a_0.

5.2.2 Elliptic Curve Cryptography

Neil Koblitz and Victor Miller were the first to propose the elliptic curve cryptosystems [26]. In ECC - Elliptic Curve Cryptography, the elliptic curves algebraic construction over a finite field plays a vital role. The ECC systems require smaller keys when compared to non-ECC systems which are based on Galois field and provides identical security. In cryptography, elliptic curve finite field is defined over a group that contains all the points on the curve satisfying the equation "$y^2 = x^3 + ax + b$ where $4a^3 + 27b^2 \neq 0$ along with a distinguished point at infinity denoted by O". The ECC security is based on the difficulty of the discrete logarithm problem defined over ellipitc curves. The Elliptic Curve Cryptosystems are hard under the discrete logarithmic problem which play a vital role in its security. Compated to a 1024-bit key RSA algorithm, A 160-bit key ECC is more secured[31].

5.2.3 Weil Pairing

One of the popular and most widely used pairings is the Weil pairing, which is used to construct an admissible bilinear pairings that can be used as the basis for cryptographic systems[45]. Let us consider prime number p so that "$p = 12q - 1$" for a random prime q and let $y^2 = x^3 + 1$ be a super singular curve E that is defined over finite field F_p. The $p + 1$ order cyclic group is formed by a group contains all rational points is given by: "E(F_p)={$(x, y) \in F_pXF_p : (x, y) \in E$}". Let P be a point on elliptic curve and generates cyclic additive group tt of the prime order p. With the same prime order let tt_1 be another cyclic multiplicative group. Let $a, b \in F_p^*$. In both the groups Discrete Logarithm Problem (DLP) is assumed to be hard. Using the groups tt and tt_1, a bilienear pairing $e : tt \times tt \rightarrow tt_1$ is formed with the following properties:

- " Bilinear: For all $R, S \in tt_1$, $e(aR, bS) = e(R, S)^{ab}$; "
- " Non-degenerate: There exists R and $S \in tt_1$ such that $e(R, S) \neq 1$; "
- " Computable: There is an efficient algorithm to compute $e(R, S)$ for all $R, S \in tt_1$ ".

The following assumptions assumptions are taken into the consideration while working with the bilinear pairings:

- "In both the groups tt_1 and tt_2, the Discrete logarithm problem(DLP) must be hard. In tt_1 DDHP which stands for The Decisional Diffie-Hellman problem should be easy. In tt_2, the DDHP and also CDHP which statnds for computational Diffie-Hellman problem should be hard".
- The bilinear pairing inversion computation should be hard, that is the BPIP (the bilinear pairing inversion problem) states that:
  - " BPIP : Given $S \in tt_1$ and $e(S, T) \in tt_2$, find $T \in tt_1$."

The Weil pairing is preferred over other pairing algorithms because their outcome is not a unique value, which is often required in applications.

5.3 Related Work

5.3.1 Identity Based Encryption (IBE)

The identity based encryption schemes were first introduced in [39] by Shamir, which is not practical in its approach. Later, Franklin and Boneh [10] devised a technique to implement IBE which was secure and practical. Their scheme [46] efficiently used the concept of bilinear mapping which plays a vital role in our work. The IBE scheme is divided in to four algorithms. They are: Setup - where the users agree on public information, Extract - to generate private keys, Encrypt - to encrypt the messages and Decrypt - to decrypt the messages.

1. Setup: In this phase, the system parameters are made public whereas the master-key is available with only PKG - Private Key Generator. This phase initially takes security parameter as an input and gives the master key and system parameters as output.

2. Extract: In this phase a private key can be obtained from a given public key. This algorithm uses the input parameters, arbitrary ID $\in \{0,1\}^*$ and master key as input and return d as output. ID represents a random string which can be used like public key and the private key d is corresponding to the public key which will be used later for decryption.

3. Encrypt: This phase takes the input parameters, message, ID as input and gives the ciphertext as output.

4. Decrypt: This algorithm takes the input parameters, cipher-text and d (private key) as input and gives the corresponding message as output.

One of the main concerns of IBE is to decentralize the job of an authority or a trusted third party among the users. As a result there were many schemes proposed which adopted the secret sharing techniques. Haas and Zhou [47] were the first to introduce such a scheme using the concept of threshold cryptography which is not that practical in its approach. Later Kong et al[28] proposed another scheme but it was insecure. Other works[25] [34] [22] distribute only a part of master key in identity-based environments. All of the above works use shamir secret sharing scheme and

whenever a new user wants to be part of the network it imposes certain limitations like having a lot of interaction with existing users or not having the same ability as compared to other users.

Blundo [9] proposed new scheme in which new users can join the network dynamically without the need of any authority by using bivariate polynomials. Some other works Anzai et. al.[5] and Daza et. al. [15] used bivariate polynomials to decentralize the role of trusted authority.

5.3.2 Decentralization

In identity based encryption the master key is available only to the PKG and it should be protected. To achieve this we will be distributing the master key among several nodes by using the concept of threshold cryptography. The users exchange a bivariate polynomial to decentralize the work of PKG.

When working in subgroups, the suggestion is to use small subgroup of a curve in order to increase the IBE system performance. Here we use Weil pairing to decentralize the PKG. In this procedure, each user's public key is transformed to a point on the group by hashing the public ID of the user to a elliptic cureve point. The main advantage of using Weil pairing is computation can be done on small order points, which eventually leads to faster computation [46].

5.4 Proposed System

In our system the role of PKG is fully decentralized as discussed in Section 5.3.2. After the initial exchange of polynomials every user holds a share of the network secret. He can communicate with other users or can perform subgroup operations using the given protocol.

5.4.1 Setup

Let U denote the set of K users of the network. This initial K users are known as founding users of the network. All those users will run the protocol designed in the initialization phase (specified in subsection 5.4.2). The main goal is to decentralize the role of the PKG by using Shamir's secret sharing scheme, Weil pairing and identity

based encryption. Groups tt_1, tt_2 are taken for pairing with their respective hash functions. Threshold values t and t^1 are used for performing subgroup operations.

5.4.2 Initialization

Our scheme will have the following parameters which are made public. a group G, which is cyclic additive group of prime order q and is generated by the point on elliptic curve P. Another group tt_1 which is cyclic multiplicative of same order is also required. In addition to the above two groups it requires a bilinear pairing e, and two cryptographically secure hash functions "$H_1 : \{0,1^*\} \rightarrow Z_q$", another hash function " $H_2 : \{0,1^*\} \rightarrow tt_1$", and a public key PK = P_{pub} as discussed in section 2.2.3.2. The hash function H_1(Str,n,hashfcn) takes the input parameters - string Str, integer n, secure hash function hashfcn and gives a value with in the range of " 0 to n-1 ". $n \leq 2^{hashlen}$ where hashlen is " the number of octets comprising the output of the hash function hashfcn ". The Merkle's method [30] is used to create the H_1 hash function and it is also provably secure hash function because of underlying hashfcn hash function. The hash function H_2(E,p,q,id,hashfcn). The input parameters to the function H_2 are : E - elliptic curve ,p and q are two primes, id - a string and hashfcn - hash function. The return value of the function is Q_{id} = (x, y) is a point on elliptic curve and has prime order q. Two threshold values t, t^1 are chosen, where the threshold value t will be used to regulate test the security of the designed network i.e it will test that maximum t-1 nodes are deceptive. Another threshold value t^1 is used for looking after the security of the threshold operations computed in the users subgroup.The required condition for security is $t^1 \leq$ t $\leq$ N.

The hash function H and a bilinear pairing e are needed to generate the individual keys based on identity or as well as to compute the threshold operations on subgroup of users. Initialization phase of our proposed algorithm is described below:

1. Each user in U chooses a random bivariate polynomial f_i(x,z)$\in F_p$[x,z] with $t-1$ degree in the variable x and z. Here, U denotes the initial set of K users in the decentralized network. Each polynomial f(x,z)=$\sum_{u_i \in N} F_i$(x,z) (Here, u_i is the i^{th} user in given initial set of K users) share the same properties. The constant term of the given polynomial is $f_{i,0} = F_i(0, 0)$.

2. Each user $u_i \in N$ secretly sends the bivariate polynomial to the other users $u_j \in N$ (founding users) in the form of $f_{ij}(x) = f_i(x, h(u_j))$. Later, user u_i computes $Y_i = f_{i,0}P$ and uses this value in every message.

3. After each user in N performs the above step, each user u_j will compute their final secret value and is given by:

   $s_j(x) = \sum_{u_i \in u} f_j(x) = \sum_{u_i \in N} f_i(x, h(u_j)) = f(x, z)$.

   Each user computes their public key and make it public based on the information received from the other users $u_j \in N$. The public key(PK) will be as follows:

   PK=sP= $\sum_{u_i \in N} f_{i,0}P = \sum_{u_i \in N} Y_i$

   Note: Implicitly secret key(s) is F(0,0). A share $[s_j] = s_j(0)$=F(0,0) which is equal to $F(0, h(u_j))$ can be computed by each user in u_j from its share $s_j(x)$.This set up runs securely only when t $\leq$ N.

5.4.3 Network Management

After the initialization procedure is completed, if user u_k, a new user wants to be part of the network then he should run the below steps:

1. The new user u_k will select a group N_m which consists minimum of t users in the network and request them to include him in their Network.

2. If any of the user in N_m (suppose u_j) agrees to include this new user(u_k)in their network then he sends the following value : " $s_j(h(u_k)) = f(h(u_k), h(u_j))$ $= F(h(u_j), h(u_k)) = s_k(h(u_j))$ "

3. When the new user u_k gets this information from t users then he uses lagrange interpolation to extract secret polynomial as follows:

   "$\sum_{u_k \in N_m} \prod_{u_i \in N_m, i \neq j} \frac{x - h(u_i)}{h(u_j) - h(u_i)} s_j(h(u_k)) = \sum_{u_k \in N_m} \prod_{u_i \in N_m, i \neq j} \frac{x - h(u_i)}{h(u_j) - h(u_i)}$ $f(h(u_j), h(u_k))$=F(x,$h(u_k)$)=$s_k(x)$".

4. Finally the share $[s_k] = s_k(0)$ is computed by u_k.

5.4.4 Secure Communication Using IBE

In IBE, the user identity is used to compute the public key N_m that is $pk_m = H(N_m) \in G$ and $H : \{0,1\}^* \rightarrow tt_1$ which is chosen as hash function during initialization phase. Since it is a decentralized network, the user u_k needs to contact other users to compute the secret key sk_m=sH(N_m), s is master secret key. The designed protocol is given below:

1. The user u_k approaches a group of users (N_m) having minimum of t users to request for their share.
2. If any of the user (u_j) in the group of users (N_m) accepts the identification of the user u_k then he sends the following value: $\sigma_j m = s_j(0)H(u_k) = f(0, h(u_j))H(u_k) \in tt.$
3. The user u_k should receive t such values to compute the secret key sk_m where $sk_m = f(0, 0)H(u_k) = sH(u_k) \in tt.$

Then the Encryption and Decryption as discussed in [14] is given below.

Encrypt:

This method takes public parameters,id, message and returns cipher text

def $Encrypt(p, q, P, id, m, Q_{id})$:

E2 = EllipticCurve(F_p[0,1]) $P2 = E2(Q_{id}.xy())$ $(qx, qy) = P_{P\ ub}.xy()$

$Q = E2(qx * z, qy)$

$g_id = P2.weil_pairing(phiQ, p + 1)$

$l = h(g_id_r, q)$

Decrypt:

This method takes public parameters,secret key and cipher text and decrypt the message .

def $Decrypt(p, q, P, C, S_{id})$:

$E2 = EllipticCurve(F2, [0, 1])$

$P3 = E2(S_{id}.xy())$

$(qx, qy) = C_1.xy()$

$phiQ1 = E2(qx * z1, qy)$

$f_id = P3.weil_pairing(phiQ1, p + 1)$

key generation

This method takes public parameters, master secret key and generates the key to respective id.

def KeyGen(E,p,q,hashfcn,msk,id):

Q_{id}=H(*E, p, q, id, hashfunction*)

$sk_{id} = msk * Q_{id}$

$sec = (Q_{id}, sk_{id})$

return sec

The two hash functions are given below as defined in RFC 5091[12].

Algorithm 2.3.1 H_1: Hash function to convert the given string into interger

Input of the function $H)_1$:

o A $|s|$ octets length string s

o n - positive integer

o hashfcn - A secure hash function

Output: of the funciton H_1

o return positive integer v, $0 \leq v \leq n - 1$

Method:

1. Let the output hashfcn consists of the hashlen number of octets
2. The initial value of a variable $v_0 = 0$
3. The intial value of $h_0 = 0x00\ldots00$
4. " For i = 1 to 2, do:

   (a) Let $t_i = h_(i - 1)||s$, which is the ($|s|$ + *hashlen*) - octet string concatenation of the strings $h_(i - 1)$ and s

   (b) Let h_i = hashfcn(t_i), which is a hashlen-octet string resulting from the hash algorithm hashfcn on the input t_i

   (c) Let a_i = Value(h_i) be the integer in the range 0 to $256^{hashlen}$ - 1 denotedby the raw octet string h_i interpreted in the unsigned big-endian convention

   (d) Let $v_i = 256^{hashlen} * v_(i - 1) + a_i$"
5. Let $v = v_l(mod n)$

Algorithm 2.3.2 H_2: Translates the given string into elliptic curve point

Input of the hash function H_2:

o p - A choosen prime number for group tt

o q - A choosen prime number for group tt_1

o id - given string

o hashfcn - secure hash function

Output of the hash fucntion H_2:

o An elliptic curve point Q_{id}

Method:

1. Take y = H_1(id, p, hashfcn), H_1 is the function discussed in Algorithm 2.3.1
2. Compute x = $(y^2 - 1)^{\frac{(2*p-1)}{3}} \bmod p$, $p \in F_p$
3. a non- zero point $Q' = (x, y) \in (F_p)$
4. A q order poitn Q = $[\frac{(p+1)}{q}]Q' \in E(F_p)$

5.4.5 Subgroup operations

As mentioned in the initialization phase, each user adopts Shamir secret sharing scheme and holds the shares of secret key of the entire system corresponding to the threshold t. These shares can be used by the users in order to perform certain operations with minimum of t nodes being involved in the network. In our system, the nodes encrypt the messages among the subgroup(sub) of users as discussed in section 5.4.4. The decryption is possible only when t^1 users in the subgroup cooperate. Now, if a member of the subgroup wants to decrypt the message then the following steps are to be followed to get the share from its secret key:

1. A user u_k approaches a group of nodes (N_m) having minimum of t^1 users.
2. Any user (u_j) in N_m accepting the identity of the new user u_k need to send the following value to u_k

   $\tau_k = s_j(h(u_k))H(ID_{sub}) = f(h(u_j), h(u_k))H(ID_{sub}) \in tt_1$, where $H(ID_{sub})$ is the unique identity of the subgroup.

3. The share of the user u_k is computed by using Lagrange's interpolation after the user u_k has received t^1 such distinct values (as in above step). The share of the user is given by:

   $[SK_{sub}]_k = f(0, h(N_m))H(ID_{sub}) \in tt.$

5.4.6 Example

The two Hash functions H_1, H_2 are defined as follows based on algorithms discussed in Section 2.3

def H_1(uid,p)
t = str(uid).hexdigest(); thash = int(hashlib.sha224(t,16)
var = modulo(thash,p)
return var

def H_2(E,p,q,uid,h)
y_1 = H_1(uid,p)
x_1 = pow(($y_2^2 - 1$),((2*p - 1)/3),p)
$Q_1 = E(x, y)$
a_1 = int($\frac{p+1}{q}$)
Q = $a_1 * Q_1$
return Q

- **Setup**

- Let $U_M = \{U_1, U_2, U_3, U_4\}$ be the initial set of users
  Number of users = 4

- The Public Parmeters available to all the users are:
  " An additive group G of prime order q = 4019.
  - The curve used is $E(F_{4019}) : y^2 = x^3 + 1$
  - The Generator is P = E(3198,578)
  - Let t = 2 (degree of polynomials) and q = 67 (q is the order of subgroup) "

- Here we use Weil Pairing as bilinear pairing

5. SUBGROUP OPERATIONS IN IDENTITY BASED ENCRYPTION(IBE) USING WEIL PAIRING IN DECENTRALIZED NETWORKS

- The two explicit hash functions which are collision resistant are - H_2(Hash the given string to Point) : $\{0,1\}^* \rightarrow tt_1$ and another hash functions H_1(Hash the given string to the Range) : $\{0,1\}^* \rightarrow tt$.

- All the users randomly selects a bivariate polynomial that has symmetry in $x_1$andx_2 in GF(67)
  " $U_1 = 3(x_1)^2(x_2) + 3(x_2)^2(x_1) + 8(x_1)(x_2) + 5(x_2) + 5(x_1) + 5$
  $U_2 = 5(x_1)^2(x_2) + 5(x_2)^2(x_1) + 3(x_1)(x_2) + 8(x_2) + 8(x_1) + 9$
  $U_3 = 8(x_1)^2(x_2) + 8(x_2)^2(x_1) + 5(x_1)(x_2) + 3(x_2) + 3(x_1) + 6$
  $U_4 = 2(x_1)^2(x_2) + 2(x_2)^2(x_1) + 4(x_1)(x_2) + 8(x_2) + 8(x_1) + 4$

- Implicitly each user has a polynomial defined in $x_1$and x_2 is given by
  $F((x_1), (x_2)) = U_1 + U_2 + U_3 + U_4$"
  = "$18(x_1)^2(x_2) + 18(x_1)(x_2)^2 + 20(x_1)(x_2) + 24(x_1) + 24(x_2) + 24$"

- The network secret S_k = F(0,0) = 24.

- Now every user privately communicates other users the univariate polynomial in x_1 as $F_{ij} = F_i((x_1), H_1(U_j))$, $1 \leq j \leq 4$.

- The hash values of the users are
  $h_{u1} = (H_1)(\text{"}User1\text{"}, q) = 37$
  $h_{u2} = (H_1)(\text{"}User2\text{"}, q) = 54$
  $h_{u3} = (H_1)(\text{"}User3\text{"}, q) = 25$
  $h_{u4} = (H_1)(\text{"}User4\text{"}, q) = 17$

- Now all the users substitues the hash values of other users identity in their polynomial and sends the following message:

- Y_1 = P * 5 is also included by u_1 that value is(152,1437), a point on curve
  $u_{11} = 53(x_1) + 44(x_1)^2 + 56$
  $u_{12} = 6(x_1) + 28(x_1)^2 + 7$
  $u_{13} = 3(x_1) + 8(x_1)^2 + 63$
  $u_{14} = 3(x_1) + 51(x_1)^2 + 23$

- Y_2 = P * 9 is also included by u_2 that value is(409,2266), a point on curve
  $u_{21} = 63(x_1) + 51(x_1)^2 + 37$
  $u_{22} = 10(x_1) + 2(x_1)^2 + 39$
  $u_{23} = 59(x_1) + 58(x_1)^2 + 8$
  $u_{24} = 30(x_1) + 18(x_1)^2 + 11$

- Y_3 = P * 6 is also included by u_3 that value is(3063,3143), a point on curve
  $u_{31} = 18(x_1) + 28(x_1)^2 + 50$
  $u_{32} = 17(x_1) + 30(x_1)^2 + 34$
  $u_{33} = 36(x_1) + (x_1)^2 + 14$
  $u_{34} = 55(x_1) + 2(x_1)^2 + 57$

- Y_4 = P * 4 is also included by u_4 that value is(3863,2497), a point on curve u_4 also includes Y_4 = 4 * P = (3863,2497)
  $u_{41} = 13(x_1) + 7(x_1)^2 + 32$
  $u_{42} = 26(x_1) + 41(x_1)^2 + 34$
  $u_{43} = 18(x_1) + 50(x_1)^2 + 3$
  $u_{44} = 51(x_1) + 34(x_1)^2 + 6$

- Each users adds the values receive form other users with his own value and computes their secret polynomial in x_1.

- $S(U1)((x_1)) = 13(x_1) + 63(x_1)^2 + 41$

- $S(U2)((x_1)) = 59(x_1) + 34(x_1)^2 + 47$

- $S(U3)((x_1)) = 49(x_1) + 48(x_1)^2 + 21$

- $S(U4)((x_1)) = 5(x_1) + 38(x_1)^2 + 30$

- " The Networks public key, PK = s * P "
  = " 24 * E(3198,578) = E(2651, 2267) "

- " PK should also equal to $Y_1 + Y_2 + Y_3 + Y_4$ "
  " =E(152,1437)+E(409,2266)+E(3063,3143)+E(3863,2497)
  = E(2651, 2267) "

5. SUBGROUP OPERATIONS IN IDENTITY BASED ENCRYPTION(IBE) USING WEIL PAIRING IN DECENTRALIZED NETWORKS

- All the users compute their share as $S_{(Ui)}(0)$.
  From the above polynomials, the users shares are
  $S_3 = 21,\ S_2 = 47,\ S_1 = 41,\ S_4 = 30$

- once all the users have their shares, they can be validated with the polynomial of the network
  $f(x_2) = F(0,(x_2))$
  " $= 24 * (x_2) + 24$ "

- Let u_5 be new user who wants to be part of the network, then he identifies himself to 3 other users and ask them to take him in. $\{U_2,\ U_3, U_4\}$
  $h_{U5} = (H_1)(\text{"}User5\text{"}, k) = 27$

- U_5 gets the following messages from users 2,3 and 4
  $S_{U25} = S_{(U2)}(27)\ modulo\ 67 = 28$
  $S_{U35} = S_{(U3)}(27)\ modulo\ 67 = 22$
  $S_{U45} = S_{(U4)}(27)\ modulo\ 67 = 62$

- " u_5 computes its secret univariate polynomial by using Lagrange interpolation "
  " $S_5((x_1)) = 17 * (x_1)^2 + 18 * (x_1) + 2$ "

OBTENTION OF INDIVIDUAL KEYS BY IndentityBasedEncryption(IBE) SCENARIO example

- Take a public parameter $P_{P\ ub} = msk * P$

- now user u2 want to send the message m = 1712 to user n1.

- u2 calls encrypt method .

- $C = Encrypt(p, q, P, \text{"}Node1\text{"}, m, Q_{id1})$;

- cipher text is (1807 ,1481) 1718

- After receiving encrypted message user n1 calls decrypt method.

- $msg = Decrypt(p, q, P, C, Sk_{id1})$.

- After decrypting message is m= 1712.

- **Threshold Decryption on Sugroup example**

- Take the shares of users N={N1,N2,N3,N4} as a subgroup.
  Each user is having its own secret polynomial.

- $s_1(x_1) = 31 + 13x_1 + 63x_1^2$

- $s_2(x_1) = 37 + 59x_1 + 34x_1^2$

- $s_3(x_1) = 11 + 49x_1 + 48x_1^2$

- $s_4(x_1) = 20 + 5x_1 + 38x_1^2$

- create an id for the sub group.
  hsg1234 = h('SG1234',q), where SG1234 is the subgroup unique identity

- To find share of jth node remaining users contribute their shares and lagranges interpolation is applied.
  Share of user1 (3125,1868)
  Share of user2(2292,3913)
  Share of user3(2350,780)
  Share of user4(163,2657)
  To verify the shares of users caluclate the hash of users they are rd1,rd2,rd3,rd4.
  sg1 = int(rd1)* H(E,p,q,'SG1234',hashfcn)
  sg2 = int(rd2) * H(E,p,q,'SG1234',hashfcn)
  sg3 = int(rd3) * H(E,p,q,'SG1234',hashfcn)
  sg4 = int(rd4) * H(E,p,q,'SG1234',hashfcn)
  The shares of the users must be (Calculated from F) Share of user1(3125,1868)
  Share of user2(2292,3913)
  Share of user3(2350,780)
  Share of user4(163,2657)

Secret of Subgroup

$14 * H(E, p, q, Stt1234, hashfcn)$

Secret of Subgroup is (3857,1351)

- secret of a subgroup is lagranges interpolation is applied on users then we get k1,k2,k3 from nodes n1,n2,n3.

  a1 = int(k1) * sg1

  a2 = int(k2) * sg2

  a3 = int(k3) * sg3

  Secret of subgroup a1+a2+a3=(3857,1351)

- Here ENCRYPTION and DECRPYTION methods are same but in decryption method we have one more parameter i.e k11 for user 1 it is formed from lagranges interpolation with t^1 users.

- Any user want to send the message to the sub group. let m=50
  This message can be encrypted as:

  $C = Encrypt(p, q, P, Stt123, m1, Q_{id})$

  Threshold is 2 so any two users compute let n1 and n2 compute l1=Decrypt(p,q,P,C,sg1,k11)

  l2=Decrypt(p,q,P,C,sg2,k22)

  r1=l1*l2

  r=h(r1,q)

  now the encrypted message with r output is message=50.

5.5 Security Analysis

For a public key encryption scheme the acceptable notion for security is cipher-text security[7] [16] [35]. But the definition concerning the chosen cipher-text should be strengthened. This is because if an adversary outbreaks the public ID of IBE system and then the adversary may get hold of the users private keys. Thus the designed system should withstand such an attack and should be secure. Here our assumption is that the IBE system can withstand the chosen cipher-text attack.

Note: Attacker A should not have any advantage against the challenger.

Setup: Initialization phase is followed by the challenger by inputing the security parameters. The system parameters are obtained by adversary .

Phase1: The adversary issues either the extraction query or the decryption query.

- Extraction Query:The extract algorithm (defined in 5.3.1) is run by the challenger. As a result of this, the private key is generated corresponding to particular public key. This private key is sent to the adversary.

- Decryption Query:The extract algorithm (defined in 5.3.1) is run by the challenger. As a result of this, the corresponding public key and privae key are generated. The decrypt algorithm is run by it using the private key for decrypting the cipher-text. The resulting plain text is sent to the adversary.

Challenge: Two equal length plain texts and ID are generated by adversary after Phase 1 is over. The ID is the parameter on which the adversary desired to be challenged. ID did not appear anywhere in the extraction of phase 1 query. A bit b $\in$ {0,1} is randomly picked by challenger and it sets C to Encrypt(parameters,ID,M_b). Challenger sends this cipher text C to adversary as a challenge.

Phase2: In this phase more and more queries are posed by the adversary and it can be either of the following:

- Extraction Query: Here ID_i ≠ ID. Then the challenger replies as in phase1.

- Decryption Query: Challenger replies as in phase1 if (ID_i, C_i) ≠ (ID,C). Here C is the cipher-text.

Guess: $b^1 \in \{0, 1\}$ is displayed by the adversary and the game is won by adversary if $b^1 = b$.

Attack Let the number of players trying to reconstruct the secret s be $\leq t - 1$. Here t denotes the threshold value.

Analysis: The recovery of the secret in the proposed scheme completely revolves around the concept of Lagrange's Interpolation polynomial. To solve for the secret, we are definitely going to need t number of equations. Therefore, it is only t or more players who can have a complete knowledge of the secret. There is no chance for less than t number of players to crack the secret.

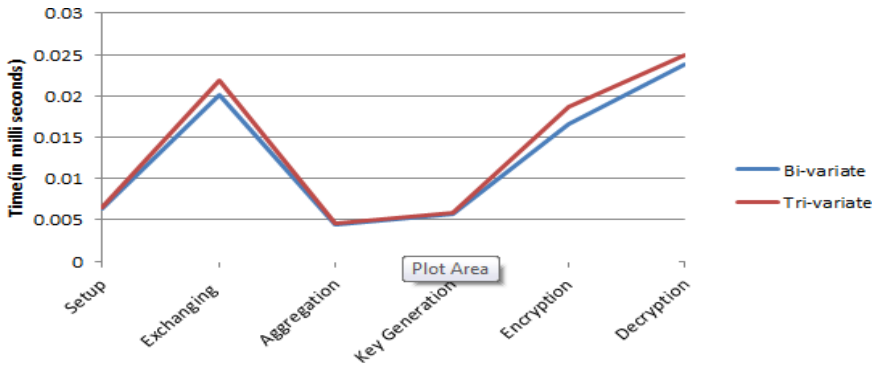
5.6 Comparison between Bi-variate and Tri-variate polynomials

Our proposed IBE subgroup protocol is implemented using Bi-variate polynomial and is compared with the existing Tri-variate polynomial implementation and found to be efficient.

Table 5.1: Identity Based Encryption Comparision of Bi-variate and Tri-variate

TIME	BI-VARIATE	TRI-VARIATE
Set Up	0.00631594657898	0.00651788711548
Exchanging	0.0201599597931	0.0219240188599
Aggregation	0.00441002845764	0.00461021175385
Key Generation	0.00565202331543	0.00592576600231
Encryption	0.016654968261	0.0186330871582
Decryption	0.023789520159	0.024987452

Figure 5.1: Identity Based Encryption Comparision of Bi-variate and Tri-variate



5.7 Conclusion

Now a days many applications demand the network without the presence of TTP - Trusted Third Party. This is achieved by distributing the role of TTP among the network users using secret sharing concept. In this chapter, we proposed an efficient way to decentralize the network and to establish a secure communication among the

users of the network using Identity based encryption. We also discussed the suitable protocol to perform sub group operations among the subset of users of a network. Our scheme is useful for the applications where secure communication is required without the presence of trusted third party.

Chapter 6

Conclusions

In this work, we addressed the problem of security in a decentralized network consisting of resource constraint devices and presented a comprehensive set of protocols that can be used for secure communication in these kinds of network. Secret sharing techniques are employed to distribute the shares of a secret key among the network members and the secret can be reconstructed only when a threshold number of users collaborate. We used BLS signature scheme to verify the certificate in the decentralized networks where each user holds a share of the network secret. The secret is distributed to the users among them selves after the few exchanges of the messages in the initial phase. Once the users have their shares, then they prepare a certificate and make them fully signed. They use BLS Signature to generate the certificate. Our scheme uses a bi-variate polynomial to reduce the communication overhead. Once all the network users have their share in the network secret, protecting their share for the entire network's life time is difficult. An attacker may collect enough number of information to gain the access of the share. To avoid this, the share of the users have to change frequently without affecting the actual secret of the network. We achieved this by using the concept of proactive secret sharing and we also used the concept of Elliptic curve cryptography to verify the user shares. To establish a secure communication among the users of the network we used the concept of lightweight cryptography. As the users of this network have less computation power, they can not use traditional cryptography algorithms because of heavy computations. In our network setup, first all the users have a share of the network's private key and they use this share to generate symmetric key using "Diffie Hellman key exchange". Then they use the secret

key with the TEA algorithm to encrypt/decrypt messages. We also discussed that the general key equivalence attack of TEA can be avoided by adopting proactive secret sharing. Now a days many applications demand the network without the presence of TTP - Trusted Third Party. This is achieved by distributing the role of TTP among the network users using secret sharing concept. We proposed an efficient way to decentralize the network and to establish a secure communication among the users of the network using Identity based encryption. We also discussed the suitable protocol to perform sub group operations among the subset of users of a network. Our scheme is useful for the applications where secure communication is required without the presence of trusted third party.

References

[1] ALEX EDWARD AFTUCK. **The Weil pairing on elliptic curves and its cryptographic applications**. 2011. (27, 30)

[2] RAM RATAN AhirwAL AND MANOJ AHKE. **Elliptic curve diffie-hellman key exchange algorithm for securing hypertext information on wide area network**. *International Journal of Computer Science and Information Technologies*, **4**(2):363–368, 2013. (20)

[3] YAIR AMIR, YONGDae KIM, CRISTINA NITA-ROTAru, JOHN L SCHULTZ, JONATHAN STANTON, AND GENE TSUDIK. **Secure group communication using robust contributory key agreement**. *IEEE Transactions on Parallel and Distributed Systems*, **15**(5):468–480, 2004. (68)

[4] SUSAN. HOHENBERGER ANNA. FErRARA, MATTHEW. GREEN. **Practical Short Signature Batch Veriftcation**. **5473**:1–24, 2009. (40)

[5] JUN ANZAI, NATSUME MATSUZAKI, AND TSUTOMU MATSUMOTO. **A quick group key distribution scheme with entity revocation**. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 333–347. Springer, 1999. (36, 54, 85)

[6] ABDUL BASIT, N CHAITanya KUMAR, V Ch VENKAIAH, SALMAN ABDUL Moiz, APPALA NAIDU TENTU, AND WILSON NAIK. **Multi-stage Multi-secret sharing scheme for hierarchical access structure**. In *Computing, communication and automation (ICCCA), 2017 International Conference on*, pages 557–563. IEEE, 2017. (53, 55, 82)

[7] Mihir BELLARE, ANAND DESAI, DAVID POINTCHEVal, AND PHILLIP ROGAWAy. **Relations among notions of security for public-key encryption schemes**. In *Annual International Cryptology Conference*, pages 26–45. Springer, 1998. (96)

[8] GEORGE ROBERT BLAKLEY ET AL. **Safeguarding cryptographic keys**. In *Proceedings of the national computer conference*, **48**, pages 313–317, 1979. (34, 37, 53, 66, 81)

[9] CARLO BLUNDO, ALFREDO DE SANTIS, AMIR HERZBERg, SHAy KUTTEN, Ugo VACCARo, AND MOTI YUNG. **Perfectly-secure key distribution for dynamic conferences**. In *Annual International Cryptology Conference*, pages 471–486. Springer, 1992. (36, 49, 54, 85)

[10] DAN BONEH AND MATT FRANKLIN. **Identity-based encryption from the Weil pairing**. In *Annual international cryptology conference*, pages 213–229. Springer, 2001. (23, 29, 84)

[11] DAN BONEH, BEN LYNN, AND HOVAV SHACHAM. **Short signatures from the Weil pairing**. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 514–532. Springer, 2001. (34, 36, 49)

[12] L. MARTIN BOYEN X. **Identity-Based cryptography standard (IBCS) 1: Super singular curve implementation of the BF and BBI cryptosystems**. 2007. (41, 56, 89)

[13] LEONARD S CHARLAP, DAVID PETER ROBBINS, AND RAYMOND COLEY. *An elementary introduction to elliptic curves*. Center for Communications Research, Institute for Defense Analysis, 1988. (19)

[14] VANESA DAZA, JAVIER HERRANZ, PAZ MORILLO, AND CARLA RAFOLS. **Cryptographic techniques for mobile ad-hoc networks**. *Computer Networks*, **51**(18):4938–4950, 2007. (88)

[15] VANESA DAZA, JAVIER HERRANZ, AND GERMÁN SÁEZ. **Constructing general dynamic group key distribution schemes with decentralized user join**. In *Australasian Conference on Information Security and Privacy*, pages 464–475. Springer, 2003. (36, 55, 85)

REFERENCES

[16] DANNY DOLEV, CYNTHIA DWORK, AND MONI NAor. **Nonmalleable cryptography**. *SIAM review*, **45**(4):727–784, 2003. (96)

[17] PAUL FELDMAN. **A practical scheme for non-interactive veriftable secret sharing**. In *Foundations of Computer Science, 1987., 28th Annual Symposium on*, pages 427–438. IEEE, 1987. (54)

[18] GERHARD FREY, MICHAEL MULLER, AND H-G RUCK. **The Tate pairing and the discrete logarithm applied to elliptic curve cryptosystems**. *IEEE Transactions on Information Theory*, **45**(5):1717–1719, 1999. (81)

[19] Goichiro HANAOKA AND JACOB CN SCHULDT. **On signatures with tight security in the multi-user setting**. In *Information Theory and Its Applications (ISITA), 2016 International Symposium on*, pages 91–95. IEEE, 2016. (36)

[20] Y-C Hu, ADRIAN PERRIG, AND DAVID B JohNSON. **Packet leashes: a de- fense against wormhole attacks in wireless networks**. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communica- tions. IEEE Societies*, **3**, pages 1976–1986. IEEE, 2003. (34)

[21] STEPHANIE ANG YEE HUNN, NORINA BINTI IDRIS, ET AL. **The development of tiny encryption algorithm (TEA) crypto-core for mobile systems**. In *Electronics Design, Systems and Applications (ICEDSA), 2012 IEEE International Conference on*, pages 45–49. IEEE, 2012. (70)

[22] STANISLAW JARECKI, NITESH SAXENA, AND JEONG HYUN Yi. **An attack on the proactive RSA signature scheme in the URSA ad hoc network access control protocol**. In *Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*, pages 1–9. ACM, 2004. (36, 84)

[23] ANTOINE Joux. **A one round protocol for tripartite Diffie–Hellman**. *Journal of cryptology*, **17**(4):263–276, 2004. (81)

[24] S KENT AND T POLK. **Public-key infrastructure (x. 509)(pkix) charter**, 2002. (33, 35)

[25] ARAM KHALILI, JONATHAN KATZ, AND WILLIAM A ARBAugh. **Toward secure key distribution in truly ad-hoc networks**. In *Applications and the Internet Workshops, 2003. Proceedings. 2003 Symposium on*, pages 342–346. IEEE, 2003. (84)

[26] NEAL KOBLITZ. **Elliptic curve cryptosystems**. *Mathematics of computation*, **48**(177):203–209, 1987. (67, 68, 82)

[27] N CHAITanya KUMAR, ABDUL BASIT, PriyADARSHI SINGH, V Ch VENKAIAH, AND YV RAO. **Node Authentication Using BLS Signature in Distributed PKI Based MANETS**. *arXiv preprint arXiv:1708.08972*, 2017. (75)

[28] HAIYUN Luo, JIEJUN KONG, PETROS ZERFOS, SONGwu Lu, AND LIXIA ZHANG. **URSA: ubiquitous and robust access control for mobile ad hoc networks**. *IEEE/ACM Transactions on Networking (ToN)*, **12**(6):1049–1063, 2004. (36, 84)

[29] ALFRED J MENEZES, TATSUAKI OKAMOTO, AND SCOTT A VANSTONE. **Reducing elliptic curve logarithms to logarithms in a ftnite fteld**. *iEEE Transactions on information Theory*, **39**(5):1639–1646, 1993. (81)

[30] R. MERKLE. **A fast software one-way hash function**. **3**:43–58, 1990. (40, 55, 73, 86)

[31] VICTOR S MILLER. **Use of elliptic curves in cryptography**. In *Conference on the theory and application of cryptographic techniques*, pages 417–426. Springer, 1985. (67, 82)

[32] MAITHILI NARASIMHA, GENE TSUDIK, AND JEONG HYUN Yi. **On the utility of distributed cryptography in P2P and MANETs: the case of member- ship control**. In *Network Protocols, 2003. Proceedings. 11th IEEE International Conference on*, pages 336–345. IEEE, 2003. (36)

[33] TATSUAKI OKAMOTO. **Cryptography based on bilinear maps**. In *Interna- tional Symposium on Applied Algebra, Algebraic Algorithms, and Error-Correcting Codes*, pages 35–50. Springer, 2006. (26, 81)

[34] JIANPING PAN, LIN CAI, XUEMIN SHERMAN SHEN, AND JON W MARK. **Identity-based secure collaboration in wireless ad hoc networks**. *Com- puter Networks*, **51**(3):853–865, 2007. (84)

[35] CHARLES RACKoff AND DANIEL R SIMON. **Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack**. In *Annual International Cryptology Conference*, pages 433–444. Springer, 1991. (96)

[36] YASSER SALEM AND MOHAMED ALI. *Implementation of Elliptic Curve Cryptography using biometric features to enhance security services*. PhD thesis, University of Malaya, 2009. (5)

[37] NITESH SAXENA, GENE TSUDIK, AND JEONG HYUN Yi. **Efficient node ad- mission for short-lived mobile ad hoc networks**. In *Network Protocols, 2005. ICNP 2005. 13th IEEE International Conference on*, pages 10–pp. IEEE, 2005. (36, 55)

[38] ADI SHAMIR. **How to share a secret**. *Communications of the ACM*, **22**(11):612–613, 1979. (34, 35, 36, 37, 53, 66, 67, 81)

[39] ADI SHAMIR. **Identity-based cryptosystems and signature schemes**. In *Workshop on the theory and application of cryptographic techniques*, pages 47–53. Springer, 1984. (81, 84)

[40] JOSEPH H SILVERMAN. **An introduction to the theory of elliptic curves**. *link: http://www. math. brown. edu/jhs/Presentations/WyomingEllipticCurve. pdf*, 2006. (6, 7)

[41] ANDREW V SUTHERLAND. **Identifying supersingular elliptic curves**. *LMS Journal of Computation and Mathematics*, **15**:317–325, 2012. (29)

[42] DAXING WANG AND JIKAI TENG. **Efficient Aggregate Signature Algorithm and Its Application in MANET**. *International Journal of Mathematical, Computational, Physical, Electrical and Computer Engineering*, **7**(11), 2013. (36)

[43] STALLINGS WILLIAM. *Cryptography and network security: principles and practices*. Pearson Education India, 2006. (35)

[44] SEUNG YI AND ROBIN KRAVETS. **MOCA: Mobile certiftcate authority for wireless ad hoc networks**. Technical report, 2004. (34)

[45] XUN YI. **An identity-based signature scheme from the Weil pairing**. *IEEE communications letters*, **7**(2):76–78, 2003. (83)

[46] QUAN YUAN AND SONGPING LI. **A New Efficient ID-Based Authenticated Key Agreement Protocol.** *IACR Cryptology ePrint Archive*, **2005**:309, 2005. (84, 85)

[47] LIDONG ZHOU AND ZYGMUNT J HAAS. **Securing ad hoc networks**. *IEEE network*, **13**(6):24–30, 1999. (34, 36, 84)

[48] MINGYUAN ZHOU, CHUNPING WANG, MINHUA CHEN, JOHN PAISLEY, DAVID DUNSON, AND LAWRENCE CARIN. **Nonparametric Bayesian matrix completion**. *Proc. IEEE SAM*, 2010. (53)

List of Publications

[49] N CHAITanya KUMAR, ABDUL BASIT, PriyADARSHI SINGH, V Ch VENKAIAH, Y. V. SUBBA RAO . **Node Authentication using BLS Signature in Distributed PKI based MANETs**. *International Journal of Network Security & Its Applications* , volume 9, pages 11-19, 2017.

[50]N CHAITanya KUMAR, ABDUL BASIT, PriyADARSHI SINGH, V. Ch. VENKAIAH. **Proactive Secret Sharing For Long Lived MANETs Using Elliptic Curve Cryptography**. *International Conference on Inventive Computing and Informatics (ICICI 2017)*, November 23-24, 2017.

[51]N CHAITanya KUMAR, ABDUL BASIT, PriyADARSHI SINGH, V. Ch. VENKAIAH. **Lightweight Cryptography for Distributed PKI Based MANETs**. *International Journal of Computer Networks & Communications*, volume 10, pages 63-69, 2018.

[52]N CHAITanya KUMAR, ABDUL BASIT, PriyADARSHI SINGH, V. Ch. VENKA- IAH. **Subgroup Operations In Identity Based Encryption (IBE) Using Weil Pairing In Decentralized Networks**. *International Journal of Network Security*, volume 21, 2019.

Appendices

Appendix A

Hash Value Computation

The hash function is defined as follows:

```
import hashlib
def hfun(id,k):
print hashlib.sha224(str(id)).hexdigest()
h = int(hashlib.sha224(str(id)).hexdigest(),16)
print 'h value of ', id , 'is ', h
val = mod(h,k)
return val
```

The hash values of the nodes are calculated as follows

k1 = 67

h_{n1} = hfun('User1',k1)

print 'Hash Value of the node 1 is ',h_{n1}

The hashed valued in hexa decimal format is

7d16c92ca29b2937d00ca2392eff7d3d67b9d7578b95a13ee547cfa4

converting the hexa decimal into integer gives

h value of User1 is

1317341000836137729849899285257766452258040528821827990166175795190 8

Hash Value of the User1 in the field k1 is 37 We used sha224 to calculate the hash value, other hashing techniques can also be used.