CAPITAL UNIVERSITY OF SCIENCE AND TECHNOLOGY, ISLAMABAD

Generalization of Certificateless Aggregated Signcryption Scheme Based on Bilinear Mapping for Internet of Things

by

Seher Urooj

A thesis submitted in partial fulfillment for the degree of Master of Philosophy

in the

Faculty of Computing
Department of Mathematics

2024

To my parents (specially my mother), teachers and friends for their support and love.

CERTIFICATE OF APPROVAL

Generalization of Certificateless Aggregated Signcryption Scheme Based on Bilinear Mapping for Internet of Things

by

Seher Urooj

(Registration No: MMT213037)

THESIS EXAMINING COMMITTEE

(a)	External Examiner	Dr. Shamsa Kanwal	FJWU, Rawalpindi
(b)	Internal Examiner	Dr. Abid Kamran	CUST, Islamabad
(c)	Supervisor	Dr. Rashid Ali	CUST, Islamabad

Dr. Rashid Ali
Thesis Supervisor
May, 2024

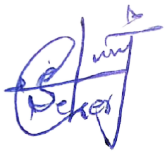
Dr. Muhammad Sagheer
Head
Dept. of Mathematics
May, 2024

Dr. M. Abdul Qadir
Dean
Faculty of Computing
May, 2024

Author's Declaration

I, **Seher urooj** hereby state that my MPhil thesis titled "**Generalization of Certificateless Aggregated Signcryption Scheme Based on Bilinear Mapping for Internet of Things** " is my own work and has not been submitted previously by me for taking any degree from Capital University of Science and Technology, Islamabad or anywhere else in the country/abroad.

At any time if my statement is found to be incorrect even after my graduation, the University has the right to withdraw my MPhil Degree.



(Seher Urooj)

Registration No: MMT213037

Plagiarism Undertaking

I solemnly declare that research work presented in this thesis titled "**Generalization of Certificateless Aggregated Signcryption Scheme Based on Bilinear Mapping for Internet of Things** " is solely my research work with no significant contribution from any other person. Small contribution/help wherever taken has been duly acknowledged and that complete thesis has been written by me.

I understand the zero tolerance policy of the HEC and Capital University of Science and Technology towards plagiarism. Therefore, I as an author of the above titled thesis declare that no portion of my thesis has been plagiarized and any material used as reference is properly referred/cited.

I undertake that if I am found guilty of any formal plagiarism in the above titled thesis even after award of MPhil Degree, the University reserves the right to withdraw/revoke my MPhil degree and that HEC and the University have the right to publish my name on the HEC/University website on which names of students are placed who submitted plagiarized work.

(Seher Urooj)

Registration No: MMT213037

Acknowledgement

At the end of my thesis, I would like to thank all those people who made this thesis possible and an unforgettable experience for me. Foremost, I would like to acknowledge and give my warmest thanks to my supervisor **Prof. Dr. Rashid Ali** who made this work possible. His guidance and advice carried me through all the stages of writing my thesis. I am very thankful for his patience, motivation, enthusiasm, continuous advice and encouragement throughout the course of this thesis. Beside this, I would also like to express my appreciation to the head of the Mathematics department **Prof. Dr. Muhammad Sagheer**. His commitment to providing a conducive and innovative learning environment has enriched my academic experience significantly. However, the most profound gratitude goes to my family, who have been my pillars of strength. This thesis is dedicated from the depths of my heart to my beloved mother and father. And specially thanks my frients (Shehar bano , Saliha, Saman)full supported me in degree of MPhil. In closing, I offer my sincere gratitude to all those mentioned above and to the Almighty God for the blessings and opportunities that have brought me to this moment. May His blessings continue to enrich the lives of all who have contributed to this journey, and may our collective endeavors continue to bear fruit in the pursuit of knowledge and understanding.

(Seher Urooj)

Registration No: MMT213037

Abstract

Generalized Signcryption is a fresh cryptographic primitive that not only can obtain encryption and signature in a single operation, but also provives encryption or signature alone when needed. This paper gives a formal definition of certificateless generalized signcryption and its security model is present. A certificateless generalized signcryption scheme is also proposed in this thesis. The increasing number of devices in the age of Internet of Thing (IoT) has arisen a number of problems related to security. Cryptographic processes, more precisely the signatures and the keys, increase and generate an overhead on the network resources with these huge connections. Therefore, in this thesis we present a signcryption framework to address the above problems. The solution highlights the use of aggregate signcryption and certificaless approach based on bilinear mappings. The use of signcryption with aggregation and certificateless authentication reduces the time consumption, overhead and complexity. The solution is also able to solve the key staling problems. Experimental results and comparative analysis based on key parameters, memory utilization and bandwidth utilization have been measured. It confirms that the presented work is efficient for IoT infrastructure.

Contents

List of Figures

List of Tables

Abbreviations

ASG	Aggregate Signcryption Generator
CLGSC	Certificateless Generalized Signcryption
CDH	Computational diffie-hellman
DBDH	Decisional bilinear diffie-hellman
GSC	Generalized Signcryption
GDH	Gap diffie-hellman
GCD	Greatest Common Divisor
KGC	Key Generation Center
PKC	Public Key Cryptography
SC	Signcryption Scheme

Symbols

$\mho$	Ciphertext
δ	Message
ϱ	Bilinear mapping
Z	Set of integer
h	Hash Function
c	generator of group
C	Cyclic group
t	Time stamp

Chapter 1

Introduction

Cryptography [1] is the practice of securing messages so that unauthorized parties cannot read them. It involves using mathematical techniques to protect information. With encryption, an original message (plaintext) is transformed into a coded message (ciphertext) using a special method (encryption algorithm) [2].

This coded message is then changed back to plaintext using a decryption algorithm, but only by the intended recipient or authorized person. Both the sender and receiver use a secret piece of information known as a key for encryption and decryption. This setup is called a cryptosystem. The security of the cryptosystem relies on the secrecy of the key.

Cryptographic methods are divided into two main types: symmetric key cryptography and asymmetric key cryptography. In symmetric key encryption, a single secret key is used by both the sender and the receiver. This key must be kept private and is known only to them. Examples of symmetric key encryption [3] are Data Encrption Standard (DES) [4] and Advanced Encryption Standard (AES) [5].

The main challenge in this approach is delivering the secret key securely between the sender and the receiver. This task becomes difficult when there are many users who need to communicate with each other. To solve this problem, Diffie and Hellman [6] came up with asymmetric key cryptography, also called Public Key Cryptography (PKC).

In this method, participants have two types of keys: a public key that can be shared openly and a private key that must be kept secret. RSA [7] and ElGamal [8] are examples of

asymmetric key cryptography. This approach makes it easier to connect with many users securely without needing to share a secret key with each one.

In the past, when people wanted to send important messages, they would physically sign the message and put it in an envelope to keep it safe from forgery and to keep its contents private. But with the advancement of technology, this process has changed.

Public key cryptography, a method discovered about twenty years ago, has changed how safe and authenticated communications happen. Now, even people who do not know each other can communicate securely over open and private networks like the internet. This modern process still follows a two-step technique.

First, the sender adds a digital signature to the message using a digital signature method. Then, they lock the message using a private key. They also use a random key to further lock the message before sending it. This is known as encryption. The special key for the random encryption is then locked again using the receiver's public key. This two-step method is called "signature-then-encryption".

However, there are some drawbacks to this approach. One is the amount of data involved, which can be a lot. Another is that it can take a lot of computer power to do all the necessary calculations. In simpler terms, a digital signature is a way to electronically sign a message to make sure it is real.

This method has evolved from physically signing paper letters to using technology. It lets people communicate safely even if they do not know each other.But sometimes this method can require a lot of data and be computationally demanding.

A digital signature is like an electronic version of a person's written signature. It is a way for the sender of a message to add a special code to the message that works like a signature. This helps ensure that the message is genuine and has not been tampered with.

To make a digital signature work, three steps are involved: making a special key, creating the signature, and checking the signature to make sure it is valid. There are different varients of signcryption. Some varients are blind signcryption, certificateless signcryption,

generalized signcryption. Blind signcryption used to shield the sender's identity and privacy from other users, particularly in electronic currency payment and voting systems.

Certificateless signcryption is a variation of ID-based signcryption. Mostly Certificate authority and key generation center are based on certificateless signcryption. Generalized signcryption is an extension of signcryption. There are three modes of generalized signcrytption. One is signature only mode, encryption only mode, and signcrytption mode.

Security attacks on cryptographic algorithm are increasing day by day. There are some attacks that concern with cryptosystem.

- Known plaintext attack
- Man in the middle attack
- Chosen plaintext attack
- Forgery attack
- Cipheretext only attack

1.1 Internet of Things and its Properties in Cryptography

The Internet of Things (IoT) refers to the network of interconnected devices and objects that communicate and exchange data with each other over the internet. These devices can include everyday objects such as home appliances, wearable devices, industrial machines, and more. Cryptography plays a crucial role in securing the communication and data exchanged within the IoT ecosystem. Here are some properties of IoT in the context of cryptography:

1. **Confidentiality:**

- **Encryption:**

  Cryptography is used to encrypt the data transmitted between IoT devices and systems. This ensures that even if the data is intercepted, it cannot be easily understood without the appropriate decryption key.

2. **Integrity:**

- **Digital Signatures:**

  Cryptographic techniques like digital signatures are employed to verify the integrity of the data.

  By signing data with a private key, the sender provides a way for the recipient to confirm that the data has not been altered during transmission.

3. **Authentication:**

- **Authentication Protocols:**

  Cryptography is used to establish the identity of IoT devices and ensure that only authorized devices can communicate with each other. This is essential for preventing unauthorized access and malicious activities.

4. **Non-repudiation:**

- **Digital Signatures:**

  In addition to ensuring integrity, digital signatures also provide non-repudiation, meaning that a sender cannot deny sending a message that they digitally signed.

5. **Key Management:**

- **Secure Key Exchange:**

  Managing cryptographic keys securely is a critical aspect of IoT security. Secure key exchange protocols are implemented to ensure that keys are exchanged between devices in a secure manner.

6. **Scalability:**

- **Efficient Algorithms:**

  Cryptographic algorithms used in IoT must be scalable to handle a large number of devices efficiently. Lightweight cryptographic algorithms are often preferred to minimize resource consumption on IoT devices, which may have limited processing power and memory.

1.2 Literature review

In 1997, Zheng [9] introduced a new idea called signcryption, aiming to make the processes of encryption and signature more efficient by combining them. This approach, called signcryption, is a way to simultaneously perform encryption and signature, offering advantages over the traditional sign-then-encrypt method.

In 2006,[10, 11] Han and his team introduced a new concept called Generalized Signcryption (GSS), which is a versatile primitive that can function as an encryption scheme, a signature scheme, or a signcryption scheme using just one algorithm. They developed a GSS scheme based on ECDSA [12].

In 2007, Wang et al [13] established the first security model and enhanced the scheme proposed. Following this, Lal and Kushwah [14] introduced the first ID-GSC scheme in 2008, along with a security model. In 2007, Au et al. [15] introduced a new Type-II adversary called a malicious-but-passive Key Generation Center (KGC), which may act maliciously during the setup stage.

Some cryptographic schemes secure against this type of attack have been developed. While the security models of previous CLGSC schemes covered honest but curious KGC, we introduce a formal security model for CLGSC schemes secure against malicious but passive KGC attacks.

Later in 2008, Barbosa and Farshim [16] expanded this concept to a certificateless cryptographic system. Signcryption becomes particularly useful in network environments where both confidentiality and authenticity are essential. However, there are situations where we may need these aspects separately.

To address this, we commonly employ three distinct cryptographic schemes: encryption, signature, and signcryption. In environments with limited bandwidth, such as smart cards and wireless sensor networks (WSN), using three distinct schemes for achieving confidentiality and authenticity separately or simultaneously might not be feasible. This is due to the constraints of low bandwidth.

However, in 2010, Yu et al. [17] revealed that the security model proposed by Lal and Kushwah was incomplete. They improved the model and proposed a secure scheme. Ji et al. [18] introduced Certificateless Generalized Signcryption (CLGSC) in 2010, defining it formally and providing a security model along with a concrete scheme. However, in 2010, Kushwah and Lai [19] identified security issues in Ji et al. [20] scheme and proposed a new secure and efficient CLGSC scheme.

In the same year, Ji et al. [20] proposed another CLGSC scheme based on a previous one, but it was later found to be insecure. To date, there are only three CLGSC schemes in the literature. Certificateless cryptography typically considers two types of attackers: Type I, who lacks access to the master secret key but can manipulate public keys, and Type II, who has access to the master secret key and can compute partial private keys.

In 2011, Kushwah and Lal [21] simplified the security model and presented an efficient ID-GSC scheme. Various other GSC schemes were proposed by different researchers, including multi-receiver schemes and multi-PKG schemes. Our proposed scheme is proven secure under certain hardness assumptions, and performance analysis demonstrates its efficiency and practicality.

In comparison to conventional signature-then-encryption techniques, signcryption can cut computational costs and transmission overheads by combining digital signature and encryption into a single logical step. Some of the crucial characteristics of signcryption include accuracy, effectiveness, security in terms of forward secrecy, and unforgeability [22].

In recent years, numerous signcryp-tion methods have been created. A common type of signencryption [23].
There are several signcryption methods based on hyper elliptic curves that have been studied and found to be effective in terms of security [24].

It is also important to note several other notable applications of signcryption in different IoT based infrastructures. Blockchain-based signcryption is also derived to improve signcryptions' performance. In order to achieve post quantum resistance, lattices are used in sign encryption[25–28].

By performing aggregations on the total number of signcryptions, aggregation algorithms can be further improved. Such advancements are a result of the goal of delivering the necessary security with less overhead and processing. It is important to discuss an IoT development for obfuscating aggregate signcryption [29].

1.3 Thesis Contribution

In this thesis, we expanded Zhang et al. [30] certificateless aggregate signcryption scheme, which is based on bilinear maps, into a new generalized signcryption scheme.

This scheme utilizes bilinear mapping in a group for secure and authenticated message transmission, offering both digital signature and encryption with reduced computational costs compared to a signature-then encryption scheme.

Its security relies on the difficulty of solving the elliptic curve discrete logarithm problem (ECDLP) and the elliptic curve Diffie-Hellman problem, which are currently more secure. The scheme ensures integrity, message confidentiality, forward secrecy, availability, unforgeability, verification, and non-repudiation security attributes.

While the computational time is slightly higher than Zheng and Imai's scheme and Zhang et al.'s scheme, our proposed scheme is more secure.

Our proposed scheme serves dual functions, seamlessly providing both confidentiality and authenticity separately without additional calculations. It encompasses all security attributes and remains unaffected by various known attacks.

1.4 Thesis layout

- In Chapter 1, Introduction, a literature review covering cryptography, Elliptic curve cryptography, Certificateless aggregated signcryption, Certificateless generalized signcryption, and cryptanalysis is provided.
- In Chapter 2, mathematical and cryptographic background is provided.
- In Chapter 3, fundamental terminology and concepts concerning certificateless aggregated signcryption are introduced.
- In Chapter 4, the proposed generalized signcryption scheme which is based on bilinear mapping will discussed.
- In Chapter 5, the conclusion and prospects for future work on the modified scheme are discussed.

Chapter 2

Preliminaries

In this chapter we will discuss some basic definitions utilized throughout the thesis will be recalled.

2.1 Cryptography

Cryptography involves the skill and discipline of converting confidential messages into an indecipherable form known as ciphertext. Only individuals possessing a secret key have the ability to decode the ciphertext and retrieve the original message. There are five fundamental components of a typical cryptosystem:

1 Plaintext space M

2 Ciphertext space C

3 Encryption algorithm E

4 Decryption algorithm D

5 Key K

Cryptography have the following types:

i Symmetric Key Cryptography(secret key cryptography)

ii Public Key Cryptography

2.1.1 Symmetric Key Cryptography

"A system in which related keys is used for both Encryption and Decryption is called symmetric key cryptography [31]. For example, Data Encryption Standard(DES) [4], Double Data Encryption standard [32], Triple Data Encryption (3DES) [32] and Advance Encryption Standard (AES) [5]. A model of symmetric key cryptography is shown in the figure 1.

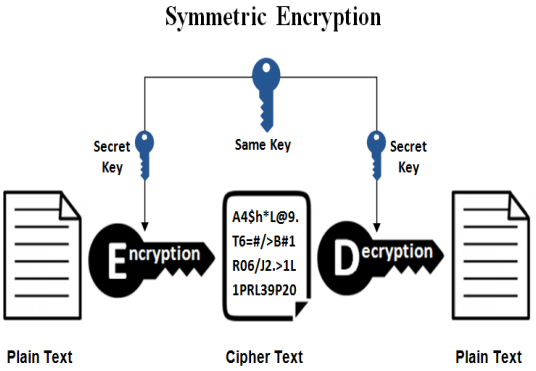


FIGURE 2.1: Symmetric-Encryption

The main disadvantage of symmetric key cryptography is key sharing which means that the secret key is to be transmitted to each party involved in the communication. Electronic communication used for this purpose may not be a secure way of exchanging keys because anyone can access to the communication channels. The only protected ways of switching keys will be to exchange them privately but it could be a very difficult task.

2.1.2 Public Key Cryptography

Public key cryptography is proposed by Diffie and Hellmen[6] in 1976. In public key cryptography [32], two keys are used for encryption and decryption, one is called public key which is known to everybody and the other is called secret key which is kept secret by user. The public key cryptography is shown in the figure 2.2

Here sender encrypt original text using public key and encryption algorithm to obtain the cipher-text. The secret key and decryption algorithm are used by the receiver end to

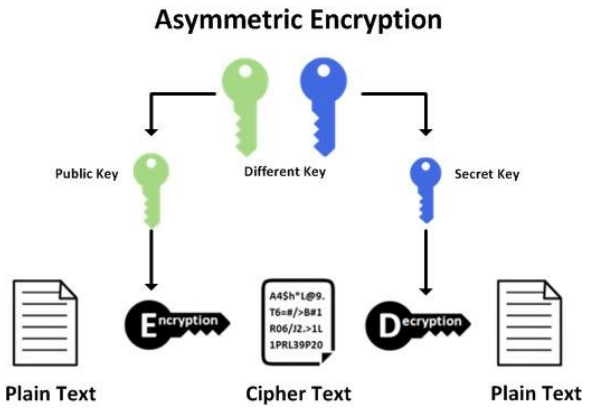


FIGURE 2.2: Asymmetric

obtain original text. RSA cryptography [4] and ElGamal cryptography [5] are examples of asymmetric key cryptography. Diffie and Hellman version of the cryptographer based on trapdoor function (which is easy to calculate in one direction but hard to calculate in other direction).Diffie and Hellman protocol relies on some hard problems which will be discussed after the mathematical background.

Definition 2.1.1. Hash function

A string of characters with a variable is known as message. A "hash function" which is mathematical function simply convert this data or string into a string with a fixed number of characters known as a hash value or just a hash. Since even a small change to the message will give output that is completely unique hash, hashing is important for verifying the legitimacy of an amount of data and that it has not been messed with.

Hashing is convenient to certify the authenticity of data and that it has not been tinker with even a small change in the message will create an entirely distinct hash string [33]. Mathematically, if the value of t is given then it is easy to calculate $h(t)$ but with given $h(t)$ it is hard to calculate the value of h. The result of hash function is named as hash value.The hash value are easy to calculate but difficult to invert. The describe properties of a hash function:

1. **Efficiency:** In a hash function, for any given output, the hash value or output is easily computed.

2. **Pre-image resistance:** In a hash function, it is impossible to determine the associated input value for any given output or hash value.

3. **Collision resistance:**

   - A collision occurs when two distinct inputs produce the same output.
   - A hash function h is collision resistant if noboby can find a collision
   - A hash function h is said to be collision resistant if it is infeasible to find two value, x_1 and x_2 such that $x_1 \neq x_2$, yet $h(x_1)$=$h(x_2)$

4. **Sensitivity:** Small change in input data creates major change in output data. The hash functions that are commonly used are secure hash algorithm SHA [34], SHA-1 [35], SHA-2 (256, 512, 348) [36], SHA-3 [37], message digest 4(MD4) and message digit 5 (MD5). Comparison of different hash functions are describe.

TABLE 2.1: Comparison of hash functions

Algorithm	Output size	Block size	Message size	Rounds	Collision
SHA	160	512	$2^{64}-1$	80	YES
SHA-1	160	512	$2^{64}-1$	80	2^{63}*Attack*
SHA-256	256	512	$2^{64}-1$	64	No
SHA-224	224	512	$2^{64}-1$	64	No
SHA-512	512	1024	$2^{128}-1$	80	No
SHA-384	384	1024	$2^{128}-1$	80	No

Types of Hash Functions

Hash function has two types

1. **Keyed Hash function**

   The one-way keyed hash value is generated by the keyed hash function using both the message and the secret key as inputs.

2. **Unkeyed Hash Function**

   The unkeyed hash function only necessitates input in the form of a message, producing a hash value without the inclusion of any secret key.

2.2 Mathematical Background

Definition 2.2.1. "A **group** G, sometime denoted by $\{G, .\}$ is a set of element with a binary operation, that associates to each order pairs of elements (x, y) in G", such that the axioms listed below are followed:

1. **Closure:** If x and y belong to G, then $x \cdot y$ is also in G.
2. **Associated:** $x \cdot (y \cdot z) = (x \cdot y) \cdot z$ for all x, y, z in G.
3. **Identity element:** There is an element e in G such that $x \cdot e = e \cdot x = x$ for all x in G.
4. **Inverse element:** For each x in G there is an element x' in G such that $x \cdot x' = x' \cdot x = e$

Definition 2.2.2. "A group G is said to be **abelian** if it satisfies the following additional condition $x \cdot y = y \cdot x$ for all x, y in G" [38].

Definition 2.2.3. "A group G is **Cyclic Group** if every element of G is a power x^k (k is an integer) of a fixed element $x \in G$. The element x is said to generate the group G, or to be generator of G".[32]

Example 2.2.4. Following are some examples of groups.

1. Set of integer $\mathbb{Z}$, real number $\mathbb{Q}$, complex number $\mathbb{C}$, are all group under binary operation addition $+$. Here the identity is 0 and the inverse of x is $-x$, Since $x + (-x) = 0$
2. The non-zero rational, real or complex numbers under multiplication. Here the identity is 1 and the inverse of x is x^{-1}, since $x.x^{-1} = 1$.
3. The set $\mathbb{Z}, \mathbb{R}, \mathbb{C}, \mathbb{Q}$ are abelian groups under addition

Definition 2.2.5. "A **Ring** denoted by $(R, +, *)$ is a set of elements together with two binary operations addition " $+$ " and multiplication "$*$" that satisfies the following properties:

1. $(R, +)$ is an abelian group.

2. $(R, *)$ is closed and associative.

3. '$*$' is distributive with respect to "+," i.e for all x,y,z $\in R$.

4. Left and right distributive laws holds in R". [39]

Example 2.2.6. The Set of integers $\mathbb{Z}$ under usual addition "+" and multiplication "." is a ring. Let $\mathbb{Z}_q = \{0, 1, 2...p-1\}$ and $q > 0$ and $q \in \mathbb{Z}$ is a ring under addition and multiplication modulo q. The set of integer $\mathbb{Z}_q$ modulo a prime q is a ring. [39]

Theorem 2.2.7. (Fermats theorem)
"Fermat's theorem states that, If q is prime and a is a positive integer coprime with q then a^{p-1} -1 $\equiv$ 0 mod q" [32].

Definition 2.2.8. Field: "A nonempty set $(\mathbb{F}, +, *)$ together with binary operations "+" and "*" is called field $\mathbb{F}$, if the following properties hold:

1. $\mathbb{F}$ is abelian under addition.

2. $\mathbb{F}$ forms an abelian group under multiplication (only non-zero elements).

3. Multiplication is distributed over addition in $\mathbb{F}$" [40].

Example 2.2.9. Here is a examples of fields.

1. Set of complex numbers $\mathbb{C}$ and set of real numbers $\mathbb{R}$ are field $\mathbb{F}$

2. Set of integers $\mathbb{Z}$ under multiplication is not a field,because inverse does not exist.

Definition 2.2.10. Galois Field A Galois field, also known as a finite field, is a mathematical structure that comprises a finite set of elements, exhibiting properties like to those of a field. Unlike an infinite field, a Galois field involves a limited number of elements. Its elements are typically prime numbers and are often represented as $\mathbb{Z}_p$. The field is essentially constructed from the set of integers under modulo p, denoted $\mathbb{Z}_p$ as , and was initially introduced by "Evariste Galois" in 1905. [41]

Example 2.2.11. Suppose we have Galois Field GF(3) which has the 3 elements only {0,1,2}. The addition and multiplication operation in this field are performed under modulo 3 which means that result of any operation will always be less than 3.

$$4 + 2 \mod 3 = 0$$

also

$$4 * 2 \mod 3 = 2$$

Definition 2.2.12. (Discrete logrithm problem)

"Given $x, y \in \mathbb{Z}_p$ such that $x^n = y \mod p$ then finding n is known as discrete logarithm problem" [42].

Algorithm 2.2.13. Euclidean Algorithm

Input: Two integers $\mathcal{U}$ and $\mathcal{V}$

Output: gcd($\mathcal{U}$, $\mathcal{V}$).

1. If $U = 0$ then $\gcd(\mathcal{U}, \mathcal{V}) = \mathcal{V}$, since $\gcd(0, \mathcal{V}) = \mathcal{V}$ and stop.
2. If $\mathcal{V} = 0$ then $gcd(\mathcal{U}, \mathcal{V}) = \mathcal{V}$, since $\gcd(\mathcal{U}, 0) = \mathcal{V}$ and stop.
3. Write $\mathcal{U} = \amalg * \mathcal{V} + r$ where $\amalg$ is quotient and r is reminder.
4. Find $\gcd(\mathcal{V}, r)$, since $\gcd(\mathcal{U}, \mathcal{V}) = \gcd(\mathcal{V}, r)$.

Algorithm 2.2.14. (Extended Euclidean Algorithm)

The above algorithm is transferred as follows and compute modular inverse of integer $\mathcal{U}$ and $\mathcal{V}$

Input: Two integers $\mathcal{U}$ and $\mathcal{V}$.

Output: gcd($\mathcal{U}$, $\mathcal{V}$.)

1. Set $(\mathcal{U}_1, \mathcal{U}_2, \mathcal{U}_3) = (1, 0, \mathcal{V})$ and $(\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3) = (0, 1, b)$.
2. If $\mathcal{M}_3 = 0$ then returns $\mathcal{L}_3 = \gcd(\mathcal{V}, b)$ no inverse of element b exist.
3. Now check If $\mathcal{M}_3 = 1$ then return $\mathcal{M}_3 = \gcd(\mathcal{V}, b)$ $B_2 = b^{-1} \mod \mathcal{V}$.
4. Now divide $\mathcal{U}_3$ and $\mathcal{V}_3$ set the quotient $R = \mathcal{U}_3$ div $\mathcal{V}_3$.

5. Now let we take (P_1,P_2, P_3) = ($\mathcal{L}_1 - R * \mathcal{M}_1$, $\mathcal{L}_2 - R * \mathcal{M}_2$, $\mathcal{L}_3 - R * M_3$).

6. Set $(\mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3) = (\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3)$.

7. Set $(\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3) = (P_1, P_2, P_3)$.

8. Goto step number 2 [32].

2.3 Elliptic curve cryptography

Cryptography with public key that involves elliptic curve is called elliptic curve cryptogrphy(ECC), and it is consist on the algebraic structure of elliptic curves over finite fields. ECC allows for smaller keys to achieve equal security compared to non elliptic curve encryption (based on plain Galois fields).[43]

Applications for elliptic curves include key agreement, digital signatures, and pseudo-random number generators. By coupling a symmetric encryption method with a key agreement, they may be used for encryption secretly.

The use of an ellpitic curves in cryptography was first suggested in 1985 Neal I.Koblitz and Victor Saul Miller. The main benefit of using an elliptic curves is that the same level of security may be attained by working in a field with 160 bits, however, solving the issue of processing complexity to obtain the appropriate level of security. An elliptic curve will be thoroughly explained in the section after this. [44]

TABLE 2.2: Comparison of Key sizes

Symmetric	RSA	ECC
56	512	112
80	1024	160
112	2048	224
128	3072	256
192	7680	384
256	15360	521

2.4 Elliptic Curve Encryption Decryption

Asymmetric key cryptography uses the elliptic curve approach. Each user must have a unique public key and private key for secure communication.

2.4.1 Global Settings

These global parameters that involved in communication between sender and receiver.

1. $\mathbb{G}$ is the base point such that $n * \mathbb{G} = \mathbb{O}$. where n is the smallest prime number and $\mathbb{O}$ is point at infinity.

2. A prime integer modulo q and constants a and b.

2.4.2 Key Generation Phase

1. Alice choose randomly secret key $n_a \in \{1, 2, ...n-1\}$ and calculate public key as P_a=$n_a * \mathbb{G}$.

2. Choose his secret key $n_b < n$ and compute his public key as $P_b = n_b * G$.

2.4.3 Encryption Phase

1. The sender uses the ECC scheme to send a message m to the receiver. For this m is converted into a elliptic curve point $\mathbb{P}_m$.

2. Sender select a random integer k and calculates the ciphertext C_m as the elliptic curve pair of points using receiver's public key $\mathbb{P}_b$ as follows.

$$C_m = (kG, \mathbb{P}_m + k\mathbb{P}_b) \mod q$$

2.4.4 Decryption Phase

After receiving ciphertext C_m, message will be decrypted back into original form by multiplying $k*G$ with private key of Bob n_B and then add the result into second ciphertext pair $(P_m + kP_B)$

$$\mathbb{P}_m + k\mathbb{P}_b - n_b(kG) = \mathbb{P}_m + k(\mathbb{P}_b) - k(\mathbb{P}_b) \mod q$$

$$= \mathbb{P}_m, \mod q$$

which is plaintext, so receiver gets the same value.

Example 2.4.1. Let us consider an elliptic curve $y^2 = x^3 - 4 \bmod 257$ that is equalvalent to $E_q(0, -4)$.

let $G = (2, 2)$ be the basepoint of an elliptic curve. Total number of points are 258 and order of this curve is 129 where $129(2, 2) = \mathbb{O}$. Let private key of receiver is $n_b = 101$ and his public key is $\mathbb{P}_b = n_b * G = 101(2, 2) = (197, 167)$

A sender wishes to send a message to receiver that is encrypted in an elliptic point $\mathbb{P}_m = (112, 26)$. Sender chooses random integer $K = 41$ and computes

$$K * \mathbb{P}_b = 41(197, 167) = (68, 84) \mod 257$$

$$K * G = 41(2, 2) = (136, 128) \mod 257$$
$$\mathbb{P}_m + k(\mathbb{P}_b) = (112, 26) + (68, 84) = (246, 174) \mod 257$$

sender sends the ciphertext to receiver,

$$C_m = (c_1, c_2) = \{K * \mathbb{G}, \mathbb{P}_m + k(\mathbb{P}_b)\} \bmod 257$$

$$C_m = \{(136, 128), (246, 174)\} \mod 257$$

then receiver receives the ciphertext and decrypt the ciphertext

$$\begin{aligned}\mathbb{P}_m =&\{\mathbb{P}_m + K(\mathbb{P}_b) - K(\mathbb{G} * n_b)\} \mod 257\\ =&\{(112,26) + (68,84) - 41(197,167)\} \mod 257\\ =&\{(112,26) + (68,84) - (68,84)\} \mod 257\\ \mathbb{P}_m =&(112,26) \mod 257.\end{aligned}$$

Definition 2.4.2. (Weierstrass equation)

The equation of the form

$$y^2 + \alpha_1 xy + \alpha_3 y = x^3 + \alpha_2 x^2 + \alpha_4 x + \alpha_5$$

defined over a some field $\mathbb{F}$, such as field of real numbers $\mathbb{R}$, complex numbers $\mathbb{C}$ or any finite field $\mathbb{F}_p$ and known as Weierstrass equation. Where α_1, $\alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6$ are called Weierstrass coefficients [45].

$$-\beta_2^2\beta_8 - 8\beta_4^3 - 27\beta_6^2 + 9\beta_2\beta_4\beta_6 \neq 0$$

where,

$$\begin{aligned}\beta_2 =&\alpha_1^2 + 4a_2\\ \beta_4 =&2\alpha_4 + \alpha_1\alpha_3\\ \beta_6 =&\alpha_3^2 + 4\alpha_6\\ \beta_8 =&\alpha_1^2\alpha_6 + 4\alpha_2\alpha_6 - \alpha_1\alpha_3\alpha_4 + \alpha_2\alpha_3^2 - \alpha_4^2.\end{aligned}$$

2.5 Elliptic curve over Finite Fields

This equation describes an elliptic curve.

$$y^2 = x^3 + ax + b^2 \mod p \tag{2.1}$$

where a and b are Weierstrass coeficients and select from finite field $\mathbb{F}_{\mathbb{P}}$. If the discriminant $4a^3 - 27b^2 \neq 0$ then curve is said to be smooth and this curve is known as an Elliptic curve.

2.5.1 Point addition

Consider two points, $P(x_1, y_1)$ and $Q(x_2, y_2)$, on an elliptic curve E. The addition of P and Q is $R(x_3, y_3)$ as shown in figure.
The following steps must be followed in order to add such points.

1. A straight line is passed from points P and Q .

2. At some point, the straight line intersects the curve, say at R of E.

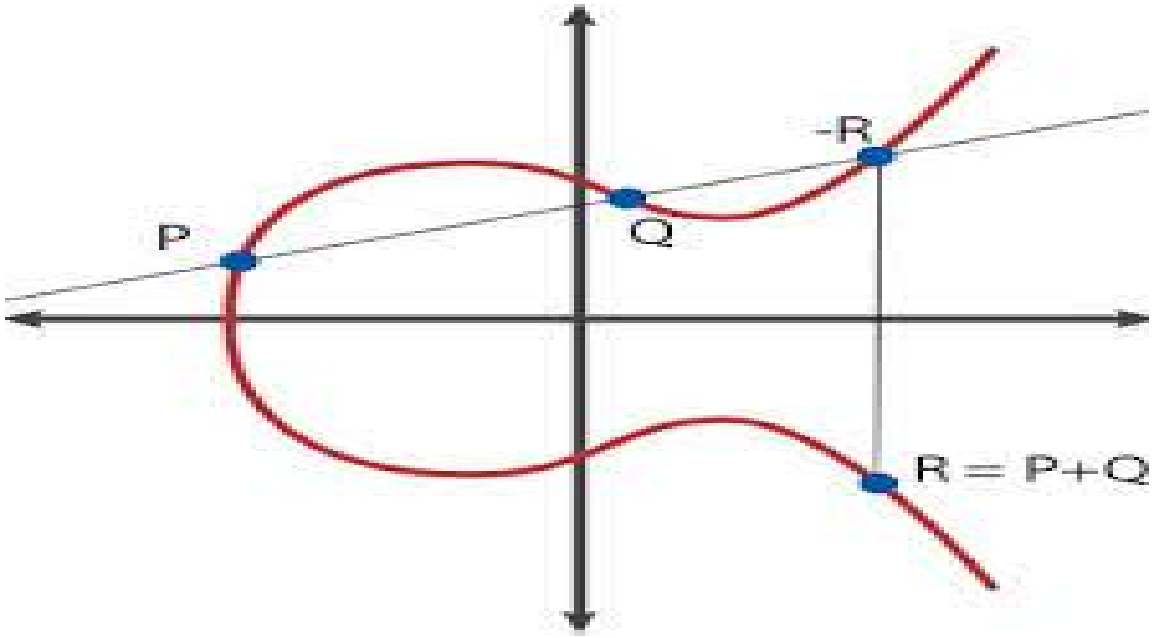


FIGURE 2.3: Point addition

3. Next, we get the point S as the product of P and Q. It is only appropriate to take the S negative, which is $-R = (x, -y)$.

2.5.2 Point doubling

Now we explain the point doubling operation of elliptic curve. Suppose we have point $P(x_1, x_2)$ on elliptic curve E. To add a point P to itself as $P + P = 2P$, we perform the following actions:

1. Draw a P-tangent $P(x_1, x_2)$ which intersects the curve at the second point of elliptic curve E.

2. It eventually crosses the curve, which is once more considered to be a R on E.

3. The next step is to simply take R negative to get at the position $S = 2P$, which is the result of multiplying P by itself.

2.5.3 point at infinity

The similar approach can be applied to the addition of P to $-P$. We are aware that $-p$ is is effectively an extension of P. So, as a straight line departs from them, it gets closer to infinity. To identify a specific infinity-bound point that is known as a point towards infinity.

2.6 Mathematical Representation

For the addition of points $P(x_1, y_1)$ and $Q(x_2, y_2)$ on elliptic curve. For graphical representation of point addition, a line must be drawn through them. Let the line pass through P and Q , the point slope form of L is

$$L : y = sx + c$$

To calculate the slope s following steps must be obeyed

$$s = \begin{cases} \frac{y_2 - y_1}{x_2 - x_1} & if P \neq Q \\ \frac{3{x_1}^2 + a}{2y_1} & if P = Q \end{cases} \tag{2.2}$$

by using basic algebra, the new point say $R^{'}(x_3, y_3)$ obtained by adding $P(x_1, y_1$ and $Q(x_2, y_2)$ has the following coordinates:

$$x_3 = s^2 - x_1 - x_2 \tag{2.3}$$

$$y_3 = s(x_1 - x_3) - y_1 \tag{2.4}$$

Example 2.6.1. Let us consider the curve over $\mathbb{F}_{11}$ that is

$$y_2 = x^3 + x + 6 \mod 11 \tag{2.5}$$

The elliptic curve points addition for $E_{11}(1,6)$ let $P(3,5)$ and $Q(7,9)$ are any two points on elliptic curve E, then formula provide us new points $R^{'}(x_3, y_3)$ as shown in table 2.3 where

$$\begin{aligned} s =& \frac{9-5}{7-3} \mod 11 \\ =& \frac{4}{4} \Rightarrow 1 \mod 11 \\ s =& 1 \mod 11 \end{aligned}$$

TABLE 2.3: Point at infinity

x	y^2	y_1	y_2	$P(x,y)$	$Q(x,y)$
0	6	–	–	–	–
1	8	–	–	–	–
2	5	4	7	(2,4)	(2,7)
3	3	5	6	(3,5)	(3,6)
4	8	–	–	–	–
5	4	2	9	(5,2)	(5,9)
6	8	–	–	–	–
7	4	2	9	(7,2)	(7,9)
8	9	3	8	(8,3)	(8,8)
9	7	–	–	–	–
10	4	2	9	(10,2)	(10,9)

Put value of s in eq 2.3

$$x_3 = s^2 - x_1 - x_2$$

$$y_3 = s(x_1 - x_3) - y_1$$

$$x_3 = 1^2 - 3 - 7 \mod 11$$

$$= -9 \mod 11$$

$$x_3 = 2 \mod 11$$

$$y_3 = 1(3-2) - 5 \mod 11$$

$$y_3 = -4 \mod 11$$

$$y_3 = 7 \mod 11 \tag{2.6}$$

so, $S'(x_3, y_3) = (2, 7)$ is the addition of points. Now let us add a point $P(3, 5)$ into itself.

$$s = \frac{3x^2 + a}{2y_1} \Rightarrow \frac{3(3^2) + 1}{2(5)} \mod 11$$

$$= \frac{3(9) + 1}{10} \mod 11$$

$$= \frac{14}{5} \mod 11$$

$$= 14(5)^{-1} \mod 11$$

$$= 14(6) \mod 11$$

$$s = 7 \mod 11$$

Now use values of s

$$\begin{aligned} x_3 &= s^2 - 2x_1 \\ x_3 &= 7^2 - 2(3) \mod 11 \\ &= 49 - 6 \mod 11 \\ &= 43 \mod 11 \\ x_3 &= 10 \mod 11 \\ y_3 &= 7(3 - 10) - 5 \\ y_3 &= 7(-7) - 5 \\ &= -49 - 5 \mod 11 \\ y_3 &= 10 \mod 11 \end{aligned}$$

so we have

$$S^{'}(x_3, y_3) = (10, 10)$$

2.7 Elliptic Curve Discrete Logarithm Problem

Given an ellpitic curve $y^2 = x^3 + ax + b^2 \mod p$ and a basis point P, we will calculate $Q = P^k$ through $k - 1$ iterative point additions. Fast algorithems for this task exist. It is hard to compute k when the point Q is known. This is known as an Elliptic Curve Discrete logarithm Problem(ECDLP). ECDLP is responsible for ECC's complete security.

2.8 Diffie-Hellman Key Exchange Based for Eliptic Curve Group

To communicate securely, Alice and Bob must disclose their keys to encrypt and decrypt the messages. Exchanging of keys over a public network without compromising security was first intoduced by Diffie and Hellman [6] in 1976. A cyclic group of elliptic curve points

is used to create the scheme, and the safety of the system depends on how challenging it is to overcome ECDLP. The Diffie-Hellman key exchange protocol is apply to exchange keys between Alice and Bob in the following method.

1. Sender and receiver mutually selects an elliptic curve E over a finite field $\mathbb{F}_{|}$ with base point of elliptic curve G of curve E.
2. Sender selects a random number $n_a \in \{1, 2, 3, ...n1\}$ as her secret key and compute her public key as $P_a = n_a * G$.
3. Bob choose his private key $n_b \in \{1, 2, 3, ...n1\}$ and calculates his public key $P_b = n_b * G$.
4. They both share their public keys P_a and P_b with each other.
5. Alice computed $k = P_{ab} = P_b * n_a = n_a * n_bG$ where P_{ab} is used to find n_a and n_b as session key security.
6. Bob computed $k = P_{ab} = n_b * P_a = n_a * n_bG$

Example 2.8.1.

Let sender A wishes to send a message $m = 23$ to recipient B. So, they must share their keys to encode and decode a message. A sender a and recipient b mutually selects an elliptic curve $y_2 = x^2 - 4 \mod 257$ that is equalvalent to $E_q(0, -4)$, where $G = (2, 2)$ is the base point of order $n = 129$ where $129(2, 2) = \mathbb{O}$. An elliptic curve has also 258 points.

1. Sender A selects secret key $n_a = 101$ and compute public key as $P_a = n_a * G = 101(2, 2) = (197, 167)$.

2. Recipient B selects secret key $n_b = 17$ and compute public key as $P_b = n_b \cdot G = 17(2, 2) = (80, 56)$.

3. They both exchange their $P_a = (197, 167)$ and $P_b = (556, 631)$ with each other.

4. Sender A computed $P_{ab} = n_a * n_b = 17(101) = 175 \mod 257$ is used to find $n_a = 101$ and $n_b = 17$ as session key security.

2.9 Elliptic Curve Encryption Decryption

Asymmetric key cryptography uses the elliptic curve approach. Each user must have a unique public key and private key for secure communication.

2.9.1 Global Settings

These global parameters that involved in communication between sender and receiver.

1. $\mathbb{G}$ is the base point such that $n * \mathbb{G} = \mathbb{O}$. where n is the smallest prime number and $\mathbb{O}$ is point at infinity.

2. A prime integer modulo q and constants a and b.

2.9.2 Key Generation Phase

1. Alice choose randomly secret key $n_a \epsilon \{1, 2, ...n1\}$ and calculate public key as P_a=$n_a *$ $\mathbb{G}$

2. Choose his secret key $n_b < n$ and compute his public key as $P_b = n_b * G$

2.9.3 Encryption Phase

1. The sender uses the ECC scheme to send a message m to the receiver. For this m is converted into a elliptic curve point $\mathbb{P}_m$

2. Sender select a random integer k and calculates the ciphertext C_m as the elliptic curve pair of points using receiver's public key $\mathbb{P}_b$ as follows.

$$C_m = (kG, \mathbb{P}_m + k\mathbb{P}_b) \mod q$$

2.9.4 Decryption Phase

After receiving ciphertext C_m, message will be decrypted back into original form by multiplying $k*G$ with private key of Bob n_B and then add the result into second ciphertext pair $(P_m + kP_B)$

$$\begin{aligned}\mathbb{P}_m + k\mathbb{P}_b - n_b(kG) =& \mathbb{P}_m + k(\mathbb{P}_b) - k(\mathbb{P}_b) \mod q \\ =& \mathbb{P}_m\end{aligned}$$

which is plaintext, so receiver gets the same value.

Example 2.9.1.

Let us consider an elliptic curve $y^2 = x^3 - 4 \bmod 257$ that is equivalent to $E_q(0, -4)$.
let $G = (2, 2)$ be the base point of an elliptic curve. Total number of points are 258 and order of this curve is 129 where $129(2, 2) = \mathbb{O}$. Let private key of receiver is $n_b = 101$ and his public key is $\mathbb{P}_b = n_b * G = 101(2, 2) = (197, 167)$.

A sender wishes to send a message to receiver that is encrypted in an elliptic point $\mathbb{P}_m = (112, 26)$. Sender chooses random integer $K = 41$ and computes

$$K * \mathbb{P}_b = 41(197, 167) = (68, 84) \mod 257$$

$$K * G = 41(2, 2) = (136, 128) \mod 257$$

$$\mathbb{P}_m + k(\mathbb{P}_b) = (112, 26) + (68, 84) = (246, 174) \mod 257$$

sender sends the ciphertext to receiver,

$$C_m = (c_1, c_2) = \{K * \mathbb{G}, \mathbb{P}_m + k(\mathbb{P}_b)\} \mod 257$$

$$C_m = \{(136, 128), (246, 174)\} \mod 257$$

then receiver receives the ciphertext and decrypt the ciphertext

$$\begin{aligned}
\mathbb{P}_m =& \{\mathbb{P}_m + K(\mathbb{P}_b) - K(\mathbb{G} * n_b)\} \mod 257 \\
=& \{(112, 26) + (68, 84) - 41(197, 167)\} \mod 257 \\
=& \{(112, 26) + (68, 84) - (68, 84)\} \mod 257 \\
\mathbb{P}_m =& (112, 26) \mod 257
\end{aligned}$$

2.10 Cryptanalysis

The study of ciphertext, ciphers, and cryptosystems is known as cryptanalysis, which aims to develop techniques for weakening or breaking them. Secure hashing, digital signatures, and other cryptographic techniques are the focus of crypto analysts, who, for example, attempt to interpret ciphertexts without being aware of the plaintext source, encryption key, or method that was used to encrypt them.

While cryptanalysis aims to detect weaknesses in cryptographic algorithms or otherwise undermine them, cryptographers make use of cryptanalysts research findings to strengthen or replace outdated techniques. Cryptography, which focuses on creating encryption ciphers and made better other techniques, It encircle both cryptanalysis and cryptography.

It is conceivable for researchers to devise techniques of attack that entirely destroy an encryption method, making it trivially simple to decrypt ciphertext encoded with that algorithm without the encryption key. When cryptanalytic output point out infirmity in the design or implementation of the technique, the total number of keys are turn down that are tried on target ciphertext.

An encryption algorithm may be completely defeated by attack techniques created by researchers, making it trivially viable to decipher ciphertext encrypted using that algorithm without the encryption key.

When cryptanalytic find out errors in the design or implementation of the algorithm, the number of keys that have to be checked on the target ciphertext might be reduced.

2.10.1 Types of attacks

There are different types of attacks. Some of them are discussed below.

2.10.2 Ciphertext Only

In this attack, the attacker knows only the ciphertext. Normally, the corresponding plaintexts are not known. To obtain the corresponding plaintexts, he utilises these known ciphertexts to break the system.

2.10.3 Known Plaintext Attack

The analyst could have approach to any or every bit of the ciphertext of plaintext in a known plaintext attack. The analyst's goal in this scenario is to locate the encryption key and decrypt the message. Once that key is discovered, an attacker can use it to decrypt all messages that were originally encrypted.

A known plaintext attack called linear cryptanalysis approximates the operation of a block cipher using a linear function. The ability to decipher or infer a portion or the full of an encrypted message, as well as the format for the original plaintext, is a prerequisite for known plaintext attacks.

2.10.4 Choosen Plaintext Attack

In a choosen plaintext attack, the interpreter is either in possession of the necessary encryption tools or is conversant with their workings. The interpreter can encrypt the selected plaintext with the selected algorithm to discover more about the key.

2.10.5 Differential Cryptanalysis Attack

Differential cryptanalysis is a specific plaintext attack for block cyphers that looks at pairs of plaintexts rather than a single plaintext to show how the targeted algorithm responds to different types of data.

2.10.6 Integral Cryptanalysis Attacks

Comparable to differential cryptanalysis attacks, integral cryptanalysis attacks use sets of plaintexts rather than pairings, with some of the plaintext remaining constant while the remainder is changed. This attack is particularly effective against block ciphers built on substitution-permutation networks.

2.10.7 Side Channel Attack

Data from the system that is used for encryption and decryption of the message is collected during a side-channel attack. Successful side-channel attacks make use of information other than the ciphertext generated by the encryption method, such as, information on how quickly a system answers to particular queries, total power the encrypting system injest, or how much electromagnetic radiation emited by the system.

2.10.8 Man-in-the-Middle Attacks

When a third party learns how to enter the communication room between two parties who are interacting with one other and want to exchange keys for encrypted communication

using the asymmetric or public keys, this is known as a man-in-the-middle attack. The attacker performs a key exchange with every single one of the original parties while the parties think they are exchanging keys with each other. The keys of an attacker are ultimately used by the two parties.

2.10.9 Digital Signature

A digital signature serves as a mathematical formula applied to validate the authenticity and integrity of digital messages or documents. With a valid digital signature, the recipient of a communication can have a high level of confidence that the message originated from a known sender (authenticity) and remained unaltered during transmission (integrity), assuming certain conditions are met.
Digital signatures are integral to most cryptographic protocol suites and find widespread use in financial transactions, software distribution, contract management, and other scenarios where detecting fraud or tampering is crucial. They are a key component of asymmetric cryptography, enhancing security and verification across insecure channels.
Properly implemented digital signatures are more difficult to forge compared to traditional handwritten signatures. These cryptographic systems require careful implementation to be effective and can also provide non-repudiation, meaning the signer cannot deny having signed a message while still keeping their private key secret [46].

Properties of Digital Signature

1. **Authenticity:** In order for a signature to be considered genuine, the communication it is attached to must have been signed knowingly.

2. **Unforgeability:** A valid signature for the related message can only be provided by the signer.

3. **Non-re-usability:** A signature from one document cannot be used on another.

4. **Non-repudiation:** The signer of a document with a valid signature cannot retract their signature.

5. **Integrity:** Make sure the information hasn't been changed.

2.10.10 Signcryption

Signcryption is a public-key primitive in cryptography that combines the capabilities of digital signature with encryption. They were considered as significant but separate components of many cryptographic systems up until 1997.

In public key systems, the traditional method is to sign a communication digitally, then encrypt it (signature, then encryption). However, this approach has two drawbacks. First, it is inefficient and ex-pensive, and second, no random strategy can provide security.

Signcryption is a relatively recent cryptographic technique that seeks to merge the digital signature and encryption operations in a single logical step, as opposed to the traditional signature-then-encryption systems. Additionally, it can drastically lower the overheads associated with communication and computing.

Rather than signing and encrypting separately, a more efficient approach, signcryption combines the benefits of digital signatures and encryption methods. This suggests that atleast some aspects of its efficiency (such as computing time) are superior to any hybrid of digital signature and encryption systems, in accordance with a particular security paradigm.

In 1997, Yuliang Zheng [9] presented the first signcryption technique. When compared to the conventional elliptic curve-based signature, then encryption systems, Zheng's elliptic curve-based signcryption approach saves 58% of computational costs and 40% of transmission costs.

Various alternative signcryption systems have also been put forth over the years, each of which comes with its own set of issues and constraints in addition to providing varying degrees of security and computational costs.

A signcryption method is often made up of the three methods key creation, signcryption, and un-signcryption. While Sincryption Scheme (SC) is frequently a probabilistic process and Unsincryption Scheme (USC) is nearly surely deterministic, and creates a pair of keys for every user [9].

2.10.11 Security attributes

- **Confidentiality:**

  It should be computationally impossible for an attacker to ex-tract even a small portion of the content of a signcrypted text without knowing the sender's and receiver's private keys.

- **Unforgeability:**

  A clever attacker shouldn not be able to produce an authentic signcrypted text that the unsigncryption algorithm can accept by disguising them-selves as an honest sender due to computational limitations.

- **Non-repudiation:**

  The signcrypted text should be conveyed, and the recipient should be able to show the sender's identity to a third party (such as a judge). Because of this, the sender's previously signed and encrypted texts cannot be dis-puted.

- **Integrity:**

  The message received should be authentic and delivered by the sender, and the recipient should be able to confirm this.

- **Public Verifiability:**

  Without knowing either the sender's or the receiver's private key, anyone may check to see if the signed text is a real signcryption of the message it relates to.

- **Forward Secrecy:**

  No body should be able to decrypt previously sign- crypted texts in plaintext regardless of whether the sender's long-term private key is as-sured. In a standard signature encryption method in the unlikely scenario that the long-term secret key is compromised, all previously issued signatures become invalid. Forward secrecy appears to be a crucial component of such systems, as more cryptographic computation are often performed on unprotected devices like cell phones, the risk of key exposure is increasing.

2.11 Different varients of signcryption

2.11.1 Identity Based Signcryption

A type of public key encryption called identity-based encryption [47] allows users to create their own public keys using well-known unique identifiers, like email addresses, and has a trusted third-party server construct the corresponding private keys from the public keys. This eliminates the requirement for distributing public keys before sharing encrypted data.

The recipients unique identifier can be used by the sender to create a public key and encrypt the contents. The PKG first provides a master public key accessible while maintaining the corresponding master private key (sometimes referred to as the master key) in order to apply this encryption technique.

Any party can determine a public key matching to an identity given the master public key by combining the master public key into a known identity value (such as an email address). The PKG generates the requested private key using its master private key after receiving a request from the owner of the identity that was utilized to generate the public key.

2.11.2 Certificateless Based Signcryption

User encryption and verification keys contain a users identity and an unauthenticated public key, as certificateless cryptography opposes the idea of key escrow. Likewise, user secret keys consist of two partial secrets:

one created by the user and the other provided by an identity-based trusted authority known as the Key Generation Centre (KGC). In scenarios addressed by certificateless security models, either a system user or the KGC itself could act as an attacker. Attackers are permitted to substitute user public keys to attempt impersonation, acknowledging that these keys lack authentication.

2.11.3 Blind Sincryption Scheme

Blind signcryption is a method used to keep the senders identity private, especially in electronic currency payments and voting systems. One blind signcryption technique, developed by Riaz Ullah et al., uses elliptic curve cryptography, which is considered secure due to the difficulty of the elliptic curve discrete logarithm problem. In this scheme, there are three participants: the sender (Alice), the recipient (Bob), and the signer.
The process involves four steps: pre-request, key creation, blind signcryption, and unsigncryption. Alice, the sender, wants to communicate with Bob, the recipient. The signer is the party responsible for signing the received message without knowing its content. In this scenario, Alice is the sender trying to contact Bob.
The security of the system relies on the complexity of elliptic curve cryptography, which adds a layer of protection to the communication process. Alice wants to send a message to Bob, but she wants to keep the message private and secure, especially when sending it over a public network where others might try to intercept it. To do this, Alice takes a step called "blinding." This is like putting a temporary cover on the message to make it unreadable. Then, she sends this blinded message to someone who can sign it, let is call them the "signer." The signer signs the blinded message without knowing what is inside. After getting the signed message back from the signer, Alice removes the blinding cover, a process called "unblinding." Now she has a signed and unblinded message, which she can safely send to Bob. This helps ensure the privacy and integrity of the message during transmission over the public network.

2.12 Gernalized Signcryption

Depending on the situation, it might be enough to just sign or encrypt the message. Unlike standard signcryption, in these cases, only one of the involved parties is necessary. This is because there are no specific key pairs for each party, making traditional signcryption ineffective.

To overcome this limitation, Zheng suggests using the Sincryption scheme, signature and ElGamal encryption in applications. However, it is important to note that in certain

scenarios, especially those involving limited space like embedded devices, this approach becomes impractical.

To address this issue, we propose a definition for generalized signcryption. This type of signcryption is more adaptable and practical, making it suitable for various applications. In generalized signcryption, the three essential functions of signing, encrypting, and signcrypting are all integrated, providing a more versatile solution [13].

There are three main scenarios: signature encryption, where both functions are needed, signature only, and encryption only. Identifying which scenario is in use is important. In public key setups, information about a sender (public and private key) is required for authentication.

2.13 Bilinear Mapping

Definition 2.13.1. It is the mapping the combining two element of two group G_1 and G_2 yield elements if third group G_3 .

"A bilinear map from $G_1 \times G_2$ to G_3 is a function $\varrho : G_1 \times G_2 \rightarrow G_3$ such that $p \in G_1$, $q \in G_2, x, y \in \mathbb{Z}$

$$\varrho(p^x, q^y) = \varrho(p, q)^{xy}$$

These map are called Bilinear Pairing because they are associate pairs of elements from G_1 and G_2 with elements of G_3" [48].

Definition 2.13.2. Admissible Bilinear Map

Let $\varrho : G_1 \times G_2 \rightarrow G_3$ be a bilinear map. Let g_1 and g_2 be the generator of G_1 and G_2, "The map ϱ is admissible bilinear map if $\varrho(g_1, g_2)$ generate G_3 and ϱ is the efficiently computable. These are the only bilinear maps" [48]

Such admossible mapping should also possess the property

1. **Non Degenerate**

   "A bilinear map $\varrho : G_1 \times G_2 \rightarrow G_3$ is non degenerate if it satisfied the condition :

   a: $\text{Ker}(\varrho) = 0$; $\forall p \in G_1$ implies v $= 0$

b: dim $G_1 = G_2$" [49]

2. **Computability**

   "There exists an efficient algorithm to compute $\varrho(g_1, g_2)$ for $g_0 \in G_1$ and $g_1 \in g_2$" [49]

3. **Computation Diffie Hellman**

   "For a cyclic group G of order q, CDH states that: For a given (g, g^x, g^y) with any random generator g ∈ G and random a, b ∈ $\mathbb{F}_{11}$, it is computationally intractable to compute g^{xy}" [49]

4. **Decisional Bilinear Diffie Hellman(DBDH) Problem**

   "Let G be cyclic group of order q and with generator g. For a given g^x and g^y with uniformly and independently chosen x, y ∈ $\mathbb{Z}_{11}$,*is to be calculated* g^{xy} *is random in G."[49]*

5. ***Gap Diffie Hellman (GDH) Problem***

   "Let G be cyclic group of order q and with generator g. Given, $(g^x, g^y) \in G_1$ *with unknown x,y* ∈ $\mathbb{Z}_{11}$,*then compute* $g^{xy} \in G_1$ *with the help of DBDH oracle." [49]*

Chapter 3

Certificateless Aggregated Signcryption Based On Bilinear Mapping

In this chapter, we discuss the review of Certificateless Aggregated Signcryption scheme that is based on bilinear mapping.[50].In the last section we will explain the experimental result and security analysis.

3.1 Aggregated Signcryption Scheme

Recall that, signcryption is a cryptographic primitive that combines digital signature and encryption functionalities in a single step, aiming for a more efficient and compact solution compared to separate application. It ensures confidentiality, integrity, authenticity, and non-repudiation in one operation. By applying the signcryption approach, the time consumption can be decreased. In cryptography, aggregates involve combining or grouping multiple cryptographic entities or operations into a single structure, enhancing efficiency, reducing overhead, and simplifying protocols. Signcryption integrates signature and encryption, providing an efficient approach to secure communication. Aggregate signcryption is used to reduce the overhead caused by cryptographic processes. By employing a certificateless technique, it is possible to reduce the complexity due to certificates.

Types of Aggregates:

1. **Signature Aggregation:** Combines multiple digital signatures into a single signature.

2. **Key Aggregation:** Combines multiple public keys into a single aggregated key.

3. **Secret Sharing and Threshold Cryptography:** Utilized in schemes where multiple parties collaborate, distribute keys or secrets among them.

Benefits:

1. **Efficiency:** Reduces data size for transmission and processing.

2. **Scalability:** Facilitates scalability in cryptographic protocols with numerous participants.

3. **Simplicity:** Simplifies the design and implementation of cryptographic systems.

3.1.1 System Model

The model proposed in [50] involves the use of two Raspberry Pi devices, three minicomputers, five mobile phones, and one desktop to create an IoT model.

One Raspberry Pi 3 functions as the client model and is equipped with DHT-11 and MQ-135 [50] sensors for collecting environmental data within an experimental room. This data is then signed and transmitted to another Raspberry Pi 3, which serves as the server for the sensor network.

Mobile phones in the setup serve as end devices, while the desktop operates as a server with the sensor network server connected to it. The desktop is responsible for verifying the received messages from the sensor network. The system model is depicted in Figure 3.1.1

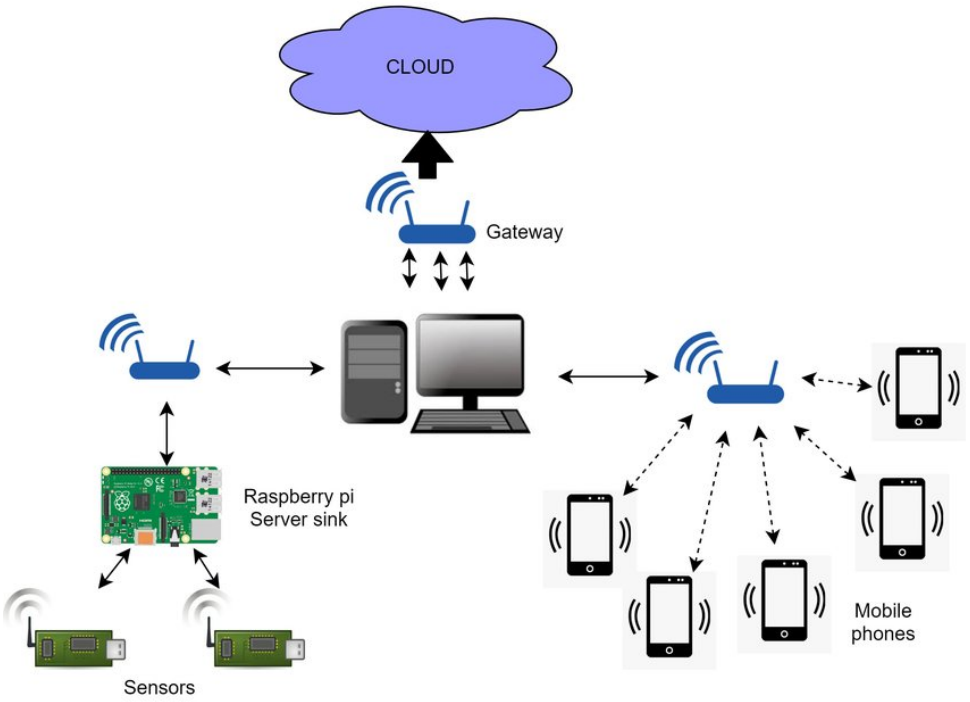


FIGURE 3.1: A system model for the suggested plan demonstrates how a device is connected to the internet via a router.

3.1.2 Functional Description

The key parties involved in the scheme are: Key Generation Center (KGC), a sender u_s and receiver u_r, an aggregating set a of n user and aggregate signcryption generator (ASG). Key generation is the responsibility of KGC. The components of communication nodes are sender and receiver where u_s, $u_r \in a$. The final signcryption is made by ASG, and the incoming signcryptions are verified [51].

Changes to the key generation and master key creation processes are addressed by updates the algorithm presented in [51].

A better foundation for IoT signcryption is thus obtained. we followed the same execution steps as those depicted in the process outlined above. The proposed framework modules are shown in Figure 3.1.2

3.1.3 Proposed Scheme

The subcategories of the proposed scheme are

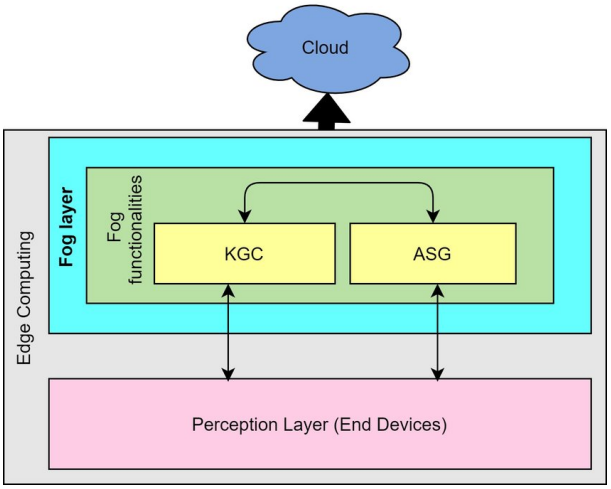


FIGURE 3.2: KGC and ASG

- Setup,
- Extract partial private key,
- User key generation,
- Signcrypt,
- Aggregate,
- Aggregate verify,
- Aggregate unsigncrypt

The following are a few of the scheme's presumptions:

- KGC is safe and dependable.
- A specialised module ASG, links to a group of users individually to aggregate signcryptions.
- The receiver unsigncrpt all messages together.

3.2 Globle Parameter

Here are some important global parameters of the scheme.

TABLE 3.1: Global Parameter

Symbol	Description
$\mathbb{Z}_p$	Finite field of integer element
G_1	Cyclic additive group
G_2	Multipicative group
ϱ	Bilinear map
g_1, g_2	Generator of grougps
msk	Master secret key
h_0, h_1, h_2	Hash functions
param	Public parameter
ID_s, ID_r	Identity of sender, Identity of receiver
$t, \delta t$	Time stamps
δ	Message
Δ	State information
Pub_s, Pub_r	Public key of sender and receiver
Piv_s, Piv_r	Private key of sender and receiver
$\mho$	Ciphertext
$\overline{\mho}$	Aggregate signcryption ciphertext

1. **Setup**

   Selecting Parameters:

   The KGC process that the function setup. It starts by choosing a suitable elliptic curve group over finite fields $\mathbb{Z}_p$ with an order $O = P^k$ where random prime number p and k is an integer. The coefficient v, $s \in \mathbb{Z}_p$, where u and s are the parameter of elliptic curve. The corresponding elliptic curve equation as $y^2 = x^3 + vx + s \bmod p$.

   The following algrothim will be used to create the msk, the master secret key, and publishes a set of system parameter "param".

Algorithm 3.2.1. Setup

Input: $y^2 = x^3 + vx + s$

Output: msk, param

1. By using a cyclic additive group G_1 from $\mathbb{Z}_p$ of prime order p with generator g_1.
2. By using the non-zero element of $\mathbb{Z}_p$, generate the cyclic multiplication group G_2 with generator g_2.
3. Consider the bilinear mapping $\varrho : G_1 \times G_1 \rightarrow G_2$
4. Select a random number $r \in \mathbb{Z}_p^* \rightarrow msk, G_1 \subseteq \mathbb{Z}_p^*$
5. Master public key $mpk = r \cdot g_2 \cdot G$
6. Choose the hash functions
   $h_1 : \{0,1\}^* \rightarrow G_1$ and $h_2 : \{0,1\}^* \rightarrow \{0,1\}^d$
7. Store
   param: $\{G_1, G_2, \varrho, g_1, g_2, mpk, h_1, h_2\}$

2. **Partial Private Key Generation**

   The system parameters, after establishing the KGC starts the key generation process for users who are registering for the network. KGC requires as input parameters,master secret key (msk), a 128-bit timestamp, and the identity of the user ID_u registering for the network. Timestamp aids in preventing key revocation. It should be noted that the suggested system leverages ICMetrics to produce user identities, which are then transformed into 128-bit binary representations $ID_u = \{0,1\}^{128}$ [52]. This identity was created by a specific user. KGC returns a partial $\widehat{piv}_u$ for the user u through an assumed secure channel.

Algorithm 3.2.2. Partial Private Key Generation

Input: params : $\{G_1, G_2, \varrho, g_1, g_2, mpk, h_1, h_2\}$,$msk$, ID_{u_i}

Output: $\widehat{piv}_{u_i}$, the private key of signcryption

1. Compute $Q_i = h_1(ID_{u_i})$.
2. $\widehat{piv}_{u_i} = h_1[(msk.Q_i)||(t + \delta t)]$.
3. Return $\widehat{piv}_{u_i}$.

3. **Users Key pair Generation**

   Upon reception of the partial private key from the Key Generation Center (KGC),

each user engages in a procedure to generate a public-private key pair. This process, taking parameters and the users identity ID_{u_i} as inputs, results in the creation of a private key Piv_{u_i} and its corresponding public key Pub_{u_i} for the user u_i. The user keeps the private key confidential, while the public key is shared openly without any certification.

Algorithm 3.2.3. User Key Pair Generation

Input: param : $\{G_1, G_2, \varrho, g_1, g_2, mpk, h_1, h_2\}$, $\widehat{piv_u}$

Output: $\{pub_u, piv_u\}$

1. select a random number $i \in \mathbb{Z}_p^*$
2. $pub_{u_i} = \widehat{piv}_{u_i} \cdot r_u$ as the users public key.
3. $pub_{u_i} = r_u \cdot g_1 \cdot g_2$
4. return $\{piv_{u_i}, pub_{u_i}\}$

4. **Signcrypt**

Whenever a registered user u wants to communicate. it executes the process of signcryption. The process takes input, of param, some state information Δ, message m, identity of the its own ID_{u_i}, public key pub_{u_i}, private key piv_{u_i}, the identity of receiver ID_{u_r} and its corresponding public key pub_{u_r}. The process output a signcrypted message $\mho$.

Algorithm 3.2.4. Signcryption Input: param: $\{G_1, G_2, \varrho, g_1, g_2, mpk, h_1, h_2\}$, Δ, ,m, ID_{u_r}, pub_{u_i}, piv_{u_i}, pub_{u_r}

Output: $\mho$

1. Choose a random number $r \in \mathbb{Z}_p^*$
2. Compute $U_i = r_2 \cdot g_1 \cdot g_2$
3. Compute $Q_i = h_1(ID_{u_r}||t)$
3. compute $T_i = \varrho(mpk, Q_i)^{r_2}$
4. Compute $H_i = h_2(U_i, T_i, i, pub_{u_r}, \triangle)$
5. $V_i = H_i \bigoplus m$

6. Compute $h_i = h_1(U_i, V_i, ID_{u_i}, pub_{u_i}, ID_{u_r}, pub_{u_r})$
7. Compute $h_\triangle = h_1(\triangle)$
8. Compute $W_i = r_i h_i + r_u h_\triangle$
9. return $\mho : \{U_i, V_i, W_i\}$

5. **Aggregate**

The Aggregate Signcryption Generator (ASG) executes the aggregation process. It takes as inputs an aggregating set i of n users, some state data $\triangle$, the identity ID_{u_i} if user u_i and public keys pub_{u_i}, signcrypted ciphertext $\mho$. It outputs an aggregate ciphertext $\bar{\mho}$.

Algorithm 3.2.5. Aggregation

Input: $a, \triangle, ID_{u_i}, pub_{u_i}, \mho, ID_{u_r}$

Output: $\bar{\mho}$

1. $\overline{W} = \Sigma_{i=1}^{n} W_i, i = 1, 2, 3.....n \in a$ where W_i is a signature if use multiple signature then use $\overline{W}$
2. Combine U_i
3. Combine V_i
4. $\overline{\mho} = \{U_1, U_2, ...U_n, V_1, V_2, ...V_n\}, \overline{W}$

(a) Return $\overline{\mho}$

6. **Aggregate Verification**

Any receiver of $\mho$ has the ability to validate the aggregated signcryption. The receiver needs the proper public keys of all the receivers for whom that $\mho$ is created as inputs for this.

The outputs are compared, and if the comparison is valid, the process of unsigncryption is continued; otherwise, the connection is terminated.

Algorithm 3.2.6. Aggregation Verification

Input: $a, ID_{u_i}, pub_{u_i}, \Delta, \overline{\mho}, ID_{u_r}, pub_{u_r}$

Output: Agree or disagree

1. For $i = 1$ to n do

   Compute $h_i = h_1(U_i, V_i, ID_{u_i}, pub_{u_i}, ID_{u_i}, pub_{u_i}, ID_{u_r}, pub_{u_r})$ End do.

2. compute $h'_{\Delta} = h_1(\Delta)$.

3. $J_1 = \varrho(\overline{W}, pubu_i)$.

4. Compute $J_2 = \varrho(\Sigma_{i=1}^{n=1}, Q_i, mpk) \prod_{i=1}^{n} \varrho(h_i, U_i)\varrho(h'_{\Delta}, \Sigma_{i=1}^{n} pub_{u_i})$.

5. if $(J_1 = J_2)$.

   then accept $\overline{\mho}$.

   else

   disagree and break

6. Return Null

7. **Aggregate Unsigncryption**

   The verification is completed for $\overline{\mho}$, The unsigncryption procedure is carried out by the recipient. The receiver use $\bar{\mho}$, the state information $\triangle$, the identity ID_{u_r}, and its symmetric-Asymmetric key pair $\{pub_{u_r}, piv_{u_r}\}, a$ all the sender identities ID_{u_i} and their corresponding Asymmetric key $pub_{u_i}$and the output number of the plaintext.

Algorithm 3.2.7. Aggregation Unsigncryption

Input: $\overline{\mho}, \triangle, ID_{u_r}, pub_r, piv_{u_r}, ID_{u_i}, pub_{u_i}$

Output: $\{m_1, m_2, m_3, ...m_n\}$

1. For i = 1 to n do
2. $T_i = \varrho(P_i, \widehat{pivu_r})$
3. Compute $H_i = \varrho(U_i, T_i, r_{u_r}, P_i, ID_{u_r}, piv_{u_r}, \triangle)$
4. $m_i = V_i \oplus H_i$
5. $\{m_1, m_2, ...m_n\}$
6. End do

A summary of the overall plan is given below. It demonstrates the relationship between KGC, users (both sender and receiver), and ASG. The operation order of the illustrated

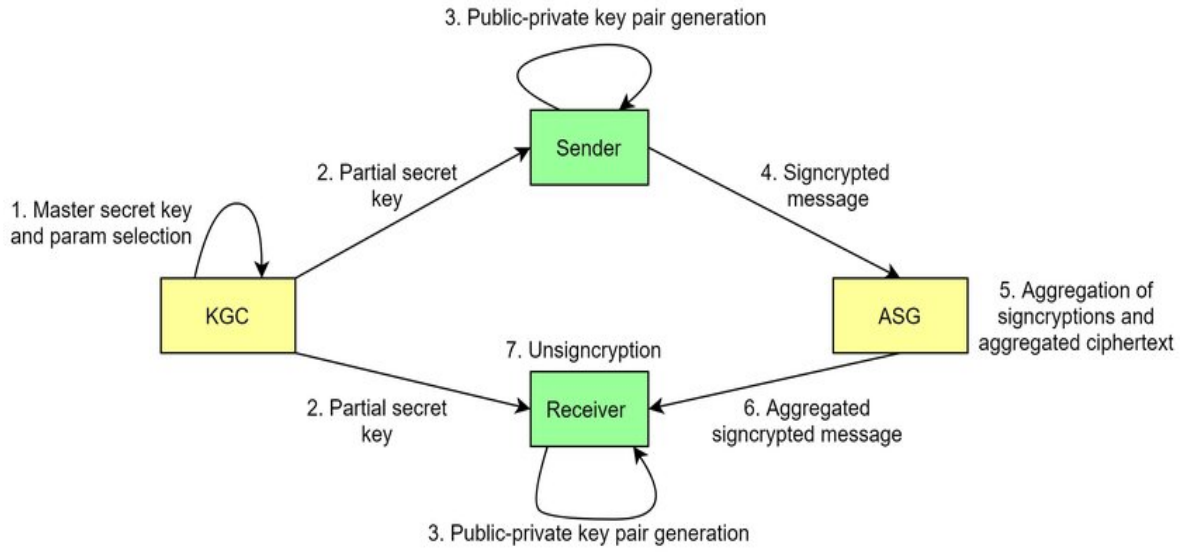


FIGURE 3.3: Sequence of operation among the framework modules: sender,receiver,KGC, and ASG

work is represented by the numbers.

Correctness of Certificateless Aggregated Signcryption scheme

The following equality proves the verifiable correctness of the proposed works. $\varnothing(W_i, g)$
$=\varrho(\sum_{i=1}^{n} W_i, g)$
$=\varrho(\sum_{i=1}^{n}(\widehat{piv_u} + ih_i + h_u Z_\triangle), g)$
$=\varrho((\sum_{i=1}^{n} h_1(msk, Q_i) \uparrow Extract(t + \delta t) + ih_i + i_u h_\Delta), g)$
$=\varrho(\sum_{i=1}^{n}(msk, Q_i) \prod_{i=1}^{n} \varrho(h_i, U_i)\varrho(h'_\Delta \cdot \sum_{i=1}^{n} piv_u)$

3.2.1 Experimental Results

In this part, the authors looked at how well the suggested plan worked. They compared it to other plans that already exist.

1. **Perfomance Metrics**

   The following metrics are used to assess the schemes performance.

   - **Throughput**

The correctly delivered number of message per unit time is called throughput. The unit of this measurement is bit per second (bps).

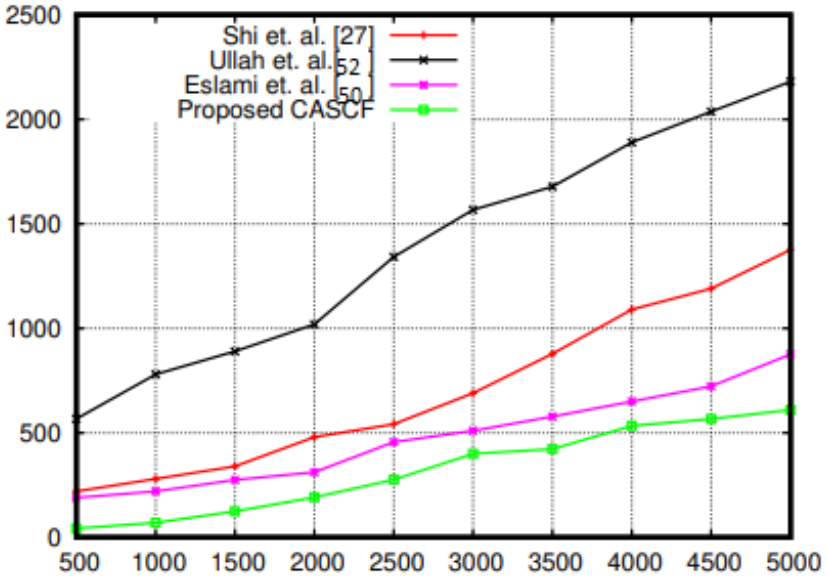


FIGURE 3.4: Number of Message and Throughput(kbps)

The experimental setup and results for a network using a CASCF (Certificateless Aggregate Signcryption scheme Framework). The comparative result is shown in Figure 3.4. The passage mentions the use of 5000 messages, varying message sizes, and comparing the performance with other schemes from references [29]-[53]-[51]. The CASCF scheme is reported to show better throughput, with improvements of 28.3%, 43.6%, and 17.9%.

- **Delay**

The networks round-trip time, is how long it takes for information to go from one place to another and then back again. This time is made up of different parts:

i. **Processing delay:**

This is the time it takes for the network devices (like computers or routers) to work on the data.

ii. **Queue delay:**

It is the time spent waiting in line (like people waiting their turn) to be processed by the network devices.

iii. **Transmission delay:**

This is how long it takes to send the data from one place to another through the network.

iv. **Propagation delay:**

This is the time it takes for the data to travel through the wires or airwaves from one point to another.

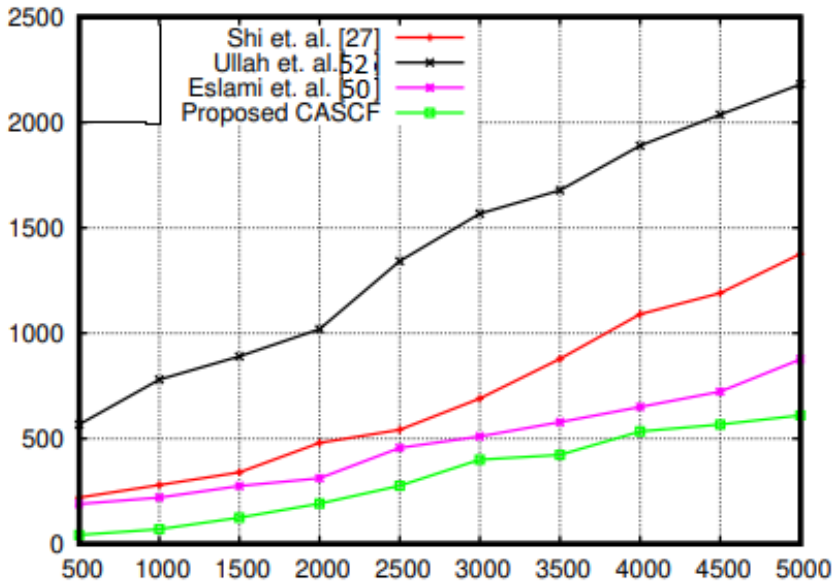


FIGURE 3.5: Number of Message and Delay(millisecond)

The size of the message does not affect the cryptographic schemes being t ested. The delay output is shown in Figure 3.5. The processing delay includes: Signcryption of the message, Aggregation, Receivers aggregate unsigncryption. Queuing delay, transmission delay, and propagation delay were excluded from the measurements.

These were assumed to be constant and the transmission channel was considered to be congestion free. CASCF (presumably one of the cryptographic schemes being tested) demonstrated a reduced processing delay compared to the other schemes. Certificateless signcryption in the experiment had an impact on the delay. This is likely because the delays associated with creating and verifying certificates were avoided in this approach. The approach mentioned in reference [51] employs a similar certificateless approach, resulting in a similar output. However, CASCF, due to specific reduction steps, produced even less dela. CASCF achieved a 25% reduction in delay compared to the other algorithms tested.

• **Energy Consumption**

The Internet of Things (IoT) includes small devices with limited power.
So security systems for IoT need to use as little energy as possible. We measure this using something called residual energy, which is shown as a percentage. This tells us how much energy is left in the device after it is been running for a while. It is provided a passage discussing the importance of energy consumption in an

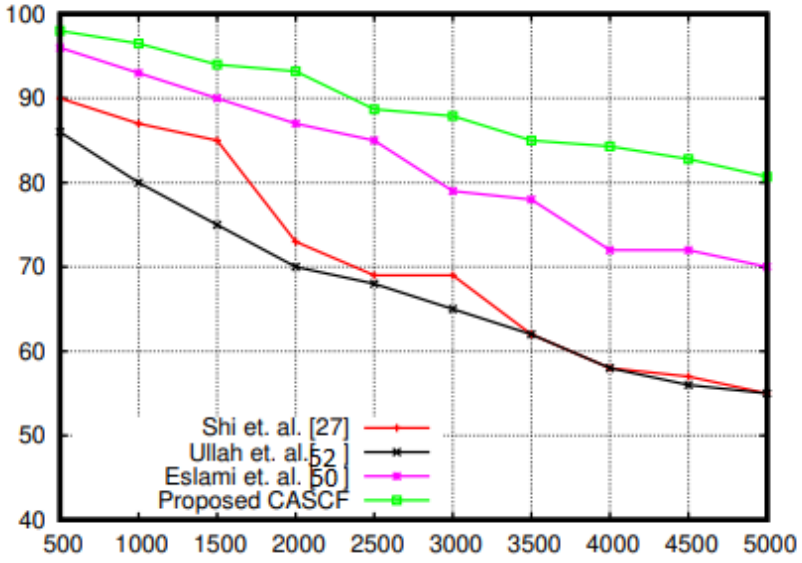


FIGURE 3.6: Residual energy comparison among CASCF and other approaches

IoT (Internet of Things) framework and comparing different algorithms in terms of their energy efficiency.The result is shown in Figure 3.6 The passage mentions that algorithms[53]-[51] consume 45% of the total energy on average, but [53] initially consumes more energy and [29] degrades more rapidly over time.

Additionally, it states that CASCF and the algorithm in [51] are more efficient due to their avoid-ance of certificates.CASCF is even more efficient, as it reduces energy consumption by 48%, 49.7%, and 15.6% compared to [29]-[53]-[51] respectively, by changing the key generation mechanism. particular attention to the space needed for certificates.

• **Memory Consumption**

This metric measures how much storage space is required for things like keys, intermediate data, and certificates. In situations where you are making comparisons, you should pay

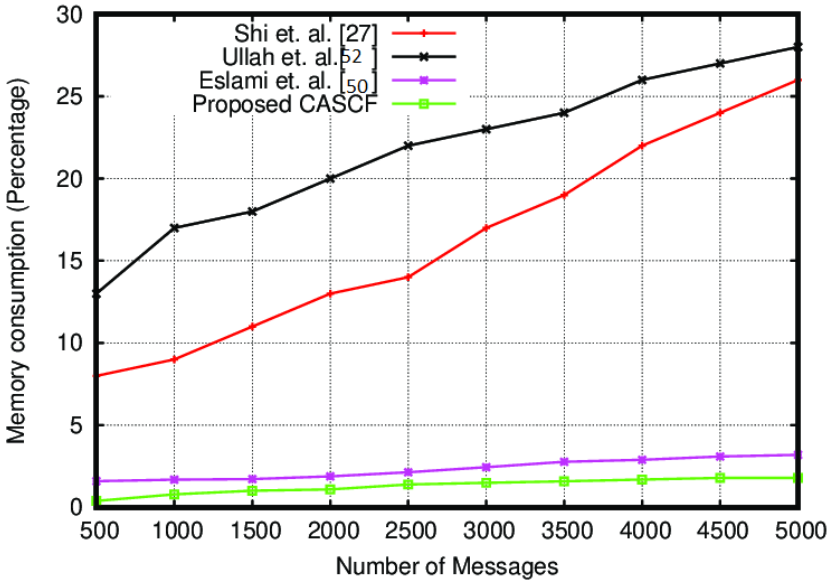


FIGURE 3.7: Memory Consumption

Memory plays a crucial role in any cryptographic process, and as the volume of messages grows, individual memory consumption naturally rises. However, a comparative analysis reveals that CASCF holds a distinct advantage in terms of memory efficiency. The amount of memory consumed in CASCF is notably lower compared to the other three algorithms, which exhibit memory usage that is 25%, 36%, and 18% higher, respectively. Consequently, when considering memory utilization, CASCF demonstrates its efficiency. The comparison result is shown in Figure 3.7

- **Complexity**

The individual operation complexity basis is a way to measure how complicated different schemes or plans are. When it comes to the Internet of Things (IoT) framework, it works better with simpler algorithms or methods.
In other words, for IoT, it is more effective to use straightforward and uncomplicated approaches.

The assessment of complexity has been divided into two components: computational complexity on the receiver side and communication complexity.
We adopted the same parameters and notation for this metric as outlined in [51].
Table 3.2 summarizes the notations, while Table 3.3 illustrates the comparison.
The comparison in Table 3.3 reveals that the communication complexity is lower compared to the schemes described in and [29]-[53]-[51].

Nevertheless, CASCF exhibits a comparable level of communication complexity.

However, in terms of sender-side complexity, CASCF tends to be higher.

Conversely, the computation complexity for the receiver and aggregator sides is the least among all methods.

Overall, CASCF proves to be efficient in terms of both computation complexity and communication cost.

Parameters for Measuring Complexity

Here are some important parameter.

Table 3.2: Paeameter

Parameter	Description
T_C^+	Time for additive cyclic group
T_C^*	Time for multiplication cyclic group
T_p	Time Pairing
T_h	time hash
T_{op}	Time operator bitwise

Camparision Complexity

This figure show the camparatively results.

3.3 Security Analysis

Here, the security analysis of CASCF with IoT infrastructure is described.It consist of Diffie-Hellman Problem and other security feature are required.

A. **Diffie-Hellman Problem (DHP)**

We have examined a decrease in CDH as if g^{an} is solvable from a given g^n then CDH problem is solvable.

Let $\mathcal{R}$ be an adversary that uses g^a for random integer a and output g^{a2} with probability P. As a result of $\mathcal{R}^{'}$ who receives $s = g^a$ and $t = g^b$ and works as follows. its input s,t and s.t on $\mathcal{R}^{'}$ and run R for n times. If $\mathcal{R}$ returns correct answer

TABLE 3.3: Comparison of complexity

	Sender side computation complexity	Receiver side computation complexity	Aggregation side computation complexity	Communication complexity
Shi et.al. [29]	$6T_C^*+4T_h+2T_p+T_C^+ +T_{op}$	(2n+1) $T_C^+ T_h$+(n+3) $T_p T_h$ + $nT_C^* T_{op}$	n($3T_h+T_p+2T_C^*$)	(n+1)$\|C_0\|$.(n+1)$\|C_1\|$
Ullah et.al.[52]	$8T_C^*+4T_h+3T_p+2T_C^+ + T_{op}$	(4n+1) $T_C^+ T_h$+(2n+3) $T_p T_h + nT_C^* T_{op}$	3(n + 1) T_h+(n + 1)T_p+$2T_C^*$	$(n+1)^2\|C_0\|.(n+1)^2\|C_1\|$
Eslami et .al[50]	$5T_C^*+3T_h+T_p+T_C^+ +T_{op}$	(2n+1) T_h+(2n+3) T_p+ $n(T_C^*+T_{op})$	(n+2) T_h+$4T_p$	(n+1)$\|C_0\|$
Proposed CASCF	$6T_C^*+4T_h+T_p+T_C^+ +T_{op}$	n($T_h+T_p+T_C^* + T_{op}$)	(n+1) T_h+$3T_p$	(n+1)$\|C_0\|$
n is the number of message. The complexity is calculated on the technical specification of the system as:4GB RAM,16GB storage based mobile device including laptop and mobile phone and then taken as average.				

every time then $\mathcal{R}'$ have $\mathcal{R} = g^{an}, \mathcal{P} = g^{bn} and \mathcal{Q} = g^{(a+b)^n}$. Its output as $n\sqrt{\frac{\mathcal{Q}}{\mathcal{R}.\mathcal{P}}}$ where $n\sqrt{}$ is the prime modulo p.If i understand correctly, you are asking about the probability of a calculation being correct when using a random generator 'g' and unknown values 'a' and 'b' become: P^n , where P = $\frac{\prod_{i=1}^{n} P(g)}{\Sigma_{i=1}^{n} \Pi a.\Pi b} \rightarrow 0$ this notation are used in Algrothim 6 step 5. This means that it confirms t he s uggested plan is very difficult to so lve us ing th e CD H pr oblem. In ot her wo rds, th e sc heme is designed to be extremely challenging for a specific type of problem called the CDH problem.

For Dicisional Bilinear Diffie Hellman, try to calculate the advantage for the adversary $\mathcal{R}$.

Its input (g, g^a, g^b, g^{ab}) and select a advantage of random g^{abc} in G.If a,b select a uniformly with random c in G, the correction of g^{abc} will find difficult.The result of this probability $\mathcal{R}$ is given as:
Adv($\mathcal{R}$) = $|P(Adv(g, g^a, g^b, g^{ab}) - P(Adv(g, g^a, g^b, g^c)|; g \leftarrow -e : G^n * G^n; a, b, c \leftarrow \mathbb{Z}^*_p$

In simpler terms, when it comes to systems that work within a certain efficient time frame (like the proposed scheme), they don't have an advantage.
This means the CASCF (which stands for Complete Active Space Configuration Interaction, a method in quantum chemistry) can not solve a particular problem

called DBDH. So, it is safe to say that DBDH remains unsolved using CASCF in this context.

"Now extending the CDH problem with a random oracle, the GDH validation is conducted as mentioned in [51]"

The attacker $\dot{R}$ employs the suggested scheme utilizing a master public key (mpk) with a generator g^a and selects a random integer $\mathcal{I} \leq qh_1$, where h_1 serves as the oracle, and q represents the maximum number of iterations in the oracle. Upon receiving a GDH tuple (g, g^a, g^b) in G_1 from adversary R, $\dot{R}$ transmits G_1, G_2, e, g, mpk to R.

With the restriction of only h_1 queries, $\dot{R}$ performs h_2 queries, verifying the DBDH condition and checking if $\varrho(U, P_u) = \varrho(U, g)$ holds true.

If confirmed, and the tuple exists with a value of h, $\dot{R}$ returns it; otherwise, it chooses a random h, updates itself, and communicates the changes to R.

Subsequently, A transmits identities of the users, public keys, messages, a forged ciphertext C^*, and some state information.

Each identity is chosen with equal probability from the set of n identities. Importantly, the aggregate verification process must validate the forged aggregated signcrypted message, ensuring $\dot{R}$ can compute g^{ab}. This affirms the security of the proposed scheme against the GDH assumption.

3.3.1 Unforgeability

A signature must have the capacity to be unforgeable. The signcryption technique must also be unforgeable because signcryption needs to implement both encryption and signatures.

Certificateless Aggregated signcrption (CLAS) Scheme has Unforgeability.
The proposed CASCF in signcryption mode is unforgeable against adaptive chosen message attacks. Two cases are considered here. A challenge-response game is initiated as mentioned in[54].A challenger $\mho$ the public paramerter and msk by use the setup algorithim 3.5.1.

Its input param to an adversary R^{-}. And the output $(ID^{'}_{s}, ID^{'}_{r})$.If it is assume that R^{-} is not extract partial private key then use the random number and time stamp with hash

function.Furthermore, the assumption is made that for the chosen message,$R^{'}$ is unable to use set keys or private key queries on $ID_s^{'}$. This implies that $R^{'}$ cannot gain unauthorized access to certain information.

Consequently, the output of aggregation verification is false, and $R^{'}$ is unable to proceed further. In an extensive scenario, if $R^{'}$ attempts to input a forged ciphertext $\bar{\Omega}$ in the aggregation, it is stated that $R^{'}$ cannot retract the key pairs.

This implies that even if $R^{'}$ manages to create a forged ciphertext, the challenger $\mho$ will still win the game because $\bar{\Omega}$ has already prevented $R^{'}$ from obtaining the partial private key or replacing the public keys.

3.3.2 Secrecy

The confidentiality of previously encrypted messages is unaffected by the disclosure of the encrypts private key, which is known as forward secrecy.

Certificateless Aggregated signcrption (CLAS) scheme has Forward Secrecy.

It is discussing a cryptographic scheme that employs a technique called Certificateless Aggregated signcrption scheme framework (CASSF). The scheme claims to provide forward secrecy, which means that even if the master secret key (msk) of the Key Generation Center (KGC) is compromised, the attacker won't be able to obtain the private key of a user.

This is achieved by generating the private key using both the master secret key (msk) and a user-specific random number i by using the algorithm 3.5.4, which is also kept secret. Since the attacker would need both the msk and the specific random number of the user to generate the private key, and the random number is private to the user, the generation of the private key becomes infeasible for the attacker.

Chapter 4

An Efficient Certificateless Generalized Signcryption Scheme based on Bilinear Mapping

In this chapter, we will talk about a special kind of security method called Certificateless Generalized Signcryption Scheme (CLGSC). First, we will clearly explain what CLGSC is. Next, we will describe the security ideas behind this new way of keeping information safe. Finally, we will introduce a new CLGSC plan or method.

4.1 Backgroud

Recall that, Signcryption is a type of security method that was suggested by Zheng in 1997 [9]. It is a way to do both encryption and creating a signature in just one step, which is faster than the usual method of doing the signature first and then the encryption.
After Zheng's idea, many other ways of doing signcryption have been suggested. In 2002, Malone-Lee [47] came up with the first signcryption method that uses identity (like a username) as a key. After that, more ways of doing identity-based signcryption have been suggested by different researchers.
The key escrow problem in identity-based cryptosystems led to the development of certificateless public key systems. Al-Riyami and Paterson [55] introduced this cryptographic

approach, where users private keys are generated by both a Key Generation Center (KGC) and the users themselves.

This innovation addresses the key escrow issue found in identity-based systems and avoids the complex certificate management problem seen in traditional public key systems [55]. In certificateless public key systems, users receive their private keys from a KGC and also contribute to the key generation process. This dual-generation eliminates the need for a central authority to hold all private keys, enhancing security.

Various certificateless signature and encryption schemes have been proposed, with the first certificateless signcryption introduced by Barbosa and Farshim [16] in 2003. Subsequent developments in certificateless signcryption followed, and the concept of generalized signcryption, introduced by Yiliang and Xiaoyuan [56] in 2006, became noteworthy.

Generalized signcryption is a special type of signcryption that provides both confidentiality and authentication simultaneously. Moreover, it allows obtaining either confidentiality or authentication alone, offering flexibility in security requirements.

Wang Xu-an et al.[13] presented the first security model for generalized signcryption and an improved version of generalized signcryption.

The progress in this field includes the proposal of the first ID-based vision by Lal and Kushwah [14].

Overall, these advancements aim to address security concerns, improve confidentiality and authentication, and simplify the management of cryptographic keys and certificates.

4.1.1 Globle parameter

The important global parameter of the proposed scheme are provided.

4.1.2 Certificateless Generalized Signcryption

A certificateless generalized signcryption scheme comprises five probabilistic polynomial-time algorithms, as defined below.

1 **Setup**:

A security system (PKG) uses this parameter to run a process and create two

TABLE 4.1: Global Parameter

Symbol	Description
$\mathbb{Z}_p^*$	Integer element
G_1	Cyclic additive group
G_2	Multiplicative group
ϱ	Bilinear map
g_1, g_2	Generator of group
D	private key
S	secret key
h_0, h_1, h_2, h_3, h_4	Hash functions
param	Public parameter
ID_s, ID_r	Identity of sender, Identity of receiver
sp	public key
m	Message
Δ	State information
Pub_s, Pub_r	Public key sender and receiver
Piv_s, Piv_r	Private key sender and receiver
$\mho$	Ciphertext
r	random integer

things: a master key and some global parameters called "params." The security system shares the global parameters with everyone but keeps the master key secret, not revealing it to anyone.

2 **Extract-Partial-Private-Key (ID_i, S, params)**:

Running the Extract-Partial-Private-Key algorithm with user identity ID_i, parameters, PKG runs the algorithm and returns a partial private key D.

3 **Set-User-Key (ID_i, D, params)**:

In simpler terms, when a user has an ID_i, a partial private key, and some parameters, they can use an algorithm. By running this algorithm, the user generates a public key *pub* associated with their identity and a secret key. The private key corresponding to this user is a combination of the secret key and the partial private key *piv*.

4 **Generalized Sincryption Scheme**:

This algorithm has 3 scenarios: signcryption, signature and encryption.

- **Signcryption**:

  If user s securely and authentically transmits the message m to user r, the input comprises (S_i, m, ID_r), resulting in the output $\mho = GSC(S_s, m, ID_r)$.

- **Signature**:

  If user s intends to sign a message m without defines reciver r, the input is represented as $((P_s, m, ID_\varnothing))$, where $(ID_\varnothing)$ signifies that the recipient is undefined. The resulting output is denoted as $(\mho = GSC(S_s, m, ID_\varnothing))$.

- **Encryption**:

  If an individual intends to transmit a confidential message, the given input comprises $(S_\varnothing, m, ID_r)$, where S_r represents the private key associated with $ID_\varnothing$. The resulting output is $\mho = GSC(S_\varnothing, m, ID_r)$.

5 **UnGeneralized Signcryption Scheme**:

If $\mho$ is valid, receiver r decrypts the ciphertext and returns the message, and optionally the signature on m by a. If $\mho$ is not valid, r returns $\perp$ to indicate failure.

4.1.3 Certificateless Generlization Signcryption Scheme (CLGSC)

We proposed the Certificateless Generalized Signcryption Scheme (CLGSC) as following.
Setup:
Provided a security parameter, the KGC select two groups G_1 and G_2 of prime order p, two random generator g_1 and g_2 of G_1 such that $g_1 \neq g_2$,
and a bilinear map $\varrho : G_1 \times G_1 \to G_2$.
Compute $\mathrm{t} = \varrho(G, g_2) \in G_1$,
define 5 hash functions as

$h_0 : \{0,1\}^* \to {Z_p}^*$,
$h_1 : G_2 \times \{0,1\}^* \to {\mathbb{Z}^*}_p$,
$h_2 : \{0,1\}^m \times G_2 \times \{0,1\}^* \times G_2 \times G_1 \times \{0,1\}^* \to {\mathbb{Z}^*}_p$,
$h_3 : {\mathbb{Z}^*}_p \times \{0,1\}^* \to {\mathbb{Z}^*}_p$,
$h_4 : G_2 \times G_2 \times G_2 \times \{0,1\}^* \to \{0,1\}^{d_1+d_2}$,

where d_1, d_2 denote the number of bits to represent G_1 and ${Z_p}^*$ elements respectively. KGC chooses random s$\in {\mathbb{Z}^*}_p$ as master secret key and set $pub = sp$. KGC publishes the system parameters as

$$\{G_1, G_2, G, g_2, pub, \varrho : G_1 \times G_1 \to G_2, p, h_0, h_1, h_2, h_3, h_4\}$$

Extract Partial Private Key:

The calculation of the partial private key for the user identified as ID_i involves the Key Generation Center (KGC) determining q_i as $h_0(ID_i)$, and computing the formula $D_i = (q_i + s)^{-1}Q$.

Set user key:

The user identified by ID_i randomly selects $t_i \in \mathbb{Z}_p^*$ and defines their private key as $piv_i =< t_i, D_i >$, along with a public key $pub_i =< pub_{i1}, pub_{i2} >$, where $pub_{i1} = g^{t_i}$ and $pub_{i2} = t_i F_i$, with $F_i = (q_i + s)p$.

Generalization of signcryption scheme:

This algorithm encompasses three scenarios: signcryption, signature, and encryption.

Signcryption:

For a given message, with sender identity s and receiver identity r, a performs the following steps:

1 A chooses random r, r$\in \mathbb{Z}_p^*$,computes $\alpha = g^r$;

2 Computes H = $h_1(\alpha, \delta, ID_s, ID_r)$ and $H_2 = h_2(\delta, \alpha, H, ID_s, pub_{s1}, pub_{s2}, ID_r)$;

3 W= $\dfrac{r}{t_a + H_2} D_s$;

4 Computes n = $h_3(h, ID_s, ID_r) \bigoplus \delta || \alpha$;

5 Computes $H_4 = h_4(g^r, (pub_{r1})^r, piv_{r1}, ID_r)$

6 Compute $y_1 = r(q_r + s)P$ and $y_2 = H_4 \bigoplus h || W$.

7 Return ciphertext $\mho = (n, y_1, y_2, ID_r)$.

Signature:

Given message m, sender identity s, A operates the following steps:

1 Step 1 is the same as in signcryption;

2 Computes H = $H_1(\alpha, \delta, ID_s, ID_r)$ and $h_2 = H_2(\delta, \alpha, h, ID_s, pub_{s1}, pub_{r2}, 0)$;

3 W= $\dfrac{r}{t_s + h_2} D_s$;

4 Sets n = $m || \alpha$;

5 Computes $H_4 = 0$;

6 Compute $y_1 = 0$ and $y_2 = H_4 \bigoplus h||W$.

7 Return ciphertext $\mho = (n, y_1, y_2, 0)$.

Encryption:

Given message m, receivers identity r, someone operates the following steps:

1 A chooses random r, r $\in \mathbb{Z}_p^*$,computes $\alpha = g^r$;

2 Computes H = $h_1(\alpha, \delta, 0, ID_r)$ and $H_2 = h_2(m, \alpha, H, 0, 0, 0, ID_r)$;

3 Computes n = $h_3(H, 0, ID_r) \bigoplus m||\alpha$;

4 Computes $H_4 = h_4(g^r, (pub_{r1})^r, pub_{r1}, ID_r)$

5 Compute $y_1 = i(q_r + s)P$ and $y_2 = H_4 \bigoplus H||0$.

6 Return ciphertext $\mho = (n, y_1, y_2, ID_r)$.

Un Certificateless Generlized Sincryption scheme:

given $\mho$, a receivers identity r, operates the following steps:

1 Computes E = $\varrho(y_1, D_r)$ and $(E)^{t_i}$(if there is no receivers identity, then $D_r = 0$, and E =1);

2 Sets $H_4 = h_4(D, (D)_i^t, piv_{r1}, ID_r)$; $(if D_r = 0$, then $H_4 = 0)$;

3 Computes $H||W = y_2 \oplus H_4$;

4 If $ID_r \neq 0$, computes $m||\alpha = n \oplus h_3(H, ID_s, ID_r)(if Z = 0$, then $ID_s = 0)$;

5 If $ID_r \neq 0$, computes $H_2 = h_2(m, \alpha, H, ID_s, pub_{s1}, pub_{s2}, ID_r)$ (if $Z = 0$, then $ID_s = 0$ and $pub_{s1} = pub_{s2} = 0)$;

6 If W $\neq 0$, then r accepts m if and only if $H = h_1(\alpha, m, ID_s, ID_r)$ and $\varrho(W, pub_{s2} + H_2(q_s + s)g_1) = \alpha$ holds. otherwise accepts m if and only if H = $h_2(\alpha, m, 0, ID_r)$.

4.1.4 Correction

The correctness of our Certificateless Generlized Sincryption scheme (CLGSC) is below:
If $W \neq 0$,
then

$$\begin{aligned}
\varrho(W, pub_{s2} + H_2(q_s + s)g_1) =& \varrho(r(t_s)) + H_2)^{-1} D_s, \\
t_s(q_s + s)g_1 + H_3(q_s + s)g_1) =& \varrho(r(t_s + H_2)^{-1}(q_s + s)^{-1} g_2, \\
(t_s + H_2)(q_s + s)g_1) =& \varrho(g_1, g_2)^r, \\
=& \alpha.
\end{aligned}$$

Chapter 5

Conclusion

In the current study, a solution for IoT security has been demonstrated, employing aggregate signcryption to improve network performance. The scheme utilizes a bilinear map, and the key generation process incorporates timestamp to ensure key freshness.

The architecture employs a group of nodes as an aggregate signature generator, specifically designed for optimal use in the fog layer of the IoT infrastructure. Performance evaluation is conducted based on throughput, delay, energy consumption, and memory consumption, with results compared to existing schemes. Scheme complexities are also compared, with the comparative analysis indicating the efficiency of the proposed scheme for IoTs. Ad-ditionally, a security analysis validates the accomplishment of the security objectives of the work.

Notably, the sender side exhibits higher computation complexity, identified as a potential area for future improvement. In this thesis, we initially provide the formal definition and security model for certificateless generalized signcryption.

Additionally, we introduce a specific certificateless generalized signcryption scheme built upon bilinear pairing. Following that, we furnish the security proof for our proposed scheme.

Bibliography

[1] B. A. Forouzan and D. Mukhopadhyay, *Cryptography and network security*, vol. 12. Mc Graw Hill Education (India) Private Limited New York, NY, USA:, 2015.

[2] R. L. Rivest, "The rc5 encryption algorithm," in *International Workshop on Fast Software Encryption*, pp. 86–96, Springer, 1994.

[3] H. Delfs, H. Knebl, H. Delfs, and H. Knebl, "Symmetric-key cryptography," *Introduction to Cryptography: Principles and Applications*, pp. 11–48, 2015.

[4] W. G. Barker, *Introduction to the analysis of the Data Encryption Standard (DES)*. Aegean Park Press, 1991.

[5] M. A. Musa, E. F. Schaefer, and S. Wedig, "A simplified aes algorithm and its linear and differential cryptanalyses," *Cryptologia*, vol. 27, no. 2, pp. 148–177, 2003.

[6] W. Diffie and M. E. Hellman, "New directions in cryptography," in *Democratizing Cryptography: The Work of Whitfield Diffie and Martin Hellman*, pp. 365–390, 2022.

[7] J. Groth, "Cryptography in subgroups of," in *Theory of Cryptography Conference*, pp. 50–65, Springer, 2005.

[8] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE transactions on information theory*, vol. 31, no. 4, pp. 469–472, 1985.

[9] Y. Zheng, "Digital signcryption or how to achieve cost (signature & encryption) cost (signature)+ cost (encryption)," in *Advances in Cryptology—CRYPTO'97: 17th Annual International Cryptology Conference Santa Barbara, California, USA August 17–21, 1997 Proceedings 17*, pp. 165–179, Springer, 1997.

[10] Y. Han, X. Yang, P. Wei, Y. Wang, and Y. Hu, "Ecgsc: elliptic curve based generalized signcryption," in *International conference on ubiquitous intelligence and computing*, pp. 956–965, Springer, 2006.

[11] Y. Han, "Generalization of signcryption for resources-constrained environments," *Wireless Communications and Mobile Computing*, vol. 7, no. 7, pp. 919–931, 2007.

[12] W. J. Caelli, E. P. Dawson, and S. A. Rea, "Pki, elliptic curve cryptography, and digital signatures," *Computers & Security*, vol. 18, no. 1, pp. 47–66, 1999.

[13] X. A. Wang, X. Yang, and Y. Han, "Provable secure generalized signcryption," *Cryptology EPrint Archive*, 2007.

[14] S. Lal and P. Kushwah, "Id based generalized signcryption," *Cryptology ePrint Archive*, 2008.

[15] M. H. Au, Y. Mu, J. Chen, D. S. Wong, J. K. Liu, and G. Yang, "Malicious kgc attacks in certificateless cryptography," in *Proceedings of the 2nd ACM symposium on Information, computer and communications security*, pp. 302–311, 2007.

[16] M. Barbosa and P. Farshim, "Certificateless signcryption," in *Proceedings of the 2008 ACM symposium on Information, computer and communications security*, pp. 369–372, 2008.

[17] G. Yu, X. Ma, Y. Shen, and W. Han, "Provable secure identity based generalized signcryption scheme," *Theoretical Computer Science*, vol. 411, no. 40-42, pp. 3614–3624, 2010.

[18] C. Zhou, "An improved lightweight certificateless generalized signcryption scheme for mobile-health system," *International Journal of Distributed Sensor Networks*, vol. 15, no. 1, p. 1550147718824465, 2019.

[19] P. Kushwah and S. Lal, "Efficient generalized signcryption schemes," *Cryptology EPrint Archive*, 2010.

[20] H.-F. Ji, W.-B. Han, and L. Zhao, "Identity-based generalized signcryption in standard model," *Jisuanji Yingyong Yanjiu*, vol. 27, no. 10, 2010.

[21] P. Kushwah and S. Lal, "An efficient identity based generalized signcryption scheme," *Theoretical Computer Science*, vol. 412, no. 45, pp. 6382–6389, 2011.

[22] Y. Yuan, "Security analysis of an enhanced certificateless signcryption in the standard model," *Wireless Personal Communications*, vol. 112, pp. 387–394, 2020.

[23] D. Dharminder and D. Mishra, "Understanding signcryption security in standard model," *Security and Privacy*, vol. 3, no. 3, p. e105, 2020.

[24] S. Ullah, X.-Y. Li, and L. Zhang, "A review of signcryption schemes based on hyper elliptic curve," in *2017 3rd International Conference on Big Data Computing and Communications (BIGCOM)*, pp. 51–58, IEEE, 2017.

[25] G. Kumar, R. Saha, M. K. Rai, R. Thomas, and T.-H. Kim, "A lattice signcrypted secured localization in wireless sensor networks," *IEEE Systems Journal*, vol. 14, no. 3, pp. 3949–3956, 2020.

[26] S. Belguith, N. Kaaniche, M. Hammoudeh, and T. Dargahi, "Proud: Verifiable privacy-preserving outsourced attribute based signcryption supporting access policy update for cloud assisted iot applications," *Future Generation Computer Systems*, vol. 111, pp. 899–918, 2020.

[27] X. Fan, T. Wu, Q. Zheng, Y. Chen, M. Alam, and X. Xiao, "Hse-voting: A secure high-efficiency electronic voting scheme based on homomorphic signcryption," *Future Generation Computer Systems*, vol. 111, pp. 754–762, 2020.

[28] N. Eltayieb, R. Elhabob, A. Hassan, and F. Li, "A blockchain-based attribute-based signcryption scheme to secure data sharing in the cloud," *Journal of Systems Architecture*, vol. 102, p. 101653, 2020.

[29] Y. Shi, J. Han, X. Wang, J. Gao, and H. Fan, "An obfuscatable aggregatable signcryption scheme for unattended devices in iot systems," *IEEE Internet of Things Journal*, vol. 4, no. 4, pp. 1067–1081, 2017.

[30] J. Huifang, H. Wenbao, and Z. Long, "Certificateless generalized signcryption," *Cryptology ePrint Archive*, 2010.

[31] M. S. Iqbal, S. Singh, and A. Jaiswal, "Symmetric key cryptography: Technological developments in the field," *International Journal of Computer Applications*, vol. 117, no. 15, 2015.

[32] W. Stallings, *Cryptography and network security, 4/E.* Pearson Education India, 2006.

[33] H. Wu, "The hash function jh," *Submission to NIST (round 3)*, vol. 6, 2011.

[34] P. Gupta and S. Kumar, "A comparative analysis of sha and md5 algorithm," *architecture*, vol. 1, no. 5, 2014.

[35] C. Xiao-hui and D. Jian-zhi, "Design of sha-1 algorithm based on fpga," in *2010 Second International Conference on Networks Security, Wireless Communications and Trusted Computing*, vol. 1, pp. 532–534, IEEE, 2010.

[36] N. Sklavos and O. Koufopavlou, "On the hardware implementations of the sha-2 (256, 384, 512) hash functions," in *Proceedings of the 2003 International Symposium on Circuits and Systems, 2003. ISCAS'03.*, vol. 5, pp. V–V, IEEE, 2003.

[37] S.-j. Chang, R. Perlner, W. E. Burr, M. S. Turan, J. M. Kelsey, S. Paul, and L. E. Bassham, "Third-round report of the sha-3 cryptographic hash algorithm competition," *NIST Interagency Report*, vol. 7896, p. 121, 2012.

[38] B. A. Forouzan, *Cryptography & network security.* McGraw-Hill, Inc., 2007.

[39] J. B. Fraleigh, *A first course in abstract algebra.* Pearson Education India, 2003.

[40] S. R. Buss, "An introduction to proof theory," *Handbook of proof theory*, vol. 137, pp. 1–78, 1998.

[41] C. J. Benvenuto, "Galois field in cryptography," *University of Washington*, vol. 1, no. 1, pp. 1–11, 2012.

[42] K. S. McCurley, "The discrete logarithm problem," in *Proc. of Symp. in Applied Math*, vol. 42, pp. 49–74, USA, 1990.

[43] K. Damasceno, A. de Oliveira, L. de Castro, *et al.*, "Alternative n-bit key data encryption for block ciphers," in *Anais do XIX Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais*, pp. 409–414, SBC, 2019.

[44] D. Mahto and D. K. Yadav, "Performance analysis of rsa and elliptic curve cryptography.," *Int. J. Netw. Secur.*, vol. 20, no. 4, pp. 625–635, 2018.

[45] J. H. Silverman, *The arithmetic of elliptic curves*, vol. 106. Springer, 2009.

[46] R. C. Merkle, "A certified digital signature," in *Conference on the Theory and Application of Cryptology*, pp. 218–238, Springer, 1989.

[47] J. Malone-Lee, "Identity-based signcryption," *Cryptology ePrint Archive*, 2002.

[48] J. Bethencourt, "Intro to bilinear maps," 2015.

[49] W. J. BUCHANAN, M. K. RAI, G. GEETHA, and R. THOMAS, "Cascf: Certificateless aggregated signcryption framework for internet-of-things infrastructure,"

[50] T.-H. Kim, G. Kumar, R. Saha, M. Alazab, W. J. Buchanan, M. K. Rai, G. Geetha, and R. Thomas, "Cascf: Certificateless aggregated signcryption framework for internet-of-things infrastructure," *IEEE Access*, vol. 8, pp. 94748–94756, 2020.

[51] Z. Eslami and N. Pakniat, "Certificateless aggregate signcryption: Security model and a concrete construction secure in the random oracle model," *Journal of King Saud University-Computer and Information Sciences*, vol. 26, no. 3, pp. 276–286, 2014.

[52] S. Yadav and G. Howells, "Analysis of icmetrics features/technology for wearable devices iot sensors," in *2017 Seventh International Conference on Emerging Security Technologies (EST)*, pp. 175–178, IEEE, 2017.

[53] S. Ullah, F. Russo, L. Marcenaro, and B. Rinner, "Aggregate-signcryption for securing smart camera iot applications," in *2018 Global Internet of Things Summit (GIoTS)*, pp. 1–6, IEEE, 2018.

[54] B. Zhang, Z. Jia, and C. Zhao, "An efficient certificateless generalized signcryption scheme," *Security and Communication Networks*, vol. 2018, 2018.

[55] S. S. Al-Riyami, K. G. Paterson, *et al.*, "Certificateless public key cryptography," in *Asiacrypt*, vol. 2894, pp. 452–473, Springer, 2003.

[56] Y. Han and X.-Y. Yang, "New ecdsa-verifiable generalized signcryption," *CHINESE JOURNAL OF COMPUTERS-CHINESE EDITION-*, vol. 29, no. 11, p. 2003, 2006.