Hybrid Identity and Credential Management in Modern Internet

Final Report

HAIDA LU JINXUAN CHEN QINGLIN ZOU
SIWEI ZHANG YUHAN MA ZIYU CHENG

Master of Science Thesis

Communication Systems
School of Information and Communication Technology
KTH Royal Institute of Technology
Stockholm, Sweden

Panos Papadimitratos, Cihan Eryonucu

Abstract

Internet security is becoming more and more important with the rapid development of information technology. However, traditional public key cryptography schemes introduce a challenging problem; user anonymity. In addition, the security of digital certificate authentication based on public key cryptography has been challenged. Traditional schemes use digital certificates issued by a certification authority to prove that the identities between entities in a data exchange system are legit. One possible problem is that if a legitimate certificate is stolen by a malicious user, it can easily compromise the security of the entire system. Besides, users within the system, identities are non-anonymous on the DS side. To solve the non-anonymity and security drawbacks of public key cryptography schemes at the same time, we propose a hybrid scheme based on the group key cryptography scheme. The group scheme ensures the anonymity of users as much as possible, while the hybrid scheme in this paper strengthens its functions. By adding another layer of authentication to the group key method, the certificate security drawback can be effectively solved. We compared the performance of the hybrid scheme with that of the other two schemes. The hybrid scheme is more efficient and secure than the group encryption scheme. Finally, we developed a mobile application based on this scheme to collect data and draw a data heat map in Stockholm.

Contents

List of Acronyms and Abbreviations

MC	Mobile Client
GM	Group Manager
CA	Certification Authority
LTCA	Long-Term Certification Authority
PCA	Pseudonym Certification Authority
gpk	Group Public Key
gsk	Group Signing Key
PK	Public Key
PR	Private Key
IdP	Identity Provider
PKI	Public Key Infrastructure
PKC	Public Key Cryptography
GKC	Group Key Cryptography

Chapter 1

Introduction

Network security has always been an important topic in the field of communication systems. Recent advances in sensoring, computing, and networking have paved the way for emerging mobile applications such as mobile payments and cloud services. Those require extensive authentication operations to ensure user authentication, integrity, and non-repudiation.

Traditional authentication is based on public key infrastructure (PKI), which uses public key cryptography. With the help of a certification authority (CA), certificates issued by different parties through the CA can quickly complete the verification steps. PKI system divides keys into public and private keys and integrates authentication and certificate management by combining CAs' certificates.

However, the PKI scheme has security drawbacks. Certificates issued by CAs are usually long-term, and all users using non-expired certificates in the PKI system can connect via the CAs. If a user is attacked and the certificate is stolen soon after it is obtained from the CA, the attacker can easily threaten the security of the entire system.

Recent studies enhanced its security by modifying the original PKI scheme and adding new structures. Based on CAs' certificates, a short-term identity management organization is added to increase the depth of the system, thus avoiding the security risks introduced by relying on long-term certificates.

1.1 Organization

This project is organized as a six-person project, building upon previously published work.

Through relevant literature, we summarize the methods and design our own security-management system. After that, we deploy the system on a client and evaluate its performance. All sub-parts are carried out according to the timetable.

1.2 Problem description

This project focus on the user privacy problem introduced by long-term certificates. Since for a CA it would be easy to trace other activities linked to the same user. So after we ensure the routine processing of the group signature scheme, we should let the users sign their own pseudonymous using gsk. The gpk authentication process also needs to be done properly. In this project, we investigate the utilization of anonymous authentication in group signature schemes.

1.3 Problem context

In the first part, we performed a literature review for traditional PKIs and privacy-preserving identity and credential management systems, and understand privacy issues of the former and how they are addressed with the latter ones.

For the next part, we implemented a hybrid identity and credential management system, integrating traditional PKI-enabled and group signature schemes. We also deployed a mobile client application (for android phones) that collects data using its sensors and authenticate their messages with group signature schemes or hybrid schemes.

At last, we performed the analysis and evaluation of the performance of hybrid schemes with experiments on client applications.

1.4 Structure of this project

Chapter 1 describes the problem and its context. The next chapter 2 provides the background necessary to understand the problem and related literature reviews for this project.

Chapter 3 describes the system architecture and methods used in this project. The final solution, a hybrid security system with applications deployed on mobile equipment is proposed in it. Chapter 4 shows the performance analysis with the experiment results. After doing necessary evaluations, we deploy this system in mobile applications and draw heat maps, which are shown in Chapter 5.

Finally, Chapter 6 concludes our project and gives suggestions for future work.

Chapter 2

Background

2.1 Public Key Cryptography

Public Key Cryptography, also known as asymmetric key cryptography, refers to the encryption method consisting of a pair of corresponding unique keys (i.e., the public key and private key). It solves the issue and management of keys and is the core of the business password system. In a public key encryption system, the private key is not public, but the public key is public. If the decryption key is public, the information encrypted with the private key can be decrypted with the public key, which the client uses to verify that the data or file published by the private key holder with integrity and authenticity. The recipient knows that the information comes from someone who has the private key. This is called a digital signature, and the public key is in the form of a digital certificate.

The main purpose of a digital signature is to indicate that the information sent has not been modified by a third party. On the sender side, we often use the hash algorithm to generate the digest from the information. The private key is used to encrypt it to obtain the encrypted digest, which is called a digital signature. Finally, the digital signature and the information are sent to the receiver together.

2.1.1 Drawback of PKC

Main drawback of the PKC scheme is that linkability exists among the users. For a user, since it uses a single key, the information encrypted by the key can be proved to be from the user, which means the user's identity is not anonymous to the server. In some cases, such as when collecting private data, this defect may bring hidden dangers.

In the actual system deployment, LTCA is often used to verify the identity of all entities in the system. This method is realized by LTCA issuing digital certificates. However, most PKC scheme doesn't check this certificate at every receiving process. Certificates issued by CAs are usually long-term, and all users using non-expired certificates in the PKI system can connect via CAs. If a user is attacked and the certificate is stolen soon after obtaining it from the CA, the attacker can easily threaten the security of the entire system.

2.2 Group Key Cryptography

Group Key Cryptography (GKC) provides security to all the members in a working group, including integrity, authentication, and non-repudiation. GKC is an extension of PKC in the process of group encryption. The implement of GKC mostly relies on Group signature primitive.

2.2.1 Group Signature

Group Signature primitive, GS, is a powerful and well-studied primitive that allows members of a group to sign messages on behalf of the group in an anonymous way. That means a verifier of a group signature is assured that it was signed by a valid member of the group, but it does not learn anything about the identity of the signer, or even whether two signatures stem from the same user. That is an important building block for privacy-enhancing applications, e.g., enabling user data to be collected in authenticated form while preserving the user's privacy.

This self-signing process makes group signatures highly suited whenever data is collected that needs to be authenticated while, at the same time, the privacy of the data sources must be respected and preserved. In particular, when data is collected from users, the protection of their privacy is of crucial importance.

When aiming to implement such techniques for privacy and data protection, one needs to find a good balance with utility though: data gets collected to be analyzed and to generate new insights. For these processes, it is usually necessary to know the correlation among different data events, as they carry a crucial part of the information. For instance, when a group of users measures and upload their blood pressure via wearable activity trackers, several high-value measurements are not critical when they are distributed over many participants, but might be alarming when originating from a single user.

Often the exact purpose of the data might not be clear at the point of data collection. In fact, given the rapid advancements in machine learning and the ubiquitously available and cheap storage, data collectors tend to gather large amounts of data at first, and only use small subsets for particular applications as they arise.

Ideally, the data should be collected and stored in authenticated and unlikable form, and only the particular subsets that are needed later on should be correlated in a controlled and flexible manner [1].

2.2.2 Drawback of GKC

Although linkability is highly avoided during designing, there still exists some kind of linkability in a real deployed GKC system. To address the tension between privacy and utility, group signatures often have built-in measures that control the linkability of otherwise anonymously authenticated information. Interestingly, despite the long line of work on this subject, none of the solutions provides the functionality to cater to the flexibility needed in practice: They either recover linkability in a privacy-invasive way or offer control only in a static manner.

Another drawback of GKC is efficiency. To ensure connectivity as much as possible, each time the user uploads data, it uses a different encryption key to sign, which means that the server may need to perform multiple authentications when receiving data, which consumes more time than the traditional cryptography scheme.

2.3 Related work

This paper [2] is the theoretical basis for the structure of this project. MOBILE crowd sensing (MCS) is currently widely used to collect information through users' devices. However, there is a significant risk of indirectly inferring the daily routines or habits of users participating in MCS applications. The authors propose a secure and accountable MCS system in this paper, which preserves user privacy, and enables the provision of incentives to the participants. The system is resilient to abusive users and guarantees privacy protection even against multiple misbehaving and intelligent MCS entities (e.g servers). These performance enhancements are achieved by adding new security mechanisms and verification steps based on traditional systems. The authors introduce group management (GM), pseudonym certification authority (PCA), reporting service, and revocation authority (RA) additionally and set established verifying processes through the

system. Through this system, each user can participate in the data collection task anonymously and get a receipt from the system to get rewards. System entities trust each other, and a single entity does not know all information about users. According to the user's behavior, the system can expel malicious users from the system. In addition, the performance of the system is analyzed and simulated.

The authors studied vehicular communication (VC) systems in this paper [3]. The development of the vehicle communication system is mainly to improve the safety and efficiency of transportation. Vehicle-to-vehicle communication, especially frequent cooperative awareness messages or safety beacons, has been considered a major method in the past few years. The author considers transportation safety and efficiency applications, as they are significant features of VC systems (compared with other mobile computing systems). This paper provides a framework to analyze the impact of a given processing load on node performance and evaluate the effectiveness of applications. The author proposes Hybrid Pseudonym Scheme based on Baseline Pseudonym Scheme and Group Signature Scheme and conducts modeling and simulation of the VC system, which proves that the performance of the Hybrid Pseudonym Scheme is better in most cases. This article was published earlier (2010). In the concept of the hybrid scheme, the author tried to introduce the method of multiple verifications and self-verifications, which is considered to be the early concept of subsequent works.

This article [4] is a practical application of inter-vehicle communication security based on literature [5]. If a vehicle runs out of pseudonyms and cannot refill its pseudonym pool, it will pose a threat to overall security. One solution is the on-the-fly generation of pseudonyms,e.g the vehicle uses a certificate for self-authentication. Based on the above points, the author proposes a new solution, RhyTHM, to keep the vehicle running when disconnected without compromising privacy: vehicles with valid pseudonyms help others enhance their private pseudonyms and align life cycles by randomly adding them to use dynamic self-authentication. The scheme enhances the privacy of disconnected users through reasonable computing costs. Compared with the above literature, the scheme proposed in this article is closer to our project, which uses the user's self-signature rather than PCA.

Based on the public-key cryptography invented by Diffie and Hellman, an anonymous credential system was introduced. In an anonymous credential system, the user will be able to verify themselves with credentials without revealing their identity. Most importantly, the user will also be able to get the credentials anonymously. This complicates the case as the normal way of getting credentials requires the user to reveal his identity. To avoid that, the user will

send the hash of its identity to identify itself. A signature scheme based on those requirements can also be extended to the case where there are several blocks of the message instead of a single block of message [6].

Dan and Hovav used a short group signature of fewer than 200 bytes, which guarantees the security of the group signature on the bilinear pair-based Decision Linear and SDH (Strong Diffie-Hellman) [7]. A scheme constructed on the assumption of SDH has the same properties as Strong-RSA and is more compact. In contrast, the use of bilinear mappings ensures that each group is added to the group signature, and the total length is shorter. This group signature is smaller than 200 bytes and does not require bilinear computation for a generation but does require verification. The construction method transforms the ZKPK (Zero-Knowledge Proof of Knowledge) into a group signature scheme using Fiat-Shamir. As a result, the group signature is guaranteed to be correct, fully anonymous, and fully traceable, with a revocation mechanism and exculpability features. They also then delved into the VLR (Verifier-Local Revocation) scheme [8]. VLR is based on short group signatures, i.e., using bilinear groups, and its security is guaranteed. They designed an RL (Revocation List) to implement this algorithm, which contains the revoked user's token. When revoking, the user only needs to send a revocation message to the signature verifier. Still, the signature needs to be verified before and after the revocation, each time in conjunction with the user's private key. The individual user site website then checks the message's authenticity for revocation (time and RL).

The authors focus on dynamic accumulators that can be undone in anonymous mode [9]. The dynamic accumulator allows inputs to be added or removed dynamically so that the cost of adding or subtracting is independent of the number of accumulated values. This scheme is to incorporate the accumulator's public key into the group manager's public key, and the accumulator's secret trapdoor into the associated key. Whenever a user joins the group, the group manager will issue him a membership certificate. A component of this certificate is the prime number e, which is added to the accumulator when a user is added and removed when the user's privileges are revoked. This scheme relies on a strong RSA assumption.

This paper designed a credential system that allows users to authenticate n times anonymously within a period [10]. This system uses group signatures and anonymous credentials because both can prove that a piece of data comes from an authorized source without revealing the identity of that particular source. To realize the anonymity and incontestability of honest users and hope to detect a lot of misleading information from rogue, the author imitates the practice of e-cash, each participant obtains a set of e-tokens from the dispenser. The user will present

an e-token to the validator to authenticate themselves, but the dispenser will automatically refresh every period. This article provides the basic and extended framework of this credential system.

The LTCA is responsible for the registration of the user and issuing LTC (long-term certificate). The user will first interact with the LTCA to get involved in the process. In the registration phase, the LTCA will register users within its domain and keep track of their identities. At the bootstrapping phase, the user will get intra-domain and inter-domain information through Lightweight Directory Access Protocol (LDAP) [11]. When a user travels to a different domain, it will first interact with the previous LTCA to get a foreign ticket without revealing which domain it's transferring to. Then the user will use the foreign ticket to interact with the LTCA in the current domain to make new registrations.

In [12], a new group signature scheme is proposed, satisfying the security definition by Bellare et al. and faster than the state-of-the-art scheme. The security of the scheme is proved in the random oracle model under the strong RSA assumption and a DDH assumption. The scheme supports dynamically joining new members to the group without changing the public key and it is possible for revoking a secret key. Five algorithms (KeyGen, Join, Sign, Verify, Open, and Revoke) are described in detail in this paper. The relationship between the IND-CCA2 Public Key cryptosystem and Full Anonymity has been proved in this paper.

In [13], a new and efficient signature scheme is put forward, which is based on the discrete-logarithm-based assumption. The security of the scheme is proved under the LRSW assumption. The scheme can be used for efficient anonymous credential systems, group signatures, and identity escrow schemes. Three signature schemes (mainly including key generation, signature, and verification) are covered in this paper and proved the correctness and security under the LRSW assumption, while schemes A and B are the special cases of scheme C.

SPPEAR [14] provides a security and privacy-preserving architecture for participatory sensing. Participatory sensing is mobile and has powerful sensing capabilities. It is of the essence for participatory sensing to protect the security and privacy of users and to provide incentives for participants. In this protocol, the GM will publish a list of tasks for the user to choose from and the user will get the group private key and authentication token from the GM. Based on this, the user authenticates to the IdP and obtains a pseudonym from the PCA to authenticate the device. The architecture allows the device to not reveal its identity and the tasks it wishes to participate in through group signature and private information retrieval

schemes. The pseudonyms with non-overlapping time validates help SPPEAR secure against Sybil attacks as well.

Chapter 3

Method and Implementation

3.1 System Architecture

The system architecture mainly includes the following four parts: Long Term Certificate Authority , Client, Group Manager, and Data Server. The LTCA is mainly responsible for signing long-term certificates and managing clients. The client is the user normally responsible for collecting data, a classic type of user is vehicles in transport management. The GM is responsible for managing the group keys. When a client's authentication is successful, without obtaining the client's specific identity, GM will add a new group member. The Data Server is a SQL database responsible for collecting data. It will record the data after it has checked its validity.

During the initialization phase, the client generates its public and private keys using OpenSSL. LTCA is trusted by both the GM and the client, which means that both the GM and the client will contain the LTCA's public key. At the same time, the data server should trust the GM, which also means that the data server should contain the GM's group public key.

The first step is when the client sends the certificate signing request containing its public key to the LTCA. Afterward, the LTCA signs the certificate signing request file using its private key and sends the certificate back to the client. Upon receiving the certificate from LTCA, which is a long-term certificate(LTC), the client initiates the second step.

During the second step, the client will send the LTC to the GM to request the group secret key. It is worth noting that the client should send the LTC along with some verification methods, such as signed timestamps to avoid potential attackers.

After the GM has proven the authenticity of the client, it'll initiate the response, in which GM will generate a new member in the group and send its group secret key to the client. The System Architecture can be seen in Figure 3.1.

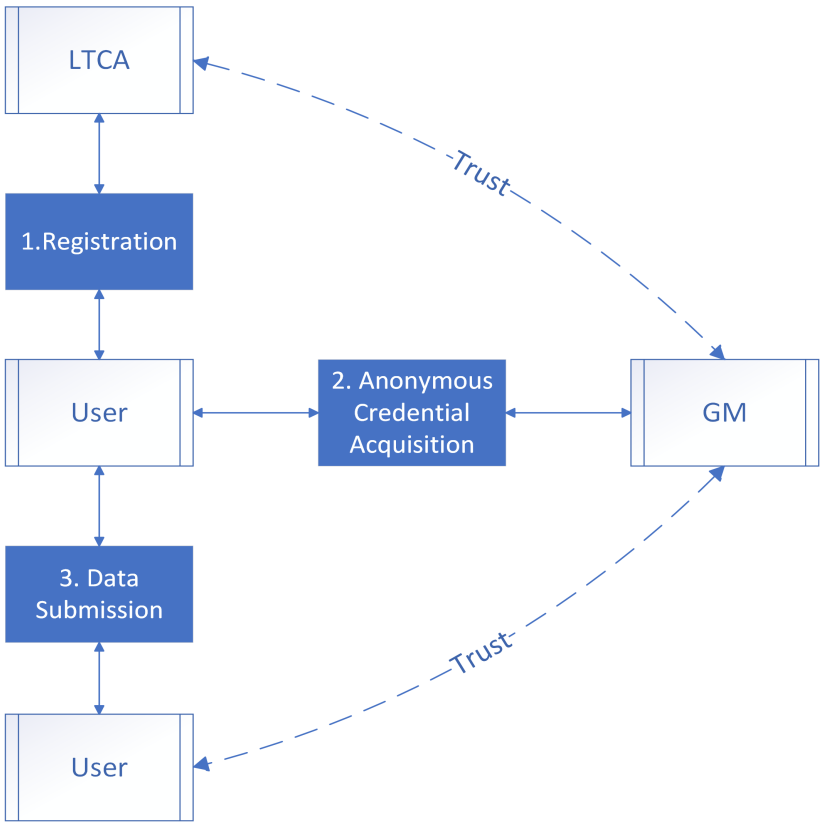

Figure 3.1: System Architecture

Upon receiving the group secret key, there are two schemes for the client to communicate with the Data Server, which is considered the third step.

What comes first is the Group Scheme. The client will simply send the message signed with the group secret key to the Data Server. The Data Server can verify the message using the group public key which is from the GM.

The second scheme is called the Hybrid Scheme. The client will generate the client signing key and client public key by itself at this stage. After the client receives the group signing key from the GM, the client hashes the plaintext information that must be sent to the data server and then encrypts the hash value

with the client signing key generated by itself. The message block sent to the data server consists of three parts: the first part is the plaintext message, the second part is the encrypted hash value, and the third part is the pseudonymous message signed with the group signing key (which contains the client public key).

In the Hybrid Scheme, a certificate called pseudonym is introduced, which is a certificate without an owner, meaning a name is usually a meaningless number, ensuring the message is still untraceable. The pseudonym will only be used for one time, so it is necessary to generate a bunch of key pairs in advance for faster action. This is why in terms of overall performance, the hybrid scheme responds faster on average than the grouping scheme. During the third step using the hybrid scheme, the client will send the message signed with the temporary private key along with the pseudonym signed with the group secret key.

3.2 LTCA

3.2.1 Implementation of LTCA

LTCA registered the user and let them gain access control towards the process by issuing LTCs. So there is two part that LTCA needs to interact with: GM and Client. And here, we need to emphasize the association again: GM needs to trust the LTCA server so that they can recognize and verify what the client sends to them. And each client needs to go for an LTC from the LTCA side and later apply for a pair of keys, the group sign key and the group public key from the GM. The processing can be seen in Figure 3.2.

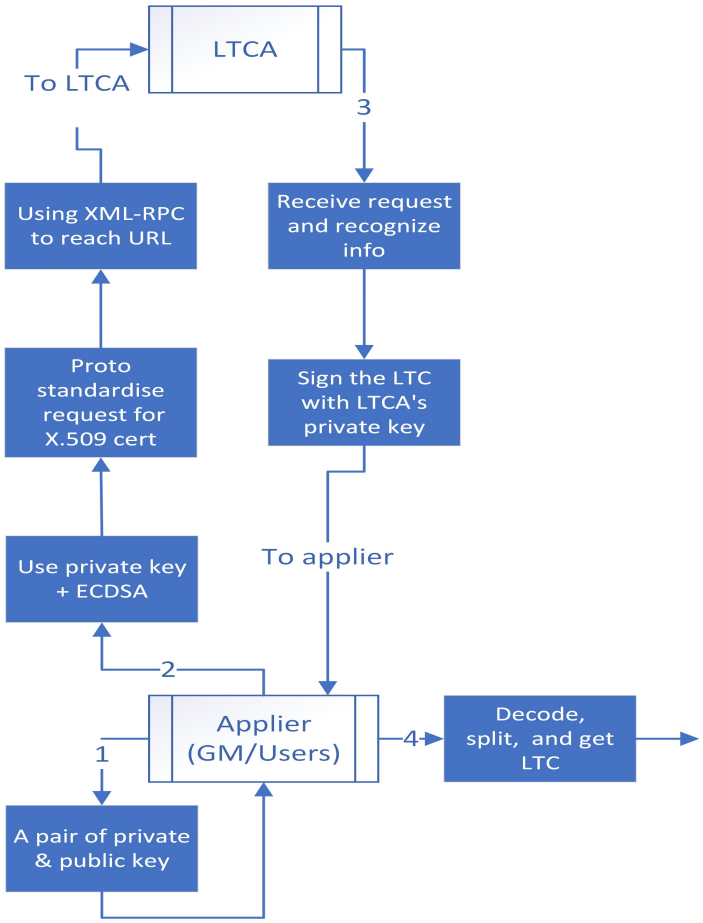

Figure 3.2: LTCA process

So first it comes to the apply part:

1) The applier should generate at least a private key with ECDSA for encryption. ECDSA is a highly secured method and consumes lesser bandwidth because of the small key size used by the elliptic curves [15, 16]. After generating the private key, what we do next is use the private key to generate the certificate request along with other detailed information such as email, organization, name, and so on.

2) Then we use the 'Proto Buffer' to standardize the request. It should include the following information:

   - Request certificate file
   - Request type

- LTCA_ID
- Nonce
- Time stamp

We set the request type as X.509 certificate. LTCA_ID can show our intention of applying for an LTC. The nonce is an random integer in 0-65535. Adding a time stamp can prevent us from man-in-the-middle attacks.

3) The communication protocol that we choose is XML-RPC.

4) After three part above have done, the protocol buffer message is encoded in Base64 format. And send to the LTCA server.

Then the LTCA server would check the request. If the message is correct, the server can sign the request with its own private key and send the response back to the applier.

The applier will receive the response. The response would include three parts: LTCA, RCA, and LTC. So what we need to do is split them and verify the LTC. So if we have the correct LTC, it should include the following extension from the issuer, the issuer's information, and the applier's information. So as Figure 3.3 this is one of the correct LTC we applied.

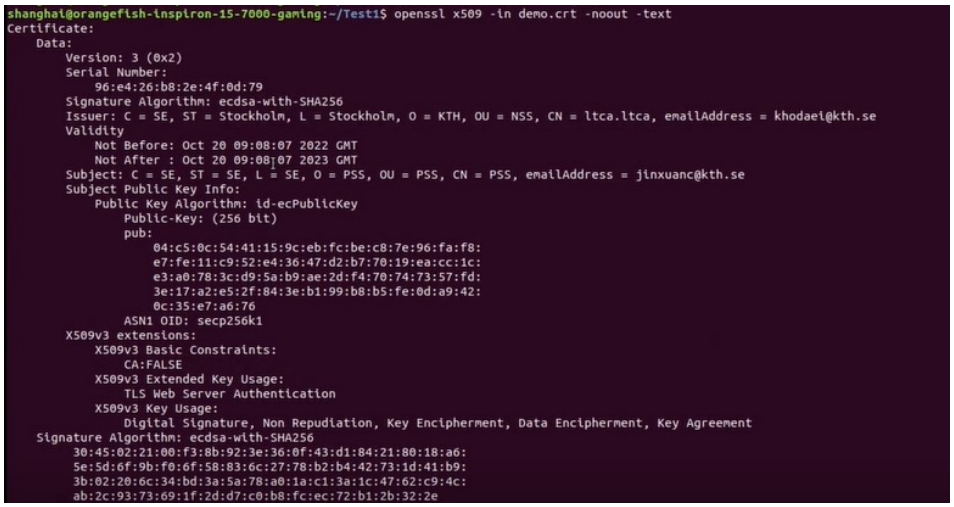

Figure 3.3: LTC example

3.3 Client

3.3.1 Interaction with Other Parts

1. Interaction with LTCA
   At first, the client uses OpenSSL to generate a certificate signing request (CSR) file by Elliptic Curve Digital Signature Algorithm (ECDSA) and includes it in the form of an X.509 certificate in the protocol file. Next, the client uses the 'Protocol Buffer' tool to convert the protocol file containing the information required by LTCA into a file that conforms to the Java language specification. Then, the client uses XML-RPC protocol to communicate with the LTCA server. The service number is 210, and a certificate containing the LTCA public key will be obtained after successful authentication.

2. Interaction with GM
   After the client gets the certificate signed by LTCA's private key, it sends the certificate to the GM server in the HTTP request process. When the verification is successful, the client will get a new gsk[userID] and the gpk from the GM. Then, the client uses gsk to generate the memkey(key for members) and uses the memkey with gpk to sign the message. At last, the client can obtain the pseudonym certificate and export it to the data server.

3. Interaction with Data Server
   To send a message to Data Server and verify itself without revealing its identity, the Client will use a pseudonymous certificate generated from gpk and gsk to sign the message. Then the client will send the message with a signature in the form of an HTTP request to the Data Server for analysis and storage.

3.3.2 Implementation of Client

The client is responsible for interacting with the LTCA, GM, and data server. It sends the csr to register itself to the LTCA and then gets the long-term certificate. Next, it can authenticate itself to the GM with its LTC and get the gpk and gsk. Then, the client can generate a self-signed pseudonymous certificate and use the gsk to sign the message and timestamps. At last, it can submit the sensory data to the data server securely and secretly.

To interact with GM, it needs to build HttpURLconnection. First, obtain the access address URL of GM and its port, and create the HttpURLConnection object. Next, set the request parameters. The major importance parameters are set as

below. After a successful connection, the client can obtain the gsk by reading the corresponding information from the connection.

- Request method-"post"
- Connection timeout-"3000s"
- Instance Follow Redirection-"true"
- Input parameters-"location of the long term certificate and the timestamps"
- Output parameters-"BufferedReader"

On the client side, it also needs to sign the message with either a group signature scheme as in Figure 4.4, or a hybrid signature scheme. For the group signature scheme, a user joins the group by engaging in an interactive protocol with the issuer. They perform algorithms Join and Issue, respectively. The initial input to Join is the gpk, whereas the initial input to issue is the issuer key, isk, and ipk. If the user's uid accepts, Join has a private output of gsk[uid]. If refuses, it will throw out. With gpk and gsk, the user can sign the message with memkey using GL19 algorithm, and then obtain the pseudonym and signature. At last, the user sends the data consisting of the hash and the signature to the data server.

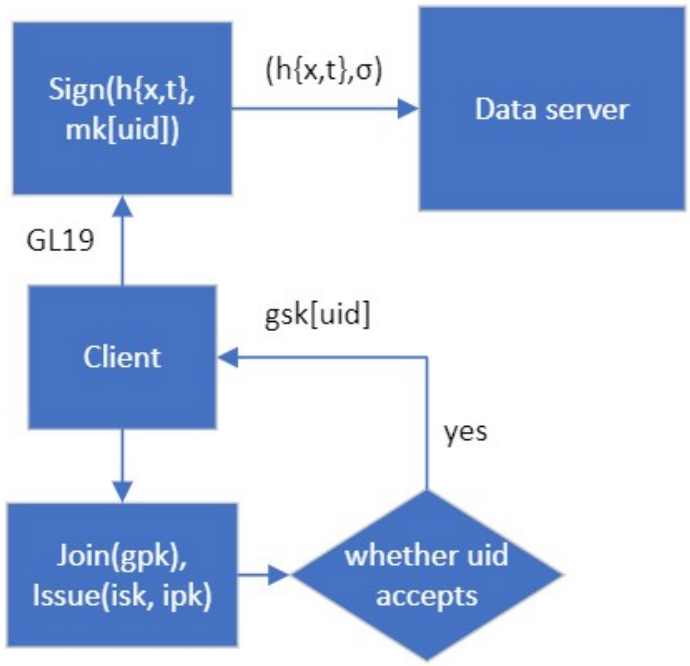

Figure 3.4: Sign the message with group signature scheme

For the hybrid scheme as is shown in Figure 3.5, firstly, use OpenSSL ecparam to generate the pseudonym private keys and the corresponding public key. Then generate the csr by OpenSSL request and sign the certificate which has the personal information. Use the scripts to repeat the steps above to generate the suitable number of the cert we need. Next, use the read buffer to obtain the data and revise the format. Now, use the GL19 algorithm to sign the self-signed certificate by gsk. What's more, use sha2 to hash the clear text and then use the generated RSA private key to digest the hash. At last, export the data block, including the clear message, digest signed by the private key, timestamp, self-signed certificate, and self-signed certificate signed by gsk, in the form of standard format to the data server to authenticate and store the information.

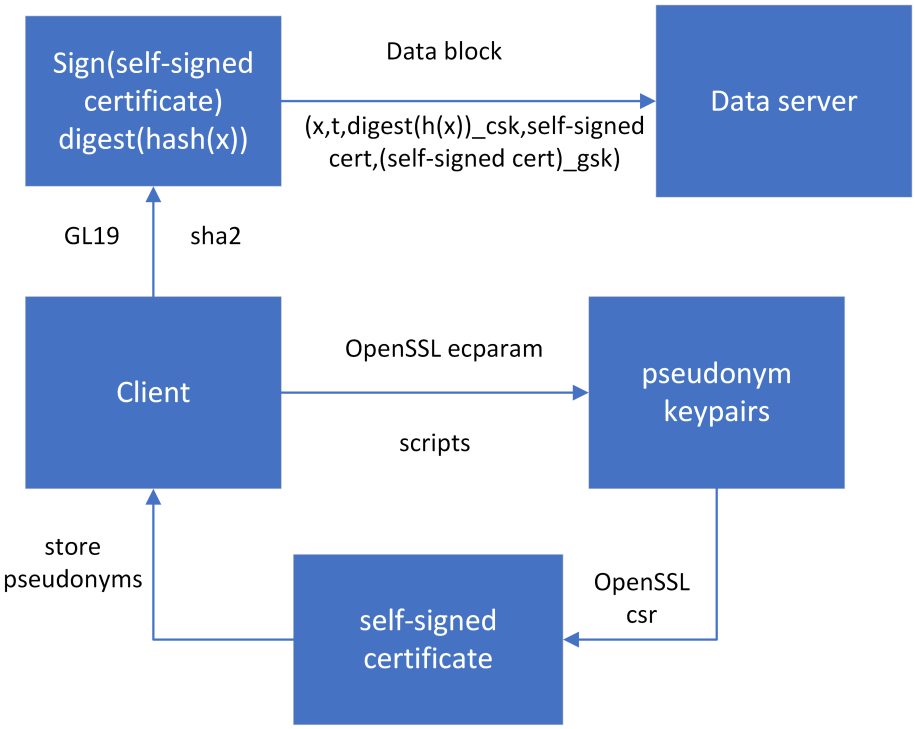

Figure 3.5: Sign the message with hybrid scheme

3.4 GM

3.4.1 Interaction with Other Parts

1. Trust between GM and LTCA
   GM will validate the certificate containing LTCA's public key from LTCA

using the same method as the client. This certificate should be signed by RCA and will be used later to verify the client's long-time certificate.

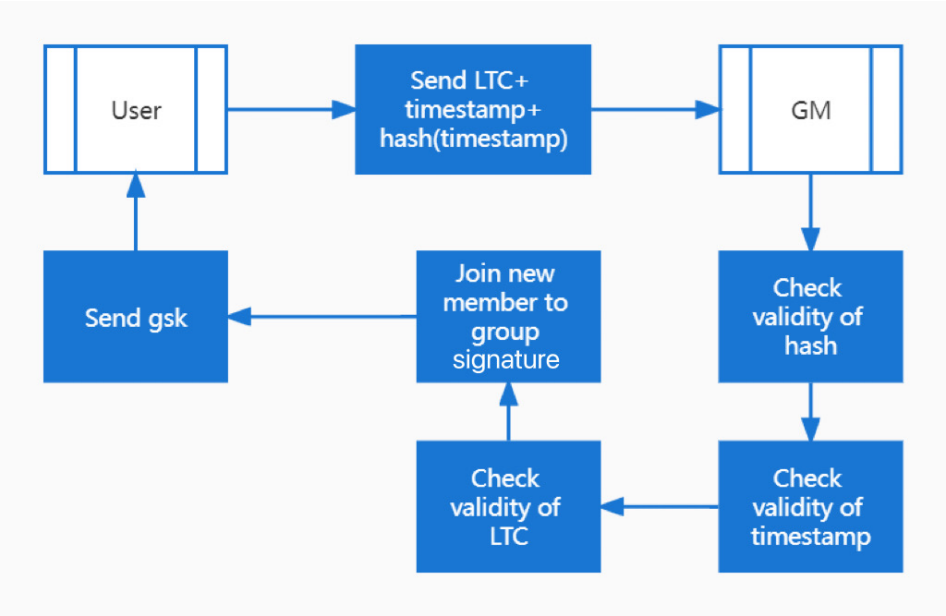

Figure 3.6: Interaction between GM and LTCA

2. Response to GM
   The GM maintains the required elements for group signature such as Issuer. The client will send an HTTP request with their LTC to the GM and GM will validate their LTC using the certificate received from LTCA. If the LTC is valid, then GM will generate a new gsk[userID] and send it to the user along with the gpk from the issuer.

3.4.2 Implementation of GM

GM is responsible for the registration of the users. GM issues anonymous credentials to the client. The anonymous credentials include a group signing key also called gsk and a group public key called gpk.

GM can validate a certificate containing LTCA's public key from LTCA using the same method as the client. This certificate should be signed by RCA and will be used later to verify the client LTC.

During the verification phase, the client will send an HTTP request to the GM,

including the LTC obtained from LTCA in this request. The GM will verify the LTC. If the LTC is valid, the GM will add a new member to the group and send the corresponding gsk to the user. The algorithm used to generate the key pair is GL19. Using GL19 to generate gsk and gpk which is Group Signatures with Selective Linkability. That is, a verifier of a group signature is assured that it was signed by a valid member of the group, but it does not learn anything about the identity of the signer, or even whether two signatures stem from the same user. The process can be seen in Figure 3.7. We model key generation using an issuer in GL19. The first input is a security parameter, and after setting up, sending the param to the issuer. After the issuer's performance, the GM can generate the group signing key and group public key.

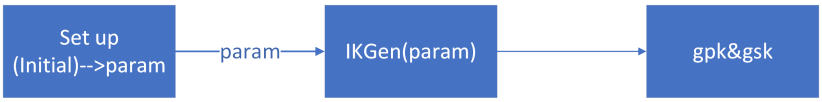

Figure 3.7: GM process

3.5 Data Server

3.5.1 Interaction with Other Parts

1. Interaction with GM
   In this project, we assume the GM has built a reliable trust link with data server. Additionally, a way to establish the trust link needs LTCA verify the LTC of data server.

   The main idea for the interaction is to get gpk from GM, which will be used to verify messages. In the system design, the DS will request the gpk from GM and save it as a permanent file. Each gpk is usable during the GM's work time. The gpk need to be updated if the GM server reboot. If the GM server reboots, the Data Server will have to request gpk from GM again.

2. Interaction with Client
   We will deploy two schemes in this system: group scheme and hybrid scheme. The main difference is the data verification step of the DS perspective.

   In group scheme: The client will send messages to the Data Server signed with its group sign key (also called memkey), a group public key can match

several group sign keys. The data server will use the gpk acquired in advance to verify these messages, and then store the messages if valid.

In hybrid scheme: Hybrid scheme offers better security performance, by adding an additional security layer. The client will send messages to the Data Server signed with pseudonymous. The message will include two signatures: one for plain text, using the client private key generated by the client, and one for security (the pseudonym). The Data server will validate the pseudonym using gpk from the GM. Then DS can extract the client public key in the pseudonym and then verify the plain text.

3.5.2 Implementation of Data Server

In general, the data server is responsible for collecting data from the trusted client's devices. Therefore, the data server will first open a port to listen for messages from clients. Upon receiving a message from a client, it verifies the integrity and authenticity of the message by verifying the signature. The verification process is shown in the figure below. The data server defines the format of the message sent by the client in advance in order to distinguish between signed and plaintext. If the authentication is successful, the data server writes the data to the database for future analysis.

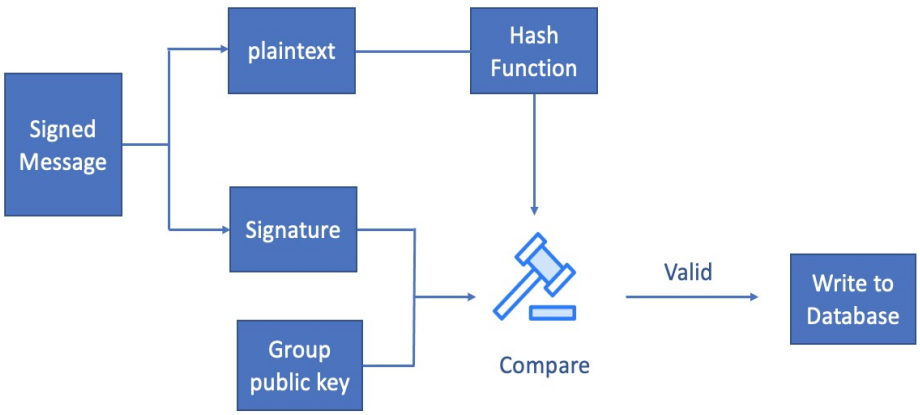

Figure 3.8: Data Server Process for A Received Message

In the deployment of the data server, the data server is still represented as an instance of GL19, which provides the Sign and Verify methods for the data. At

the same time, it acts as a server listening for messages from clients, which is implemented in Java by the SocketServer class. The data server defines the format of the message sent by the client in order to extract the signature and the plaintext, after which the data server uses the gpk and verify methods obtained by the GM to verify the signature. If the signature is valid, it writes the data to the database via Java Database Connectivity (JDBC). The database of choice is MySQL, which is easy to use and check the saved data.

Chapter 4

Experiments and Analysis

This chapter shows the evaluation of the performances with experiment results. Including two parts: security performance (section 4.1)and system performance. The rest part of this chapter is system performance evaluation, focusing on two main parts: total cost and system security.

4.1 Security Performance Analysis

Compared with the traditional asymmetric encryption scheme, the group signature scheme provides unlikability, which makes the message receiver DS unable to know who the sender is through the verification process of the received message, thus protecting user privacy. However, the problem of long-term certificate theft has not been solved - a malicious user who steals LTCA and successfully interacts with GM can send data to damage the overall data on the DS side.

The hybrid scheme solves this problem on the basis of the group signature scheme. A malicious user obtains the gpk and gsk. Because he/she does not know how to use gsk in the hybrid scheme and the generation method of pseudonyms, the probability of the generated garbage messages passing the authentication of the DS side is very low. If a user fails too many attempts, the security system has the ability to remove the user from it through the cooperation of all ends (further expansion is required).

Communications take place over secure channels.This ensures communication confidentiality and integrity.Furthermore, digital signatures are generated with keys known only to the device and thus, non-repudiation is achieved.

4.2 Effect of RSA key length on generating time

The choice of RSA key length is directly related to security performance and operation speed, so choosing an appropriate length of RSA key length is an important evaluation item. We simulate the running time of the RSA algorithm for different key lengths and evaluate it. The result obtained is shown in the figure 4.1

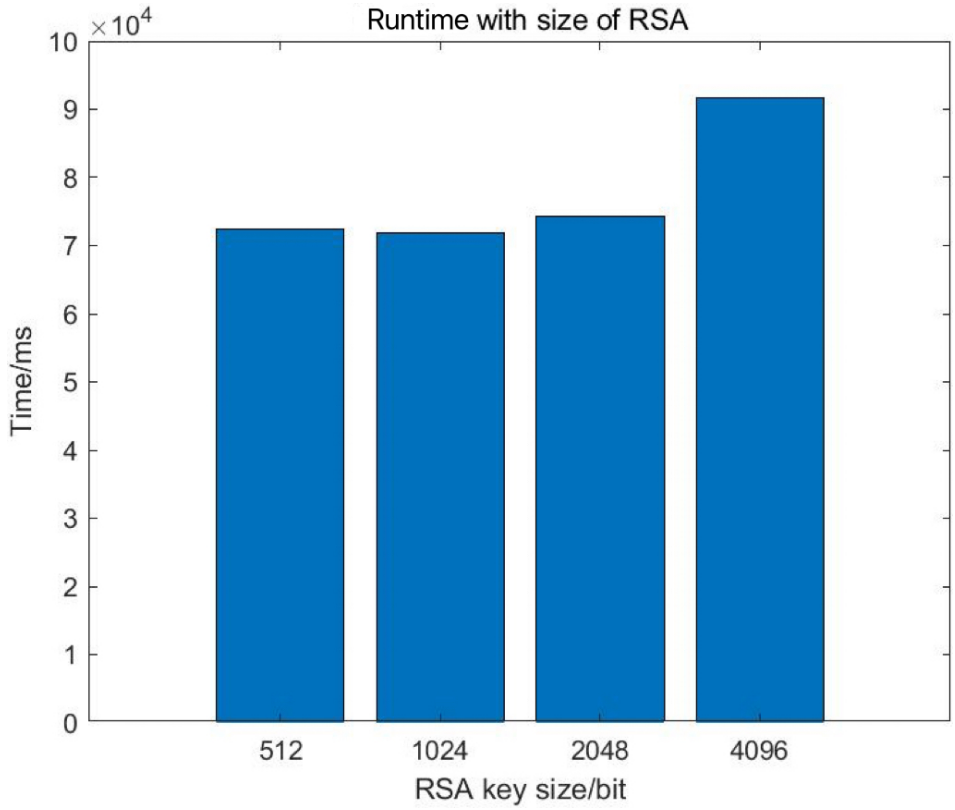

Figure 4.1: Runtime of generating with Different RSA Key lengths

Figure 4.1 shows the generation time when the key length of the RSA algorithm is 512, 1024, 2048 and 4096 bits respectively.
The unit of the key length is Bit. We use plaintext information of the same size to perform 100 repetitions of RSA encryption, and calculate the average of 100 running times as the running time presented in the figure. From the test results, it can be seen that there is almost no difference in the generation time of 512-bit, 1024-bit and 2048-bit RSA keys, but the generation time of 4096-bit RSA keys is significantly longer than the other three. Because the length of the encrypted plaintext information required by this project does not reach the level that requires a 4096-bit RSA key. For this project, a 1024-bit or 2048-bit RSA key can take into

account both security and efficiency [17].
The time it takes to generate an RSA key pair increases as the key length increases. This is because more computational power is required to perform the necessary mathematical operations for generating longer keys. As a result, it will generally take longer to generate a 4096-bit RSA key than a 2048-bit key, which will take longer than a 1024-bit key, and so on. However, the difference in generation speed between different key lengths may not be significant for small key sizes (e.g. 512-bit and 1024-bit keys), but it can become more pronounced as the key size increases (e.g. 2048-bit and 4096-bit keys). This is because the time required to generate a key increases exponentially with the key size.
It is important to note that while longer RSA keys may provide stronger security, they also come with a performance cost. This can make them less practical to use in certain situations, such as when real-time encryption and decryption is required. In general, it is a trade-off between security and performance, and the appropriate key length should be chosen based on the specific requirements of the application.

4.3 Effect of number of users on sending delay

Sending delay is a crucial part of user experience as it is linked directly to the response time of the application. We have simulated different users sending message to the GM at the same time and evaluate there average delay. The result obtained is shown in figure 4.2.

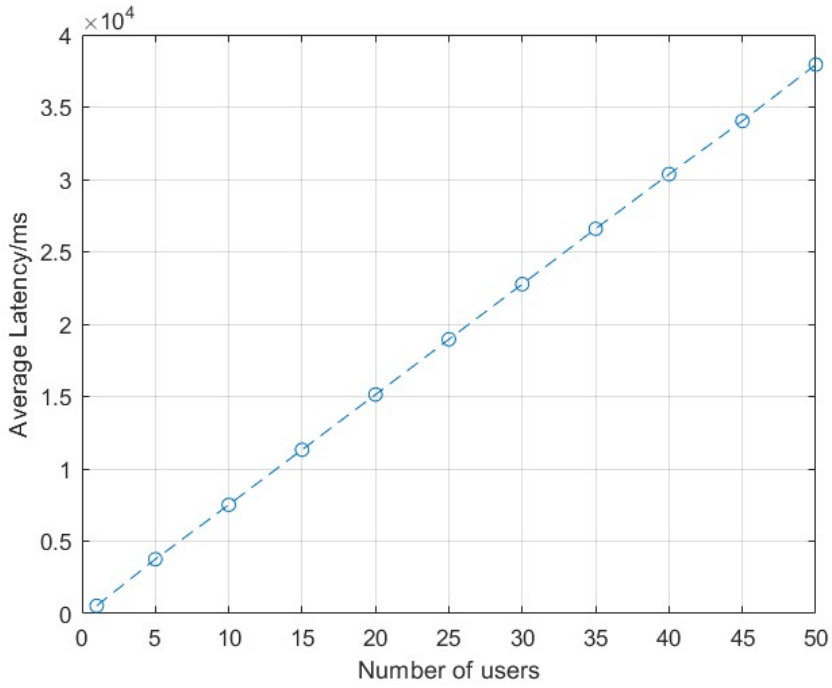

Figure 4.2: Sending Delay Varies with Number of Users

Figure 4.2 shows the average delay of users sending message at the same time. The figure shows a standard linear growth with the increment of the number of users. This indicates that the GM server(e.g. a HTTP server in this case) only process the incoming messages as serial operation and does not have the ability to reply them in parallel. When users are sending messages at the same time, such as at peak hours, the GM server will simply utilize a data structure like an array blocking queue to store the massage and process them one at a time. However, to cope with multiple users at the same time without harming its performance, it is possible to start several GM servers. It is important that those GM servers aren't allocated a specific region or it'll harm the unlinkability of the group signature.

4.4 Effect of the Hash algorithm on average latency

Hash function is a one-way function that takes an input, such as a password, and produces a string. For a well-constructed hashing function, it is not possible to compute the password from its hash. Thus it is important for us to choose a better performance hash function for the digest to the test, and the corresponding integrity check.

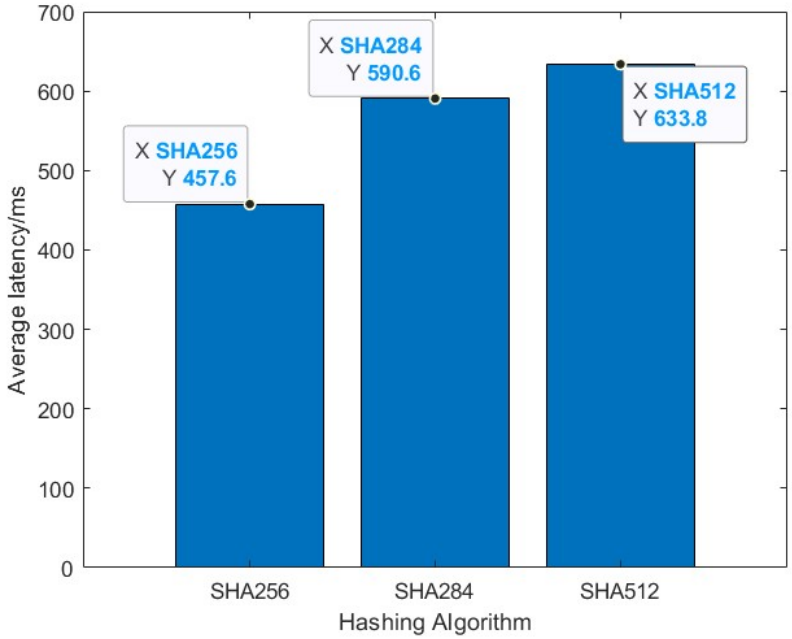

Figure 4.3: The Latency Between Hash Algorithms

Figure 4.3 shows the time taken to repeat these mainstream algorithms 1 million times. The same message is summarised using the same RSA key of length 1024 and a different hashing algorithm. Firstly with either algorithm, it is clear that data digesting can be done very quickly without putting too much load on the application or the machine. Instead of the two remaining algorithms, the shorter and more secure SHA256 was chosen for digesting.The SHA256 takes longer due to fluctuating CPU usage, but it is clear to see that all three are close in time when repeated so many times. The running time of a hash algorithm refers to the amount of time it takes to compute the hash of a given input. In general, the running time of a hash algorithm is influenced by several factors, including the size of the input data, the speed of the hardware on which the hashing is being performed, and the specific implementation of the algorithm.

As a general rule, the running time of a hash algorithm is largely independent of the length of the hash value produced. This means that the running time of SHA256, SHA385, and SHA512 should be roughly the same since they all produce hash values of the same length (256 bits). However, it is essential to note that the running time of a hash algorithm can vary depending on the specific implementation and the hardware on which it is being run. Some implementations of a particular algorithm may be optimized for certain hardware architectures and may therefore be faster on that hardware than other implementations. It is also

possible that one algorithm may be faster on some types of input data than another algorithm, due to differences in how the algorithms process the data. Overall, the running time of a hash algorithm is a complex issue and cannot be accurately predicted without considering the specific details of the implementation and the hardware on which it is being run.

4.5 Comparison Between Group Scheme and Hybrid Scheme

An essential evaluation of this project is to compare the performance of group scheme and hybrid scheme. Therefore, group scheme will be used as the baseline. This experiment compares the time performance from user signature to data server verification between the two schemes with different size of messages separately and in total. The result obtained is shown in figure 4.4 as the client side, figure 4.5 as the Data Server side, and figure 4.6 as the total for both sides.

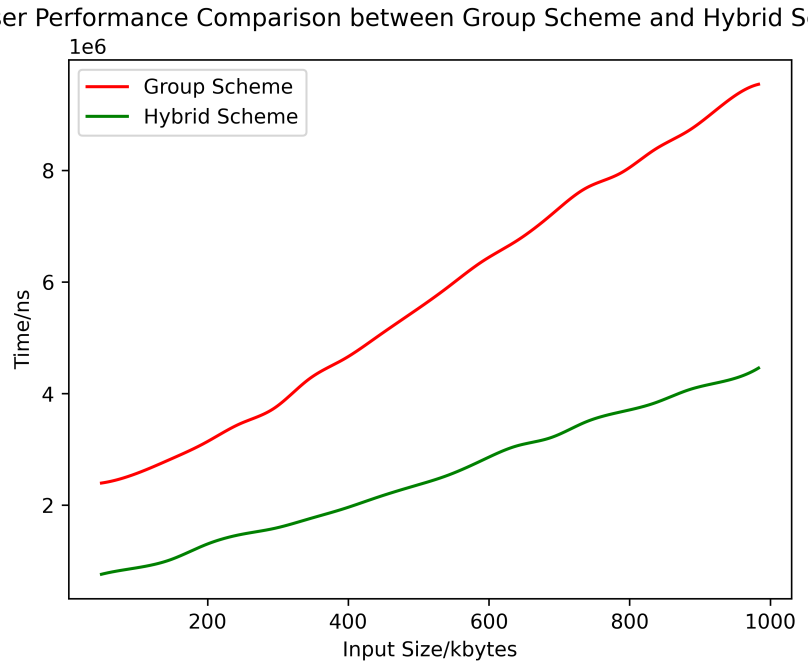

Figure 4.4: User Side Comparison Between Group Scheme and Hybrid Scheme

The figure 4.4 shows the hybrid scheme will always take less time than the group scheme for the user side. As the group scheme, the time includes the hash process for the plain text and the process for the group sign key to sign the newly generated

hash. While as the hybrid scheme, the time includes the time for the pseudonym private key to sign and using SHA256 to hash the plain text. With the gradual increase in the input size, the difference between these two are more evident.

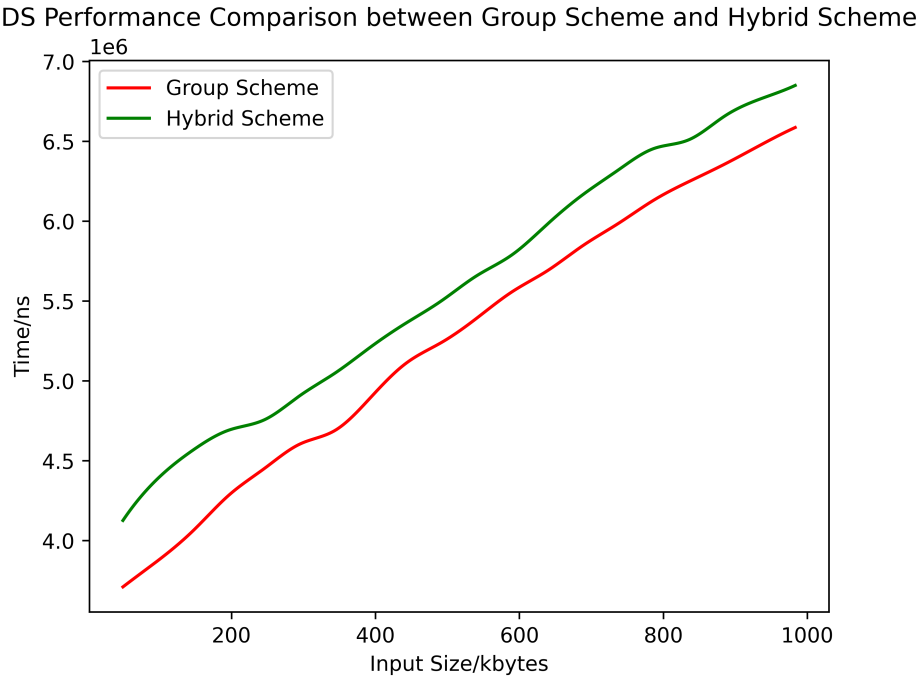

Figure 4.5: Total Comparison Between Group Scheme and Hybrid Scheme

The figure 4.4 shows the hybrid scheme takes more time than the group scheme on the Data Server side. But this is due to the hybrid scheme having an extra steep of verifying the digest using the pseudonym certificate that the user side sends. And these two lines can be considered parallel. The hybrid scheme will always take about 300ns more than the group scheme. Thus it will not affect too much for the total graph.

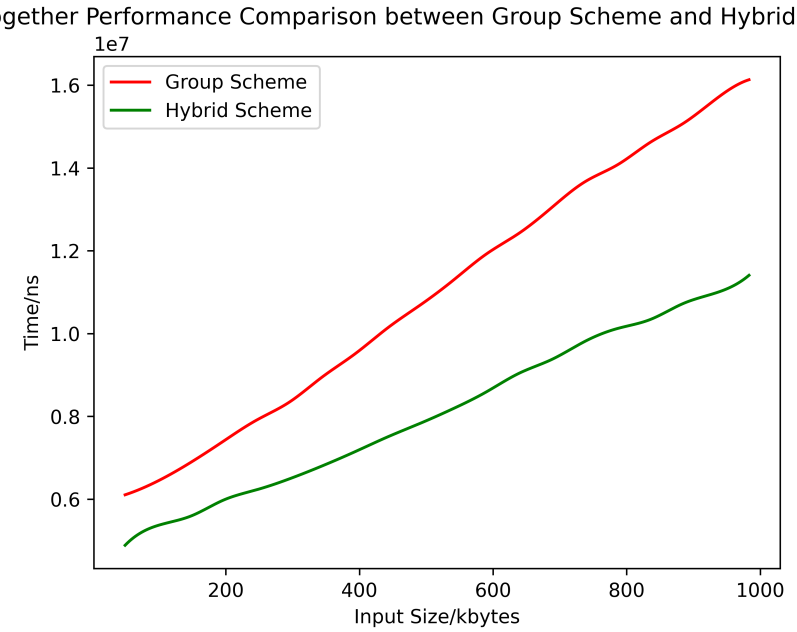

Figure 4.6: Total Comparison Between Group Scheme and Hybrid Scheme

Figure 4.6 shows the time performance by group scheme and hybrid scheme to perform the process from user signature to data server verification with the size of the message. As the figure shows, the hybrid scheme will always take less time than the group scheme for the user side. This result can be considered as the sum of the previous two components if in the same Internet environment. In addition, as the message size gets larger, the difference between the two schemes becomes larger and larger.

For group scheme, the message will be signed and verified by the group signature scheme. Therefore, its time performance is more significant with message size. For the hybrid scheme, the group signature scheme is used for pseudonyms, i.e., self-signed certificates. The signing and verification of the message is performed by ECDSA. Therefore, when the message is larger than a certain value, the group scheme will take more time than the hybrid scheme.

4.6 Mutiple Users and Corresponding CPU Load

The effect on total CPU load by the increase of numbers users are shown in figure 4.7

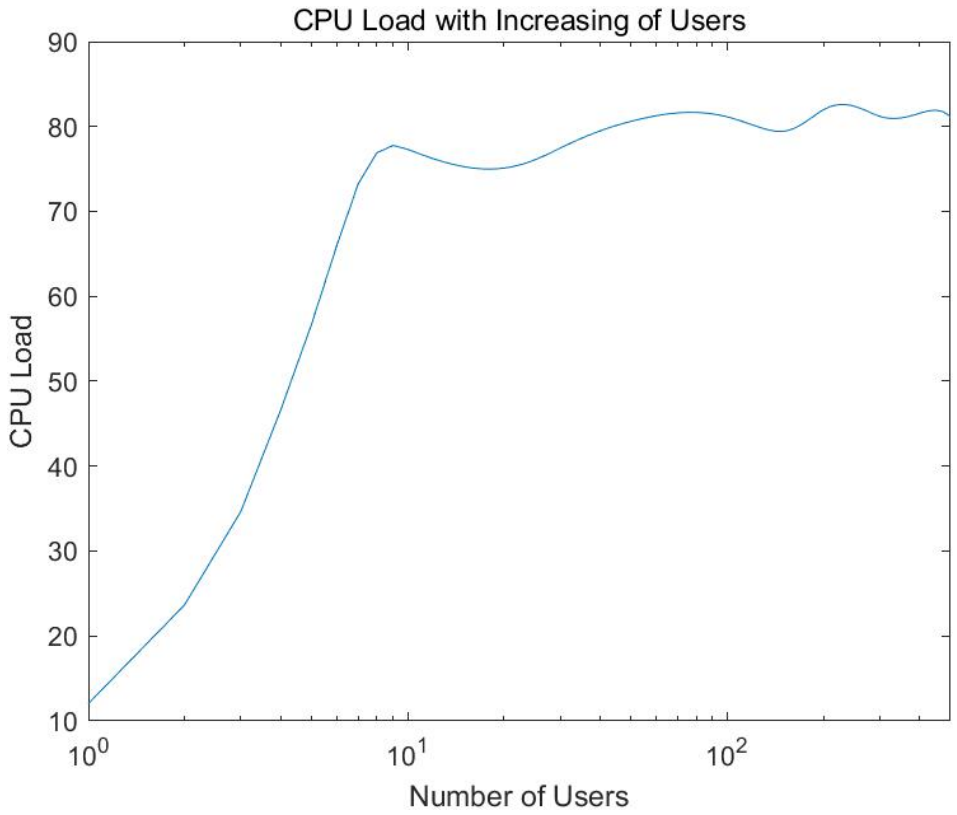

Figure 4.7: CPU load with different users

Figure 4.7 shows that the total CPU load changes non-linearly with the number of users. The change in CPU load can be divided into 2 intervals: the number of users is below 10 and greater than 10.

Now let's observe the first interval, in which the CPU load is approximately linear with the number of users. As the number of users increases from 1 to 10, the load grows approximately uniformly, from 0.15 to 0.75(notice that the x-axis uses log(x)). This shows that when the load is small, the server uses a parallel mode to process multiple users at the same time, so its curve has a nearly linear feature. Since we use a laptop to simulate the operation of the server, the maximum number of parallel processing supported is small, about 10 users Load with Increasing of users.

The next section, the number of users above 10, shows that the load rate of the CPU increases slowly with the growth of users, and it shows a non-linear feature. We think that after the number of users reaches the maximum parallel processing capacity supported by the server, the newly added users will be put into the waiting

queue, and the sent messages will not be subject to security verification the first time. Therefore, the CPU utilization is always slightly higher than 0.75, and eventually reaches 0.82. The reason for the increase is that the sent messages will not be processed immediately, but they still use CPU resources to queue.

In addition, it can be observed that CPU load fluctuates in this range. The arrival of user messages is approximately a random process, so the randomness generates fluctuations. In short, the server processes user information in parallel before reaching a certain load, after which new requests will enter a queue process.

4.7 Effect of the number of different group members on runtime

In the hybrid scheme, members of each group use the same pseudonym. In this experiment, we will explore the relationship between the number of group members and latency.

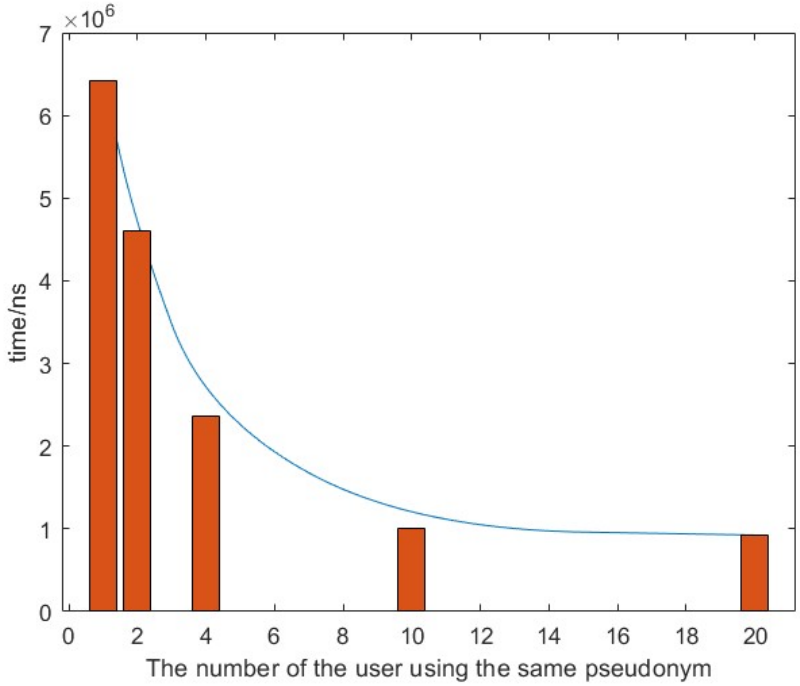

Figure 4.8: Runtime with the number of each group

Figure 4.8 shows that as the number of team members in each group increases,

the running time decreases gradually. In this experiment, we used a total of 20 pseudonym certificates. Therefore, the group member of each group should be a factor of 20. We set the number of group members to 1, 2, 4, 10, and 20 in turn. It can be seen that as the number of group members increases, the run time decreases. This is because Data Server will store the certificate every time after verifying a new pseudonym certificate, and this step will be omitted after the next use of the same pseudonym certificate, so the run time will be shortened.

Chapter 5

Mobile Deployment and Results

After testing the group scheme and hybrid scheme, we implemented this system to mobile applications. This chapter shows how we do the implement and draw the heat maps using data it collected.

5.1 Mobile Client Deployment

The project uses Android Studio to develop mobile applications. Due to time constraints, we have completed the development of the functions most related to the project goals.

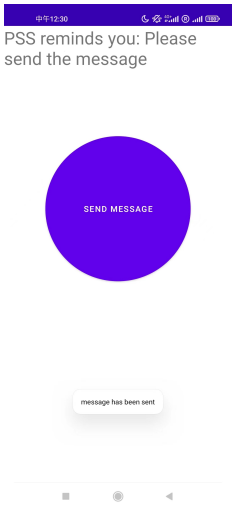

Figure 5.1: Schematic diagram of mobile application

Figure 5.1 shows that the button in the middle of the mobile application integrates the operation of the client in the design. When the user clicks the button, the

information will be sent to the Data Server, and the words that the message has been sent will be displayed on the screen.

5.2 Heatmap

This part of the project is mainly about the application after collecting the data from the user. When the user is able to use the group scheme and hybrid scheme to sign the message, it sends the data, including clear text, timestamps, hash of clear text signed by csk, the clear certificate and the certificate signed by gsk to the data server. After the data server verifies the reliabiliy of the user, it collects the data information to store in the Mysql. The information includes age, gender, temperature, humidity, and the location (different areas in Stockholm) of the users.

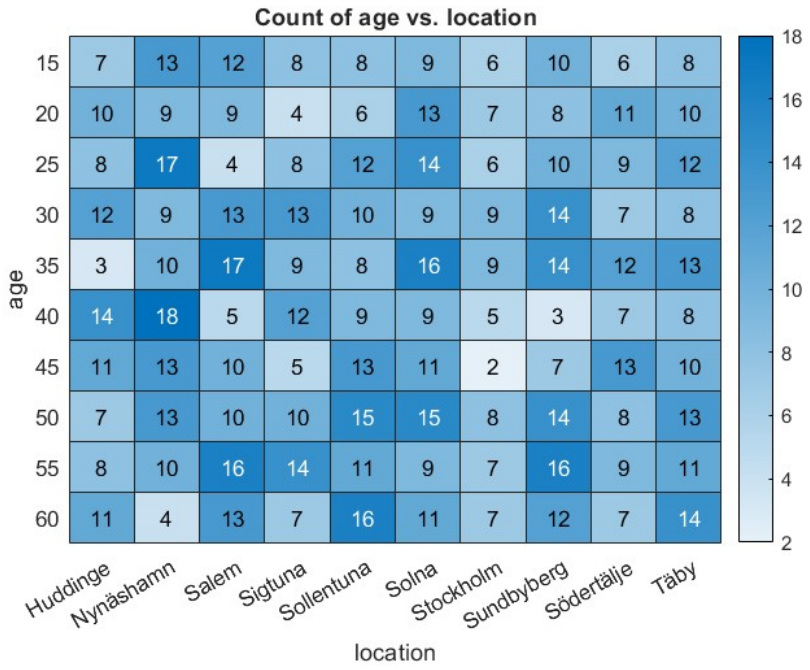

Figure 5.2: Age-Location

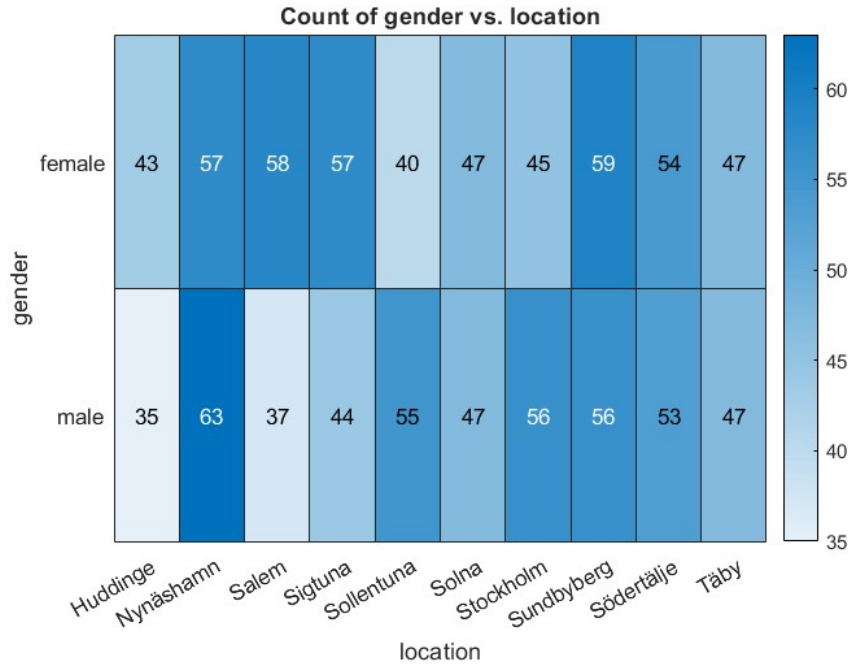

Figure 5.3: Gender-Location

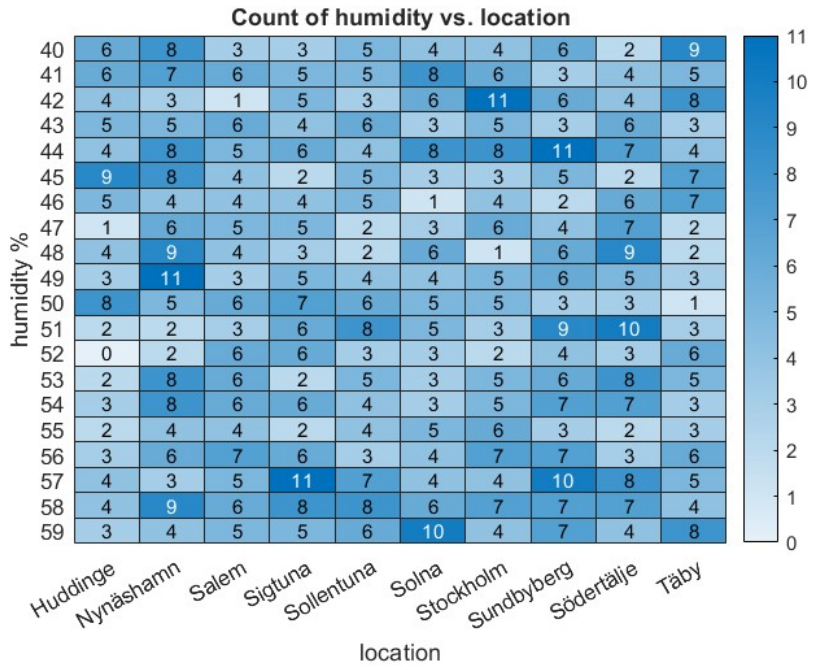

Figure 5.4: Humidity-Location

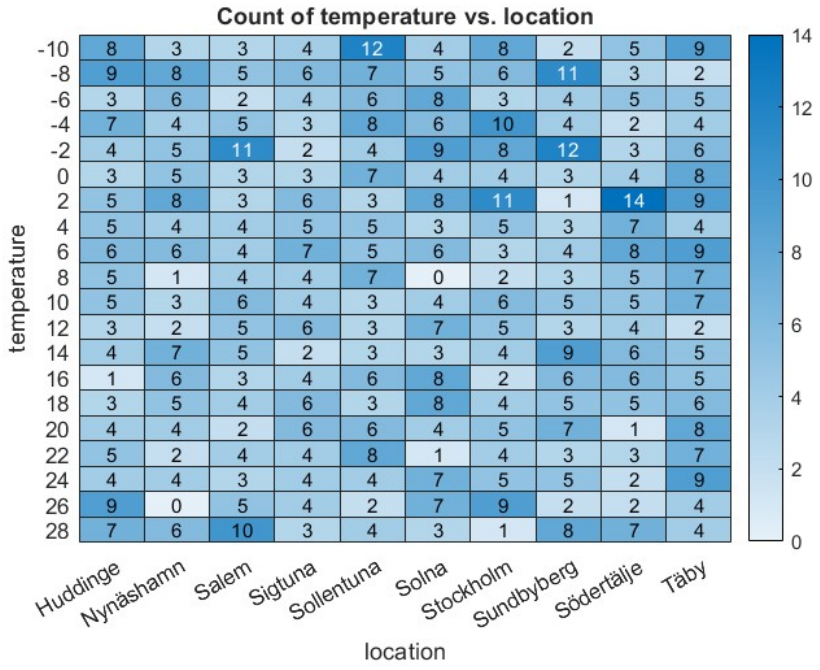

Figure 5.5: Temperature-Location

Chapter 6

Conclusions

This chapter explains the conclusions obtained throughout the design, development, and evaluation described in this thesis and proposes a number of improvements, extensions, or complements that may be of interest in order to continue this work.

6.1 Conclusion

The project includes the implement of two schemes, performance analysis and application deployment. Our achieved goals are shown below.

When implementing these two schemes, there are several steps in common:

- setting up group manager, client, and data server in our computers.
- interaction with LTCA to get long time certificates.
- interaction between GM and Client, includes group signature generation.
- data storage in data server.

For the group signature scheme, we consider about:

- message signing process with gsk in client, using self-signed certificates.
- message verification process with gpk in data server.

For the hybrid scheme, we consider about:

- message signing process with in client, including the generation of self-sign pseudonymous.

- message verification process leveraging self-sign pseudonymous certificates in data server.

After the implementation is complete, we have measured and compared the processing delays on the clients and the server. It's necessary to collect different message arrival rates and message sizes. We used different security levels and compare the efficiency in computation and communication overhead, including memory consumption, CPU consumption, etc.

When the performance analysis part is done, the whole client is compiled into an application. The format of sensor data in transportation is more normal to collect and store data in the data server. Using these data, we have drawn some heat maps of Stockholm.

6.2 Limitations

For building the project, the limitation mainly lies in communicating with open-source code developers, especially when trying to configure the right environment for libgroupsig, which is a cryptographic library developed by IBM. Some other group signature schemes might also be beneficial, but considering that building a new environment is too time-consuming we decided to move forward with the work.

More importantly, we didn't have the budget to rent a real-time server for the servers like the GM server and DS server, so the synthesis of this project didn't include problems like communicating through a long distance or with various network disturbances. We also tried establishing trust among different GM servers so that the user can change to another region when traveling, but it also remains to be improved.

For final results, due to time constraints, the test for a large number of real users remained to be tested due to limited time and resources.

6.3 Future work

Based on the hybrid-scheme security system and performance tests we have completed, the main directions for future improvements are as follows:

First, the data we used in our tests used the converted temperature (decimal) and the geographic location represented by the area name. Metadata obtained by

sensors often needs to be transformed, such as latitude and longitude representations commonly used for geographic locations. Therefore, data processing of different sensors may be a problem worth exploring.

Second, hybrids often need to generate self-signed certificates ahead of time to save running time. On this basis, a pseudonym can be applied to multiple messages, so the number needs further testing. In addition, it is worth investigating what to do if the pseudonyms are exhausted during operation.

The third point is about the application of the scheme to weather data collection. Because the performance of a server is limited and servers need to be placed in different regions, a suitable scheme is needed to build the trust links and do key distribution between GM and Data Server across regions.

6.4 Sustainable Development

With the increase in global network traffic, the number of network security attacks is increasing sharply, so network security is becoming more important. For a network system, if it does not provide users with reliable security, it will have a lower probability of being selected. We propose a scheme for the key distribution part of the network security system, which is more secure than the traditional mode. Then, this advantage will bring more profitable products to manufacturers and better security for people who interact with the network daily.

In addition, the performance of the scheme is better. Our solution not only provides better security but also faster computing performance, which leads to a good user experience. Since a large network system distributes keys thousands to tens of thousands of times a day, shorter computation time can accumulate significant energy savings. In this way, system providers can maintain lower energy consumption and help reduce server load and environmental impact finally. Sustainable development of group signature systems involves ensuring that these systems are able to meet the needs of users over the long term. This includes ensuring the efficiency and security of group signature systems, as well as ensuring that these systems are scalable and able to handle a large number of users and signatures. It also involves ensuring that group signature systems are designed with privacy in mind, and that they comply with relevant laws and regulations. In order to achieve sustainable development, it is important to regularly audit group signature systems and address any identified issues or vulnerabilities. This may involve implementing new technologies or updating existing systems in order to improve their efficiency and security. Additionally, it may involve

working with regulatory bodies to ensure that group signature systems comply with relevant laws and regulations. By taking a holistic and proactive approach to the development and maintenance of group signature systems, it is possible to ensure their sustainable use and continued usefulness over time.

6.5 Ethical Consideration

One of the key ethical considerations in network security is the issue of privacy. In the context of network security, privacy refers to the protection of personal information from being accessed or disclosed without the individual's consent. This includes sensitive information such as financial data, medical records, and personal identification. It is important to ensure that personal information is protected and that individuals are aware of how their information is being used and shared.

Another ethical consideration in network security is the issue of security versus convenience. In many cases, increasing security measures can be inconvenient for users, and it is important to strike a balance between the two. For example, implementing strong passwords or requiring multi-factor authentication can increase security, but may also be inconvenient for users. It is important to weigh the potential risks and benefits of security measures and to consider the needs and preferences of users.

A third ethical consideration in network security is the issue of accountability. It is important to ensure that those responsible for maintaining and securing a network are held accountable for any breaches or failures. This includes both individual users and organizations, as well as any third parties involved in the security of the network.

In addition to these general ethical considerations, there are also specific ethical issues that may arise in the context of network security. For example, the use of surveillance and monitoring technologies raises questions about the extent to which individuals' privacy should be protected. Similarly, the use of hacking or other cyber attack techniques raises questions about the appropriate use of such tactics, and whether they are ever justified.

In order to address these ethical issues, it is important for organizations and individuals to develop and implement clear policies and procedures for network security. This includes establishing guidelines for the use and protection of personal information, as well as establishing protocols for responding to security

breaches. It is also important for organizations to educate their employees about the importance of network security and to provide them with the tools and resources they need to protect sensitive information.

Ultimately, ethical thinking about network security requires a balance between protecting sensitive information and ensuring the convenience and effectiveness of networks. By considering the ethical implications of network security measures and developing clear policies and procedures, organizations and individuals can ensure the safety and privacy of their networks and the individuals who use them.

Bibliography

[1] L. Garms and A. Lehmann, "Group signatures with selective linkability," in *Public-Key Cryptography – PKC 2019*, D. Lin and K. Sako, Eds. Cham: Springer International Publishing, 2019. ISBN 978-3-030-17253-4 pp. 190–220.

[2] S. Gisdakis, T. Giannetsos, and P. Papadimitratos, "Security, privacy, and incentive provision for mobile crowd sensing systems," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 839–853, 2016. doi: 10.1109/JIOT.2016.2560768

[3] G. Calandriello, P. Papadimitratos, J.-P. Hubaux, and A. Lioy, "On the performance of secure vehicular communication systems," *IEEE Transactions on Dependable and Secure Computing*, vol. 8, no. 6, pp. 898–912, 2011. doi: 10.1109/TDSC.2010.58

[4] M. Khodaei, A. Messing, and P. Papadimitratos, "Rhythm: A randomized hybrid scheme to hide in the mobile crowd," in *2017 IEEE Vehicular Networking Conference (VNC)*, 2017. doi: 10.1109/VNC.2017.8275642 pp. 155–158.

[5] S. Gisdakis, T. Giannetsos, and P. Papadimitratos, "Security, privacy, and incentive provision for mobile crowd sensing systems," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 839–853, 2016.

[6] J. Camenisch and A. Lysyanskaya, "A signature scheme with efficient protocols," in *International Conference on Security in Communication Networks*. Springer, 2002, pp. 268–289.

[7] D. Boneh, X. Boyen, and H. Shacham, "Short group signatures," in *Annual international cryptology conference*. Springer, 2004, pp. 41–55.

[8] D. Boneh and H. Shacham, "Group signatures with verifier-local revocation," in *Proceedings of the 11th ACM conference on Computer and communications security*, 2004, pp. 168–177.

[9] J. Camenisch and A. Lysyanskaya, "Dynamic accumulators and application to efficient revocation of anonymous credentials," in *Annual international cryptology conference*. Springer, 2002, pp. 61–76.

[10] J. Camenisch, S. Hohenberger, M. Kohlweiss, A. Lysyanskaya, and M. Meyerovich, "How to win the clonewars: efficient periodic n-times anonymous authentication," in *Proceedings of the 13th ACM conference on Computer and communications security*, 2006, pp. 201–210.

[11] M. Khodaei, H. Jin, and P. Papadimitratos, "Secmace: Scalable and robust identity and credential management infrastructure in vehicular communication systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 5, pp. 1430–1444, 2018.

[12] J. Camenisch and J. Groth, "Group signatures: Better efficiency and new theoretical aspects," in *International Conference on Security in Communication Networks*. Springer, 2004, pp. 120–133.

[13] J. Camenisch and A. Lysyanskaya, "Signature schemes and anonymous credentials from bilinear maps," in *Annual international cryptology conference*. Springer, 2004, pp. 56–72.

[14] S. Gisdakis, T. Giannetsos, and P. Papadimitratos, "Sppear: Security privacy-preserving architecture for participatory-sensing applications," in *Proceedings of the 2014 ACM Conference on Security and Privacy in Wireless Mobile Networks*, ser. WiSec '14. New York, NY, USA: Association for Computing Machinery, 2014. doi: 10.1145/2627393.2627402. ISBN 9781450329729 p. 39–50. [Online]. Available: https://doi.org/10.1145/2627393.2627402

[15] A. Khalique, K. Singh, and S. Sood, "Implementation of elliptic curve digital signature algorithm," *International journal of computer applications*, vol. 2, no. 2, pp. 21–27, 2010.

[16] N. Potlapally, S. Ravi, A. Raghunathan, and N. Jha, "Analyzing the energy consumption of security protocols," in *Proceedings of the 2003 International Symposium on Low Power Electronics and Design, 2003. ISLPED '03.*, 2003. doi: 10.1109/LPE.2003.1231830 pp. 30–35.

[17] D. Giry, "Cryptographic key length recommendation," [EB/OL], https://www.keylength.com/en/4/ Accessed March 1, 2020.

Appendix A

A.1 Authors

Siwei Zhang (siweizh@kth.se)
Siwei is responsible for the management of the whole team‘s work region. He works on building our privacy management system. He reviews literature 3 and 4.

Yuhan Ma (yuhanma@kth.se)
Yuhan Ma’s work is mainly about the implementation of the Group Manager with necessary sensors and reviews literature 5 and 6.

Jinxuan Chen (jinxuanc@kth.se)
Jinxuan’s work is mainly about the deployment of client and connecting it to sensors. He also participates in the hybrid system scheme. Jinxuan reads literature 7 and 8.

Ziyu Cheng (zcheng@kth.se)
Ziyu is responsible for the client and the testing of our system. She evaluates the performance of schemes through client applications. She reviews literature 9 and 10.

Haida Lu(haida@kth.se)
Haida schedules the timetable for the whole group. He contributes to the signature scheme with gsk. He makes final revisions to each weekly report and submits them. Haida reads literature 11 and 12.

Qinglin Zou(qinglinz@kth.se)
Qinglin’s work is mainly on the implementation of Data server perspective. He is responsible for doing preliminary checks (like spelling and format) before each final report. He reviews literature 1 and 2.