Security in Emergency Situations

A Final Report

Jun Yi, Yiming Peng

Abstract

Nowadays, more and more advanced emergency response systems based on wireless technology are used in emergency situations. This kind of system improves the efficiency of emergency response and management, but also faces a great deal of potential security problems, like user end forgery, communication eavesdropping, masquerade, and data stealing, at the same time. It is important and necessary to make sure the security of emergency response system and provide a secure communication system for emergency situation.

In this project, we design a complete set of mechanism to protect the emergency response system. In user end, we provide access control and user authority mechanism. In transmission, we provide communication encryption using Kerberos algorithm. In server end, we provide database encryption. Additionally, we provide friendly UI for server administrator, security token for administrator login and three levels of security protection. By covering every component of system, the security in emergency situations can be guaranteed.

Keyword: emergency response system, security, Kerberos, three levels protection

1. Specific Aims

The goal of this project is to build a security system, which enhances the emergency response system (e.g. DIORAMA) by offering three levels of security protection, which refers to user end, server end, communication process, and administrator end. There is an overview of the whole system design below.

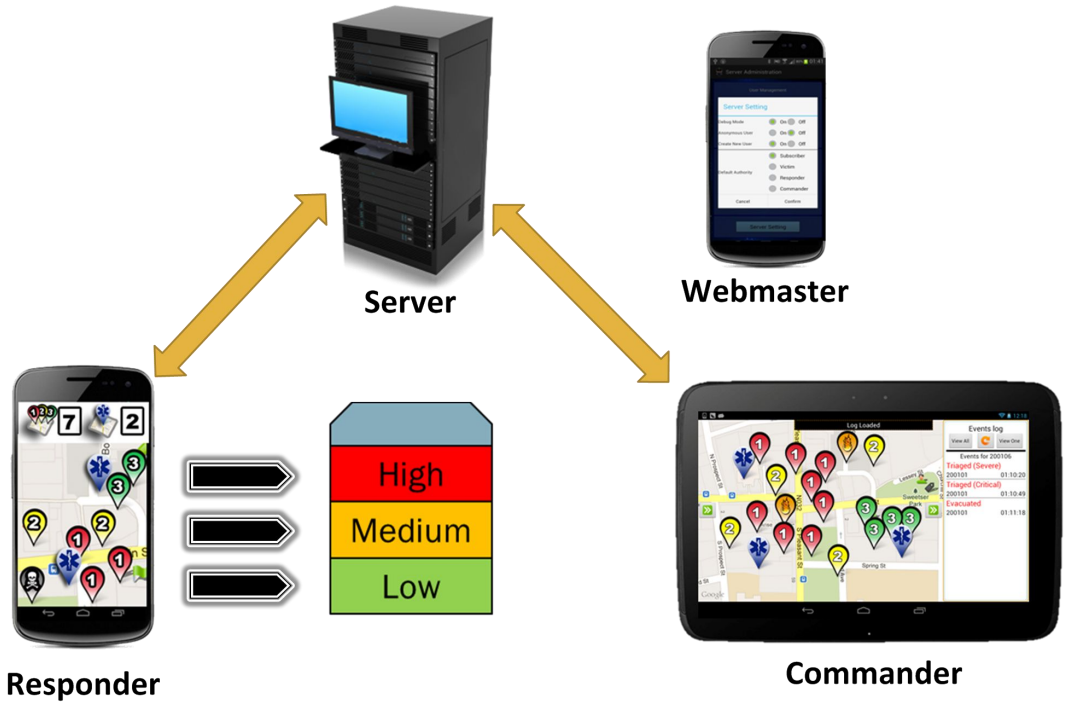

Figure 1: Overview of the security system

1.1 Access control

The objective of this part is to design a user authorization system between user end and server end. Every responder who logins in will be allocated a certain authority. The information responders with different authorities get access to are different. And for incidents in medium and high security level, it is necessary to provide a password to login. By this way, we can make sure the confidentiality of the emergency response system.

1.2 Communication encryption

The objective of this part is to build an encryption system in transmit, based on the network authentication protocol-Kerberos algorithm, to encrypt the whole communication process between user and server. A set of Kerberos algorithm models including Authentication Server (AS), Ticket Granting Server (TGS), and Kerberos Database need to be established.

Using the protocol, we can provide a more secure communication process, making it impossible for the unauthorized users to get access to the protected data.

1.3 Database protection

To protect the data of the database, we can offer AES encryption to all information stored in the data server and keep the encryption key separate with the data. The goal of this part is to prevent the information in the database from being revealed by unauthorized identity, even if the attacker gained full access to the database.

1.4 Security token

In order to increase the security of server administrator, we introduced Security Token, which is also named Authentication Code. Using this code, the attacker cannot login successfully even if he/she has the password of the administrator-The corresponding authentication code can be only generated by the device of the authentic administrator.

1.5 Three levels of security protection

There are three levels of security protection for incidents offered. The purpose of this design is to get a compromised strategy to balance between responding time (server resources) and security provided. Each incident has a security level separated from others. The security level of an incident is decided by user when created. Once determined, the security level can not be changed.

1.6 Security test

Finally we will conduct a serious of attacking trials to test the security of our system. The test includes username forgery, network sniffing and data server intrusion.

2. Background and Significance

2.1 Background of emergency response system

Emergency response systems (e.g. DIORAMA) are designed to assist the incident commanders in the management of a mass casualty incident, and make emergency responders rescue more efficient.

Emergency response and recovery efforts require timely interaction and coordination of emergency services in order to save lives and property, thus, to prove the data authenticity, integrity, and confidentiality in the emergency system is our motivation to make this design.

2.2 Theory and methods

2.2.1 Kerberos algorithm

Kerberos is a computer network authentication protocol which works on the basis of 'tickets' to allow nodes communicating over a non-secure network to prove their identity to one another in a secure manner. Its designers aimed it primarily at a client-server model and it provides mutual authentication-both the user and the server verify each other's identity.

A session key is the core of Kerberos, by which all of the data in transmit are encrypted. By this way, eavesdropping and interception do not make sense.

Figure 2 shows the specific diagram of algorithm.

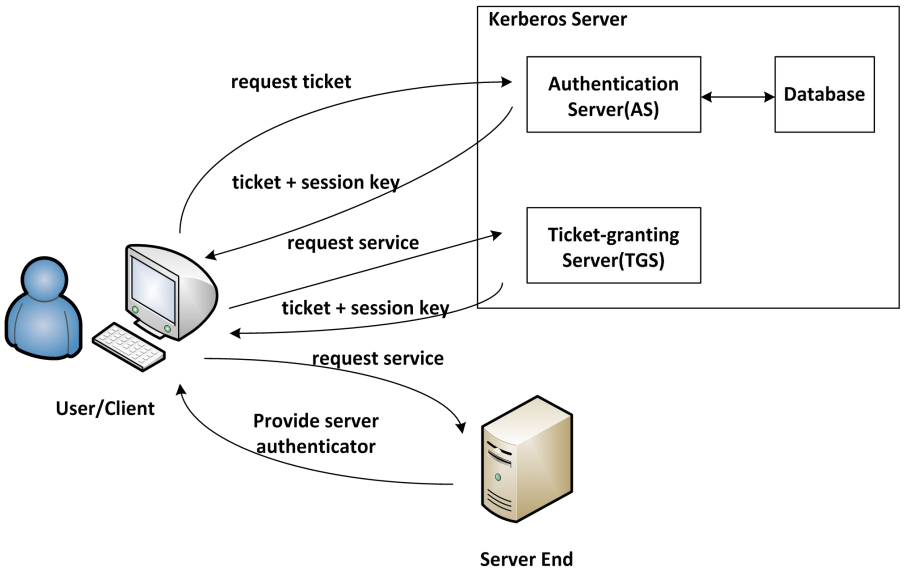

Figure 2: Kerberos Algorithm

2.2.2 MD5 Salt algorithm

MD5 Salt algorithm is one of the most popular encryption algorithm improved from MD5 message-digest algorithm, which is an irreversible algorithm. MD5 will produce a 128-bit (16-byte) hash value, typically expressed as a 32 digit hexadecimal number through a cryptographic hash function. By adding Salt, we can improve the result's randomness and uncertainty.

MD5 Salt is used as login encryption in this project. Of course, there are a lot knowledge about how to select a parameter as the value of Salt, which will be discussed in sections below.

2.2.3 AES algorithm

The Advanced Encryption Standard (AES) is a specification for the encryption of digital data, which is a variant of Rijndael with a fixed block size of 128 bits, and a key size of 128, 192, or 256 bits. Four rounds, including SubBytes, ShiftRows, MixColums, AddRoundKey, make up AES. AES provides a protection for server database in this project.

2.2.4 Security token

Security token is widely used in the area of network security. In our life, the security token may be a physical device that an authorized user of computer services is given to ease authentication. It's used to prove one's identity electronically (as in the case of a customer trying to access their bank account) and is used in addition to or in place of a password to prove that the customer is who they claim to be.

2.3 Significance

The project is designed to build a security system in emergency situation. Different from other security systems, our design provides a complete set of security mechanism, covering user end, transmission process, server end, and administrator end. Our design considers almost all kinds of attacks, including Forgery, Stealing, Tracking, Masquerade, Eavesdropping, Interception, Modification, Breaking password and so on. By this way, we can make sure the incident commander and emergency responder take measures efficiently and securely.

3. Preliminary Studies

- [1] An example for specific application is given to clarify the processes of information security emergency response in this paper.

Based on the fact that traditional information security emergency response system was unable to dynamically take the initiative to prevent network attacks, this platform adds expert system based system incident intelligent response platform to the overall framework of emergency response system, realizing the dynamic updating and learning of incident response strategy, achieving the function of incident analysis and diagnosis and attack forensics combined with status monitoring network subsystem (that is, to Recognize) the attack response subsystem realizing the security function of network system disaster recovery technology(that is, to adapt and resist), the Intelligent Backup and Rapid Recovery Subsystem achieving the function of system intelligent backup and rapid recovery, applying CORBA-based intelligent agent technology as information communication platform to realize the framework of inspection of security incidents, emergency response and disposal, and disaster recovery in information network.

The platform consists of the following four core components: Incident Control Management Center, Network Status Monitoring Subsystem, Attack Response System, System Backup and Rapid Recovery System.

- [3] This article deals with authentication protocols based on secret password information. The article points to the absence of a protocol that would not only be able to authenticate the entity, but also allocate special one-time user permissions. The article deals with authentication mechanisms which are used in the PAP (Password Authentication Protocol) and CHAP (Challenge-Handshake Authentication Protocol) protocols and with the specific implementations of authentication in C4 security system. This article provide possible solution for the absence cited before and its use in particular computer networks including potential advantages and disadvantages can be also found in the article.

4. Research Design and Methods

4.1 Client application design

4.1.1 Conceptual framework

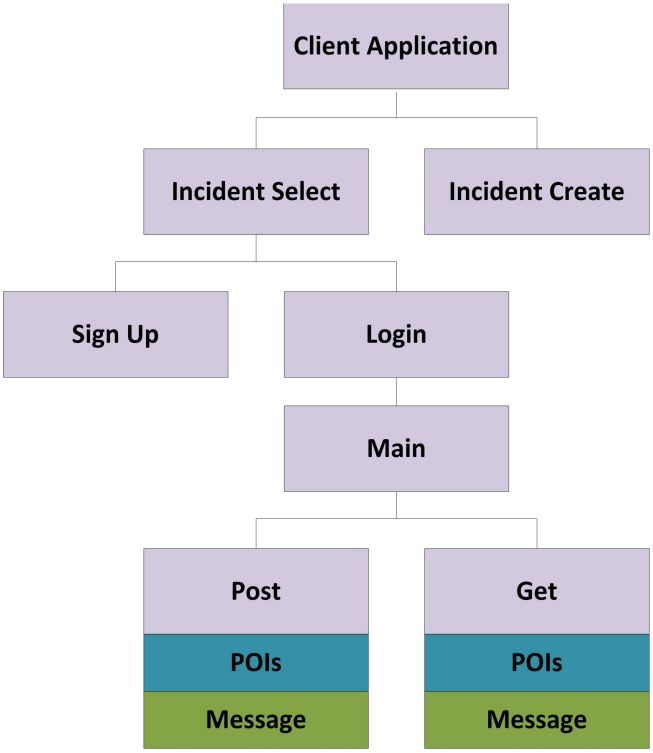

Figure 3: Conceptual framework

Figure 3 shows the conceptual framework of client application, that is, the user end. As we can see, our design provides a series of complete and rigorous operation settings for the emergency responder.

4.1.2 Security level partition

In this project, incidents are divided into three security levels, from low to high (level 0 to level 2). The security system provides a corresponding security protection for each security level. Table 1 lists what we provide in detail.

Level	Access Control	Transmit Encryption	Database Protection
Low	➢ Username ➢ Basic Authority ➢ Anonymous Allowed	Plaintext	Plaintext

Medium	➢ Username ➢ Password ➢ Authentication ➢ Authority	➢ Encrypted by a session key ➢ Authenticator and ticket required	Plaintext
High	➢ Username ➢ Password ➢ Authentication ➢ Authority	➢ Encrypted by a session key ➢ Authenticator and ticket required	➢ Encrypted ➢ Key never stored with data ➢ Certification

Table 1: Security levels table

As we can see, for incidents in different security levels, there are different corresponding mechanisms in access control, transmit encryption, database protection.

4.2 Administrator app design

Administrator app is designed to offer the server manager a friendly user interface to control the server directly. Its functions include assigning different authorities to different users, locking an incident to make it read-only, allowing or forbidding anonymous users or enabling creation of new users etc. Since the privilege of the web master is such high, it would be a disaster if an attacker breaks the account of an administrator.

Security token is used in administrator app to protect the security of administrator account. Before the encryption and authentication of Kerberos process, the admin should provide a correct authentication code, which can only be generated in his or her own device, together with his password. Only when a user possesses a right device and provides a right password will the TGS ticket be issued to him. Using the authentication code, the account of the admin is highly protected by the system and cannot be easily broken through.

5. Implementation Details

5.1 Client application

Figure 4 shows a complete running process of client application.

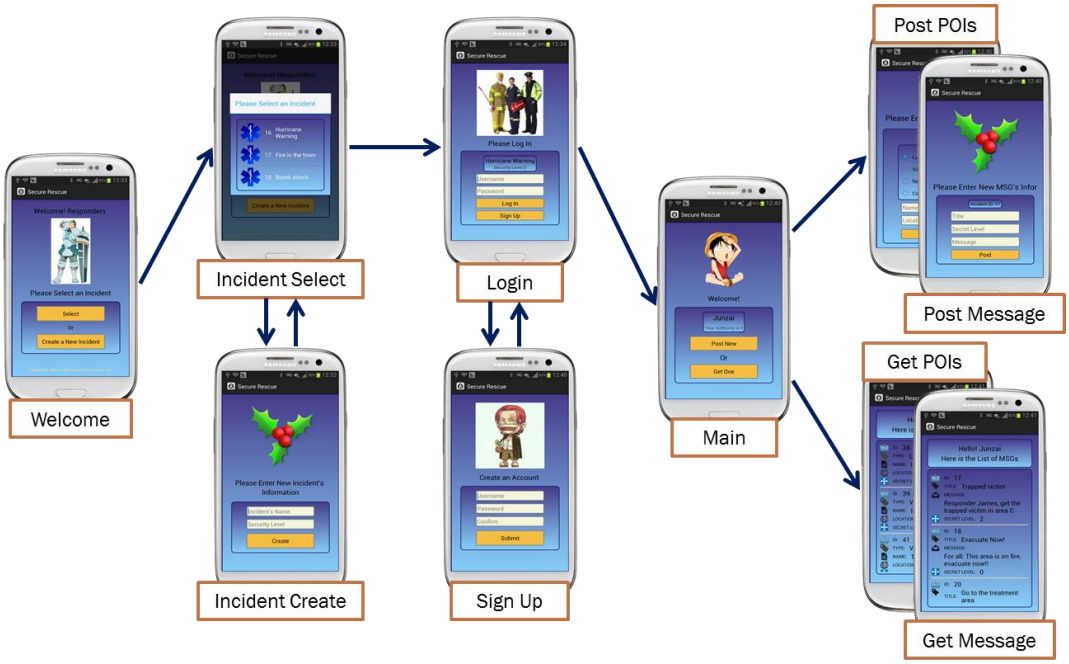

Figure 4: Client application

5.1.1 Flow chart

Figure 5 shows a program flow of client application design. To implement communication with server end through API, thread covers the whole operations, like incident select, incident create, login, sign up, post, and get.

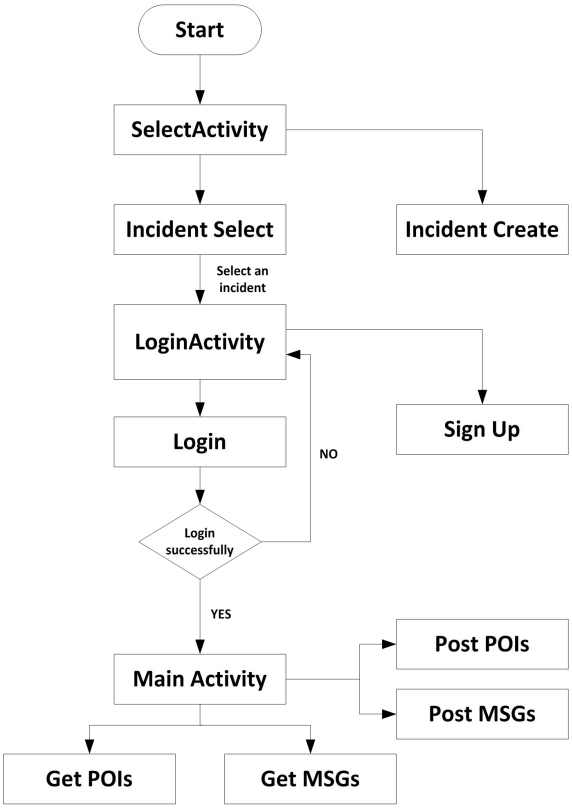

Figure 5: Basic flow chart

5.1.2 Three levels of security protection

5.1.2.1 Security level low

Security level low is basically designed for the incident that does not need much confidential or authentication. At this level, user only has to input a username while password input field is invalidated. Request is sent to data server in plaintext, and authority is checked in server.

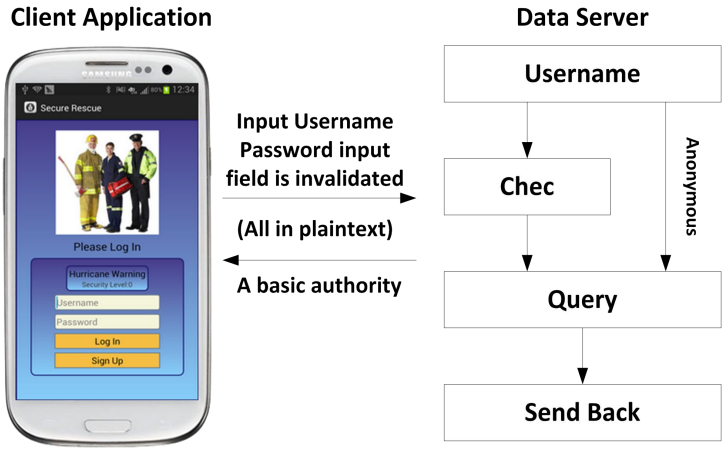

Figure 6: Security level low

This level admits anonymous login, which is an option to the server administrator. Once it is turned on, user can input any username to receive data and is treated as authority 0 in server. This function is useful in the situation of message broadcasting to all people: Everyone can see the broadcast without the need to create a new account. In this kind of scenario, of course, only authenticated users can send data.

To summarize, security level low offers the simplest login procedure for user who do not always need to create an account. It is best used at alert broadcast, EMT training etc. But at this level, forgery is easy when you get someone's username and so is eavesdropping.

5.1.2.2 Security level medium

Medium level of security protection provides the system communication encryption and user authentication using the process based on Kerberos. In our project, we use two servers: data server and Kerberos server (which will be described in 5.3).

The process of Kerberos algorithm is shown in figure 2. When a user wants to access a service (say he wants to get a list of POIs) he needs to provide a password as well as the username. The password is hashed using MD5 in combination of a salt value and stored in the server when the user is created. The client then sends the username to Kerberos Authenticate Server (AS) and tries to get a Ticket Generate Server (TGS) ticket. Both AS and TGS is a service offered by Kerberos Server. The AS issues to the client a session key $K_{c,tgs}$ encrypted with the hash value of the user's password along with a TGS ticket which is sealed using a key only known to AS and TGS. This ticket includes user's information such as username, MAC address and authority and will be used to authenticate user in the next step.

After successful decryption of the session key $K_{c,tgs}$, client will connect to TGS using the TGS ticket and an authenticator generated and encrypted by it. Ticket Generate Server examines both TGS ticket and the authenticator to check if they match. When they do, TGS will issue a Service Ticket according to the service its requesting and generate another session key $K_{c,v}$. This key is also transmitted to client securely by encryption and will be used in future communication.

After this process is done (or the client gets the service ticket), the process of authentication and key exchange is finished. In user interface, a message will be displayed indicating the

user the success of login (Shown in figure 4). The session key $K_{c,v}$ then will be used to encrypt the following communication.

5.1.2.3 Security level high

At this level, besides communication encryption, database is also encrypted using AES. The data encryption key is stored in Kerberos server and never kept in the data server. During the Service Ticket issue stage, TGS will also get the corresponding data encryption key from its database and seal it in the Service Ticket. This key can be picked up by data server after unsealing the ticket and be used to decrypt code. Therefore the data server is able to decrypt the data only when an authenticated user brings a ticket.

Certificate code is also used in every row of database to help detect the unauthorized changes of data. It is the hash value of the concatenation of all plaintext information in that row, which is generated every time when a new piece of data is inserted into the database and checked in every query operation.

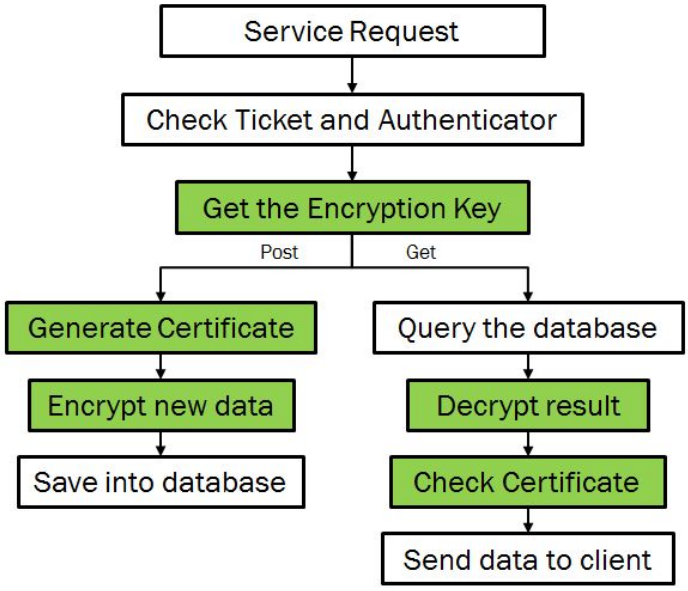

Figure 7: Database Encryption

As Figure 7 shows, after the ticket and authenticator is checked, server will retrieve the data encryption key from ticket and use it to decrypt the database. To create a new row, server will 1) concatenate all data input by user and generate a certificate using a hash function; 2) encrypt the data using the data encryption key and 3) save both certificate and encrypted data into the database. The process of getting data is similar, while the server will examine

the certificate after decryption. If the certificate matches, the data will be sent to the client; otherwise an error log will be recorded into log file to alarm the administrator.

5.2 Administrator app

5.2.1 Functions of administrator app

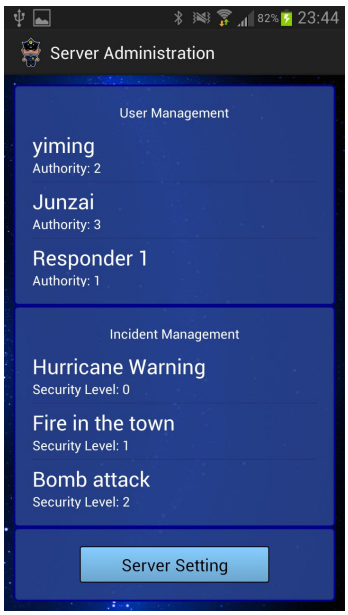

Figure 8-1: Administrator App

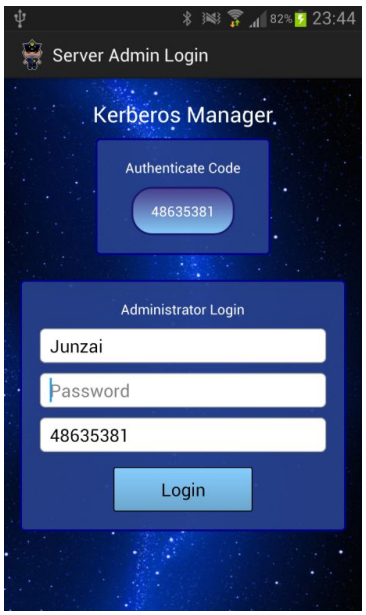

Figure 8-2: Security Token

Figure 8-1 showed the basic functions of the administrator app. The administrator main interface can be divided into three parts, user management, incident management and server settings.

- User management: Administrator can view all the users registered in the system and assign the authority of each one. The update will be posted to the server and stored in the database.
- Incident management: Incidents can be locked by the administrator in this part and will become read-only. In the server database, the incident will be marked and all the updates to it will be denied. In this way, the administrator can protect the precious data after each incident and do more further analysis.
- Server settings: As shown in figure 8, server setting offered a serious of settings that is global to all incidents. Debug mode is used only in system debug that programmer can query the data in the database directly to examine the code; Anonymous user login is only allowed when this switch is turned on; if “Create New User” switch is

turned off, no more new users can be created; default authority is the authority when a user firstly created. This is useful when a group of responders wants to register into the system.

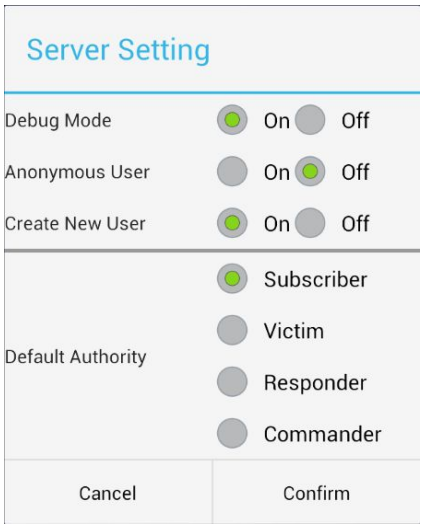

Figure 9: Server Setting

5.2.2 Security token implementation

Security token in our app is generated every 30 seconds by an algorithm, using a seed value, date and time and client's MAC address as inputs. The seed is pre-saved in both server and client (which will be describe later), date and time is synchronized at the beginning of each process and client's MAC address is also stored in server database. Hence, if the seed value in both server and client match, they can generate the same code at the same time without changing any secret message except time; also if the seed value is kept secret both on client and server, it's impossible for an attacker to generate the same code to deceive the server.

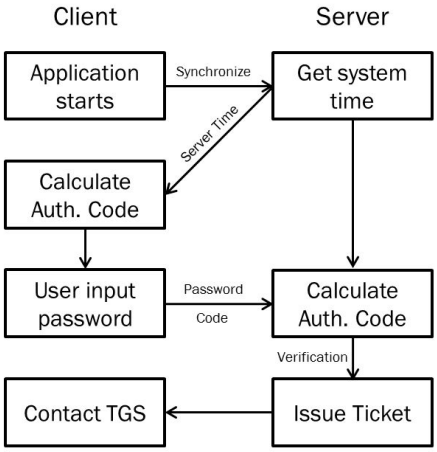

Figure 10: Security Token

As figure 10 shows, when the administrator app starts, it will connect to the server and synchronize the time. After receiving the time from server, client will calculate the authenticate code and display it (as figure7-2 shows). The user inputs password and the code and send both of them to the server to ask for TGS ticket. When the server receives the request, it queries the database and finds the seed of the user. Then it calculates the authenticate code and compare to the user's code. The TGS ticket will only be issued after the authenticate code is verified.

Note that the code can be used only once, which means a previous replayed request will be denied by the server. Also, in our system, where the network condition might be unstable; server accepts the code within the range of 30 seconds. Thus, if the time synchronization is also delayed by network for less than 30 seconds, the code generated will still be accepted by the server.

Initialization of the security token is somehow more complicated. As figure 11 shows, when a new admin account is created (perhaps by a previous admin), the server will generate a secret key (an English sentence or just a random string). The key should be passed to client secretly (By telephone or face-to-face) for the client to initialize. Then both server and client can use this key to generate the seed and store it secretly.

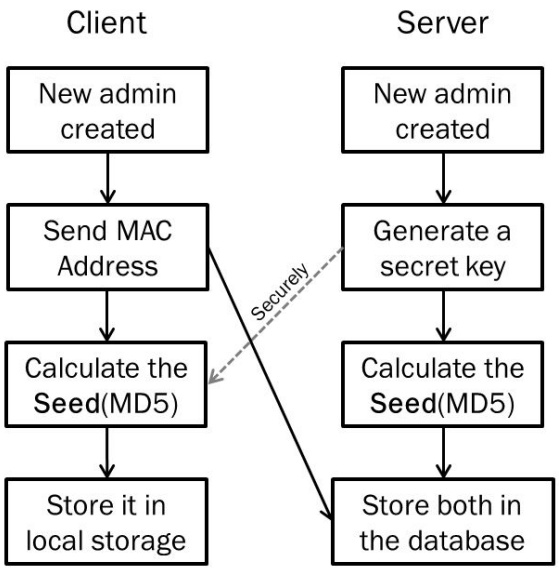

Figure 11: Security Token Initialization

5.3 Server Implement

5.3.1 Kerberos Server

Kerberos Server provides the security service to the system. It includes two databases: Authenticate Service database and Ticket Generate Service database.

1) Authenticate Service database

Authenticate Service is responsible for user authentication and issuing TGS ticket. There are two tables in this database, which will be described below:

- User Table

User Table stores the information of users, which includes username, password etc.

Column Name	Data Type	Detail
UserId	Int	Index, auto-generated
Username	String	Username
Password	String	Password, stored in ciphertext
Authority	Int	Authority, from 0 to 4
C_Time	DateTime	Creation time of the account
Detail	String	Currently not used

Table 2: User Table

- Incident Table

Incident Table contains the information of every incident. The "*IncidenId"* column of all POIs and Messages should all have an entrance in this table.

Column Name	Data Type	Detail
IncidentId	Int	Index, auto-generated
Name	String	Incident Name
Detail	String	Currently not used
Security Level	Int	Security Level of this incident, from 0 to 2
isLocked	boolean	Indicate if this incident is read-only

Table 3: Incident Table

2) Ticket Generate Service database

Ticket Generate Service (TGS) checks the TGS ticket and issues service ticket to the client. To do this job, TGS keeps and only keeps one table containing the keys of all the services. In the Kerberos process, TGS checks the validation of TGS ticket and authenticator, queries the database for the service key required by client and issues the service ticket. After this process, a new session key is generated and used to encrypt all communication later on.

5.3.2 Data Server

Data server is the server that stores all information about all incidents. The information includes all the locations of victims, responders and commanders as well as all message sent during each incident. Therefore the data server basically contains two tables: POI table and Message table. In addition, in Kerberos process, every service and Kerberos server share a secret key which is called Service Key. This key is also stored in the data server.

- POI Table

POI Table stores the information about every Point Of Interest such as: landmarks, victims, responders and commanders. This kind of data needs to be protected by authority because sometimes a commander may not want to be seen by unconcerned person. The structure of this table is shown below:

Column Name	**Data Type**	**Detail**
id	Int	Index, generated automatically
POIType	Int	Type of the POI: 1-Victim; 2-Responder; 3-Commander; 4-Hazard
POIName	String	The name of the POI
POILocation	String	Location, can be described as text or latitude and longitude
SecretLevel	Int	Authority requirement. Only the user with the privilege no less than this number can access this data
IncidentId	Int	Id of the incident
Certificate	String	Certificate of all the data in that row (Only used in security level High)

Table 4: POI Table

- Message Table

Message Table is the table storing all message sent from user in an incident. This includes the broadcast message sent to all users, the commands sent from a commander to a specific responder or help request sent from any victim etc. Table X shows the structure of the message table, which also contains an authority field to limit the message to be revealed only to a certain group of people.

Column Name	**Data Type**	**Detail**
id	Int	Index, generated automatically
MsgTitle	String	Title of the message
MsgContent	String	Content of the message

SecretLevel	Int	Authority requirement. Only the user with the privilege no less than this number can access this data
IncidentId	Int	Id of the incident
Certificate	String	Certificate of all the data in that row (Only used in security level High)

Table 5: Message Table

- Service Key Table

Service Key table stores the service keys of each service, which are synchronized with the keys stored in Kerberos Ticket Generate Server (TGS). These keys are used to unseal the service ticket passed from TGS.

- Query Process

Figure 12 shows the process of querying data from data server in security level Medium. In this process, server will unseal the ticket using the key stored in Service Key Table and obtain the session key Key_c_v. Then the authenticator is decrypted using this session key and compared to the information in the ticket to verify the client. If all information matches and the authenticator is not expired, server will query the database according to the authority and send the information back to client encrypted by Key_c_v. In security level high, as mentioned in 5.1.2.3, the encryption and decryption is also executed before inserting into database and after acquiring from database.

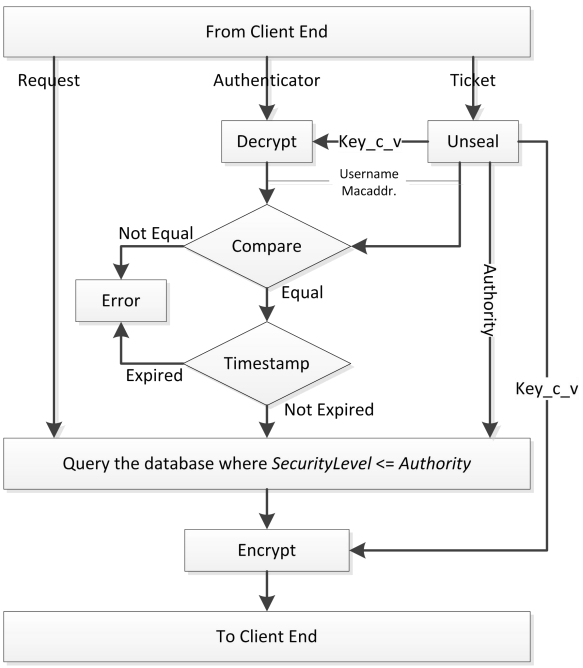

Figure 12: Query Progress

6. Literature cited

[1] Yongjun Peng, Zili Zou, Xi Guo, Jinming Chen, Yang Min, “Research on Architecture and Key Technology of Information Security Emergency Response”, Communication Command Academy of PLA., School of Computer, Wuhan University, Wuhan, China

[2] Zhao Hu, Yuesheng Zhu, and Limin Ma, “An Improved Kerberos Protocol based on DiffieHellman-DSA Key Exchange”, Shenzhen Graduate School, Peking University

[3] Radek Pospíšil, “Authentication in Computer Networks and Proposal of One-Time Increase of User Permissions”

[4] Maurizio Casoni, and Alessandro Paganelli, “Security Issues in Emergency Networks”, Vignolese 905, 41125 Modena (MO), Italy

[5] Yunhua XIANG, Qian ZHANG, and Liwei ZHANG, “To Strengthen the Role of Social Security in Emergency Management”, Wuhan University

[6] Qin Li, Fan Yang, Huibiao Zhu, and Longfei Zhu, “Formal Modeling and Analyzing Kerberos Protocol”, Shanghai Key Laboratory of Trustworthy Computing, Software Engineering Institute, East China Normal University, Shanghai 200062, China

[7] Alan H. Harbitter, and Daniel A. Menasce, “Performance of Public-Key-Enabled Kerberos Authentication in Large Networks”, PEC Solutions, Inc., George Mason University

[8] Konrad Lorincz, David J. Malan, Thaddeus R.F. Fulford-Jones, Alan Nawoj, Antony Clavel, Victor Shnayder, Geoffrey Mainland, Steve Moulton, and Matt Welsh, “Sensor Networks for Emergency Response: Challenges and Opportunities”, Harvard University, Boston University

[9] Prashant Rewagad, and Yogita Pawar, “Use of Digital Signature with Diffie Hellman Key Exchange and AES Encryption Algorithm to Enhance Data Security in Cloud Computing”, Dept of Computer Science & Engineering, G.H.Raisoni Institute of Engg and Management, Affiliated to North Maharashtra University, Jalgaon, India

Appendix

Contribution of team member

1) Yiming Peng
    - Designed the user interface of the client app
    - Implemented the functionality of the client app, including POIs display and create, Kerberos algorithm etc.
2) Jun Yi
    - Designed the server end
    - Designed the Administrator App