Secure Messaging Service on Google AppEngine using ECC and RSA Encryption

Keshav Agarwal

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela-769 008, Orissa, India

Secure Messaging Service on Google AppEngine Using ECC and RSA

Thesis submitted in partial fulfillment
of the requirements for the degree of

Bachelor of Technology

in

Computer Science and Engineering

by

Keshav Agarwal
(Roll: 108CS075)

under the guidance of

Prof. S. K. Jena
NIT Rourkela

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela-769 008, Orissa, India

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela-769 008, Orissa, India.

May 14, 2012

Certificate

This is to certify that the work in the thesis entitled ***Secure Messaging Service on Google AppEngine using ECC and RSA Encryption*** by ***Keshav Agarwal*** is a record of an original research work carried out under my supervision and guidance in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering. Neither this thesis nor any part of it has been submitted for any degree or academic award elsewhere.

S. K. Jena
Professor
CSE department of NIT Rourkela

Acknowledgment

I express my sincere gratitude to Prof. S. K. Jena for his motivation during the course of the project which served as a spur to keep the work on schedule. I also convey my heart-felt sincerity to Prof. S.Panigrahi for his constant support and timely suggestions, without which this project could not have seen the light of the day.
I convey my regards to all the other faculty members of Department of Computer Science and Engineering, NIT Rourkela for their valuable guidance and advices at appropriate times. Finally, I would like to thank my friends for their help and assistance all through this project.

Keshav Agarwal

Abstract

The idea behind this project was to develop a simple Secure Message Passing System based on Public Key encryption schemes of ECC and RSA. A Secure Messaging System is used to share information beyond corporate boundaries. Such information is highly sensitive and thus must be kept secret. Advantages of use of Secure Messaging Service over a generic Email service is that no extra software installation and key sharing will be necessary to kept the message safe againt any kind of eavesdropping.The motivation behind the project was the need for a message passing system completely free from any leaks in information.

The System was to be developed on the cloud platform thus harnessing its Software as a Service feature. Google App Engine was chosen for the deployment of the Application.

Contents

List of Figures

Chapter 1

Introduction

The idea behind this project was to develop a secure messaging service. Such services follow a server based approach to protect sensitive messages and data beyond the corporate borders. Advantages over classical secure-Email services are that confidential and authenticated communication can be started immediately over the internet since there is no requirement to install any software nor to obtain or distribute any cryptographic keys beforehand.

1.1 Functionality

Secure messaging service works as an online service. Users enroll into the messaging service. Users can then log into their account using the username and password similar to any web based Email account. They are now use the service to send messages to any other user. The transmission over the internet can be secured by SSL. The message at the server is encrypted and only the authenticated user can decrypt and read the original message.

1.2 Security

The Emails on the server are encrypted using Elliptic Curve Cryptography and RSA encryption. There is no need to share any cryptographic key among the users as the encryption, decryption and cryptographic keys are managed by the server. Further, the use of public key encryption negates the need for sharing the cryptographic key. An added choice with the key sizes is also provided for the user to choose depending

of the sensitivity of the data being shared.

1.3 Email and Secure Messaging

Secure messaging is complete change in protocol and paradigm as compared to the generic Email Service. Secure messages are encrypted and stored on the network or internet server and decrypted only on the request of the authenticated user. The decrypted message is again not stored on the server but is removed after the page reloads.

1.4 System Design

The system was built on the skeletal model of an Email server. It can be broadly divided into three modules- the Encryption and Decryption module, The Mail Server and the DataStore. The block diagram of the complete system is shown in fig: 1.1

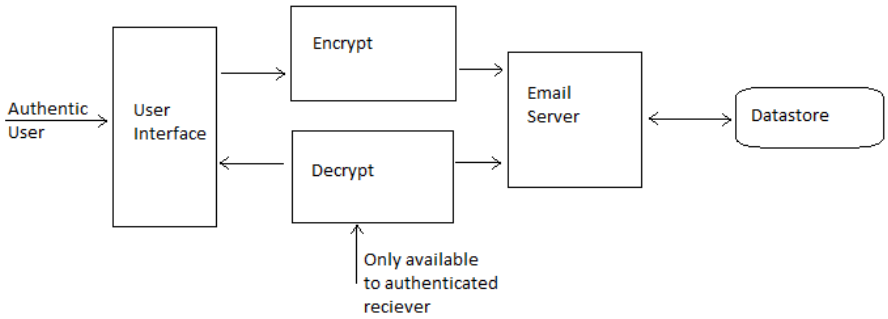

Figure 1.1: Block Diagram of the Message Passing System

Chapter 2

Encryption and Decryption Module

The term encryption refers to converting ordery information in unintelligible gibberish called ciphertext. Decryption is the reverse, in other words, moving from the unintelligible ciphertext back to plaintext. A cryptosystem is the ordered list of elements of finite possible plaintexts, finite possible ciphertexts, finite possible keys and Encryption and Decryption Algorithms. The modern field of cryptography can be divided into two broad classes- Symmetric-key or the private key cryptography and the public-key cryptography.

Symmetric key cryptography refers to the encryption methods in which both the sender and the receiver share the same key. In public key cryptography two different but mathematically related keys are used. The key pair is generated secretly, as an interrelated pair. The key system is so contructed that calculation of one keyfrom other is infeasible. One key from the pair is used for encryption and is made public while the ither is used for decryption and is kept private.

In the Secure Messeging system two public key algorithms were used- RSA and Elliptic Curve Cryptography.

2.1 RSA Encryption

The RSA cryptosystem, named after its inventors R. Rivest, A. Shamir and L. Adleman, is a widely used public key Cryptosystem. The RSA Cryptosystem is based on the mathematical property of integer factorization. It is based on the intractability

of the integer factorization.

2.1.1 Key Generation

Two very large prime numbers, normally of equal length, and both relatively prime to each other are randomly generated. These are multipled to get the field size

$$N = P \times Q$$

$$phin = (P - 1) \times (Q - 1)$$

A third number E is chosen to be between 3 and N. E should be relatively prime to 'phin' (ie. there should be no common factors between phin and E) and is termed as the public key for encryption. The private key D is then calculated as

$$D = E^{-1} mod(phin)$$

These generated public and private keys are then used for encryption and decryption of the text.

2.1.2 Encryption and Decryption

The input stream is encrypted or Decrypted one character at a time. The incoming character is mapped to the ASCII value and the following operations are done on them

For Encryption :

$$Ciphertext = (plaintext)^E mod(N)$$

For Decryption :

$$Plaintext = (Ciphertext)^D mod(N)$$

2.1.3 Security of RSA

The security of RSA depends on the ability of the attacker to factorize the numbers. New, faster and better methods for factorizing numbers are constantly being devised. A large integer is more difficult to factorize and so better is the security. The disadvantage of using long keys is the computational overhead involved in Encryption and

Decryption. The proposed Secure Message Service uses keys of length 512 bits and 1024 bits.

2.2 Elliptic Curve Cryptography

Elliptic Curve Cryptography works with points on an elliptic curve. The security of this type of public-key cryptography depends on the elliptic curve discrete logarithm problem. The idea of using Elliptic Curves in cryptography was introduced by Victor Miller and N. Koblitz as an alternative to established public-key systems such as the DSA and RSA. The Elliptical Curve Discrete Logarithm Problem(ECDLP) makes it difficult to brak an ECC encryption as compared to RSA. A significantly smaller parameters (such as the key) can be used in ECC to achieve better security. This also helps reduce the computation overhead.

2.2.1 Elliptic Curves over Finite Fields

An Elliptic Curve E(F_p) over a finite field F_p is defined by the parameters a,b $\in F_p$ (a, b satisfy the realtion $4a^3 + 27b^2 \neq 0$), consists of the set of points (x, y) $\in F_p$,

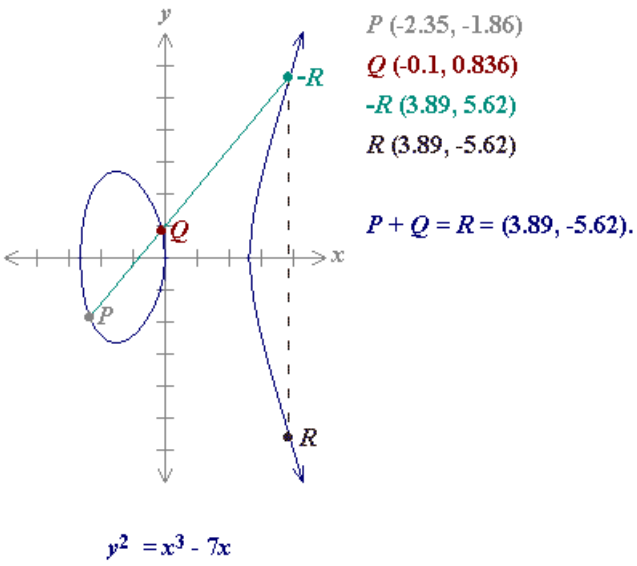

Figure 2.1: Addition of any two points on an Elliptic Curve

satisfying the equation $y^2 = x^3 + ax + b$. The set of points on $E(F_p)$ also include point O, which is the point at infinity and which is the identity element under addition. The addition operator is defined over $E(F_p)$ and it can be seen that $E(F_p)$ forms an abelian group under addition. The addition operation in $E(F_p)$ is as follows:

- $P + O = O + P = P, \forall P \in E(F_p)$
- if $P=(x,y) \in E(F_p)$, then $(x,y) + (x,-y) = O$. The point $(x,-y) \in E(F_p)$ and is called the negetive of P and is denoted as $-P$.
- if $P=(x,y) \in E(F_p)$ and $Q=(l,m) \in E(F_p)$ and $P \neq Q$, then $R = P + Q=(x_3,y_3) \in E(F_p)$.

Thus, the sum of two points on an Elliptic Curve can be visualized as the point of intersection of a straight line passing though those points with the curve. when P and Q refer to the same point then the operation is termed as doubling of a point

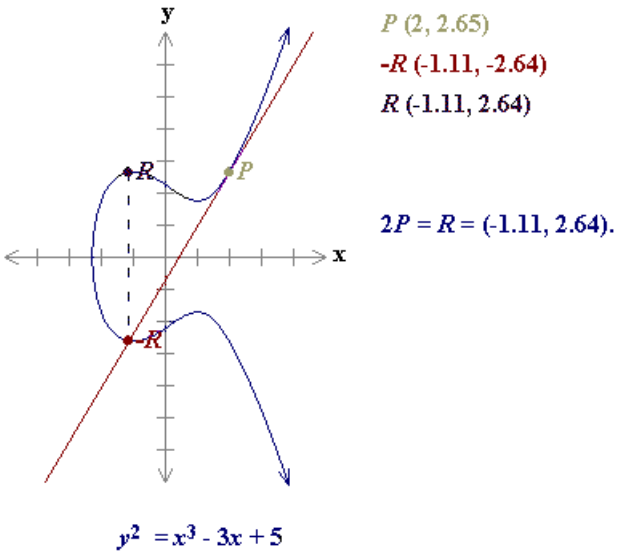

Figure 2.2: Addition a point to itself Elliptic Curve (Doubling)

This operation can be visualized as the point of intersection of the elliptic curve and the tangent at P.

2.2.2 Key Generation

Let us consider an Elliptic Curve of the form $y^2=x^3 + ax+ b$. Parameters a and b are chosen to be large random numbers following the stated condition. A random point on the curve(Q) is chhosen as the base point. **User A Key Generation**

1. Select a random number K_A as the private key, such that $K_A < n$
2. Calculate public key P as

$$P = K_A \times Q$$

User B Key Generation

1. Select a random number K_B as the private key, such that $K_B < n$
2. Calculate public key P as

$$M = K_B \times Q$$

2.2.3 Encryption and Decryption Procedure

1. Consider a message P_m sent from A to B. A chooses a random positive integer k, a private key n_A and generates the public key as

$$P_A = n_A \times G$$

and produces the ciphertext C_m consisting of the pair of points $(kG, P_m + kP_B)$ where G is the base point on the curve P_B is the public key of B with private key n_B

2. To decrypt the ciphertext, B multiplies the 1^{st} point in the pair by B's secret and subtracts the result from the secong point

$$P_m + kP_B - n_B(kG) = P_m$$

2.2.4 Security of ECC

To crack an ECC encryption the attacker needs to generate all possible A_s from the pair of AI and F that it has (as AI=A $\times$ F). and this is very difficult to generate. The number of discrete points on the curve is the called the order of the curve. If the order of a point F on the curve id 160 then 2^{80} combinations of A_S can be formed and generation of all these A_S will take a very long time. Thus, Elliptic Curve is a far better and secure way of message encryption.

Chapter 3

The Email Server

The complete mail server system was set up on the google App-Engine. It offered the very basic functions of message send and read to the logged in and authenticated users. The main view page shows the inbox of the user containing encrypted messages. The messages, if decrypted, are shown on the sceen and remain there only till the page is not refreshed or reloaded. The decrypted mails are not stored. The basic activity diagram for the system is shown in Figure 3.1

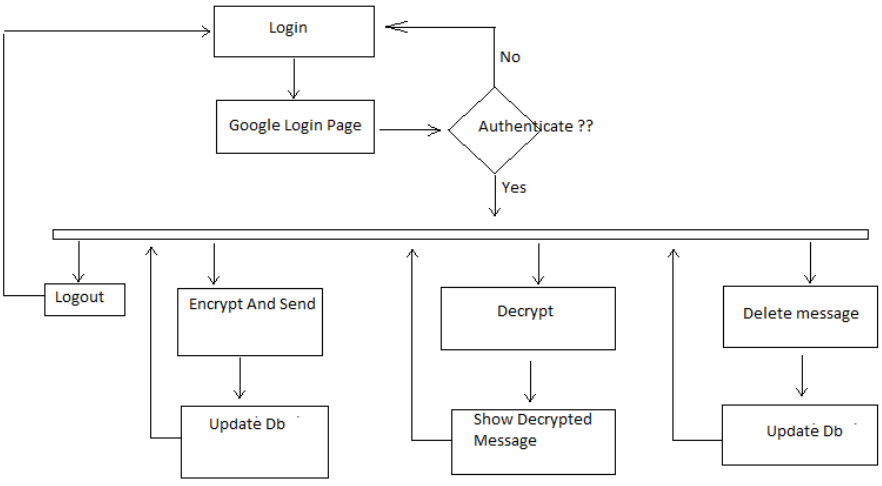

Figure 3.1: Block Diagram of the Email Server

Using this secure messaging system does not require any kind of software installation or key exchange. The system has been build on the very model of an Email Server.

3.1 User Authentication

The complete mail server system was developed on the Google AppEngine. Gmail login and authentication system was used for authentication. This platform also allowed us to use the Gmail database for user authentication. Thus, Logging into this system would require for a user to have a Gmail account.

3.2 Send Message

Sending of only Encrypted messages to other users was possible. The user could choose between the two encryption algorithms and the various key sizes depending on the sensitivity of his messages. The standard SSL communication was used to get the messages to the server. At the server, the messages were encrypted and then only were used to update the database. The receiving user could only see these encrypted messages.

3.3 Decrypt

The user could only see the inbox of his service page with encrypted messages. Plaintext or the original message was made available when the user chose to decrypt. The plaintext was displayed on the screen below the inbox pane. The original message was not saved hence the message would only appear on the screen till the page is not refreshed. If the user wants to read any message he will have to decrypt even if the particular message had already been decrypted earlier.

Chapter 4

Datastore

The App Engine Datastore is a schemaless object datastore providing robust, scalable storage to applications on the Engine. The datastore holds objects known as entities. An entity can have one or more properties, named values of one or more supported data types: for instance, a property can be a string, an integer, or a reference to another entity. Each entity is identified by its kind and also has a key which identifies an entity among its kind. The google App Engine does not support the more schema based SQL database to store information.

4.1 Entities

Each datastore entity entity is of a particular kind, which categorizes the entity for the purpose of queries: for instance, a human resource application might represent each employee with an entity of kind Employee. In addition each entity has its own ley which uniquely identifies it. The key has following components

- The Entity's kind
- An identifier

The identifier is assigned when the entity is created. It is a part of the entity key, associated with the Entity as a whole, and hence cannot be changed.
Entities used in the development of Secure Message Service can be grouped under two keys- messages and rsaKey. These Entities include

- Entity with key Message to store the messages sent by users with the following properties

  1. Name of the user sending the message
  2. Name of the user for whom the messagel was intended
  3. The Encrypted message
  4. Time and Date when the message was sent
  5. A unique identifier to identify the message

- Entity with key RSAkey to store the key parameters for RSA encryption

  1. Random numbers P and Q
  2. the Key pair E and D

A view of the datastore for Entity with key 'Message' is shown in figure 4.1. It shows the preview of the datastore as vied on google Datastore viewer.

ID/Name	content	date	fromUser	id
id=1001	~!--~2~"~¡~;~>H"ß~$~¯~ª~ rç~¤~¶çqw·Láè~§~¦~µ~!~&~8~$¦~°~"E§é~>°@¯~!¦ZÇ.~§oäB0ø;Õ~ ¹~ ÏL¸ç±~¢î~(9~7F~*~!Aâ~´~(~;¬î5vÊG^-:~ ~@~µ~ºðlv~=¦DÀæO¾Æ~ ~»äÄéE-D«a~«ê~8~µ~ nv~¥á~"~7~ r~)²~&xð~´	2012-04-05 05:35:26.810000	skp.nitrkl@wmail.com	1333604126810
id=1	~ ~·µb£ÿî~¢~»¼~¼è©ün~ ¸zªµf>Õ)îÓ6~9~¨è~!w~¾$À4~¦Ê~¡~£åwöÃ~!uÇXq~¬¦~@v}?O~¢Ù~¨~ ~.`~1Û$Ù¯rÊ~%¦~>~¦ô~!Së;²êö1iÎËpã§£	2012-04-04 22:18:52.863000	keshavaga@wmail.com	1333577932863
id=11005	~ WMï~(~¨É~9wî3~3~;ã$=~»íj~!0~ ?úx`~<l]~µ¸$+¢~¸¼ÄÇölv~º~ Ð8~ à~¥[=«`ÑEÐ´í~ ±4*u~ $~¥Ø~: £~¼Ô#2[T~®ö~ ~ªÿÓlè6~ ö~?~2ñÇfñ©ë~5%~*v~1îÙ¤¢~¯~ ?úx`~<l]~µ¸$+¢~¸¼ÄÇölv~º~ L~7~£Å÷å~/ªyÚ~"ûÔ~ºZ*ç<~:ô~!~·4]½ÿ~#p~¬öïÒè~7Èâil~¢~´Ç~ L~7~£Å÷å~/ªyÚ~"ûÔ~ºZ*ç<~:ô~...	2012-05-07 09:05:02.400000	keshavaga@wmail.com	1336381502400
id=11007	~ àüD¼©l*~©~8v$a~1`~?~¤l§±^~ ~,~<~«b®~,~/~ QG¸~ÄÖwz2~$#ð~!å~§&ö~£Ô~£~(f~>6ßÎ~%3ò2~1´m	2012-05-07 10:20:39.059000	keshavaga@wmail.com	1336386039059
id=27001	~ o~+~¥¬+¯uëô`VÉÕ55º$}1+~ ðÒ;~2»~¥)~¬µü½ð~®Õ{fô^î~ º(~´ nB~´Eúy~±~·÷~¦K~9Û©'~!ç¼V÷~ ¬Đnà"~ººÚLèiË£~¯u+~!ç¼V÷~¬Đnà"~ººÚLèiË£~¯u+~!~9è~6HÂ¹¶KÇ¸®ÞèÝ~?hUôë~ª	2012-05-07 08:59:49.725000	keshavaga@wmail.com	1336381189725
id=1	~ :Z¦Ô&~¬X~)~±=¶ø>y	2012-04-05 14:43:53.032000	kkttgg@wmail.com	1333637033032
id=2003	~ ~3'~¤õ~¥ª~'~!ûZ~%²%~!=(ð~0~)~¢|«RGJÉú~ =Úlð~,\ÃxÆÚHL~//~ PªÂ~.ÙÙôo¹~2¶9~¶Ü~ PªÂ~.ÙÙôo¹~2¶9~¶Ü~!«âP~¤~©~<´Ð*P~@¿d~5~!¢l¿~,P~±~-Õ~¨l2~ªlZ~!¸~µh´~µ~ì~ £2hcl~=T~!«âP~¤~©~<´Ð*P~@¿d~5~ ~4ç~¼(é~&~?ªU~#~+~:?~6~ ~èÄó~£{ò;ç~·ec~ ~¡~!¢l¿~,P~±~-Õ~¨l2~ªlZ...	2012-04-04 22:22:44.390000	manabesh.mandal@wmail.com	1333578164390
id=2004	~!~µ~¯ëS!ªÎÇWØ/¢JU~ Ì§?QZ~?~3è#;~¸ó~3~>~!´P¥k[ÜGuäïBnSë~!ÚV~ª~´ÉñÜ<N'g¼ ¥»~!~¾~=9B~+ª±µhª!þÕ<~ Ì§?QZ~?~3è#;~¸ó~3~>~!¡Pt½~7ùú4ÆM~ªï§~(~!Ì~7'~ (¿"~)`N_ÇfÊÖ~!¡Pt½~7ùú4ÆM~ªï§~(	2012-04-04 22:24:33.295000	keshavaga@wmail.com	1333578273295
id=1	~ 1£¦iXb?~8|©RÉtó~!~½UÉm9Gû~¶~?óg÷ºÉ~!´]~½~¦á}À"_~,~+yjÿ~!~#@~9~¾~"±r#tJ(/b~ T»~£¥[½ã,- $ø]àæ~!´]~½~¦á}À"_~,~+yjÿ~!]À~ D§~£Ðð@t6«~0F~ T»~£¥[½ã,$ø]àæ~!´]~½~¦á}À"_~,~+yjÿ~ ÁH±c~¬QèK'ýîVóÅ~ T»~£¥[½ã,$ø]àæ~!~¡e~?B=£ègØ{ê~é~ T»~£¥[½ã,$ø]àæ~!]À~ D§...	2012-04-05 05:11:06.085000	skp.nitrkl@wmail.com	1333602666085

Figure 4.1: Datastore preview for Entity with key 'Messages'

4.2 Queries and Indexes

Entities can be retrieved from datastore by its key and also by query on the values of properties. A query operation on entites is very similar to query on batabase tables. Indexes are, like, tables containing the query's potential results in a desired order. Thses help in fast retrieval of information from the datastore.

Chapter 5

Implementation

The System was developed as a web interface using javascript. The main page of the Messaging Service is shown in figure 5.1

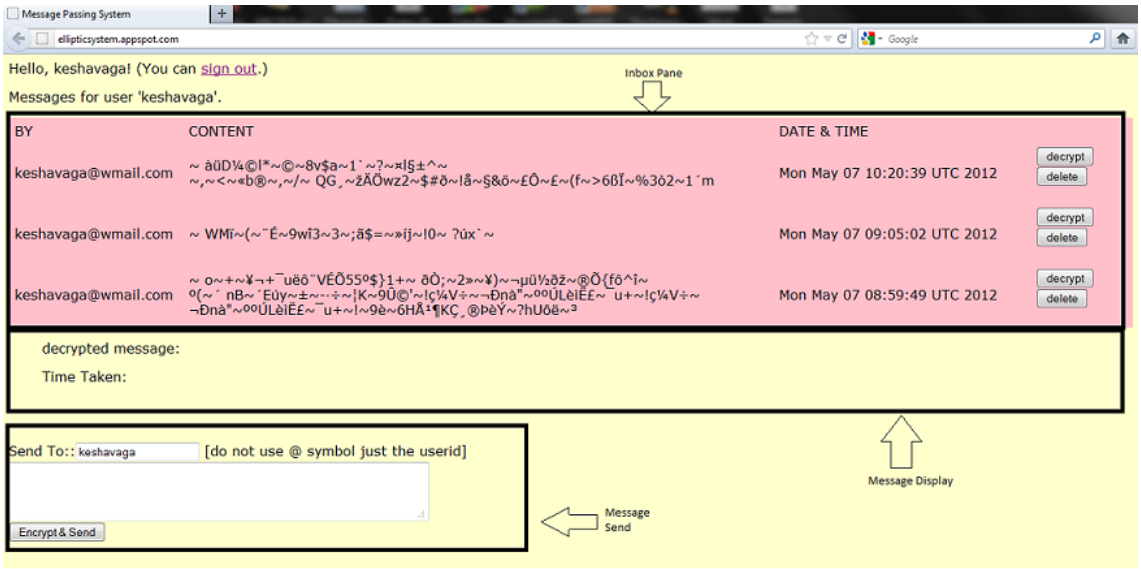

Figure 5.1: View of the main User Page

The figure shows that the user interface has three main regions

1. The inbox pane: where the user sees all the incoming encrypted messages
2. Messages Pane: where the user will see the plaintext or the original message after Decryption
3. Send Pane: from where the user sends messages to other users

The complete background system for encryption and message storage and forwarding was coded in java. The messages were transfered using POST method in java.

5.1 Inbox Pane

This part of the user interface shows the messages sent to the current user by all the other users. This is divided into 4 colums showing the names of the sending user's, the encrypted message content, The date and time when the message was sent and the final column contains buttons to either decrypt the message or to delete it.View of the inbox pane is shown in figure **??**

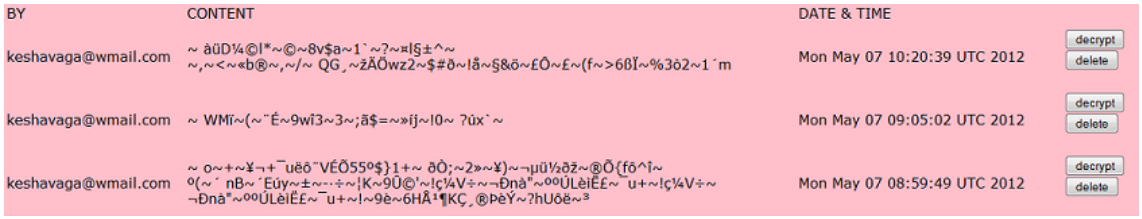

Figure 5.2: View of the Inbox Pane

5.2 Message Pane

This is the part of the interface which displays the message to the user after decryption. It also displays the time taken to encrypt or decrypt a message. The figure **??** shows the content of this pane when an inbox message is decrypted.

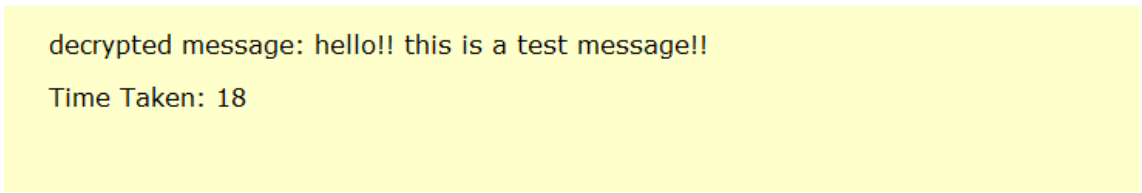

Figure 5.3: View of the Message Pane

5.3 Send Pane

This part of the interface is used to send message to any user on the system. It takes two inputs- the message and the user id of the recieving user. Figure **??** shows the send pane.

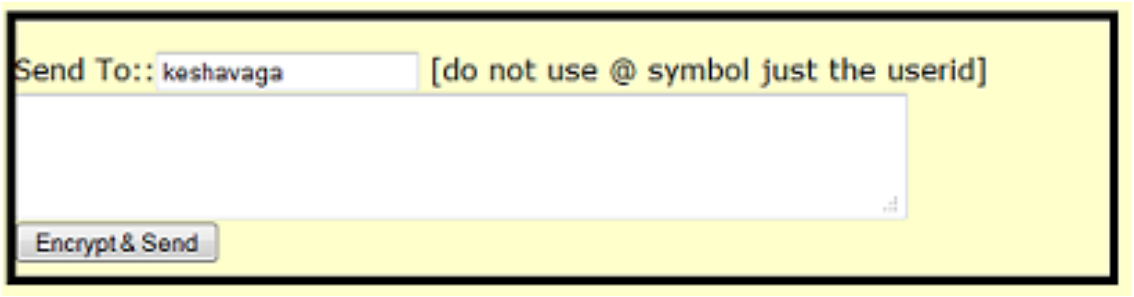

Figure 5.4: View of the Send Pane

5.4 System Test

The System was rigorously tested on App Engine for message sizes ranging from 1 chracter to 2000 characters. The results of Encryption and Decryption for the said message size range was accurate for both the algorithms and all key sizes implemented (sizes: 112 bits, 160 bits, 256 bits for ECC and 1024 and 512 bits for RSA). The time taken for encryption increases with the key size and the message size. An encryption time as high as 4 seconds was noted for message 1000 characters long.The use of SSL during transmission and encryption during storage ensures the security of the sent message.

No redundancy was recorded- the ciphertext generated was always a different one even for the same message being sent twice.

Chapter 6

Conclusion

The Message Passing System can be said to be completely secure as the original message was not stored anywhere on the system. The mail server database also stores on the Encrypted message. Thus, compromising the server and its database will not effect the security of the messages. Further, SSl communication ensures message security during transmission.

It also has certain drawbacks,as in, the message contents get encrypted before any other third party processing. Hence, the spam and mallware also get encrypted and the detection of such threats become very difficult. Scan of the message before encryption again has the risk of exposing the sensitive information.

The time taken by the algorithm keeps on incresing with the key size and the message size. A considerable delay of more than 4 second is recorded for message size greater than 1000 characters. This delay is undesirable and needs to be redused by optimizing the encryption algorithm.

Bibliography

[1] W. Stallings. *Cryptography and Network Security.* Prentice Hall, 1999.

[2] Scott Vanstone Darrel Hankerson, Alfred J. Menezes. *Guide to Elliptic Curve Cryptography.* Springer-Verlag New York, Inc., 2004.

[3] Bruce Schneir. *Applied Cryptography.* John Wiley & Sons, 1996.

[4] Documentatio, http://code.google.com.

[5] Elliptic tutorials, http://www.certicom.com/index.php/ecc-tutorial.

[6] V. Miller. Uses of elliptic curves in cryptography. In *Advances in Cryptography.* Springer Verlag.