8-2017

A New Method IBE Interfaced with Private Key Generation and Public Key Infrastructure to Achieve High Data Security

Phanitha Kommareddy
St. Cloud State University, pkommareddy@stcloudstate.edu

A New Method IBE Interfaced with Private Key Generation and Public Key Infrastructure to Achieve High Data Security

by

Phanitha Kommareddy

A Starred paper

Submitted to the Graduate Faculty of

St. Cloud State University

in Partial Fulfillment of the Requirements

for the Degree

Master of Science

in Information Assurance

August, 2017

Starred Paper Committee:
Dennis Guster, Chairperson
Mark Schmidt
Balasubramanian Kasi

Abstract

Identity based public key encryption supports basic presentation of public key cryptography by allowing a component's public key to be gotten from an optional recognizing verification worth, for instance, name or email address. The fundamental of identity based cryptography is in tremendously reducing the prerequisite for, and reliance on, public key verifications. Although some captivating character based frameworks have been made already, none are great with prominent public key encryption calculations. In addition, it is in a general sense hard to encourage fine-grained conflict with character based cryptography. Intervened RSA (mRSA) is a fundamental and rational system for section a RSA (Rivest, Shamir and Adelman) private key between the client and a Security Mediator (SEM). Neither the customer nor the SEM can cheat each other since each crypto-reasonable operation (stamp or unscrambling) incorporates both sides. mRSA allows quick and fine-grained control of clients' security benefits. In any case, mRSA still relies on upon conventional public key announcements to store and grant public keys. In this paper, we show IB-mRSA, an essential variety of mRSA that joins identity based and mediated cryptography. Under the discretionary prophet demonstrate, IB-mRSA with OAEP (Optimal asymmetric encryption padding) is showed up as secure against adaptable picked cipher text strike as standard RSA with OAEP. Also, IB-mRSA is direct, reasonable, and consummate with current public key establishments.

Keywords: Cipher text, Encryption algorithms, Identity-based mRSA, public key encryption, private key encryption, SEM.

Table of Contents

Chapter Page

List of Figures

Chapter I: Introduction

A safe server despite giving a guaranteed foundation to empower your Web applications, and Web server plan expect an essential character in your Web application's security. Gravely organized server can affect unapproved access. Understanding the threats to your Web server and having the capacity to recognize appropriate countermeasures licenses you to suspect various attacks and miracle the routinely creating amounts of aggressors. This system gives bidirectional encryption of correspondences between a customer and server, which guarantees against listening stealthily and upsetting and/or fabricating the substance of the correspondence (Lewko & Waters, 2011).

The site that I proposed to talk with furthermore protecting that the substance of correspondences between the client and the site can't be scrutinized or made by any outsider. Secure Server Plus application has principally twofold login security. That is, in the wake of marking into the application customer gets a mystery key on his selected Gmail ID. This mystery key must be embedded in the pop-up box appeared in the wake of marking into SSP Application. This application has two functionalities, one is Encryption and the other is Decryption. Encoding is the handiness in which the record to be systematized over the mail in firstly isolated in 4 an equalization of in byte arrangement and a while later encoded using unmistakable encryption computations (Boneh & Hamburg, 2008).

After Encryption records would be sent to the recipient through Gmail at the recipient end, he will download the archives and using SSP Application data as a piece of reports would be unscrambled and mixed.

Client security is likewise required in cloud. By using assurance, the cloud or distinctive customers do not have the foggiest thought regarding the identity of the other hub. The cloud can contain the hub introduces the information in the cloud, and in like way, to give advantages the cloud itself is mindful. The genuineness of the customer who stores the data is likewise bolstered. There is likewise a necessity for law approval isolated from the specific responses for surety security and safe house. Various encryption systems have been utilized to secure data on cloud to examine the data while doing computations on the information (Waters, 2011).

Distinctive strategies have been recommended to safeguard the data substance assurance by method for affirmation control. Identity based encryption (IBE) was at first introduced by Shamir, in which the sender of a message can demonstrate a character such that exclusive a recipient with organizing identity can unscramble it. A few years sometime later, Fuzzy Identity-Based Encryption is proposed, which is generally called Attribute-Based Encryption (ABE). In such encryption contrive, an identity is viewed as a plan of clear attributes, and deciphering is possible if a decrypt's character has a couple covers with the one indicated in the ciphertext. Ahead long, more wide tree-based ABE arranges (Hajny & Malina, 2012), Key-Policy Attribute-Based Encryption (KP-ABE) and Ciphertext-Policy Attribute-Based Encryption (CP-ABE), are acquainted with express more expansive condition than direct 'cover'. They are accomplices to each different as in the alternative of encryption technique (who can or cannot translate the message) is settled by different social affairs (Li et al., 2011).

In the KP-ABE, a ciphertext is connected with a game-plan of characteristics, and a private key is associated with a monotonic access structure like a tree, which portrays this present client's personality (e.g., IIT AND (Ph.D. OR Master). A client can unscramble the ciphertext if and just if the path tree in his private key is fulfilled by the characters in the ciphertext. In whatever claim, the encoding system is portrayed in the keys, so the Encrypter does not have full control over the encoding access. He needs to trust that the key generators issue keys with the right structures to the privilege customers. Besides, when a re-encryption happens, a large portion of the customers in the same system must bear their private keys, re-issued remembering the final objective to go to the re-encoded circles, and this procedure causes immense issues in the usage. Of way, those occasions and working cost are all esteemed in the CP-ABE. In the CP-ABE, ciphertexts are made with an entry structure, which shows the encryption approach, and private keys are made by qualities (Li, Ren, Zhu, & Wan, 2009). A client can unravel the ciphertext if and just if his characteristics in the private key fulfill the section tree exhibited in the ciphertext. Along these lines, the Encrypter holds a complete power about the encoding system. Moreover, the beginning now issued private keys will never be adjusted unless the entire system reboots.

In an average open key foundation (PKI) setting, a client's open key is expressly encoded in an open key authentication which is, basically, an authoritative between the declaration holder's personality and the guaranteed open key. This basic model requires general trust in testament guarantors (Certification Authorities or CAs). It has some surely understood and vexatious symptoms, for example, the requirement for cross-space trust and testament denial (Camenisch, Neven, & Rückert, 2012). The fundamental issue, in any case,

is the essential supposition that all testaments are open, universal and, consequently, promptly accessible to anybody. We watch that this suspicion is not generally sensible, particularly, in remote (or any shortcoming inclined) systems where network is sporadic. Interestingly, personality based cryptography changes the way of getting open keys by building a coordinated mapping amongst characters and open keys. Character based cryptography in this way extraordinarily diminishes the requirement for, and dependence on, open key testaments and affirmation powers. As a rule, character based encryption and personality based marks are helpful cryptographic instruments that encourage simple presentation of, and/or transformation to, open key cryptography by permitting an open key to be gotten from subjective distinguishing proof values, for example, email addresses or telephone numbers. In the meantime, personality based strategies extraordinarily rearrange key administration since they diminish both: the requirement for, and, the quantity of, open key declarations (Shahandashti, & Safavi-Naini, 2009).

The idea of character based open encryption built up an exquisite Identity-Based Encryption system (BF-IBE) in view of Weil Pairing on elliptic bends. BF-IBE speaks to a noteworthy development in cryptography. By and by, a personality based RSA variation has stayed slippery for the straightforward reason that a RSA modulus n (a result of two substantial primes) cannot be securely shared among different clients. Another prominent downside of current personality based cryptographic strategies is absence of backing for fine-grained denial. Denial is regularly done through Certificate Revocation Lists (CRLs) or comparative structures. Nonetheless, IBE expects to disentangle endorsement administration

by getting open keys from characters, which makes it hard to control clients' security privileges (Baudron, Pointcheval, & Stern, 2000).

In this paper, we propose a straightforward character based cryptosystem created on some Mediated RSA (mRSA). mRSA is a down to earth and RSA-good technique for part a RSA private key between the client and the security middle person, called a SEM. Neither the client nor the SEM knows the factorization of the RSA modulus and neither can decode/sign message without the other's assistance. By righteousness of requiring the client to contact its SEM for every unscrambling and/or signature operation, mRSA gives quick and fine-grained renouncement of clients' security benefits. Based on top of mRSA, IB-mRSA mixes the elements of character based and interceded cryptography furthermore offers some useful advantages. Like mRSA, it is completely perfect with plain RSA. Except for the personality to-open key mapping, it requires no uncommon programming for imparting parties. IB-mRSA additionally permits discretionary open key testaments which encourages simple move to an ordinary PKI. All the more for the most part, IB-mRSA can be seen as a basic and handy method between operable with regular present day PKIs (Bellare, Boldyreva, & Micali, 2000).

In the meantime, IB-mRSA offers security tantamount to that of RSA, gave that a SEM is not traded off. In particular, it can be demonstrated that, in the arbitrary prophet model, IB-mRSA with OAEP is as secure – against versatile picked ciphertext assaults–as RSA with OAEP (Bellare, Desai, Pointcheval, & Rogaway, 1998).

Need of the Study

1. To study the technology of cloud data security.
2. To study the emergency of developing public and private data security in cloud computing.
3. To analyze the architecture and present the output results.

Problem Statement

The research problem statement is as follows:

1. To implement Identity based encryption system in a private and public key generation system.
2. To provide data security and privacy in cloud computing
3. To analyze and present the program in Java.

Hypothesis

The research is mainly focuses on the developing a secure data storage and sharing using private and public key system by implementing Identity based algorithm.

Advantages

- Unique identifier of the recipient (email) is used to calculate a public key.
- A trusted third-party server, uses a cryptographic algorithm to calculate the corresponding private key from the public key.
- Recipients can generate their own private keys directly from the server as needed.
- Keys expire, so they don't need to be revoked.
- Enables automatic expiration, rendering messages unreadable after a certain date.

- Compared with typical public-key cryptography, this greatly reduces the complexity of the encryption process for both users and administrators.

Limitations

The research has some limitation as follows:

1. The research limits in analyzing the cloud computing in networking.
2. The study mainly focuses on implementation of private and public key using services in cloud technology.
3. The data collection for the research and practical programming may not be accurate.

Research Methodology

The research sources are divided into two sources:

Secondary sources: Secondary data for the research will be gathered from the

- Books,
- Magazines and Articles,
- IEEE Paper

Primary sources: A primary resource for the research includes websites:

- World Wide Web

Chapter II: Literature Review

Bitar, Gringeri, and Xia (2013) look into says cloud today defy a couple of challenges when encouraging line-of-business applications in the cloud. Key to an extensive number of these troubles is the limited support for control over cloud system limits, for instance, the ability to ensure security, execution sureties or division, and to adaptably intercede middle boxes in application associations. In this paper, we demonstrate the arrangement and utilization of a novel cloud sorting out structure called CloudNaaS. Customers can impact CloudNaaS to pass on applications extended with a rich and extensible course of action of system limits, for instance, virtual system detachment, custom tending to, organization partition, and versatile intercession of various middle boxes. CloudNaaS primitives are particularly executed inside the cloud structure itself using quick programmable system segments, making CloudNaaS exceptionally profitable.

According to Varghese and Vigila (2015), distributed computing a conveyed organize for sharing information over web, serves as an online information reinforcement with versatility. The paper portrays different classes of mists relying upon the utilization of cloud furthermore on the administrations gave by the cloud. Information security is one of the significant difficulties confronted by cloud suppliers and cloud clients. Cryptography is proposed as the proper answer for securing the cloud information. Audit on a portion of the current cryptographic techniques for securing the information put away in the cloud is likewise incorporated into this paper. The information proprietors can transfer information on to the cloud; can likewise make authorizations on the transferred information to control its entrance by different sorts of clients.

As indicated by Schoo et al. (2011), cloud processing is for the most part considered as an engaging organization display taking after the customer's obligations regarding endeavor and operations are limited, and costs are in quick association with use and intrigue.

As per Bitar et al. (2013), server ranch and cloud designs continue progressing to address the requirements of far reaching scale multi-inhabitant server homesteads and fogs. These requirements depend on seven estimations: flexibility in figuring, stockpiling, and information exchange limit, adaptability in system organizations, adequacy in resource utilization, deftness in organization creation, cost efficiency, organization faithful quality, and security. This article focuses on the underlying five estimations as they identify with systems organization. In a server ranch, server and limit resources are interconnected with package switches and switches that suit the information transmission and multi-tenant virtual systems organization needs.

Server homesteads are interconnected over the wide zone system through guiding and transport advances to give a pool of advantages, known as the cloud. Quick optical interfaces and thick wavelength-division multiplexing optical transport are used to suit high-confine transport intra-and between datacenter. This article reviews diverse trading, coordinating, and optical transport advancements, and their fittingness in tending to the systems organization needs of immeasurable scale multi-inhabitant server ranches.

As per Zissis and Lekkas (2012), the late advancement of dispersed processing has unquestionably adjusted everyone's perspective of base structures, programming transport and change models. Reckoning as a transformative wander, taking after the move from incorporated PC PCs to client/server game plan models, dispersed registering incorporates

parts from system enrolling, utility figuring and autonomic handling, into an imaginative association development demonstrating.

This paper proposes showing a Trusted Third Party, entrusted with ensuring specific security qualities inside a cloud circumstance. The proposed plan calls upon cryptography, especially Public Key Infrastructure cooperating with SSO and LDAP, to ensure the approval, respectability and mystery of included data and correspondences. The game plan, demonstrates a level of organization, open to every included component, that comprehends a security arrange, inside which essential trust is kept up.

As indicated by Subashini and Kavitha (2011), distributed computing is a way to deal with grow the point of confinement or incorporate capacities intensely without placing assets into new structure, get ready new work compel, or allowing new programming. It expands Information Technology's (IT) existing limits. In the latest couple of years, disseminated registering has created from being an ensuring business thought to one of the rapidly creating areas of the IT business. In any case as more information on individuals and associations are placed in the cloud, concerns are beginning to wind up distinctly about precisely how safe an area it is. Despite all the development including the cloud, try customers are as yet reluctant to send their business in the cloud. Security is one of the critical issues which diminishes the improvement of conveyed figuring and burdens with data assurance and data protection continue plaguing the business division. The happening to a moved model should not mastermind with the obliged helpfulness and capacities appear in the present model. Another model centering at upgrading highlights of a present model must not risk or cripple other indispensable highlights of the present model.

The auxiliary designing of cloud stances such a hazard to the security of the present advancements when passed on in a cloud space. Cloud organization customers should be watchful in appreciation the threats of data breaks in this new condition. In this paper, a survey of the various security risks that speak to a peril to the cloud is presented. This paper is a review more specific to the particular security issues that has transmitted in light of the method for the organization movement models of an appropriated figuring system.

Houidi, Mechtri, Louati, and Zeghlache (2011) inquire about presents work-in-progression on the cloud organization provisioning across over various cloud providers. The work acknowledges the improvement of Cloud Brokers amidst customers and cloud providers. The delegates part customer requests and assurance provisioning from various providers. A cautious part computation is created to capably part the cloud requests among the various cloud stages with the purpose of decreasing the cost for customers. This part is figured as a Mixed Integer Program and this is united with Open Flow and NOX advancements that achieve stream based between cloud sorting out. Another controller module is made and joined in NOX to organize the Open Flow switches for between cloud way establishments.

As per Potlapally, Ravi, Raghunathan, and Jha (2006), security is turning into an ordinary sympathy towards an extensive variety of electronic frameworks that control, impart, and store delicate information. A vital and developing classification of such electronic frameworks are battery-fueled versatile machines, for example, personal digital assistant (PDAs) and mobile phones, which are extremely compelled in the assets they have, in particular, processor, battery, and memory. This work concentrates on one essential

imperative of such gadgets battery life-and looks at how it is affected by the utilization of different security systems. In this paper, we first present a far reaching examination of the vitality necessities of an extensive variety of cryptographic calculations that shape the building squares of security systems, for example, security conventions.

Chapter III: System Analysis

Existing System

As per the pervious study, Brent Waters propose a Multi-Authority Attribute-Based Encryption (ABE) structure. In our hypothetical record, any social affair can transform into a force and there is no prerequisite for any overall coordination other than the yield of a hidden arrangement of typical reference parameters. A social event can basically go around as an ABE power by hitting an open key and issuing private keys to a few customers that reflect their traits. A customer can encode data with respect to any boolean condition over attributes issued from any picked set of capacities. Finally, our structure does not require any central force. In building up our system, our greatest specific impediment is to make it plan safe.

Nevertheless, in our structure each part will create from a possibly particular force, where we acknowledge no coordination between such powers. We make new techniques to tie key fragments together and hinder interest attacks between customers with different overall identifiers (Bellare et al., 1998). They demonstrate our system secure using the late twofold structure encryption strategy where the security check works by first changing over the test ciphertext and private keys to a semi-pragmatic structure and a while later fighting security. We accept after a late variety of the twofold system proof strategy as a result of Lewko, gathers and collect our structure using bilinear social affairs of composite sales.

As demonstrated give a general structure to building character based and broadcast encryption systems. In particular, we get a general encryption structure called spatial encryption from which various systems with a blend of properties take after. The cyphertext size in each one of these structures is independent of the amount of customers included and is

just three social affair parts. Private key size creates with the multifaceted nature of the fabric. One motivation behind these results gives the main show HIBE system with short ciphertexts. Broadcast HIBE deals with a trademark issue doing with identity based mixed email (Bellare & Rogaway, 1995).

Proposed System

The need to make accessible bona fide duplicates of substances, public key is a drawback to use public key cryptography (PKC). The standard strategy for doing this is to use the general public key structures, in which certification control issues a confirmation which ties a client's identity with his/her public key. With ID-based cryptosystems, this coupling is redundant as the character of the element would be his/her public key. In ID-based PKC, everybody's public Keys are fated by data that interestingly distinguishes them, for example, their email address (Boneh, Ding, & Tsudik, 2002).

This idea is unique inspiration for ID-based encryption to untangle support organization in email systems. Every substance in the system sends his/her identity to a trusted outsider called the Key Generation Center (KGC), to get the private key. The private key is figured utilizing the private key of the KGC and the identity of the client. Key escrow is natural in ID-based frameworks since the KGC knows all the private keys. For different reasons, this makes execution of the innovation much less demanding, and conveys some additional data security advantages. ID-based PKC (ID-PKC) remained a hypothetical idea until were proposed. A portion of the issues to be tended to contrast the ID-based frameworks and the standard PKI maintained public key cryptography (Boneh, Ding, Tsudik, & Wong, 2001).

Algorithm

Setup→ The setup estimation takes no information other than the specific security parameter. It renders individuals as a rule parameters PK and a specialist key MK.

Encode (PK, M, A) →The encryption computation takes as data individuals by and large parameters PK, a message M, and a passage structure, an over the universe of attributes. The count will encode M and produce a ciphertext CT such that only a customer that holds a game plan of characteristics that satisfies the passageway structure will hold the ability to decode the message. We will expect that the ciphertext irrefutable contains A.

Key Generation (MK, S) → The key time figuring takes as data the master key MK and an arrangement of characteristics S that portray the key. It bears a private key SK.

Decode (PK, CT, SK) →The unscrambling count takes as information the all-inclusive community parameters PK, a ciphertext CT, which contains a passage system An, and a private key SK, which is a private key for a set S of qualities. For the situation that the set S of qualities satisfies the passageway structure A then the computation will translate the ciphertext and return a message M.

Delegate (SK, $\tilde{S}$) →The representative count takes as data a puzzle key SK for some arrangement of properties S and a set $\tilde{S} \subseteq S$. It moves over a secret key SK for the game plan of $\tilde{S}$ qualities S (Boneh & Franklin, 2001).

Particularly we set emerge the advantage of the record access, and we evaluated a perfect chance to touch base at one advantage tree and depend on its affirmation parameter. When all is said in done, the computation overhead of Li is much higher than others in light of the way that their outline incorporates various more exponentiations and bilinear mappings in

light of the commitment. The encryption/unscrambling under different archive sizes did not demonstrate colossal differences when record sizes are significant (≥20MB), in light of the way that the run times are controlled by the symmetric encryption (AES-256). At last, however our run times are plotted in light of the fact that the advantage creation is the additional routine in our outline.

Identity-Based Public Key Cryptography

One of the troubles characteristic in running a PKI is in the overseeing of the authentication and related key. Identity based cryptography was made as a method for defeating this issue. The plan gave a mark calculation, however couldn't be utilized for encryption. It is just as of late that an effective character based encryption framework was proposed.

The center contrast between an ID-PKC and a conventional unbalanced calculation in the method for producing the keys. The distinction is identifiable in two ways:

- As said above, in both the mark and encryption variations, people in general keys are created from openly identifiable data. This permits a customer A to produce general society key of another customer B without doing an inquiry in a catalog or approach B for a duplicate of their key.
- Because of the science that support the calculations, the production of the private key requires the learning of an expert mystery that is held by the Trusted Authority (TA), who is the simple of the CA in a PKI.

As of late, it has been perceived that an identity need not be the main determinant of a customer's public key. For instance, data, for example, the customer's position inside an

association, the legitimacy time frame for the keys, and so forth can be incorporated into the information used to infer the key pair. This outcome in the more extensive idea of identifier-based public key cryptography.

Since the TA is clearly in charge of the era of the private key in an ID-PKC instrument, there is an inborn escrow office in the framework. This could possibly be alluring framework. In a PKI, the CA is worried with accepting the realness of the data present in the declaration, while, in an ID-PKC the TA is specifically in charge of creating and disseminating all keying material inside the framework (Coron & Naccache, 2000). There is likewise the prerequisite that TA and customer can set up a free secure channel for the dispersion of private key material. This channel needs to secure both the realness and secrecy of the private key.

Despite the fact that utilizing a customer's way of life as the base for their key pair is exceptionally engaging, it doesn't come without results. The two principle issues that will impact the discourse in the rest of this paper are as per the following:

Coping with the items of common sense of execution are not inconsequential. In the event that we take renouncement as an illustration, since we cannot repudiate a man’s character, there is a prerequisite for extra contribution to the key era process. On the off chance that we incorporate legitimacy dates, key use, and so on then a push toward more extensive utilization of distinguishing data results, driving actually to identifier-based cryptography. We will come back to repudiation issues.

The credibility of the data that is utilized as the character or identifier is presently urgent to the security of the framework. In a PKI, the declaration should show the realness of

distinguishing data. In ID-PKC, in light of the fact that a private key might be produced after general society key, the TA might not have approved the realness of the data identifying with the key pair preceding the general population key's utilization. For instance, A might utilize data it supposes is legitimate to create a public key for B, yet the data An utilizations could either identify with the wrong B, or might be totally invalid according to the TA.

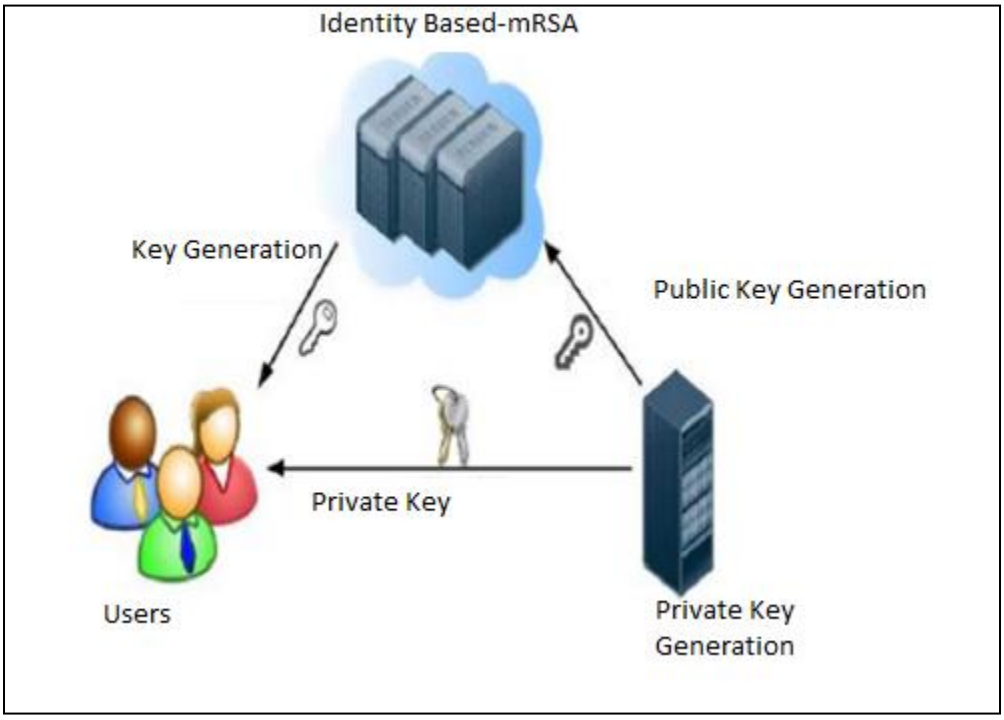


Figure 1: System Architecture

Identity-Based MRSA

The primary component of character based encryption is the sender's capacity to scramble messages utilizing people in general key got from the beneficiary's personality and other open data. The character can be the recipient's email address, client id or any quality exceptional to the beneficiary; basically, a subjective string. To process the encryption key, a productive (and open) mapping capacity KG must be set already. This capacity must be a

balanced mapping from character strings to open keys. The fundamental thought behind character based mRSA is the utilization of a solitary regular RSA modulus n for all clients inside a system (or area). This modulus is open and contained in a system wide endorsement issued, of course, by some Certificate Authority (CA).

To scramble a message for a specific beneficiary (Bob), the sender (Alice) first registers eBob=KG(IDBob) where IDBob is the beneficiary's personality quality, for example, Bob's email address (Fujisaki, Okamoto, Pointcheval, & Stern, 2001). From that point, the pair (eBob, n) is dealt with as a plain RSA open key and typical RSA encryption is performed. On Bob's side, the decoding procedure is indistinguishable to that of mRSA. We push that utilizing the same modulus by numerous clients in a typical RSA setting is totally shaky. It is subject to an unimportant assault whereby anyone using one's information of a solitary key-pair–can essentially consider the modulus and register the other client's private key. Be that as it may, in the present setting, we make a critical presumption that.

All through the lifetime of the system, the enemy can't trade off a SEM. Clearly, without this suspicion, IB-mRSA would offer no security what server: a solitary SEM soften up combined with the trade-off of only one client’s key offer would bring about the bargain of all clients' (for that SEM) private keys. The IB-mRSA supposition is somewhat more grounded than its mRSA partner. Review that, in mRSA, every client has an alternate RSA setting, i.e., a one of a kind modulus. Along these lines, to trade off a given client an enemy needs to break into both the client and its SEM. We now swing to the point by point depiction of the IB-mRSA plan (Ganesan, 1995).

We actualized IB-mRSA for the motivations behind experimentation and acceptance. The product is made out of three sections:

1. CA and Admin Utilities: area endorsement, client key era, (discretionary) declaration issuance and renouncement interface.
2. SEM daemon: SEM process.
3. Customer libraries: IB-mRSA client capacities open by means of an API.

The code is based on top of the well-known OpenSSL library. OpenSSL fuses a huge number of cryptographic capacities and substantial number-crunching primitives. Notwithstanding being productive and accessible on numerous regular equipment and programming stages, OpenSSL sticks to the basic PKCS principles and is in the general population space. The SEM daemon and the CA/Admin utilities are actualized on Linux, while the customer libraries are accessible on both Linux and Windows stages. In the instatement stage, a CA introduces the space wide cryptographic setting, in particular (n, p, q, p', q') and chooses a mapping capacity (presently defaulting to MD5) for all area customers. For every client, two structures are sent out:

1) SEM bundle, which incorporates the SEM's half-key dSEM i, and
2) client group, which incorporates dui and the whole server pack.

The server pack is in PKCS#7 position, which is fundamentally a RSA envelope marked by the CA and encoded with the SEM's open key. The customer pack is in PKCS#12 design, which is a common key envelope additionally marked by the CA and scrambled with the client supplied key which can be a pre-set key, a secret key or a pass-expression. (A client is not expected to have a prior open key.) After issuance, every client group is circulated in an

out-of-band design to the fitting client. Before endeavoring any IB-mRSA exchanges, the client should first decode and check the group. A different utility project is accommodated this reason. With it, the pack is decoded with the client supplied key, the CA's mark is checked, and, at long last, the client's half-key are removed and put away locally. To unscramble a message, the client begins with sending an IB-mRSA ask for, with the SEM group piggybacked. The SEM first check the status of the customer (Oberheide, Veeraraghavan, Cooke, Flinn, & Jahanian, 2008).

Just when the customer is esteemed to be a true blue client, does the SEM procedure the solicitation utilizing the pack contained in that. As said before, to scramble a message for an IB-mRSA, that client's space endorsement should be acquired. Administration of space testaments is thought to be done in a way like that of typical endorsement, e.g., by means of LDAP or DNS.

Chapter IV: Implementation

Screen Shots

Home page.

Figure 2: Home Page

PKG login.

Figure 3: PKG Login

The above figure explains that the page is an admin login page where the admin log's-in by entering his user Id and password.

KU-CSP login.

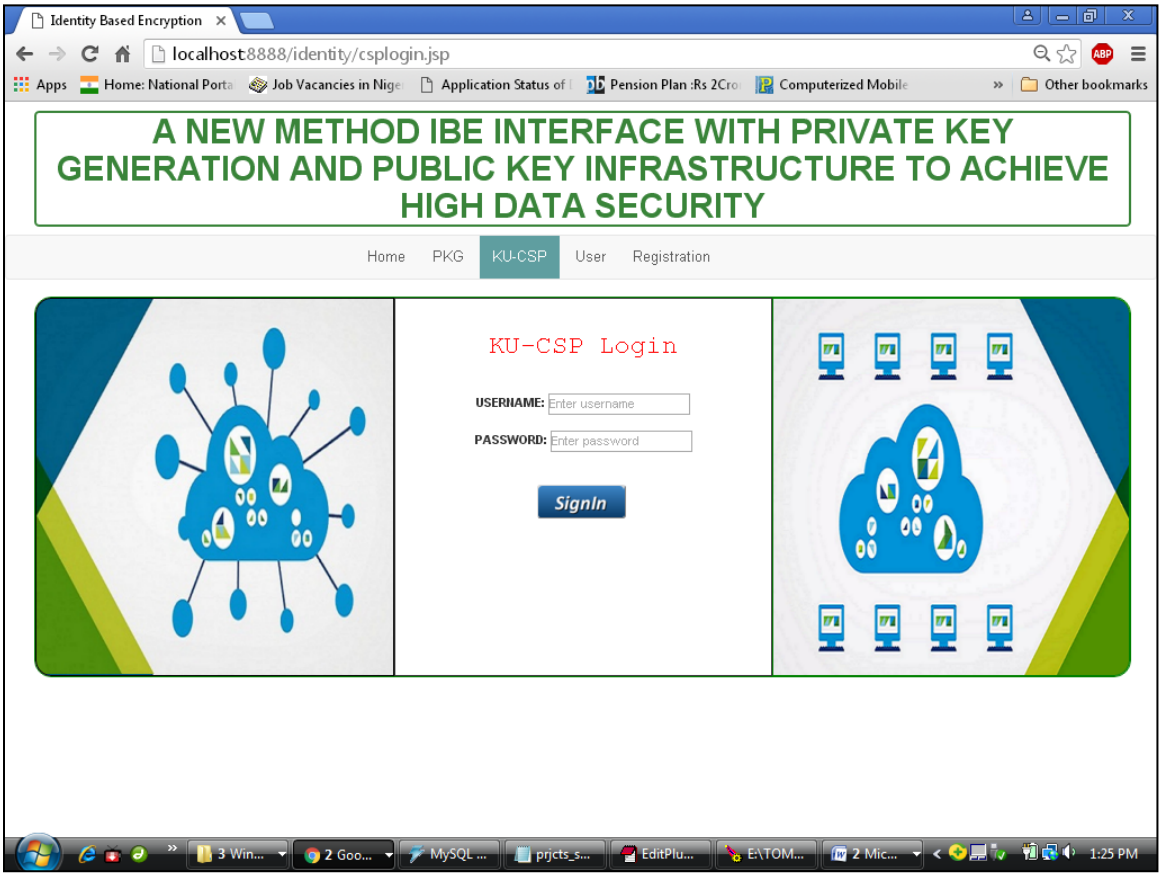


Figure 4: KU-CSP Login

The above figure explains that the page is an admin Ku-CSP login page where the admin log's-in by entering his user Id and password.

User registration.

Figure 5: User Registration

The above figure explains that the page is a user registration by entering his/her details.

User login.

Figure 6: User Login

The above figure explains that the page is a user login page where the user log's-in by entering his user Id and password

Admin home page.

Figure 7: Admin Home Page

Outsource key page.

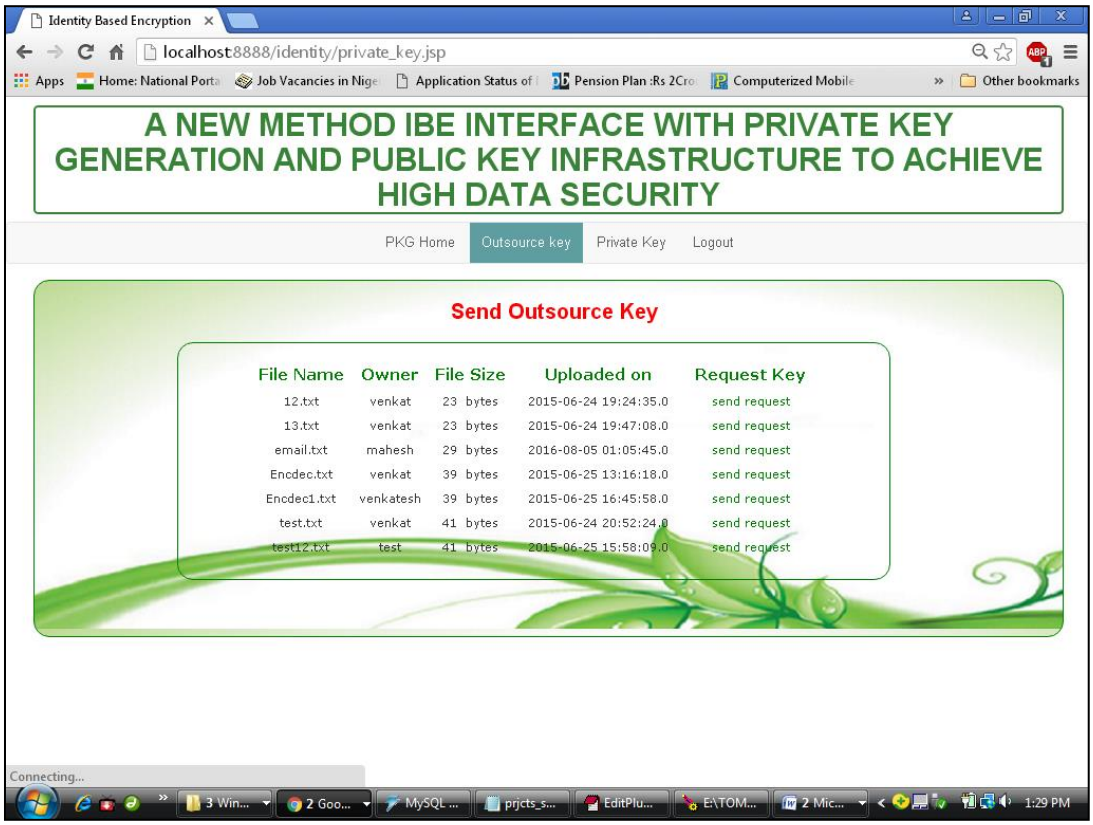


Figure 8: Outsource Key Page

The screen shows the outsource key which sent's public key to the user.

Key distribution.

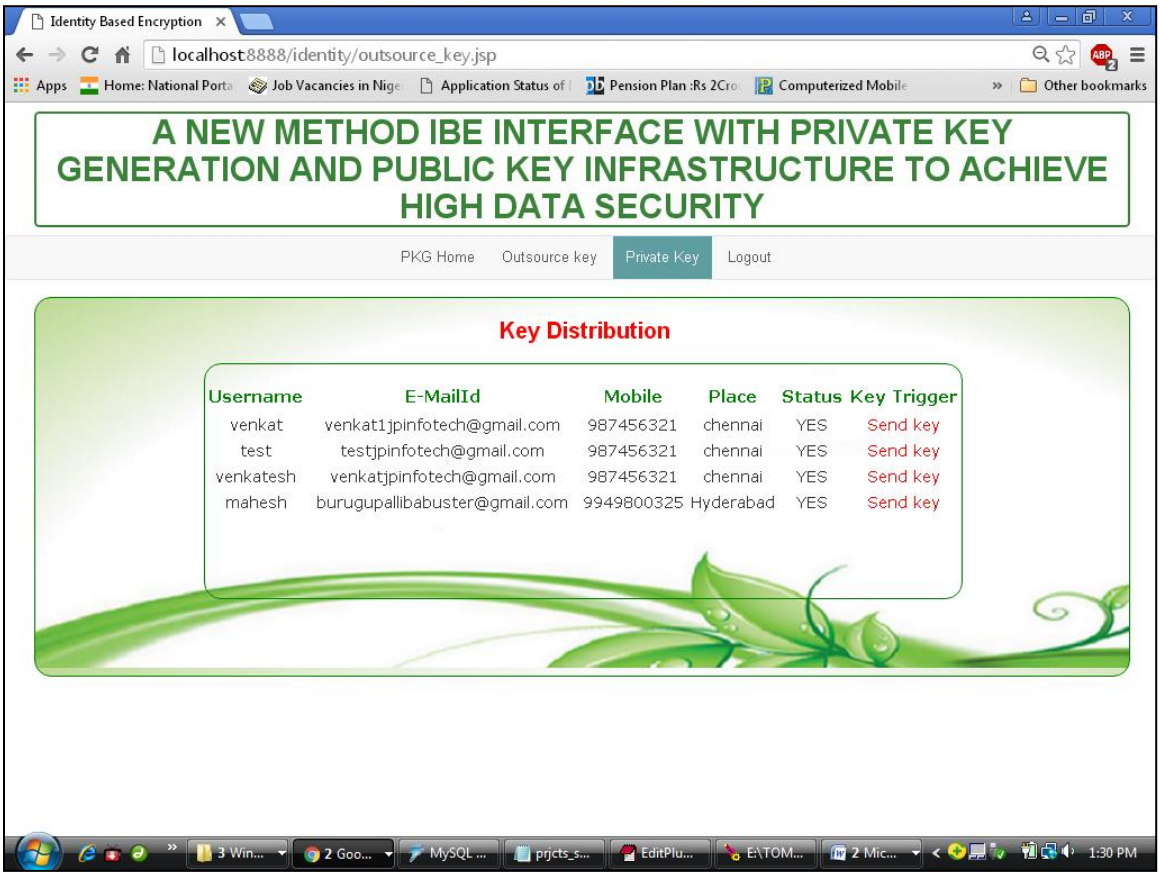


Figure 9: Key Distribution

Here the private key is sent to the user to download the file from the cloud.

CSP home page.

Figure 10: CSP Home Page

User faculty.

Figure 11: User Faculty

Cloud downloads.

Figure 12: Cloud Downloads

Updated key distribution.

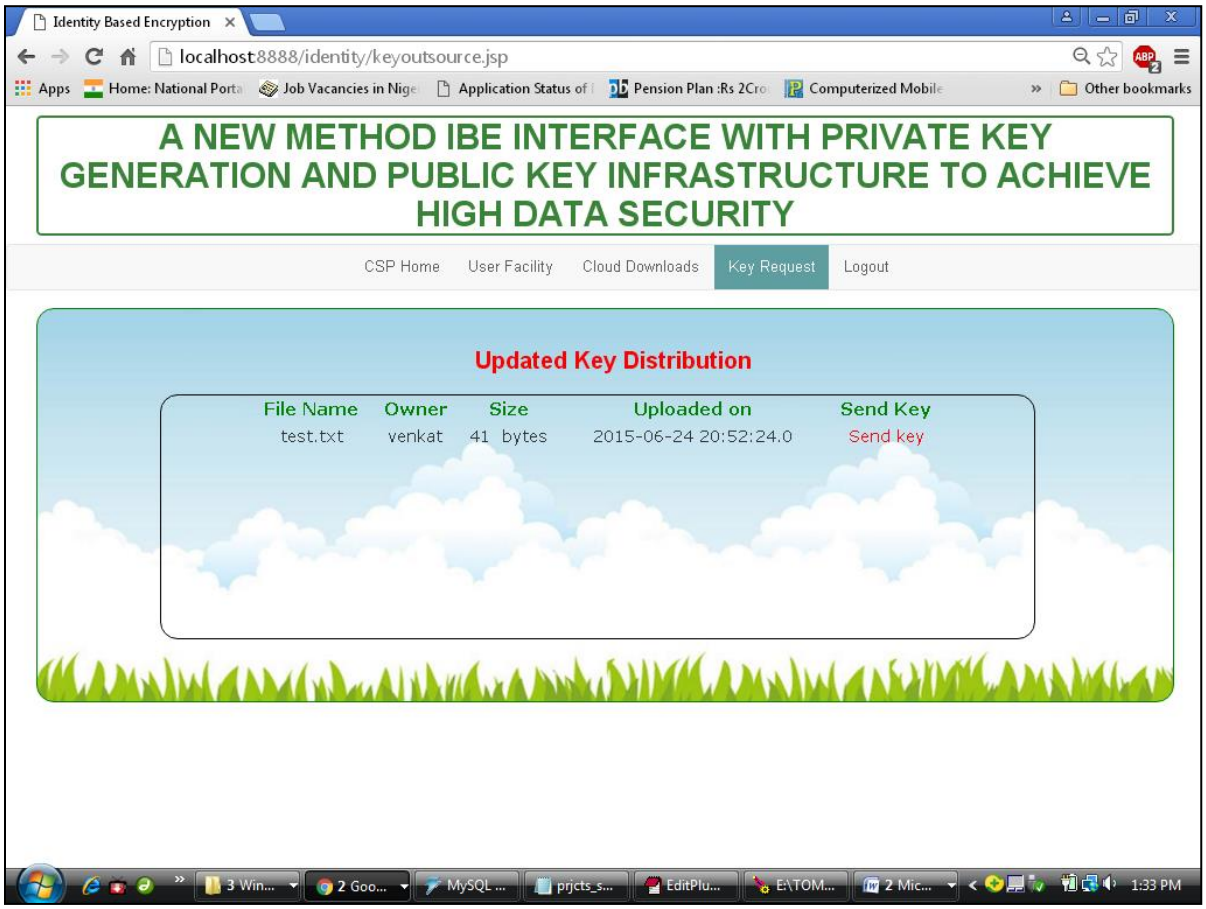


Figure 13: Updated Key Distribution

Upload page.

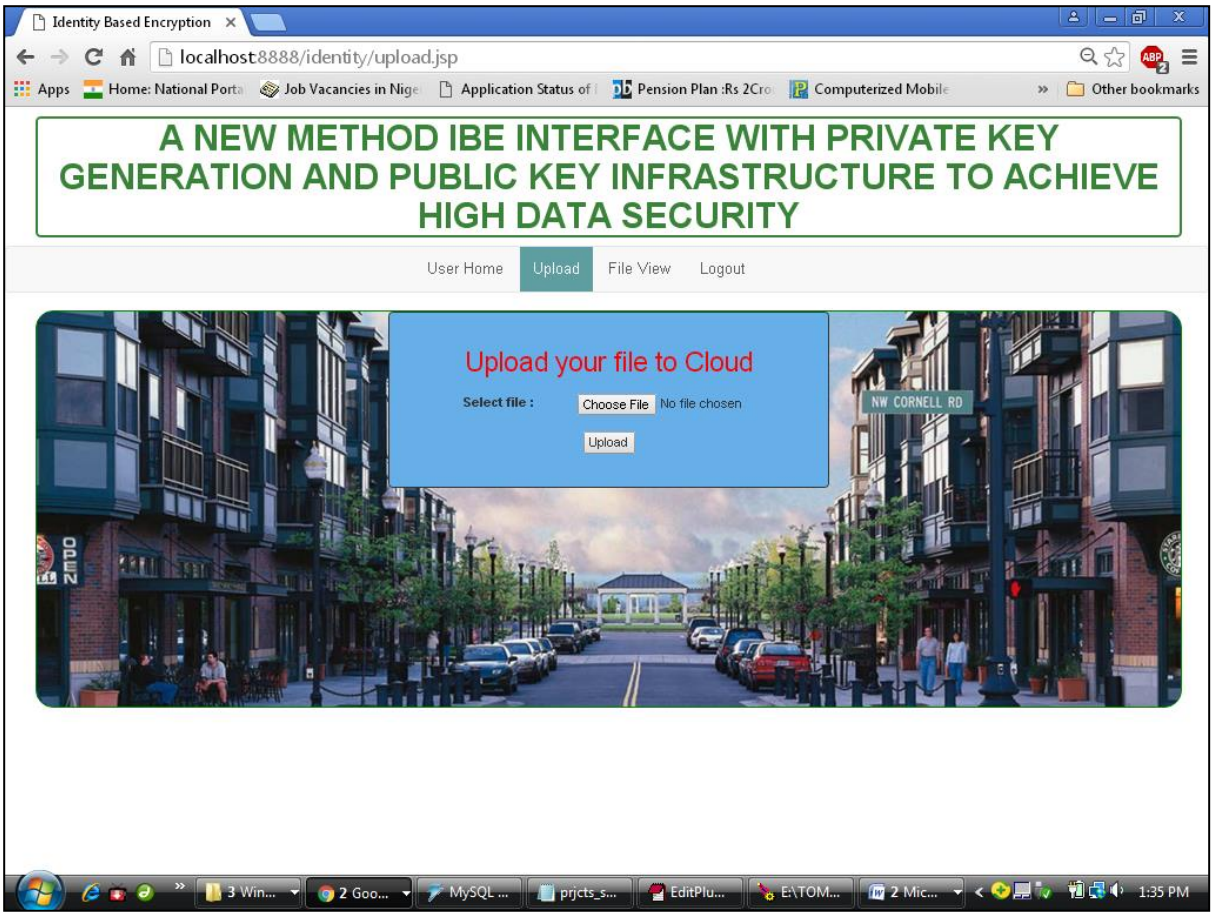


Figure 14: Upload Page

Insert private key.

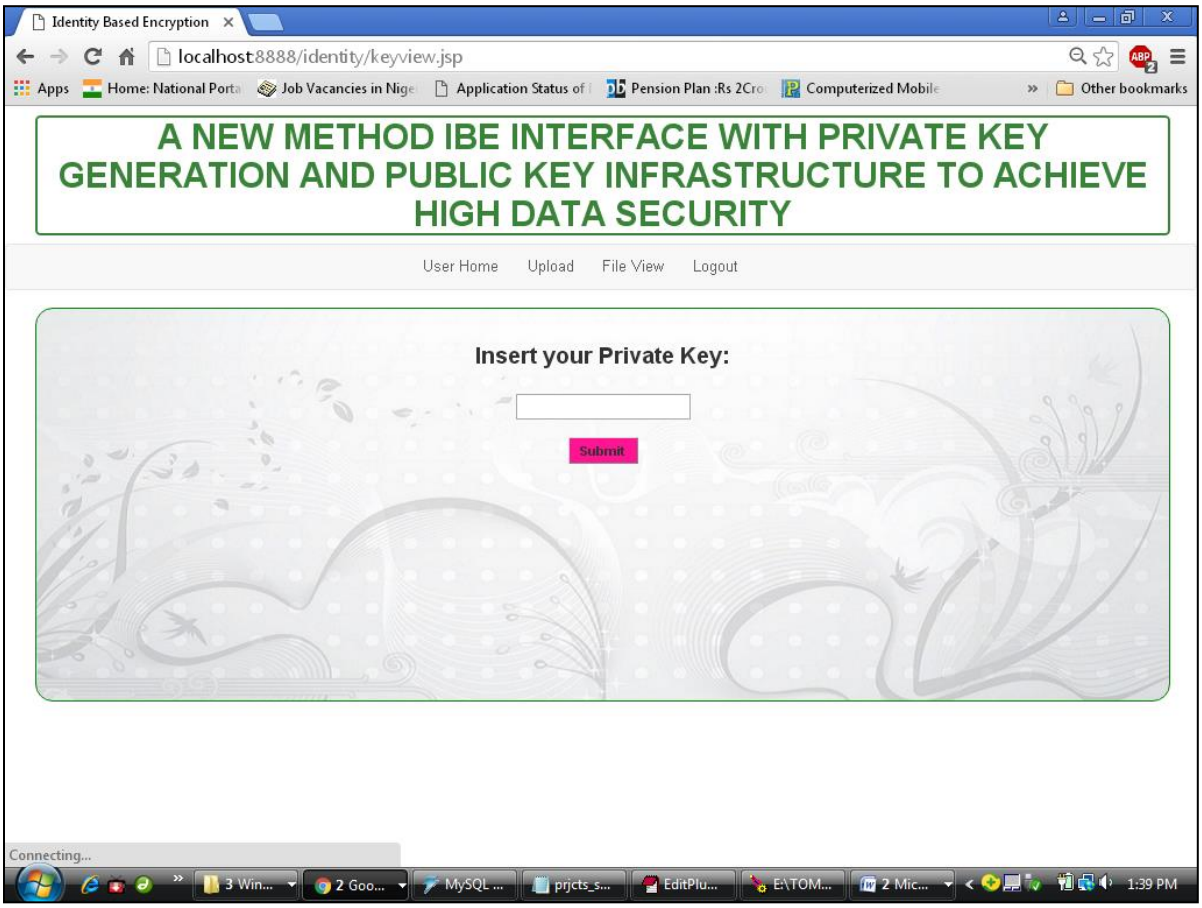


Figure 15: Insert Private Key

Code

Key verifier.jsp.

```
<%@page import="pack.Dbconnection"%>
<%@page import="java.sql.Connection" %>
<%@page import="java.sql.Statement" %>
<%@page import="java.sql.ResultSet"%>
<%
    String userName = request.getParameter("username");
    String userToken = request.getParameter("currentId");
    try{
        Connection conn = new Dbconnection().getConn();
        Statement smt = conn.createStatement();
        ResultSet rs = smt.executeQuery("select * from user where
username ='"+userName+"' and prikey ='"+userToken+"'");
        if(rs.next()){

response.sendRedirect("inbox.jsp?Private_key_verified");
        }else{
          response.sendRedirect("keyview.jsp?error");
        }
    }catch(Exception ex){
        ex.printStackTrace();
    }
%>
```

UploadDoc.jsp.

```
<!--
Author: W3layouts
Author URL: http://w3layouts.com
License: Creative Commons Attribution 3.0 Unported
License URL: http://creativecommons.org/licenses/by/3.0/
-->
<!DOCTYPE html>
<html>
    <head>
        <title>Identity Based Encryption</title>
        <link href="css/bootstrap.css" type="text/css"
rel="stylesheet" media="all">
```

```
        <link href="css/style.css" type="text/css" rel="stylesheet"
media="all">
        <!--web-font-->
        <!--<link
href='http://fonts.googleapis.com/css?family=Playfair+Display:400,70
0,900,400italic,700italic,900italic' rel='stylesheet'
type='text/css'>
        <link
href='http://fonts.googleapis.com/css?family=Roboto+Condensed:300ita
lic,400italic,700italic,400,300,700' rel='stylesheet'
type='text/css'>-->
        <!--//web-font-->
        <!-- Custom Theme files -->
        <meta name="viewport" content="width=device-width, initial-
scale=1">
        <meta http-equiv="Content-Type" content="text/html;
charset=utf-8" />
        <meta name="keywords" content="Excursion Responsive web
template, Bootstrap Web Templates, Flat Web Templates, Andriod
Compatible web template,
              Smartphone Compatible web template, free webdesigns
for Nokia, Samsung, LG, SonyErricsson, Motorola web design" />
        <script type="application/x-javascript">
addEventListener("load", function() { setTimeout(hideURLbar, 0); },
false); function hideURLbar(){ window.scrollTo(0,1); } </script>
        <!-- //Custom Theme files -->
        <!-- js -->
        <script
src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.
js"></script>
        <!-- //js -->
        <!-- start-smoth-scrolling-->
        <script type="text/javascript" src="js/move-
top.js"></script>
        <script type="text/javascript" src="js/easing.js"></script>

        <!--//end-smoth-scrolling-->
        <script>
            function validate() {
                var uname = document.name.userid.value;
                var pass = document.name.password.value;
                if (uname === 0) {
                    alert("Enter your Userid");
```

```
                    document.name.userid.focus();
                    return false;
                }
                if (pass === 0) {
                    alert("Enter your password");
                    document.name.password.focus();
                    return false;
                }
            }
        </script>
        <%
            String userId = (String)session.getAttribute("UserId");
        %>
    </head>
    <body>
         <%
//  if(request.getParameter("msg")!=null){
//      out.println("<script>alert('Invalid Password')</script>");
//  }
//  if(request.getParameter("msg")!=null){
//      out.println("<script>alert('user not exist')</script>");
//  }
//  if(request.getParameter("timeerror")!=null){
//      out.println("<script>alert('Time Scheduler block.you are not
in the respective time slot..Bye')</script>");
//  }
  %>
        <div style="margin: 10px; border: solid 3px; border-color:
#398439; border-radius: 6px; width: 1300px; margin-left: 35px;">
            <center><h3 style="font-size: 40px;color: #398439;font-
weight: bold">A NEW METHOD IBE INTERFACED WITH PRIVATE KEY
GENERATION AND PUBLIC KEY INFRASTRUCTURE TO ACHIEVE HIGH DATA
SECURITY</h3></center>
        </div>
        <!--navigation-->

        <div class="top-nav">
            <nav class="navbar navbar-default">
                <div class="container">
                    <div class="navbar-header">
                        <button type="button" class="navbar-toggle
collapsed" data-toggle="collapse" data-target="#bs-example-navbar-
collapse-1">
```

```
                            <span class="sr-only">Toggle
navigation</span>
                            <span class="icon-bar"></span>
                            <span class="icon-bar"></span>
                            <span class="icon-bar"></span>
                        </button>
                    </div>
                    <!-- Collect the nav links, forms, and other
content for toggling -->

                    <div class="collapse navbar-collapse" id="bs-
example-navbar-collapse-1" style="margin-left: 350px;">
                        <ul class="nav navbar-nav navbar-left">
                            <li><a href="user_home.jsp">User
Home</a></li>
                            <li><a href="upload.jsp"
class="active">Upload</a></li>
                            <li><a href="keyview.jsp">File
View</a></li>
                            <li><a href="userlogin.jsp">Logout
</a></li>
                        </ul>
                        <div class="clearfix"> </div>
                    </div>
                </div>
            </nav>
        </div>

        <!--navigation-->
        <!--header-->
        <div class="header">
        </div>
        <!--//header-->
        <div  style="border-radius: 20px; border: solid 2px; border-
color: green;height: 450px; width: 1300px; margin-left: 35px;margin-
bottom: -10px; background-image: url('images/bannerupl.jpg')">
            <center>
            <div style="background-color: white; background-color:
#66afe9;border: solid 1px; border-radius: 5px; width: 500px; height:
200px;">
                <br /><br /><h2><p style="color: red;font:
bold;">Upload your file to Cloud</p></h2><br />
```

```
                <form action="uploadProcess.jsp" method="post"
enctype="multipart/form-data">
                <label style="margin-left: -250px;">Select file :
</label><input type="file" name="upload" style="margin-left: 200px;
margin-top: -20px"><br />
                <input type="submit" value="Upload">
                </form>
            </div>
                </center>
        </div>
    </body>
</html>
```

Register action.

```
<%@page import="pack.sendmail"%>
<%@page import="java.util.Random"%>
<%@page import="java.sql.DriverManager"%>
<%@page import="java.sql.Statement"%>
<%@page import="java.sql.Connection"%>
<%@page import="java.net.ConnectException"%>
<%
    String uname = request.getParameter("uname");
    String mailid = request.getParameter("mail");
    String pass = request.getParameter("password");
    String ph = request.getParameter("phoneno");
    String lo = request.getParameter("location");
    Random s = new Random();
    int a = s.nextInt(100000 - 5000) +5000 ;
    System.out.print(a);
//    String key =a+"";
    String key = String.valueOf(a);

    try {
        Class.forName("com.mysql.jdbc.Driver");
        Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/ibe",
"root", "root");
        Statement st = con.createStatement();
        int i = st.executeUpdate("INSERT INTO user
(username,password,mailid,phoneno,status,location) VALUES('" + uname
+ "','" + pass + "','" + mailid + "','" + ph + "','no','" + lo +
"')");
```

```
        con.close();
        if (i != 0) {
//            sendmail send = new sendmail();
//            send.mailsend(key,uname,mailid);

response.sendRedirect("userlogin.jsp?msg=registered_sucessfully..!")
;
        }

    } catch (Exception e) {
        out.println(e.getMessage());
    }
%>
```

Email finder.

```
package util;
import pack.Dbconnection;
import java.sql.Connection;
import java.sql.Statement;
import java.sql.ResultSet;
public class EmailFinder{

    public static String findEmail(String uid){
        String emailId = null;
        Connection conn =null;
        Statement smt = null;
        ResultSet rs = null;
        try{
            conn =  new Dbconnection().getConn();
            smt = conn.createStatement();
            rs = smt.executeQuery("select mailid from user where
username ='"+uid+"'");
            while(rs.next()){
              emailId = rs.getString("mailid");
              System.out.println("email id is "+emailId);
            }
        }catch(Exception ex){
            ex.printStackTrace();
        }
        return emailId;
    }
}
```

Email sender.

```
package util;
import java.util.Properties;
import javax.mail.Message;
import javax.mail.PasswordAuthentication;
import javax.mail.Session;
import javax.mail.Transport;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;
import pack.TokenUpdator;

public class EmailSender {

    public static boolean sendKey(String emailId, String uid){

          boolean bool = true;
          final String password = "demomail";
          String from   = "demomail9849@gmail.com";
          String to   = emailId;

         Properties props = new Properties();
         props.put("mail.smtp.host", "smtp.gmail.com");
         props.put("mail.smtp.socketFactory.port", "587");
         props.put("mail.smtp.socketFactory.class",
"javax.net.tsl.TSLSocketFactory");
         props.put("mail.smtp.auth", "true");
         props.put("mail.smtp.port", "587");

         Session session = Session.getDefaultInstance(props,
         new javax.mail.Authenticator() {
         protected PasswordAuthentication getPasswordAuthentication()
{
         return new
PasswordAuthentication("demomail9849@gmail.com",password);
         }
         });

             try{
                        String secreteKey = new
TrippleDes().encrypt(uid);
                        TokenUpdator.updateToken(uid, secreteKey);
                        System.out.println("within try");
```

```
                        Message message = new MimeMessage(session);
                        message.setFrom(new InternetAddress(from));

message.setRecipients(Message.RecipientType.TO,
                        InternetAddress.parse(to));
                    message.setSubject("This is your permission key");
                    message.setText(secreteKey);
                        Transport.send(message);
                }catch(Exception ex){
                    bool = false;
                    ex.printStackTrace();
                }
            return bool;
        }

        public static boolean resend(String emailId){

             boolean bool = true;
             final String password = "demomail";
             String from   = "demomail9849@gmail.com";
             String to   = emailId;

            Properties props = new Properties();
            props.put("mail.smtp.host", "smtp.gmail.com");
            props.put("mail.smtp.socketFactory.port", "587");
            props.put("mail.smtp.socketFactory.class",
"javax.net.tsl.TSLSocketFactory");
            props.put("mail.smtp.auth", "true");
            props.put("mail.smtp.port", "587");

            Session session = Session.getDefaultInstance(props,
            new javax.mail.Authenticator() {
            protected PasswordAuthentication getPasswordAuthentication()
{
            return new
PasswordAuthentication("demomail9849@gmail.com",password);
            }
            });

                try{
                        Message message = new MimeMessage(session);
                        message.setFrom(new InternetAddress(from));
```

```
message.setRecipients(Message.RecipientType.TO,
                        InternetAddress.parse(to));
                message.setSubject("Deactivated");
                message.setText("You are deactivated..");
                        Transport.send(message);
            }catch(Exception ex){
                bool = false;
                ex.printStackTrace();
            }
        return bool;
    }
    public static boolean sendKey(String emailId, String fileName,
String fileKey){

         String str = fileName+" : "+fileKey;
         boolean bool = true;
         final String password = "demomail";
         String from   = "demomail9849@gmail.com";
         String to   = emailId;

        Properties props = new Properties();
        props.put("mail.smtp.host", "smtp.gmail.com");
        props.put("mail.smtp.socketFactory.port", "587");
        props.put("mail.smtp.socketFactory.class",
"javax.net.tsl.TSLSocketFactory");
        props.put("mail.smtp.auth", "true");
        props.put("mail.smtp.port", "587");

        Session session = Session.getDefaultInstance(props,
        new javax.mail.Authenticator() {
        protected PasswordAuthentication getPasswordAuthentication()
{
        return new
PasswordAuthentication("demomail9849@gmail.com",password);
        }
        });
            try{
                        Message message = new MimeMessage(session);
                        message.setFrom(new InternetAddress(from));

message.setRecipients(Message.RecipientType.TO,
                        InternetAddress.parse(to));
```

```
                message.setSubject("File name and access Key:");
                message.setText(str);
                        Transport.send(message);
            }catch(Exception ex){
                bool = false;
                ex.printStackTrace();
            }
        return bool;
    }
}
```

Chapter V: System Design

Data Flow Diagram

1. The DFD is additionally called as Bubble Diagram. It is a straightforward graphical formalism that can be utilized to speak to a system regarding input information to the system, different preparing completed on this information, and the yield information is produced by this system.
2. The data flow diagram (DFD) is a standout amongst the most vital demonstrating apparatuses. It is utilized to show the system parts. These parts are the system procedure, the information utilized by the procedure, an outside substance that cooperates with the system and the data streams in the system.
3. DFD shows how the data travels through the system and how it is altered by a progression of changes. It is a graphical method that delineates data stream and the changes that are connected as information moves from contribution to yield.
4. DFD is otherwise called bubble diagram. A DFD might be utilized to speak to a system at any level of deliberation. DFD might be apportioned into levels that speak to expanding data stream and utilitarian detail.

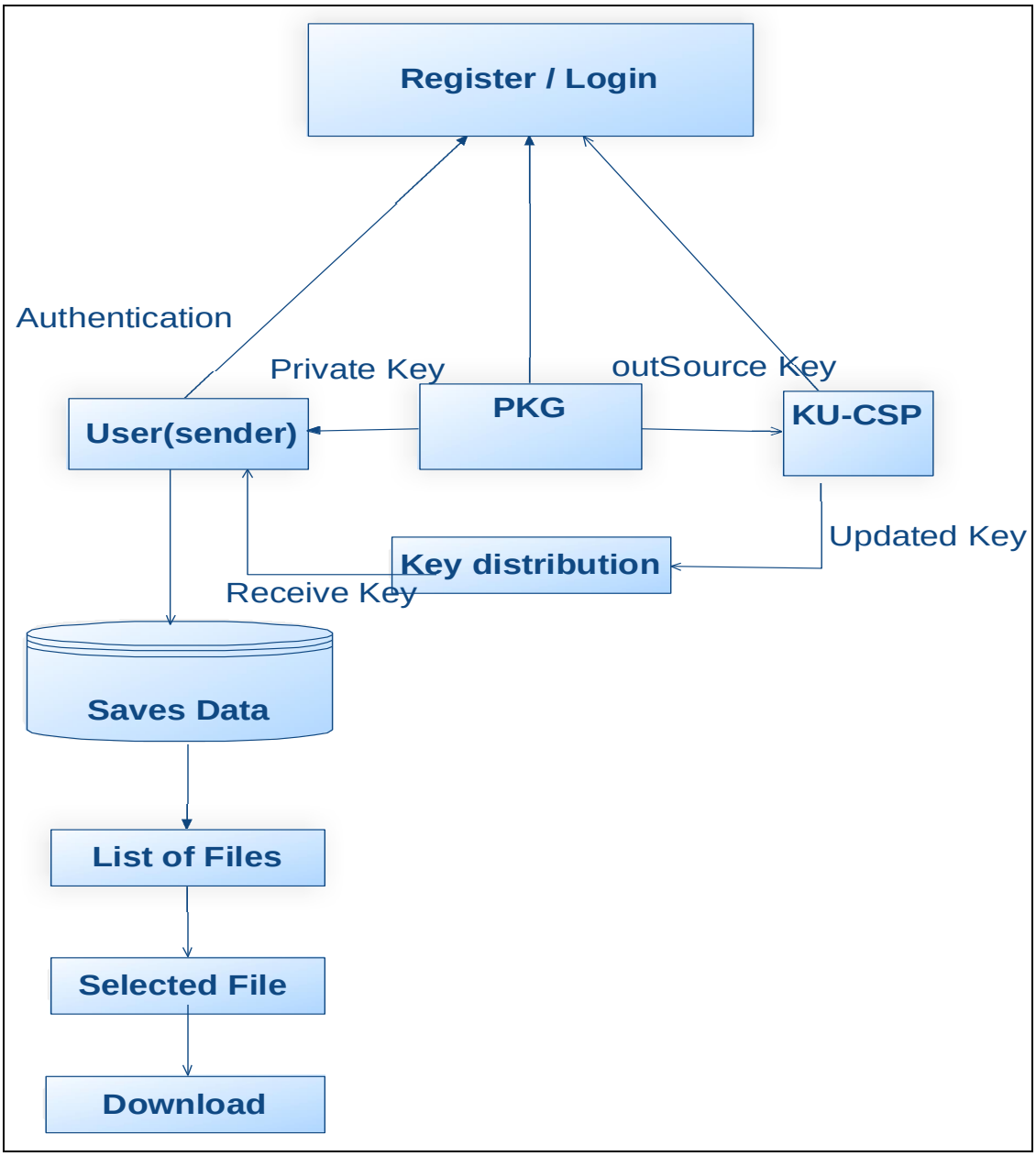


Figure 16: Data Flow Diagrams

UML Diagrams

A use case diagram in the Unified Modeling Language (UML) is a sort of behavioral graph characterized by and made from a Use-case investigation. Its motivation is to display a graphical outline of the usefulness gave by a system as far as on-screen characters, their

objectives (spoke to as utilize cases), and any conditions between those utilization cases. The principle motivation behind utilization case graph is to show what system capacities are performed for which on-screen character. Parts of the performers in the system can be delineated.

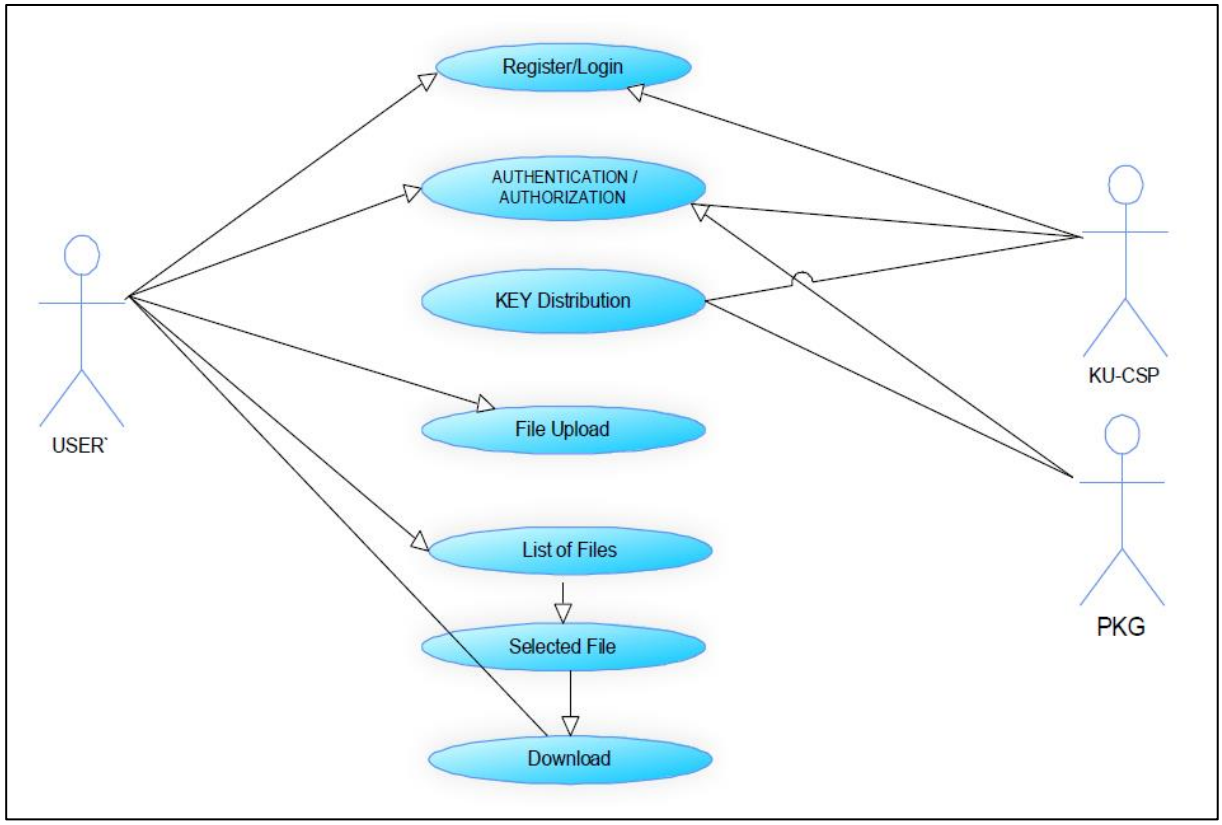


Figure 17: UML Diagrams

Class Diagram

In programming building, a class outline in the Unified Modeling Language (UML) is a kind of static structure graph that depicts the structure of a system by demonstrating the system's classes, their properties, operations (or strategies), and the connections among the classes. It clarifies which class contains data.

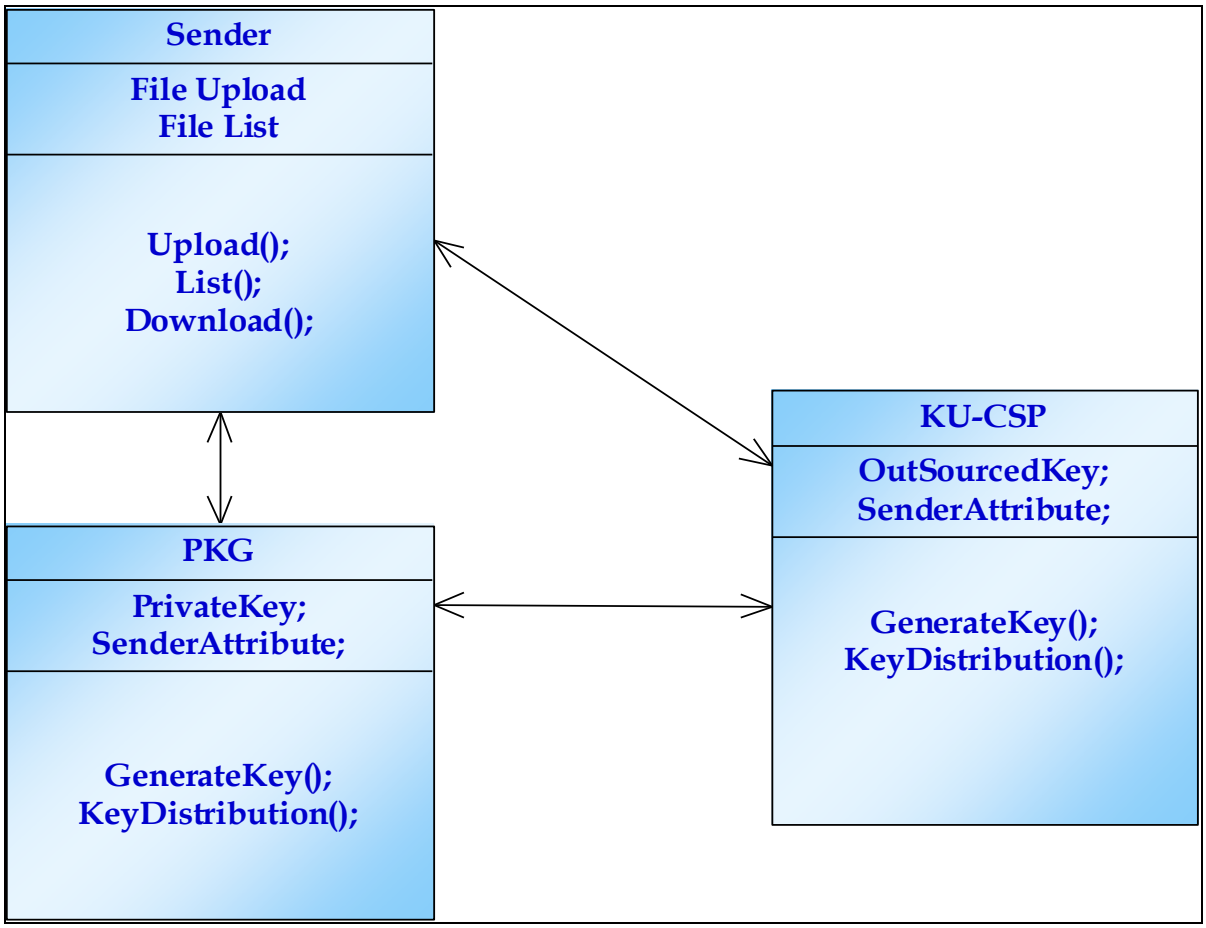


Figure 18: Class Diagram

Sequence Diagram

An arrangement outline in Unified Modeling Language (UML) is a sort of association graph that shows how forms work with each other and in what arrange. It is a development of a Message Sequence Diagram. Succession outlines are here and there called occasion graphs, occasion situations, and timing diagrams.

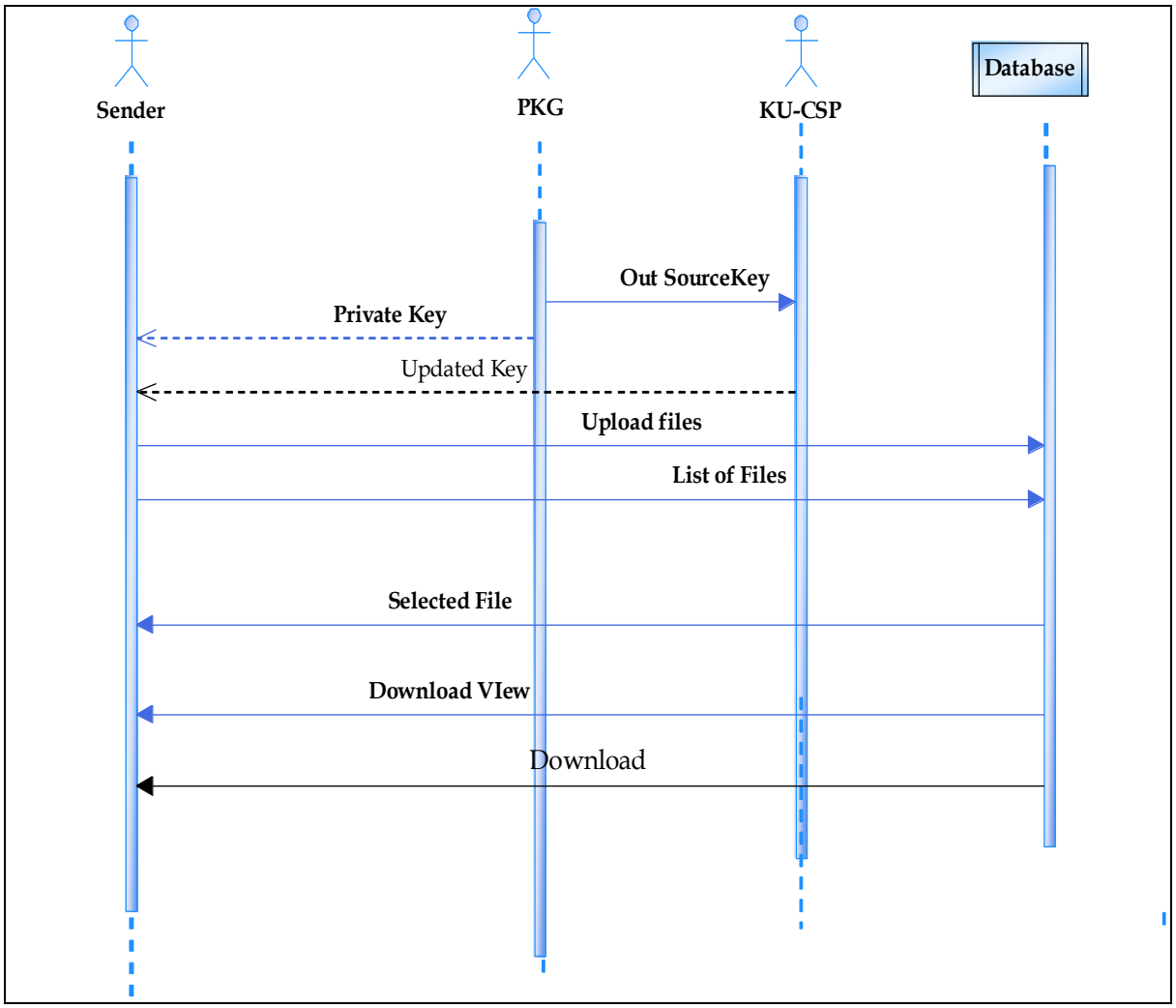


Figure 19: Sequence Diagram

Activity Diagram

Activity diagrams are graphical portrayals of work processes of stepwise exercises and activities with support for decision, emphasis and simultaneousness. In the Unified Modeling Language, action graphs can be utilized to portray the business and operational well-ordered work processes of parts in a system. A movement graph demonstrates the general stream of control.

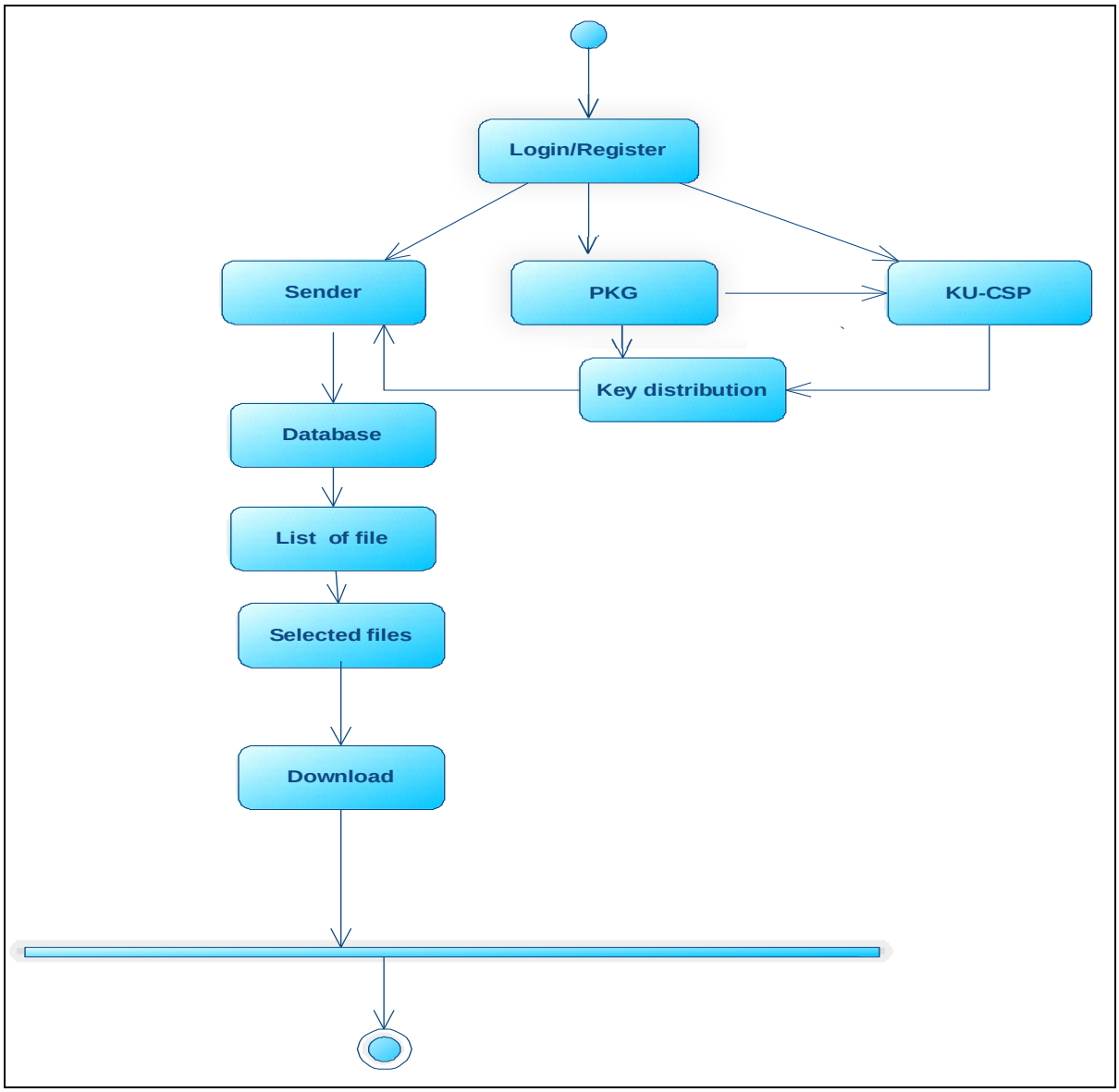


Figure 20: Activity Diagram

Chapter VI: Software Requirement

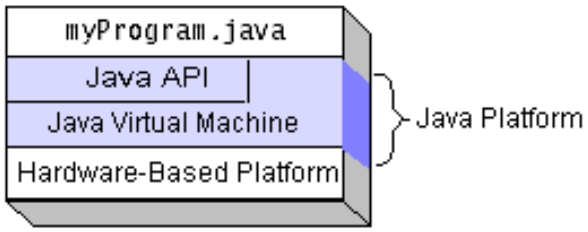
The JAVA Platform

A software is the hardware or programming condition in which a system runs. We have starting at now said probably the most visible stages like Windows 2000, Linux, Solaris, and MacOS. Most stages can be described as a mix of the working framework and hardware. The Java stage contrasts from most unique stages in that, it's just stage that continues running on top of other stages.

The Java platform has two segments:

- The Java Virtual Machine (Java VM)
- The Java Application Programming Interface (Java API)

You have starting at now been familiar with the Java VM. It is the base for the Java software and is ported onto diverse based stages. The Java API is a broad social event of moment programming parts that give various supportive capacities, for instance, graphical UI (GUI) devices. The Java API is gathered into libraries of related classes and interfaces; these libraries are known as bundles. The accompanying section, What Can Java Technology Do? Highlights what handiness a bit of the bundles in the Java API give. The going with figure depicts a venture that is running on the Java stage (Murphy, Kersten, & Findlater, 2006).



A Native code cannot avoid being code that after you gather it, the requested code continues running on a specific hardware stage. As a stage independent condition, the Java stage can be a bit slower than nearby code. In any case, smart compilers, especially tuned go between, and at the last possible second byte code compilers can pass on execution close to that of nearest code without undermining negotiability.

What Can Java Technology Do? The most generally perceived sorts of tasks written in the Java programming language are applets and applications. In case you have surfed the Web, you are likely formally familiar with applets. An applet is a venture that sticks to particular conventions that allow it to continue running inside a Java-enabled program.

In any case, the Java programming language is not exclusively to compose enchanting, drawing in applets for the Web. The all-around helpful, irregular state Java programming language is moreover a proficient programming stage. Using the liberal API, you can make various deals with of activities. An application is an independent program that runs particularly on the Java stage. An excellent kind of utilization known as a server serves and sponsorships clients on a framework. Instance of servers are Web servers, mediator servers, mail servers, and print servers. Another particular venture is a servlet. A servlet can basically be considered as an applet that continues running on the server side. Java Servlets are a well-known choice for building natural web applications, supplanting the usage of CGI scripts. Servlets resemble applets in that they are runtime extensions of uses. As opposed to working in projects, be that as it may, servlets continue running inside Java Web servers, organizing or fitting the server (Lewis & Loftus, 2009).

- How does the API bolster every one of these sorts of projects? It does as such with packages of programming segments that gives an extensive variety of usefulness. Each full usage of the Java platform gives you the accompanying components:
    - **The essentials:** Objects, strings, strings, numbers, info and yield, information structures, system properties, date and time, et cetera.
    - **Applets**: The arrangement of traditions utilized by applets. A Java applet is a little application which is composed in Java and conveyed to clients as bytecode. The client dispatches the Java applet from a page, and the applet is then executed inside a Java Virtual Machine (JVM) in a procedure separate from the web program itself. A Java applet can show up in an edge of the website page, another application window, Sun's AppletViewer, or a stand-alone instrument for testing applets. Java applets were presented in the principal variant of the Java language, which was discharged in 1995.
    - **Networking**: URLs, TCP (Transmission Control Protocol), UDP (User Data gram Protocol) sockets, and IP (Internet Protocol) addresses.
    - **Internationalization**: Help for composing programs that can be restricted for users around the world. Projects can naturally adjust to particular areas and be showed in the fitting language.
    - **Security**: Both low level and abnormal state, including electronic marks, public and private key administration, access control, and endorsements.
    - **Software components**: Known as JavaBeans, can connect to existing component models.

- **Object serialization**: Permits lightweight tirelessness and correspondence via Remote Method Invocation (RMI).
- **Java Database Connectivity (JDBC™)**: Gives uniform access to a wide range of social databases.

In what manner will Java technology change my life? We cannot guarantee you distinction, fortune, or even an occupation in the event that you take in the Java programming language. Still, it is prone to improve your projects and requires less exertion than different languages. We trust that Java innovation will help you do the accompanying*:*

Get started quickly: Although the Java programming language is a powerful object-oriented language, it's easy to learn, especially for programmers already familiar with C or C++.

ODBC and JDBC

ODBC. Microsoft Open Database Connectivity (ODBC) is a standard programming interface for application architects and database frameworks providers. Before ODBC transformed into a genuine standard for Windows ventures to interface with database frameworks, programming engineers expected to use limited languages for each database they expected to connect with. By and by, ODBC has settled on the choice of the database framework skirting on immaterial from a coding perspective, which is as it should be. Application engineers have significantly more basic things to worry over than the sentence structure that is required to port their venture beginning with one database then onto the following when business needs unexpectedly change.

Through the ODBC Administrator in Control Panel, you can show the particular database that is associated with a data source that an ODBC application framework is formed to use. Consider an ODBC data source as a gateway with a name on it. Each door will lead you to a particular database. For example, the data source named Sales Figures might be a SQL Server database, however the Accounts Payable data source could imply an Access database. The physical database implied by a data source can harp wherever on the LAN.

The ODBC framework reports are not presented on your framework by Windows 95. Or, on the other hand possibly, they are presented when you setup an alternate database application, for instance, SQL Server Client or Visual Basic 4.0. Exactly when the ODBC image is presented in Control Panel, it uses a record called ODBCINST.DLL. It is in like manner possible to control your ODBC data sources through a remaining solitary program called ODBCADM.EXE. There is a 16-bit and a 32-bit variation of this framework and each keeps up an alternate once-over of ODBC data sources.

From a programming perspective, the greatness of ODBC is that the application can be created to use a similar game plan of limit calls to interface with any data source, paying little notice to the database dealer. The source code of the application doesn't change whether it speaks with Oracle or SQL Server. We simply indicate these two for example. There are ODBC drivers available for a couple of dozen surely understood database frameworks. To be sure, even Excel spreadsheets and plain substance records can be changed into data sources. The working framework uses the Registry information formed by ODBC Administrator to make sense of which low-level ODBC drivers are required to talk with the data source, (for instance, the interface to Oracle or SQL Server). The stacking of the ODBC drivers is clear to

the ODBC application program. In a client/server condition, the ODBC API even handles a substantial part of the framework issues for the application programming engineer (Guan, Ip, & Zhang, 1998).

JDBC. With an ultimate objective to set a free database standard API for Java; Sun Microsystems made Java Database Connectivity, or JDBC. JDBC offers a nonexclusive SQL database get to framework that gives a relentless interface to a variety of RDBMSs. This relentless interface is expert utilizing "module" database accessibility modules, or drivers. In case a database vender wishes to have JDBC reinforce, he or she ought to give the driver to each stage that the database and Java continue running on.

Few programming bundles are laid out without destinations as an essential concern. JDBC is one that, thus of its various targets, drove the headway of the API. These goals, in conjunction with early pundit input, have settled the JDBC class library into a solid framework for building database applications in Java. The destinations that were set for JDBC are basic. They will give you some learning with reference to why certain classes and functionalities bear in transit they do. The eight layout goals for JDBC are according to the accompanying:

SQL level API. The originators felt that their fundamental objective was to characterize a SQL interface for Java. In spite of the fact that not the most minimal database interface level conceivable, it is at a sufficiently low level for more elevated amount setup and APIs to be made. Alternately, it is at a sufficiently high level for application software engineers to utilize it. Achieving this objective takes into account future sellers to "produce" JDBC code and to conceal a large number of JDBC’s complexities from the end client.

SQL conformance. SQL structure differs as you move from database seller to database merchant. With an end goal to a wide classification of sellers, JDBC will permit any inquiry explanation to be gone through it to the fundamental database driver. This permits the availability module to handle non-standard usefulness in a way that is appropriate for its clients.

JDBC must be implemental on top of common database interfaces. The JDBC SQL API must "sit" on top of other regular SQL level APIs. This objective permits JDBC to utilize existing ODBC level drivers by the utilization of a product interface. This interface would make an interpretation of JDBC calls to ODBC and the other way around.

TOMCAT Web Server

Tomcat is an open source web server created by Apache Group. Apache Tomcat is the servlet compartment that is utilized as a part of the official Reference Implementation for the Java Servlet and JavaServer Pages advancements. The Java Servlet and JavaServer Pages details are created by Sun under the Java Community Process. Web Servers like Apache Tomcat bolster just web segments while an application server underpins web segments and business segments (BEAs WebLogic, is one of the well-known application server). To build up a web application with JSP/servlet introduce any web server like JRun, Tomcat and so on to run your application (Hunter & Crawford, 2001).

Chapter VII: System Testing

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies, and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

Types of Tests

Unit testing. Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing. Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields.

Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct

and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test. Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input: Identified classes of valid input must be accepted.

Invalid Input: Identified classes of invalid input must be rejected.

Functions: Identified functions must be exercised.

Output: Identified classes of application outputs must be exercised.

Systems/Procedures: Interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System test. System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach. Field testing will be performed manually and functional tests will be written in detail.

Test objectives.

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested.

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g., components in a software system or–one step up–software applications at the company level–interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements (Pacheco & Ernst, 2007).

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Chapter VIII: Conclusion

Despite the fact that exploration enthusiasm for ID-PKC is exceptionally solid right now, it is a moderately new innovation in contrast with PKI. In our article, we have tried to investigate what isolates ID-PKC from PKI. Our underlying judgment, in fact made with regards to almost no business organization of ID-PKC frameworks, is that there is next to no to isolate the two. Maybe the imperative info when choosing whether to embrace PKI or ID-PKC is the diverse path in which the two advances normally produce and confirm rights and keys. Similarly, as with symmetric and hilter kilter cryptography, the central elements when picking amongst PKI and ID-PKC are prone to be ecological. This impact of the limitations encompassing the usage are prone to be more noteworthy given that there doesn't appear to be such a solid isolating element as the capacity to give non-revocation is amongst symmetric and topsy-turvy cryptography.

The paper portrays the IB-mRSA, a commonsense and secure character based encryption plan. It is perfect with standard RSA encryption and offers fine-grained control (renouncement) of client’s security benefits. A few issues stay for future work. It is indistinct whether IB-mRSA can be indicated secure under the standard model (our contention uses the arbitrary prophet setting). Also, we require a more formal investigation of semantic security. Another issue identifies with IB-mRSA execution. Utilizing a hash capacity for open key mapping makes encryption more costly than RSA since people in general type is arbitrary (and on the normal portion of the bits are set). We have to examine elective mapping capacities that can create more "effective" RSA types.

References

Baudron, O., Pointcheval, D., & Stern, J. (2000, July). Extended notions of security for multicast public key cryptosystems. In *27th International Colloquium on Automata, Languages and Programming (ICALP '2000), no. 1853 in lecture notes in computer science*. Berlin, Germany: Springer-Verlag.

Bellare, M., Boldyreva, A., & Micali, S. (2000). Public-key encryption in a multi-user setting: Security proofs and improvements. In B. Preneel (Ed.), *ERUOCRYHPT 2000. LNCS, 1807*, 259–274. Springer, Heidelberg.

Bellare, M., Desai, A., Pointcheval, D., & Rogaway, P. (1998). Relations among notions of security for public-key encryption schemes. In H. Krawczyk (Ed.), *Advances in cryptology–CRYPTO '98, no. 1462 in lecture notes in computer science* (pp. 26-45). Berlin, Germany: International Association for Cryptologic Research, Springer-Verlag.

Bellare, M., & P. Rogaway, P. (1995). Optimal asymmetric encryption—how to encrypt with RSA. In A. D. Santis (Ed.), *Advances in cryptology–EUROCRYPT '94, no. 950 in lecture notes in computer science* (pp. 92-111). Berlin, Germany: International Association for Cryptologic Research, Springer-Verlag, 1995.

Bitar, N., Gringeri, S., & Xia, T. J. (2013). Technologies and protocols for data center and cloud networking. *Communications Magazine, IEEE, 51*(9), 24-31.

Boneh, D., Ding, X., & Tsudik, G. (2002, August). Identity based encryption using mediated RSA. In *3rd Workshop on Information Security Application.* Jeju Island, Korea, KIISC.

Boneh, D., Ding, S., Tsudik, G., & Wong, C. M. (2001, August). A method for fast revocation of public key certificates and security capabilities. In *10th USENIX Security Symposium.* Washington, DC: USENIX.

Boneh, D., & Franklin, M. (2001). Identity-based encryption from the Weil Pairing. In J. Kilian (Ed.), *CRYPTO: Lecture notes in computer science* (vol. 2139, pp 213-229). New York: Springer.

Boneh, D., & Hamburg, M. (2008). Generalized identity based and broadcast encryption schemes. In *Advances in cryptology-ASIACRYPT 2008* (pp. 455-470). Springer Berlin Heidelberg.

Camenisch, J., Neven, G., & Rückert, M. (2012). Fully anonymous attribute tokens from lattices. In *Security and cryptography for networks* (pp. 57-75). Springer Berlin Heidelberg.

Coron, J.-S., & Naccache, D. (2000). Security analysis of the Gennaro-Halevi-Rabin signature scheme. In B. Preneel (Ed.), *Advances in cryptology* (pp. 91-101). New York: Springer.

Fujisaki, E., Okamoto, T., Pointcheval, D., & Stern, J. (2001). RSA-OAEP is secure under the RSA assumption. In J. Kilian (Ed.), *Advances in cryptology* (vol. 21, pp. 260-274). Princeton, NY: Springer.

Ganesan, R. (1995, February). Augmenting kerberos with public-key cryptography. In T. Mayfield (Ed.), *Symposium on network and distributed systems security.* San Diego, CA: Internet Society.

Guan, H., Ip, H. H., & Zhang, Y. (1998). Java-based approaches for accessing databases on the Internet and a JDBC-ODBC implementation. *Computing & Control Engineering Journal*, *9*(2), 71-78.

Hajny, J., & Malina, L. (2012). Unlinkable attribute-based credentials with practical revocation on smart-cards (pp. 62-76). In S. Mangard (Ed.), *CARDIS 2012. LNCS, 7771*. Berlin: Springer, Heidelberg.

Houidi, I., Mechtri, M., Louati, W., & Zeghlache, D. (2011, July). Cloud service delivery across multiple cloud platforms. In *Services Computing (SCC), 2011: 8th International Conference on Services Computing* (pp. 741-742). IEEE Xplore.

Hunter, J., & Crawford, W. (2001). *Java servlet programming: Help for server side Java developers*. Sebastopol, CA: O'Reilly Media, Inc.

Lewis, J., & Loftus, W. (2009). Java software solutions: Foundations of program design. Boston: Pearson/Addison-Wesley.

Lewko, A., & Waters, B. (2011). Decentralizing attribute-based encryption. In *Advances in cryptology–EUROCRYPT 2011* (pp. 568-588). Springer Berlin Heidelberg.

Li, J., Huang, Q., Chen, X., Chow, S. S., Wong, D. S., & Xie, D. (2011, March). Multi-authority ciphertext-policy attribute-based encryption with accountability. In *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security* (pp. 386-390). ACM.

Li, J., Ren, K., Zhu, B., & Wan, Z. (2009). Privacy-aware attribute-based encryption with user accountability. In *Information security* (pp. 347-362). Springer Berlin Heidelberg.

Murphy, G. C., Kersten, M., & Findlater, L. (2006). How are Java software developers using the Elipse IDE? *IEEE Software, 23*(4), 76-83.

Oberheide, J., Veeraraghavan, K., Cooke, E., Flinn, J., & Jahanian, F. (2008, June). Virtualized in-Cloud security services for mobile devices. In *Proceedings of the First Workshop on Virtualization in Mobile Computing* (pp. 31-35). New York, NY: ACM.

Pacheco, C., & Ernst, M. D. (2007, October). Randoop: Feedback-directed random testing for Java. In *Companion to the 22nd ACM SIGPLAN Conference on Object-oriented Programming Systems and Applications Companion* (pp. 815-816). Montreal, Canada, October 21-25, 2007.

Potlapally, N. R., Ravi, S., Raghunathan, A., & Jha, N. K. (2006). A study of the energy consumption characteristics of cryptographic algorithms and security protocols. *IEEE Transactions on Mobile Computing, 5*(2), 128-143.

Schoo, P., Fusenig, V., Souza, V., Melo, M., Murray, P., Debar, H., ... & Zeghlache, D. (2011). Challenges for Cloud networking security. In *International Conference on Mobile Networks and Management* (pp. 298-313). Springer Berlin Heidelberg.

Shahandashti, S. F., & Safavi-Naini, R. (2009). Threshold attribute-based signatures and their application to anonymous credential systems. In *Progress in cryptology–AFRICACRYPT 2009* (pp. 198-216). Springer Berlin Heidelberg.

Subashini, S., & Kavitha, V. (2011). A survey on security issues in service delivery models of Cloud computing. *Journal of Network and Computer Applications, 34*(1), 1-11.

Varghese, S., & Vigila, S. M. C. (2015). A comparative analysis on cloud data security. In *2015 Global Conference on Communication Technoloies (GCCT)*, April 23-24, 2015, Thuckalay, India.

Waters, B. (2011). Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In *Public key cryptography–PKC 2011* (pp. 53-70). Springer Berlin Heidelberg.

Zissis, D., & Lekkas, D. (2012). Addressing Cloud computing security issues. *Future Generation Computer Systems*, *28*(3), 583-592.