CHAPTER 1: INTRODUCTION

The world of Computer and Internet has changed from its emergence. There have been Substantial advancements in computing power, storage, and networking technology. The computing resources are advancing towards web for better utilization. Now, the traditional IT resources setups are being replaced with CLOUD COMPUTING technology. Cloud computing consists of large-scale distributed and virtual machine computing infrastructure. It offers dynamically scalable resources such as computational power, storage, hardware platforms and applications as services over the Internet.

Some of the key attributes of cloud computing are ease of provisioning, on-demand availability, dynamic and virtual inline scalability. Many organizations can use such cloud systems and services while migrating all or some information services to the cloud computing environment. Various benefits include increases in flexibility and cost savings by minimizing investments in hardware and software. As cloud computing has high economical benefits, various organizations has showed interest in it. Many organizations are adopting this paradigm.

While there are a huge number of benefits associated with cloud computing, many issues are also present such as policy, standards, security, reliability etc. Along with the technology, Security issues in cloud computing evolves. More and more personal information and potentially secure data is now stored on cloud. This increasing usage of cloud necessitates the adoption of powerful security measures. Security is considered as the most critical issue because the security measures being followed are not sufficient. It restricts many leading and large scale organizations to completely adopt this technology. So, it’s highly needed to address security challenges in cloud computing to make it more secure as well as reliable.

1.1 MOTIVATION

From last few decades, there have been major changes and advancements in technologies. Cloud computing, being an emerging technology, is now very popular because of its obvious benefits. Many organizations are moving to cloud. Major Tech. companies such as Google, Microsoft, Amazon are actively engaged in developing and providing cloud services e.g. Amazon EC2, Google Compute Engine, and Microsoft Windows Azure. These companies are investing a significant part of their budget in cloud R&D. While cloud computing is reaching to new levels day by day, there has been a lot of issues that cannot be ignored. Because of these issues, many organizations are hesitant to adopt this paradigm. Security is one of the most critical issues among other issues such as reliability, policy, standards and so on. These issues must be given proper attention in order to make cloud computing an acceptable and adoptable technology.

1.2 PROBLEM STATEMENT

Security is one of the major challenge in cloud computing. The measures followed are not sufficient enough to provide high degree of security. While the technology is becoming popular day by day, there have been many instances of security breaches. Security issues are major factors in slowing down the acceptance of cloud computing. Among many security issues, **Authentication** is considered one of the major key issues.

Cloud computing demands efficient and strong authentication techniques. While it's important to authenticate user to a server before allowing him/her to access the services, server authenticating to user is equally important for achieving high degree of security. There is need of a strong and efficient yet user friendly Mutual Authentication technique because the existing techniques are not well enough. The technique should be simple, cost effective, easily implementable, user friendly and strongly efficient of course.

1.3 AIM OF THESIS

To achieve success, cloud computing paradigm needs efficient solutions to the major challenges. The aim of thesis is to provide such a solution to one of the most critical security issue i.e. **Mutual Authentication.** Many schemes have been proposed till now, but we need an approach which is strong and efficient but simple, easy to implement, user friendly as well.

This thesis proposes a strong mutual authentication scheme using factors-based approach for cloud computing. The proposed scheme makes use of factors from two categories i.e. Knowledge Factor and Possession Factor. The proposed scheme uses factors i.e. Alphanumeric Password, Stego-Image, One Time Password (OTP) and Digital Signature from two categories which make the scheme stronger. The proposed protocol provides identity management, mutual authentication and session key agreement between user and service provider. The scheme provides facilities for changing credentials such as password, Stego-image file etc. The scheme also provides facility to get new credentials if the user looses/forgets any credential. Security analysis shows that our scheme can withstand common attacks. The proposed protocol is strong, efficient and well suited for cloud computing.

1.4 ORGANIZATION OF THESIS

This thesis is organized in the following manner:

Chapter 1 is the introduction part of thesis. It includes motivation, problem statement, aim of thesis, and organization of thesis.

Chapter 2 presents the literature survey. It summarizes the existing literature, all the work done in this field.

Chapter 3 provides the research background. It describes cloud computing, its service model, deployment model and characteristics in detail. It also explains cloud security and two factor mutual authentication.

Chapter 4 describes the factors used in the proposed scheme in detail.

Chapter 5 describes the proposed scheme. Every phase is described in detail.

Chapter 6 contains the implementation and results of the proposed algorithm.

Chapter 7 presents the in-depth security analysis of our proposed scheme.

Chapter 8 contains conclusions and future work possible in this area.

CHAPTER 2: LITERATURE SURVEY

Cloud computing suffers from many issues. Security is one of the major challenges in success of this emerging technology. Strong authentication scheme is a vital requirement for cloud computing. Many solutions has been given by researchers for providing authentication in cloud computing.

Plain Password based authentication is one of the simplest techniques. Although being most commonly used and simple, this technique suffers from many attacks such as dictionary attack. This is a weak technique. Passwords can be only a combination of symbols that are on keyboard and that lead to another kind of attack, known as brute force attack. Also, users tend to select passwords that are easy to remember such as nicknames, phone numbers etc. The attacker can easily make a table of significant words and this is known as dictionary attack. There are other kind of attacks as well such as replay attack, Man-In-Middle attack etc [17].

Alphanumeric password is one of the most used components in authentication. But it is not secure enough. An authentication scheme using **graphical password** was proposed by Guo, Liaw, hasio and yen [9]. The notion of Graphical password is first proposed by Bolnder [20]. User have to select some square blocks sequentially as password and then have to reproduce the same sequence at the time of login phase. The proposed scheme claims to achieve resistance against some common attacks such as replay attack, sniffer attack, impersonation Attack.

It is very difficult to memorize different passwords for different web services while keeping the same password for all web services makes the password vulnerable. A password manager is needed that can store and retrieve multiple existing passwords using only one master password. Yang and chu [2] proposed such a **cloud password manager** scheme using privacy enhanced biometrics. A cloud service is used to synchronize all local password managers in encrypted form. The proposed scheme claims to be efficient and secure.

Yang, chang and Huang [5] proposed an **ID-based** user authentication scheme on multi-server environments for the cloud computing. Use of ID based authentication in proposed scheme makes it suitable for multi server environment. The proposed scheme uses **XOR and one-way hash function**. This reduces the computation cost significantly. The scheme claims to prevent some of the common attacks such as insider attack, impersonation attack and replay attack.

Elliptic Curve Cryptosystem (**ECC)** is a well known cryptosystem because of its low computing cost as compared to others. Chang and yang [16] proposed ID –based mutual authentication scheme for mobile devices using ECC but the scheme suffered from insider attack and impersonating attack. To overcome these issues, chen, yeh and shih [12] proposed a scheme which claims to resist these attacks and provides mutual authentication between server and user based on **dynamic ID** and **ECC**.

Shoup and Rubin [19] proposed an extension of Bellare-Rogaway model [21]. The extended model is based upon **three party key distribution protocol.** The proposed model uses **smartcard** to store the secret keys and to prevent adversaries. The author makes the assumption that the smartcard will never be compromised.

Shailaja, Kumar and saxena [18] proposed a mutual authentication scheme which is based on **bilinear pairings** using **smart card**. The scheme proposes use of two servers, a registration server which registers new users and authentication server which comes into play at the login phase to authenticate the registered users. To prevent insider attack, two secrets are used for mutual authentication between user and authentication server. Further, it is claimed that the proposed scheme prevents many common attacks such as replay attack, Man-In-Middle attack, forgery, insider attack. Although it resists some attacks, many attacks such as smart card anonymity, parallel session attack, smart card reader authentication are still possible.

Tsague and Nelwamondo [10] proposed a mutual authentication scheme which uses **3DES**. The proposed scheme doesn't only enhance the security levels but it also lessens the load of computation on **smart cards**. The proposed scheme provides authentication of user to server and server to user along with generation of secure session key. It also authenticates smart card reader.

It also allows user to change their password. The proposed scheme claims to resist against forgery attack, replay attack, parallel session attack, password guessing attack, platform impersonation attack.

Lee, Ong, Lim and Lee [15] have proposed a authentication scheme for cloud computing which uses **public key** and **mobile out of band**. But the proposed scheme has many flaws such as the data (ID, PWD etc.) are transmitted in plaintext form which is easily attackable. The scheme also has some major flaws such as user was not allowed to change their password. This scheme was not suitable for real time cloud computing.

Choudhury, Kumar, Sain, Lim, and H. Jae-Lee [13] proposed a authentication framework for cloud computing. User has to prove his identity to enter into cloud. The proposed scheme is based on **password, smartcard and out of band.** The verification is done in two-steps. The proposed scheme generates session key as well. In addition, the scheme is claimed to be secure against many popular attacks.

Hao, Zhong and Yu [11] proposed a time bound **ticket** based mutual authentication scheme. The scheme provides mutual authentication and it also generates Session key. The scheme claimed to resist several attacks such as lost smart card attack, replay attack, lost ticket attack but it was found that the scheme is vulnerable to Denial of Service attack.

To overcome limitations of [11], Jaidhar [6] proposed an improved mutual authentication scheme. The scheme first of all verifies the authenticity of user as well as server then only it generates session key. The scheme inherits all the merits of [11].The scheme claims to resist several attacks such as insider attack, offline password attack, replay attack, parallel session attack. During password change phase, user can change password at smart card without need of any involvement of cloud server.

Yeh [14] proposed a scheme to protect data by authentication and secret sharing (**PASS**). The scheme does not store the secret key anywhere on the cloud. The proposed scheme uses public key cryptosystem to encrypt secrets which increases the cost. The client has to store the secret key and the secret key is vulnerable because if the client device is compromised, then secret key can be easily leaked out.

Yang and Lai [3] proposed a authentication and key agreement scheme to strengthen PASS [14] scheme. For secure cloud computing (SCC), the scheme uses **Elliptic Curve Diffie-Hellman (ECDH)** and **symmetric bivariate polynomial based secret sharing**. The proposed scheme provides two types of SCC. The first type of SCC requires trusted third party (TTP) and the other does not. Also, SCC provides mutual authentication to avoid connecting the fake server. The proposed scheme provides mutual authentication and data privacy. The proposed secure cloud computing can be made fit to an environment where multiple servers exists to provide service by extending it to multi server SCC. The scheme claims to resists against insider attack, outsider attack, client side attack, server side attack.

Yassin, Jim, Ibrahim and Zou [8] proposed a mutual authentication scheme which uses **zero knowledge proof.** Zero knowledge based authentications provides better password based authentication because it offers to preserve the privacy. The scheme also uses **homomorphic** encryption and decryption. The secret credentials are saved and stored in USB stick or iPhone at the registration phase and the credentials are retrieved at the time of login. The proposed scheme claims to resist against off-line guessing attack, Man-In-Middle attack, insider assisted attack.

Jivanadham, Islam, Katayama and Baharun [4] proposed a authentication model, **Cloud Cognitive Authenticator (CCA)**, for cloud computing. **CCA** make use of Advance Encryption Standard (**AES**) and one round zero knowledge proof (**ZKP**). The scheme claims to enhance the security in public, private or hybrid clouds by two levels of authentication and encryption algorithms. The novel idea in CCA was to use Electro Dermal Responses (**EDR**) for the first level authentication. The scheme claims to resists wrapper attack, SQL injection attack, Man-In-Middle attack, Square attack.

Nimmy K. and Sethumadhavan [1] proposed a mutual authentication scheme which uses **steganography** as additional encryption method. The proposed technique used the concept of secret sharing in which there are two parts of secret. One is kept by user and other by cloud provider. Both parts are needed to make the complete secret. This leads to mutual authentication and prevents Man-In-Middle attack. **Out of band** communication is also used which makes the technique better. Flexibility to change the password is also offered. The proposed technique claims prevention of some attacks such as MITM, Replay attack, Masquerade attack.

Yassin, Jim, Ibrahim and Zou [7] proposed a mutual authentication scheme based on **schnorr digital signature** and **biometrics**. The proposed scheme uses fingerprint as biometric. The scheme extracts level-3 features from fingerprints. The scheme doesn’t need additional device or software. Session key is also generated during authentication phase. It is claimed that scheme resists different malicious attacks such as parallel-session attack, off-line attack, MITM attack, dictionary attack, insider attack, and replay attack.

CHAPTER 3: RESEARCH BACKGROUND

This chapters contains a brief overview of cloud computing and security. It includes definition, deployment models, characteristics of cloud computing. Two factor authentication is discussed along with explaining categories of factors.

3.1 CLOUD COMPUTING

A pool of virtual resources is called cloud. The term "cloud" is adopted from the fact that it

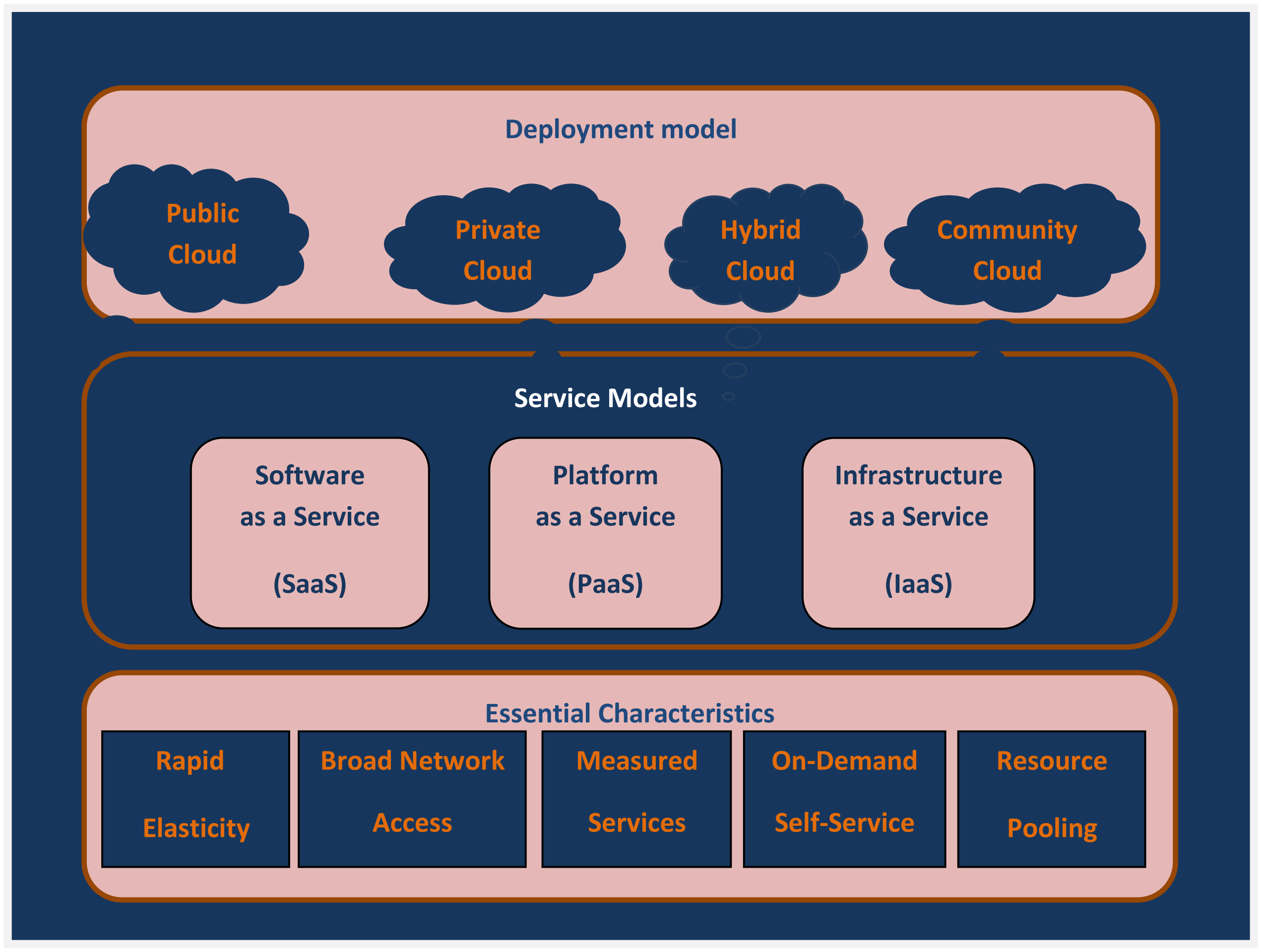


Figure 1: Overview of Cloud Computing

hides the complex details, infrastructure of system. While in traditional IT setup, user needs to be present at the location of service requested, cloud takes away this constraint. The delivery of cloud services over internet is called cloud computing.

Cloud computing allows a user to access services, information and resources from anywhere provided network connection is available. Various services such as space for data storage, network, computing processing power, customized user applications etc. are provided by cloud computing. Users can use software and hardware that is managed by third party providers from remote location. Online file storage, web mail, online business applications etc. are some of the examples of cloud services. Some of the key features of cloud computing are agility, cost reduction, multitenancy etc. Cloud computing consists of four deployment models i.e. public cloud, private cloud, hybrid cloud and community clouds. There are three types of service models in cloud computing i.e. software as a service, platform as a service and infrastructure as a service. Complete overview of cloud computing is depicted in Fig. [1].

3.2 CLOUD DEPLOYMENT MODELS

The four deployment models of cloud computing are described below. They are used to make cloud services available. Organizations and users can select any according to their needs. The deployment models of cloud are depicted in Fig. [2].

3.2.1 PUBLIC CLOUD

The service providers provide resources, applications and services to general public using this model. Examples of such services that are aimed at general public are social networking sites, online photo storage, web mail etc. In fact, public clouds can also offer services to enterprises. Various companies like Google, Microsoft, and Amazon owns infrastructure set up of public clouds. They operate and provide service over the internet.

3.2.2 PRIVATE CLOUD

The cloud infrastructure that is established specially for a specific group is called private cloud. This model is operated exclusively for a group of users or organization. Private clouds are managed by third parties or internally by same organization. It requires high level of skills to establish and manage such set up because there are many security issues that should be addressed.

3.2.3 HYBRID CLOUD

The cloud infrastructure combining the different deployment models in order to achieve advantages of those models is called hybrid model. This model combines benefits of other models and provides high degree of fault tolerance. Both, offsite cloud infrastructure and online resources are required by hybrid clouds architecture. Combining public and community cloud is an example of hybrid cloud.

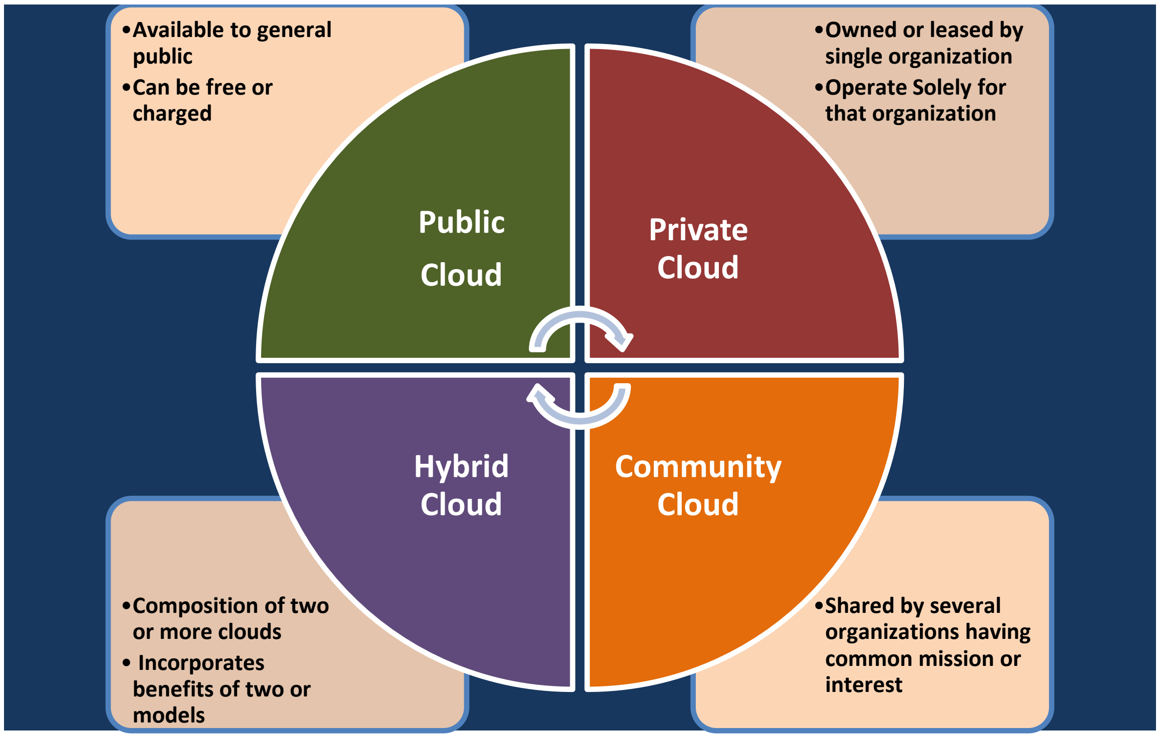

•Available to general public
•Can be free or charged
•Owned or leased by single organization
•Operate Solely for that organization
Public Cloud
Private Cloud
Hybrid Cloud
Community Cloud
•Composition of two or more clouds
• Incorporates benefits of two or models
•Shared by several organizations having common mission or interest

Figure 2: Cloud Deployment Models

3.2.4 COMMUNITY CLOUD

The cloud infrastructure which is shared by a number of organizations is called community cloud. Generally, organizations having common interest such as security use this model.

Community clouds are managed by service providers, third party or organizations itself. It includes more number of users than private clouds but fewer as compared to public cloud.

3.3 CLOUD SERVICE MODEL

There are basically three service models used by cloud service providers as discussed below and shown in Fig. [3]. The difference of services lies in the amount of control a user can have on their own data and types of services delivered by providers.

3.3.1 SOFTWARE AS A SERVICE (SaaS)

In Software as a Service model, users are given facility to access application software, databases. The infrastructure and platform that runs application are managed by cloud providers. It is also called "on-demand software". In this model, the application is installed and operated on cloud while the application is accessed using cloud clients by users. The users don't need to be bothered about managing the platform or application software. This model uses load balancers that distribute the work over many virtual systems. In general, pay-per-basis or subscription fee is paid to use service. It helps to cut down the IT set up cost with the help of outsourcing hardware and software. Also, the pricing model is scalable and adjustable if some users are added or removed at any point.

Examples of SaaS are Google apps, Microsoft office 365 etc.

3.3.2 INFRASTRUCTURE AS A SERVICE (IaaS)

In Infrastructure as service model, Resources such as computers, physical or virtual machines, and other kind of resources are offered to variety of users. Hypervisors runs virtual machines as guest. The pool of hypervisors has the ability to run a number of virtual machines together and to scale them up and down according to varying requirements of users. Other resources such as firewalls, load balancers, IP addresses etc. are also offered by IaaS.
Windows Azure virtual machines, HP clouds etc. are some of the examples of IaaS.

3.3.3 PLATFORM AS A SERVICE (PAAS)

In Platform as a service model, computing platforms such as operating system, web server, programming language execution platform, database etc. are offered to users. Software solutions can now be built upon clouds without need to buy and manage required hardware and software. With some PaaS providers, scaling of resources is also done automatically according to demand so a user doesn't need to allocate manually.

Examples of PaaS are Google App Engine, Cloud Foundry, windows Azure Compute etc.

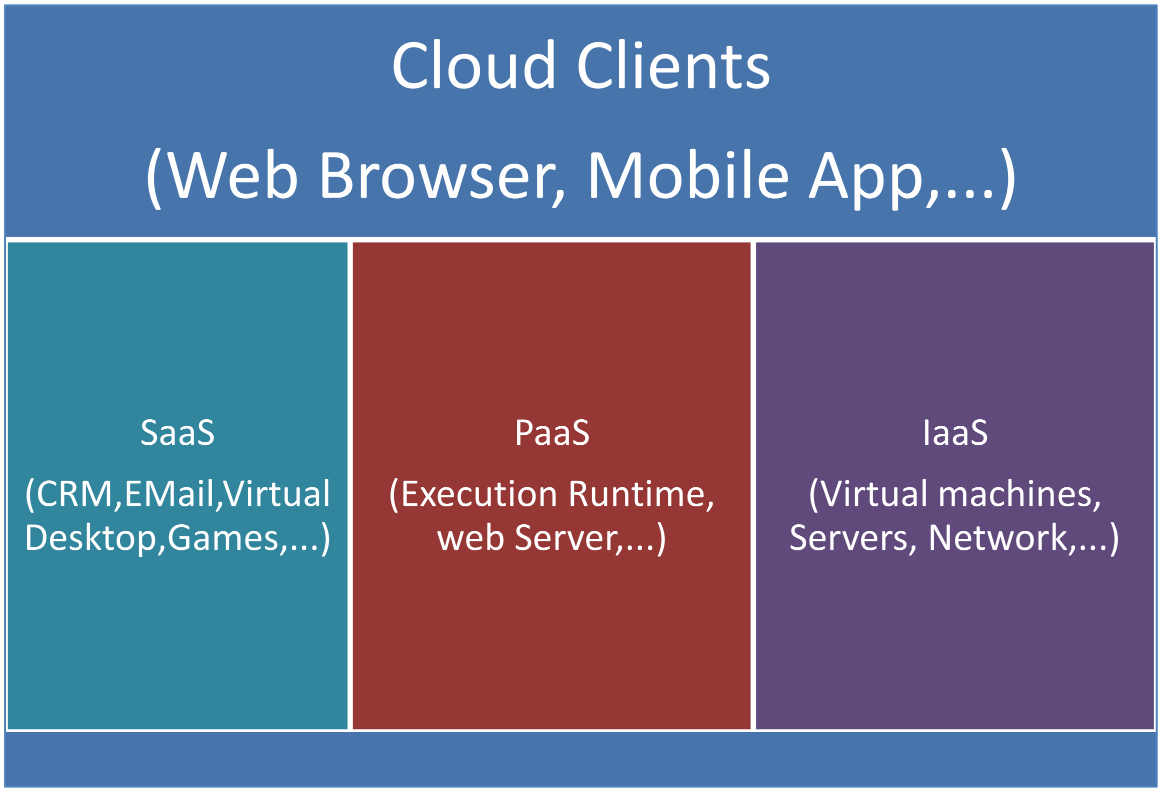

Cloud Clients
(Web Browser, Mobile App,...)
SaaS
(CRM,EMail,Virtual Desktop,Games,...)
PaaS
(Execution Runtime, web Server,...)
IaaS
(Virtual machines, Servers, Network,...)

Figure 3: Cloud Service Models

3.4 CHARACTERISTICS OF CLOUD COMPUTING

According to definition of cloud computing given by **National Institute of Standards and Technology,** the five most essential characteristics of cloud computing are as follows:

- **On Demand Self Service:** Users have access to services, resources etc. whenever needed without any human interaction with service provider. Users are billed on the basis of pay-for-what-you-use model or have to pay subscription fee.

- **Broad Network Access:** Users can access services, resources, and business management solutions over Internet using their laptops, smart phones, tablets, and office computers. Services can be used remotely i.e. from anywhere having online access point.

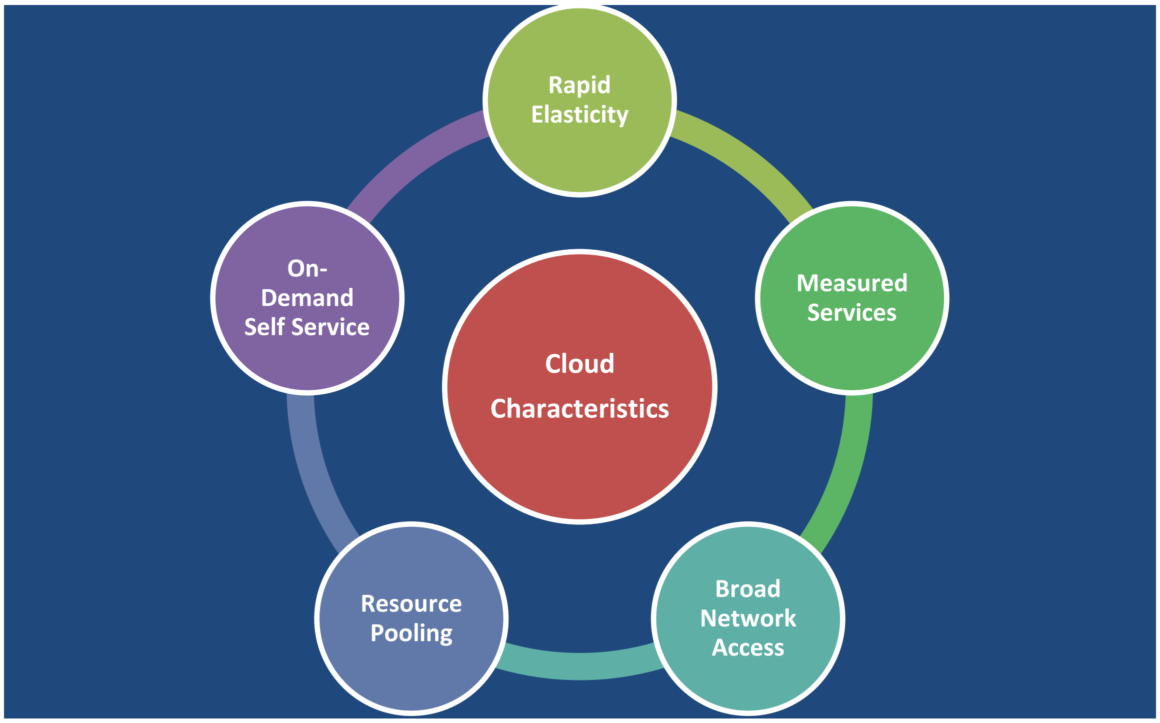


Figure 4: Cloud Key Characteristics

- **Rapid Elasticity:** The services and capabilities can be acquired and released automatically with change in requirements. Scalability and flexibility are key features of cloud computing. For users, the services should be made available whenever needed, in any quantity and at any time.

- **Resource Pooling:** The multi-tenant model is used to offer services to a large number of users. The computing resources are pooled and on the basis of demand, these physical and virtual resources are dynamically assigned and re-assigned.

- **Measured Service:** The user and service provider can control and monitor the resources by measuring services such as storage levels, processing, bandwidth, and the number of active user account. The users have to pay only for what they use. This provides transparency to both, users and service providers.

3.5 CLOUD COMPUTING SECURITY

Cloud computing security is an emerging sub-domain of information security. It is basically refers to an extensive set of policies, technologies that are deployed to protect data, application and related infrastructure of cloud computing. Cloud computing faces many security issues. Security of cloud can be measured against following parameters:

- **Confidentiality**
  Cloud computing provides storage as one of its services. More and more personal information and potentially secure data is now stored on cloud which demands confidentiality. To provide confidentiality, data encryption is commonly used. Data must be protected from unauthorized access in order to ensure confidentiality.

- **Availability**
  Availability is an important pillar of information security. It is the promise of reliable access to information and services by authorized users. It is said to be compromised when the server cannot make services available as per expectation. VM denial of service attack is one such availability attack in which VM occupies all reachable resources such that hypervisor cannot hold up more VMs and accessibility is endangered.

- **Integrity**
  Data integrity can be defined as the accuracy and consistency of stored data. It can be compromised if unauthorized users are given access to data stored on cloud. Data integrity should be ensured by cloud services. Measures to ensure data integrity includes file permission and access control.

3.6 AUTHENTICATION

To provide security, it is important that only legitimate users are allowed to access services. Authentication is the process to determine whether someone or something is, in fact, who or what it is claims to be. While it's important to authenticate user to a server before allowing him/her to access the services, server authenticating to user is equally important for achieving high degree of security. The authentication process in which both parties in communication link authenticate each other is called **Mutual Authentication.**

3.6.1 TWO FACTOR AUTHENTICATION

Two factor authentication provides unambiguous identification of users by combining two different components. The factors are broadly categorized in three categories as shown in Fig. [5]

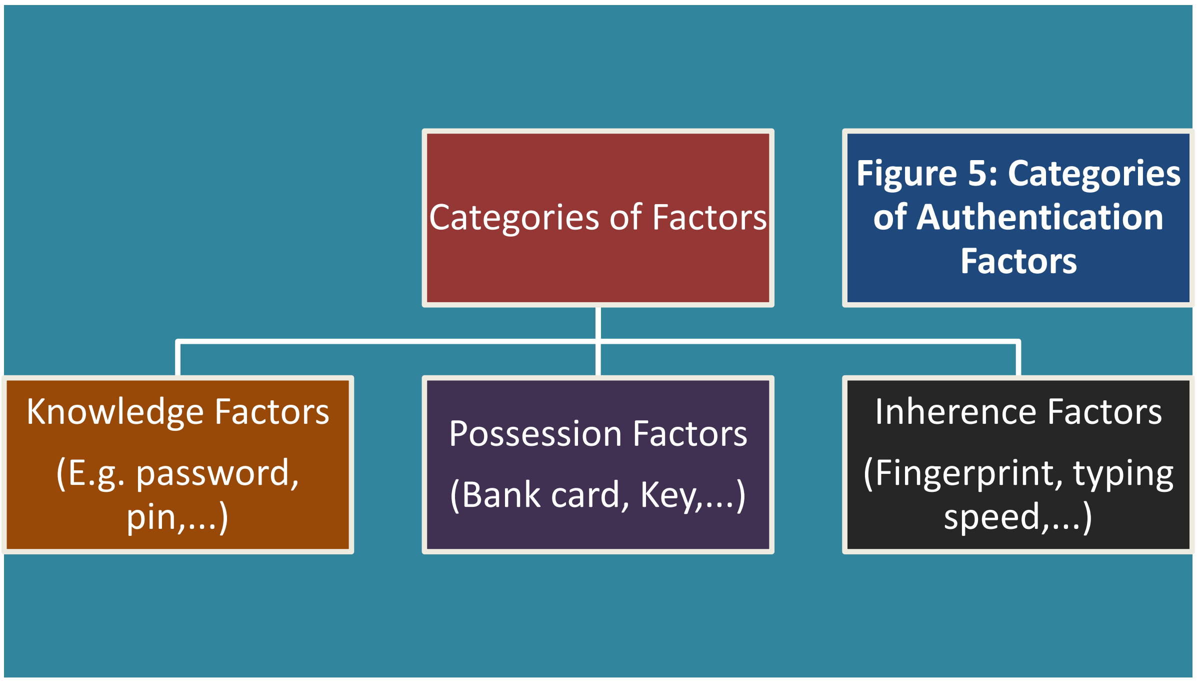

Categories of Factors
Figure 5: Categories of Authentication Factors
Knowledge Factors (E.g. password, pin,...)
Possession Factors (Bank card, Key,...)
Inherence Factors (Fingerprint, typing speed,...)

Figure 5: Categories of Authentication Factors

- **Knowledge Factors:** It is the most commonly used factor in authentication. It is something that user knows such as password, personal identification number (PIN) ,security questions.

- **Possession Factors:** It is one of the oldest factors. It is like having a key for a lock and only user can open that lock. It is something only the user possesses. Examples of this factor are bank card, key, USB stick with secret token.
- **Inherence Factors:** It is something that is inseparable from user. Basically these are biometrics such as eye iris, typing speed, fingerprint etc.

A simple example to explain two factor authentication is withdrawing money from cash machines. A user having correct combination of bank card (possession factor) and PIN (Knowledge factor) can perform transactions.

CHAPTER 4: THE FACTORS

This chapter describes all the factors used in the proposed scheme i.e. Alphanumeric Password, Stego-Image, One Time Password and Digital Signature in detail.

4.1 FACTORS USED

A factor based authentication scheme comprises of key factors which when combined give an unambiguous identification to user. The factors can be something that user knows, something that only user possesses or something that user inherits and cannot be separated from user. The proposed scheme uses following factors:

1. Alphanumeric Password
2. Stego-image
3. OTP
4. Digital signature

4.1.1 ALPHANUMERIC PASSWORD

This factor comes under the category “knowledge factor”. Alphanumeric password is one of the most used components in authentication scheme. An alphanumeric password is a secret word or a string of character which consists of lowercase letters, uppercase letters, special characters (e.g. @, #, &), numerals. These four types of characters are collectively known as complex characters. The passwords should be of high entropy. Long passwords using complex character set increases the entropy but at the same time, passwords should be easy to remember. This factor, along with other factors described below, gives a strong mutual authentication scheme.

4.1.2 STEGO-IMAGE

This factor comes under the category “possession factor”. The proposed scheme use steganography technique to create the second key factor. Steganography is an ancient technique to conceal information. **Digital Steganography** is the practice of hiding secret information within digital media such as video, image or any other file. There are a large number of steganography algorithms. In proposed scheme, Least Significant Bit algorithm is used for steganography. At the time of registration, data owner generates a STEGO-IMG which needs to be uploaded every time when user wants to login. If the correct STEGO-IMG is uploaded then only further verification will be carried out. Considering the possibility that somehow attacker managed to get STEGO-IMG, only the legitimate user who is assumed to possess private key of public cryptosystem used and correct password, can only get to subsequent verification steps.

4.1.3 ONE TIME PASSWORD (OTP)

This factor comes under “possession factor” category. One time password is a password which is valid only for a single session or login. Now-a-days, mobile phones are very popular among people. The proposed scheme uses OTP as one of the key factor to authenticate user. This is another layer of security. OTP is sent on the user’s registered mobile number after the user has correctly entered password, username and uploaded the image file.

4.1.4 DIGITAL SIGNATURE

This factor provides more strength to authentication scheme. We are using digital signature when user receives OTP. Considering the possibility that the user may have lost his/ her mobile phone and attacker is getting access to mobile phone, only the legitimate user, who is assumed to have private key of digital signature key-pair, can sign the OTP and get verified at the service provider end.

CHAPTER 5: THE PROPOSED SCHEME

This chapter describes the proposed scheme explaining the phases i.e. registration phase and log-in phase in detail. Diagrammatic representation is also given for each phase to increase the understanding of proposed algorithm.

5.1 THE ENTITIES

The proposed scheme consists of mainly three entities:

- User: User is the entity who wants to access the cloud services.
- Data Owner: Data owner is the entity which owns the data such as images, video, database that is stored on cloud.
- Service Provider: Service provider is the entity that allows only legitimate users to log-in and access the cloud services.

5.2 THE SCHEME

This section presents the proposed scheme to achieve mutual authentication and session key agreement. Before explaining the scheme in detail, all the notations used in the scheme is summarized in table [1]. Some assumptions have to be made which should never be violated during execution of scheme. The Notations, assumptions and complete scheme is described in subsequent sections.

5.2.1 NOTATIONS USED

This section summarizes all the notations used in our proposed scheme along with their brief description. The notations along with description are shown in table [1].

5.2.2 THE ASSUMPTIONS

The proposed scheme has made some assumptions which are as follows:

- All the entities i.e. users, service provider and data owner should be honest during registration phase.
- After completion of registration phase, users are not trusted. Users have to authenticate themselves to access service to the service provider.
- After successful mutual authentication between both parties, user and service provider, service provider is trusted and it is assumed that the service provider never compromises with adversaries.

Notation	Description
ID	Unique Username
PWD	Alphanumeric Password
H()	One- way –Hash Function
STEGO_IMG	Stego image
EXOR	Boolean function XOR
E(x, y)	Symmetric Encryption (AES) of x using y as key
D(x, y)	Symmetric Decryption (AES) of x using y as key
Enc(x, y)	Asymmetric Encryption (RSA) of x using y as key
Dec(x, y)	Asymmetric Decryption (RSA) of x using y as key
ST_{priv}	Asymmetric Private Key used in decrypting ENC_KEY_USER
ST_{pub}	Asymmetric Public Key used in encrypting KEY_USER
DS_{priv}	Private key used in Digital signature DSA
DS_{pub}	Public key used in Digital signature DSA
OTP	One time password
||	String concatenation
DTS	Data to be signed digitally
SIGN	Digitally signed DTS
r, k, α	Random numbers generated during Login phase
SN, SR, g	Random Numbers generated during registration phase
SN_PWD_HASH	Secret kept by service provider (generated using H(),SN, PWD)
Key_Session	Session key established between user and service provider

Table 1: Description of Notations

5.2.3 THE DETAILED SCHEME

Our scheme consists of four phases: Registration phase, Log-In phase, Change credentials phase

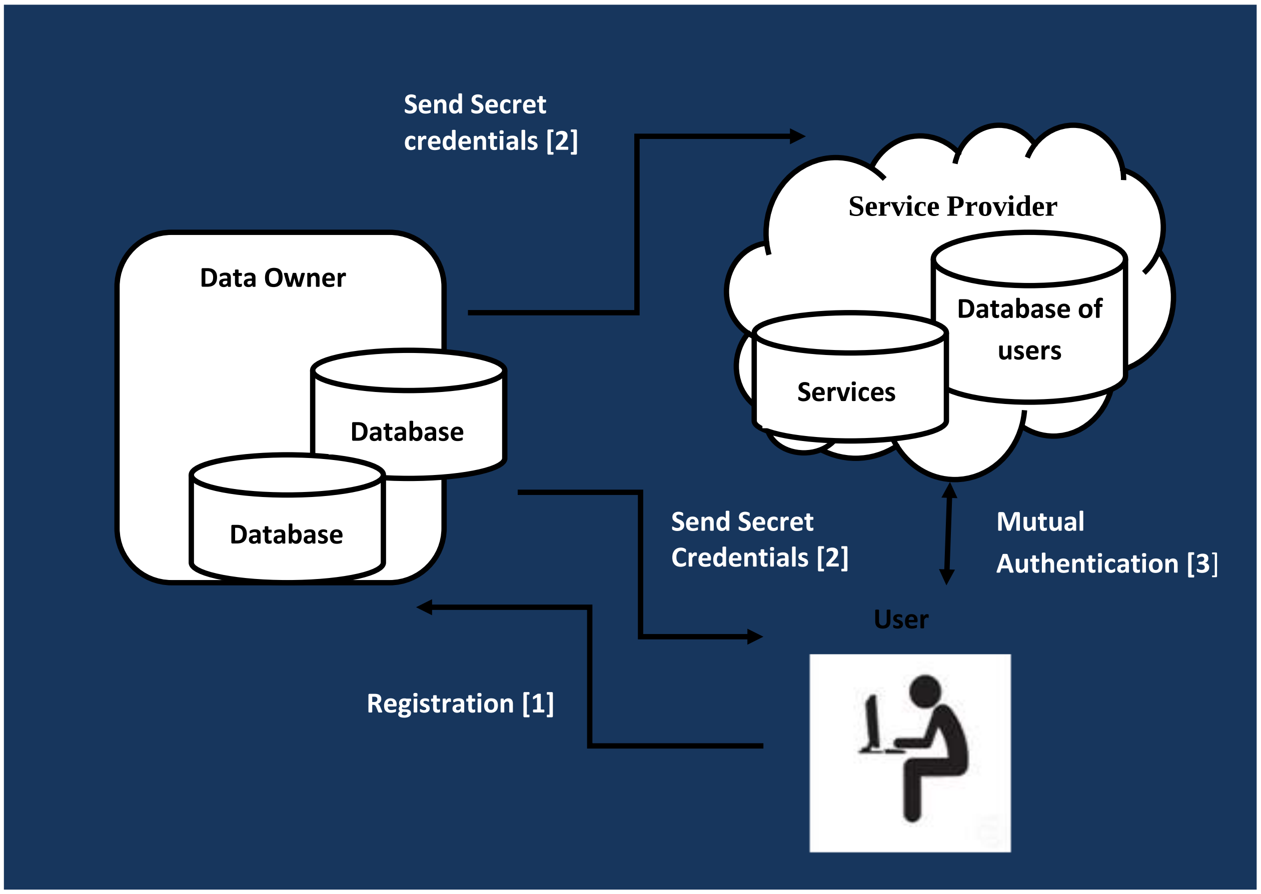


Figure 6: Basic Architecture of Proposed Model

and Forget/Lost credentials phase. Among all phases, registration and Login are elementary phases. Basic architecture of proposed scheme is depicted in Fig. [6]. A user who wants to access cloud services needs to get registered. The registration phase is executed only once while log in phase is executed for every log in request. Change credentials phase includes changing password request, changing STEGO_IMG file request, changing registered email id request and changing registered mobile number request. Change credentials phase can always be entered only after user successfully log-in. Forget credentials phase can be executed by only registered user in case of forgetting password or loss of image file or both.

5.2.3.1 REGISTRATION PHASE

The registration phase includes all the three entities i.e. user, service provider and data owner. User needs to fill fields such as user name, password etc. on registration page. Then, data owner generates some secret credentials and send them to user and service provider. Along with secret credentials, STEGO_IMG is also generated and given to user. The complete process of registration phase is depicted in Fig. [7].

AT USER END:

First of all, the user needs to choose username, password and fill them along with registered email-id and mobile number. At the user side, hash of username and password is calculated. The complete steps at user side are as follows:

Step 1: User Inputs his/ her username i.e. “ID”.

Step 2: User select alphanumeric password i.e. “PWD”.

Step 3: User enters registered email-id and mobile no**.**

Step 4: Hash of password and username i.e. H (PWD), H (ID) is calculated.

Step 5: Now, the credentials of users i.e. H (ID), H (PWD), registered mail id and mobile no are sent to data owner after user presses “Register” button.

AT DATA OWNER END:

Data owner has the database which consists of every registered user’s H (ID), H (PWD) registered email-id and Mobile number. After receiving the credentials sent by new user, data owner performs following steps:

Step 6: First of all, data owner checks the availability of username selected by user. If the username is not available then a message is displayed to notify user so that he/ she can choose other username and try to register again. If username is available, then a flag message is sent to user to generate public- private key pair ST_{priv}, ST_{pub}.

AT USER END:

Step 7: Generate Public-private Key pair i.e. ST_{priv},ST_{pub} using “RSA” algorithm for asymmetric encryption decryption.

Step 8: ST_{pub} is sent to data owner.

AT USER END:

Step 9: Data owner stores credentials of user in database.

Step 10: Data owner generates three unique random numbers i.e. SN, SR, g and public-private key pair i.e. DS_{pub} and DS_{priv} for digital signature.

Step 11: Now, with the help of hash function some key credentials are generated as follows:

$$SN\_PWD\_HASH = H(H(SN) \| H(PWD))$$

$$KEY\_USER = H(H(H(g) \| H(SR)) \| H(ID))$$

Step 12: Data owner sends a set of credentials i.e. [SN_PWD_HASH, $DS_{pub,}$ H (ID), g, Mobile number] to Service provider.

Step 13: Now, the data owner encrypts KEY_USER using ST_{pub} as key shown below:

$$ENC\_KEY\_USER = Enc(KEY\_USER, ST_{pub})$$

Step 14: After encrypting KEY_USER, data owner selects an image from database and embed ENC_ KEY_USER into the selected image IMG using steganography algorithm and generate STEGO_IMG. Here, we are using Least Significant Bit algorithm during implementation for demonstration purpose only. For real-time implementation, some other advanced steganography algorithm can be used.

Step 15: Now, Data owner sends a set of credentials to user i.e. [DS_{priv}, g, SN, STEGO_IMG].

<u>AT USER END:</u>

Step 16: After receiving credentials from data owner, user executes extracting algorithm of steganography on STEGO_IMG and extracts ENC_KEY_USER.

Step 17: Now, user decrypts ENC_KEY_USER and get KEY_USER.

$$KEY\_USER = Dec(ENC\_KEY\_USER, ST_{priv})$$

Step 18: Using KEY_USER, user encrypts [SN, g, DS_{priv}] and stores them in database of local system. After this, ST_{priv} is encrypted using H (H (PWD) || H (ID)). It is assumed that user now saves the STEGO_IMG secretly.

$$D1 = E(SN, KEY\_USER)$$

$$D2 = E(g, KEY\_USER)$$

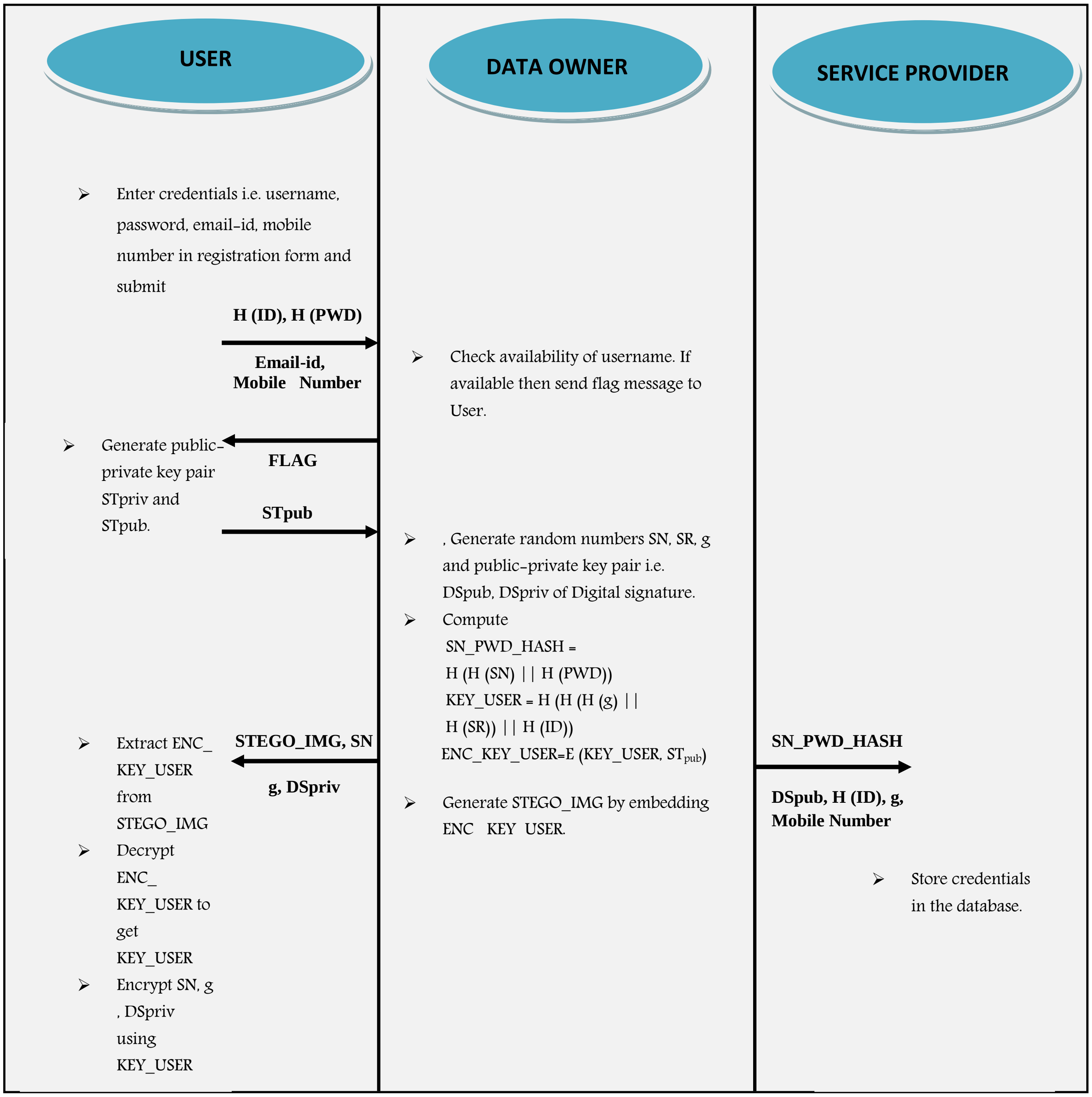

USER
DATA OWNER
SERVICE PROVIDER
Enter credentials i.e. username, password, email-id, mobile number in registration form and submit
H (ID), H (PWD)
Email-id, Mobile Number
Check availability of username. If available then send flag message to User.
Generate public-private key pair STpriv and STpub.
FLAG
STpub
, Generate random numbers SN, SR, g and public-private key pair i.e. DSpub, DSpriv of Digital signature.
Compute SN_PWD_HASH = H (H (SN) | | H (PWD)) KEY_USER = H (H (H (g) | | H (SR)) | | H (ID))
ENC_KEY_USER=E (KEY_USER, ST_pub)
Generate STEGO_IMG by embedding ENC KEY USER.
STEGO_IMG, SN
g, DSpriv
Extract ENC_ KEY_USER from STEGO_IMG
Decrypt ENC_ KEY_USER to get KEY_USER
Encrypt SN, g , DSpriv using KEY_USER
SN_PWD_HASH
DSpub, H (ID), g, Mobile Number
Store credentials in the database.

Figure 7: Registration Phase

$$D3 = E\ (DS_{priv},\ KEY\_USER)$$

$$D4 = E\ (ST_{priv},\ H\ (H\ (PWD)\ ||\ H\ (ID)))$$

AT SERVICE PROVIDER END:

After receiving credentials i.e. [SN_PWD_HASH, $DS_{pub,}$ H (ID), g, Mobile Number] from data owner, service Provider stores them in its database.

5.2.3.2 LOG-IN PHASE

During log-in phase, user and service provider interact. The authenticity of user and service provider is to be proved to each other. The proposed algorithm uses alphanumeric password, image file STEGO_IMG, OTP and digital signature as key factors to authenticate user. The secret credentials sent to user and service provider by data owner at the time of registration are used in algorithm. To prove his/her identity, user needs to have correct credentials along with key factors. Also, the secret credentials are used in such a way that they prove the legitimacy of service provider to user. Complete process is explained below:

AT USER END:

If a registered user wishes to access services provided by cloud provider, he/she needs to Log-in every time and prove his/ her identity. User must have alphanumeric password that he/ she chose at the time of registration and the image file, STEGO_IMG, provided by data owner at the time of registration is also needed. User must have access to his/her mobile phone because the OTP is sent to his/her phone during Log-in phase. At the same time, service provider also proves its identity to the user. During Log-in phase session Key agreement is also done. The Login phase consists of two parts: Mutual authentication and updating credentials.

PART 1: MUTUAL AUTHENTICATION

The first part of Login phase is Mutual Authentication. Both parties, user and service provider, authenticate to each other. Every step of this part is explained in detail. Complete process of mutual authentication is depicted in Fig. [8].

Step 1: Registered User enters his username “**ID**”.

Step 2: User enters password “**PWD**” that was chosen during registration phase.

Step 3: User needs to upload image file “**STEGO_IMG**”.

Step 4: After providing password and image file, user needs to submit by clicking on "Submit" button.

Step 5: Now, steganography extraction algorithm is executed on STEGO_IMG at client side. ENC_KEY_USER is extracted from image. H (H (PWD) || H (ID)) is calculated and used to decrypt $ST_{priv.}$

$$ST_{priv} = D\ (D4,\ H\ (H\ (PWD)\ ||\ H\ (ID)))$$

Now, KEY_USER is extracted from ENC_KEY_USER by applying decryption algorithm.

$$KEY\_USER = Dec\ (ENC\_KEY\_USER,\ ST_{priv})$$

Step 6: Using KEY_USER, credentials stored in the database are decrypted.

$$SN = D\ (D1,\ KEY\_USER)$$

$$g = D\ (D2,\ KEY\_USER)$$

$$DS_{priv} = D\ (D3,\ KEY\_USER)$$

Step 7: If the image file is correct then only this step is reached. Else the process would have been terminated with "Unsuccessful Log-In" message. Now, if the process is successful till now, two random numbers "r" and "k" are generated.

Step 8: Now, first of all the authenticity of service provider need to be verified. So, data which will act as challenge for service provider is generated. The data to be sent to service provider i.e. K and E1 is generated as follows:

$$E1 = E\ (H\ (r),\ g)$$

$$K = E\ (H\ (k),\ g)$$

g is used as key in encryption algorithm to compute E1 and K.

Step 9: H (ID), K and E1 are sent to Service provider.

<u>AT SERVICE PROVIDER END:</u>

After receiving credentials from user, Service provider performs following steps:

Step 10: Using H (ID), service provider extracts credentials related to user i.e. g, SN_PWD_HASH stored in database.

Step 11: Now, service provider decrypts K and E1 as follows:

R1= D (E1, g)

R2= D (K, g)

Step 12: After decrypting K and E1, Service provider use R2 as a key to encrypt R1 as follows:

R = E (R1, R2)

R is sent to user. R is basically acts as response to the challenge sent by user.

<u>AT USER END:</u>

After receiving R from service provider, following steps are performed at client side:

Step 13: Decryption of R with H (k) as key is done as follows:

R' = D (R, H (k))

Step 14: R' is compared with H(r). If both are equal then next step is executed else log in process is terminated and "Unsuccessful log-in" message on the screen is displayed.

Step 15: Now, Key_Session, X, E2 and E3 are calculated as shown below:

Key_Session = H (H (ID) || g) EXOR H(r)

X = H (H (H (SN) || H (PWD)) || g)

E2 = E (H (H (ID) || g), g)

E3 = E(X, Key_Session)

Step 16: After completing step 3, E2 and E3 are sent to service provider.

<u>AT SERVICE PROVIDER END:</u>

Step 17: Service provider decrypts E2 using g as key

E2' = D (E2, g)

Step 18: Key_Session is computed by applying EXOR operation on E2' and R1.

$$Key\_ Session = E2' \text{ EXOR } R1$$

Step 19: Now, X is calculated using SN_PWD_HASH and g as shown below:

$$X= H (SN\_PWD\_HASH \| g)$$

Step 20: After computing X, E3 is decrypted using Key_Session as key to get X.

$$X'= D (E3, Key\_Session)$$

Step 21: Now, X and X' are compared to see if both are equal. If both are equal then only next step is executed else the process is terminated with "unsuccessful login" message displayed on screen.

Step 22: Now, one time password (**OTP**) is generated and sent to the user's registered mobile phone.

<u>AT USER END:</u>

Step 23: After receiving **OTP**, data to sign i.e. DTS is computed as follows:

$$DTS =H (H (H (SN) \| H (PWD)) \| OTP)$$

Step 24: After calculating DTS, it is digitally signed using private key of digital signature algorithm DS_{priv} to get signature i.e. SIGN as shown below:

$$SIGN = DS (DTS, DS_{priv})$$

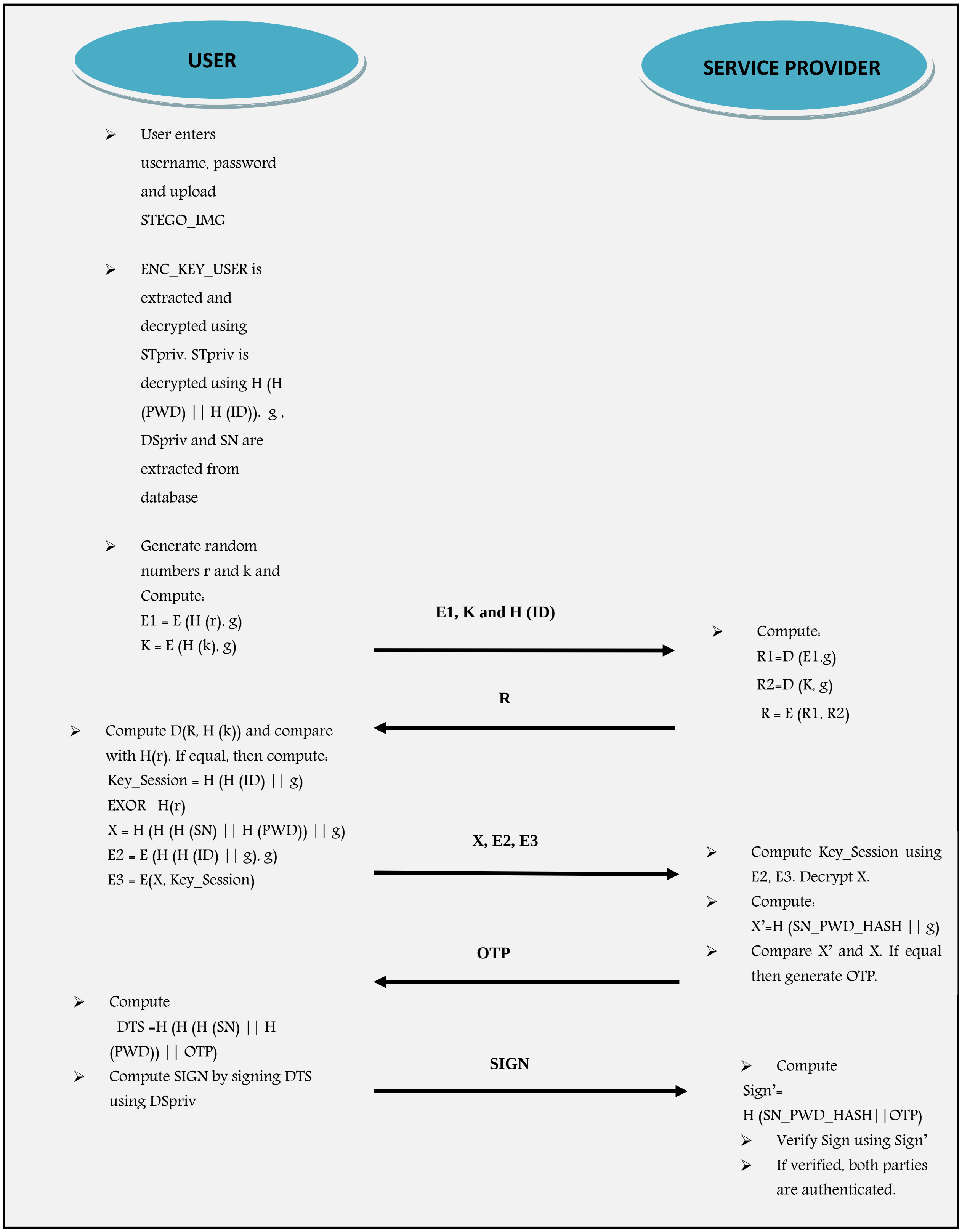
Step 25: Now, SIGN is sent to service provider to get verified.

<u>AT SERVICE PROVIDER END:</u>

Service provider receives signature SIGN from user which needed to be verified.

Step 26: After receiving SIGN, Service Provider first computes SIGN'

$$SIGN' = H (SN\_PWD\_HASH \| OTP)$$



USER

- User enters username, password and upload STEGO_IMG
- ENC_KEY_USER is extracted and decrypted using STpriv. STpriv is decrypted using H (H (PWD) || H (ID)). g , DSpriv and SN are extracted from database
- Generate random numbers r and k and Compute:
  E1 = E (H (r), g)
  K = E (H (k), g)

SERVICE PROVIDER

- Compute:
  R1=D (E1,g)
  R2=D (K, g)
  R = E (R1, R2)

USER

- Compute D(R, H (k)) and compare with H(r). If equal, then compute:
  Key_Session = H (H (ID) || g) EXOR H(r)
  X = H (H (H (SN) || H (PWD)) || g)
  E2 = E (H (H (ID) || g), g)
  E3 = E(X, Key_Session)

SERVICE PROVIDER

- Compute Key_Session using E2, E3. Decrypt X.
- Compute:
  X'=H (SN_PWD_HASH || g)
- Compare X' and X. If equal then generate OTP.

USER

- Compute
  DTS =H (H (H (SN) || H (PWD)) || OTP)
- Compute SIGN by signing DTS using DSpriv

SERVICE PROVIDER

- Compute Sign'=
  H (SN_PWD_HASH||OTP)
- Verify Sign using Sign'
- If verified, both parties are authenticated.

Figure 8: Log-In Phase (Mutual Authentication)

Step 27: Service provider verifies SIGN using public key of digital signature algorithm DS_{pub} and SIGN'. If verification successful, next step is executed else the process is terminated with "Unsuccessful login" message displayed on screen.

Step 28: If this step is reached with successful completion of all the previous steps from start of the login phase, we say that both user and service provider has proved their authenticity to each other. After successful completion of part 1, part 2 is started.

PART 2: UPDATING CREDENTIALS

After successful completion of mutual authentication part of login phase, next part of updating credentials is started. SN, SN_PWD_HASH is updated using "α". The complete process is depicted in Fig. [9].

AT SERVICE PROVIDER END:

Step 1: A random number **"α"** is generated and sent to user.

AT USER END:

After receiving "α", following steps are executed:

Step 2: Now, a new random number is generated and replace SN with this new random number.

Step 3: Now, new SN_PWD_HASH is calculated as follows:

$$SN\_PWD\_HASH = H\ (H\ (SN)\ ||\ H\ (PWD))$$

Step 4: After this, E4 is calculated as shown below:

$$E4 = E\ (SN\_PWD\_HASH,\ Key\_Session)$$

E4 is sent to service provider and database is updated.

AT SERVICE PROVIDER END:

After receiving E4, following steps are performed:

Step 5: Decryption of E4 is performed using Key_Session as key.

$$E4' = D\ (E4,\ Key\_Session)$$

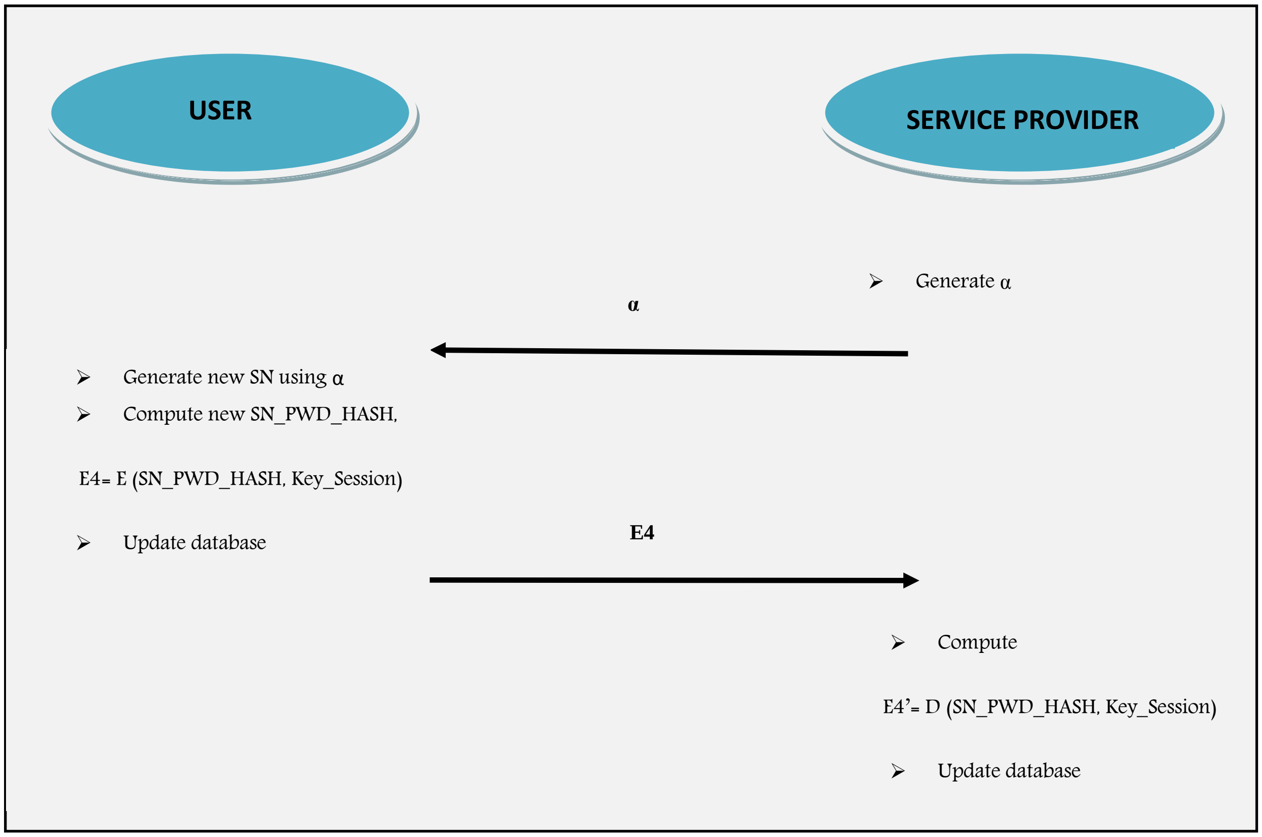

USER
SERVICE PROVIDER
➢ Generate α
α
➢ Generate new SN using α
➢ Compute new SN_PWD_HASH,
E4= E (SN_PWD_HASH, Key_Session)
➢ Update database
E4
➢ Compute
E4'= D (SN_PWD_HASH, Key_Session)
➢ Update database

Figure 9: Log-In Phase (Updating Database)

Step 6: Now, update SN_PWD_HASH with E4' i.e. new SN_PWD_HASH.

Step 7: After completion of step 2, user is allowed to access Cloud services.

5.2.3.3 CHANGE CREDENTIALS PHASE

A user can enter in this phase only after successfully log-in. This phase provides users the flexibility to change their credentials. This phase consists of four types of request:

- Change password
- Change STEGO_IMG file
- Change Registered email-id
- Change Registered mobile number

In order to complete each of these requests, some steps has to be followed. Every request along with the steps is described below.

5.2.3.3.1 CHANGE PASSWORD

In change password request, user needs to enter password again. The steps are as follows:

<u>AT USER END:</u>

Step 1: User has to enter current password again.

Step 2: If verified, then user can choose new password and have to submit.

Step 3: After submitting the new password, new SN_PWD_HASH is calculated using SN and new password. Encrypt ST_{priv} with newly calculated H (H (PWD) || H (ID)) and update database.

Step 4: Now, new SN_PWD_HASH is sent to service provider and H (PWD) to data owner.

<u>AT SERVICE PROVIDER END:</u>

Step 5: Service provider updates SN_PWD_HASH.

Step 6: "password is updated successfully" message is displayed.

<u>AT DATA OWNER END:</u>

Step 7: data owner updates database.

5.2.3.3.2 CHANGE STEGO_IMG FILE

User needs to enter password again to change STEGO_IMG file.

<u>AT USER END:</u>

Step 1: User has to enter current password again.

Step 2: If verified then connection to data owner is established.

Step 3: User sends H (ID) to data owner.

<u>AT DATAOWNER END:</u>

Step 4: New SR, g are generated.

Step 5: New KEY_USER is calculated using H (ID), SR, g.

$$KEY\_USER = H(H(H(g) \| H(SR)) \| H(ID))$$

Step 6: Now, the data owner encrypts KEY_USER using ST_{pub} as key shown below:

$$ENC\_KEY\_USER = Enc(KEY\_USER, ST_{pub})$$

Step 7: After encrypting KEY_USER, data owner selects an image from database and embed ENC_ KEY_USER into the selected image IMG using steganography algorithm as used at the time of registration and generate new STEGO_IMG.

Step 8: Now, Data owner sends new STEGO_IMG, g to user.

Step 9: New g is sent to service provider.

<u>AT USER END:</u>

Step 10: After receiving new STEGO_IMG from data owner, user executes extracting algorithm of steganography on STEGO_IMG and extracts ENC_KEY_USER.

Step 11: Now, D4 is decrypted to get ST_{priv} as shown below:

$$ST_{priv} = D\,(D4,\ H\,(H\,(PWD)\ ||\ H\,(ID)))$$

Using ST_{priv}, ENC_KEY_USER is decrypted:

$$KEY\_USER = Dec\,(ENC\_KEY\_USER,\ ST_{priv})$$

Step 12: Using new KEY_USER, user encrypts [SN, g, DS_{priv}] and stores them in database of local system.

$$D1 = E\,(SN,\ KEY\_USER)$$

$$D2 = E\,(g,\ KEY\_USER)$$

$$D3 = E\,(DS_{priv},\ KEY\_USER)$$

<u>AT SERVICE PROVOIDER END:</u>

Step 13: service provider updates g in the database.

5.2.3.3.3 CHANGE EMAIL-ID

<u>AT USER END:</u>

Step 1: User has to enter current password again.

Step 2: If verified then user needs to enter new email-id.

Step 3: User sends H (ID) and new email-id to data owner.

<u>**AT DATA OWNER END:**</u>

Step 4: Data owner updates email-id in the database.

5.2.3.3.4 CHANGE MOBILE NUMBER

<u>**AT USER END:**</u>

Step 1: User has to enter current password again.

Step 2: If verified then user needs to enter new email-id.

Step 3: User sends H (ID) and new phone number to data owner and service provider.

<u>**AT DATA OWNER END:**</u>

Step 4: Data owner updates phone number in the database.

<u>**AT SERVICE PROVIDER END:**</u>

Step 5: Service provider updates phone number in the database.

5.2.3.4 FORGET/LOST CREDENTIALS PHASE

Only a registered user can enter in this phase. This phase provides users the flexibility to get new credentials i.e. alphanumeric password and STEGO_IMG file in the event of forgetting or loosing those credentials. This phase consists of three types of request:

- Forgot password
- Lost STEGO_IMG file
- Both credentials Lost

In order to complete each of these requests, some steps are needed to be followed. Every request along with the steps is described below.

5.2.3.4.1 FORGOT PASSWORD

In the event of forgetting password, user needs to enter his/her username. New password is sent to his/her email-id. User needs to submit new password in order to use it. Along with new password, STEGO_IMG is also need to be uploaded.

AT USER END:

Step 1: user enters his/her username “ID”.

Step 2: H (ID) is sent to data owner.

AT DATAOWNER END:

Step 3: First of all, data owner checks whether request is coming from a registered user or not.
Step 4: If registered user is sending the request then data owner generates new password and sends it to user’s registered mail- Id.

AT USER END:

Step 5: User need to enter new password twice.

Step 6: User needs to upload STEGO_IMG file and click on submit.

AT DATA-OWNER END:

Step 7: Data owner check the validity of entered password. If valid then only next steps are executed else process is terminated with appropriate message.
Step 8: data owner generates new SN. SN, old H (PWD) is sent to user and SN_PWD_HASH is sent to service provider.

AT USER END:

Step 9: Now, extraction algorithm is executed and ENC_KEY_USER is extracted from image file. After this, ENC_KEY_USER is decrypted to get KEY_USER.

$$ST_{priv} = D\ (D4,\ H\ (H\ (PWD)\ \|\ H\ (ID)))$$

$$KEY\_USER = Dec\ (ENC\_KEY\_USER,\ ST_{priv})$$

Step 10: Database is updated with new credentials encrypted with KEY_USER. After this, ST_{priv} is encrypted with new H (H (PWD) || H (ID)). If the uploaded STEGO_IMG file is not valid, then user cannot update database and process terminated.

AT SERVICE PROVIDER END:

Step 11: service provider updates database.

5.2.3.4.2 LOST STEGO_IMG FILE

To get new STEGO_IMG file, user needs to enter username.

AT USER END:

Step 1: user enters his/her username "ID" and password "PWD".

Step 2: H (ID), H (PWD) is sent to data owner.

AT DATA OWNER END:

Step 3: First of all, data owner checks whether request is coming from a registered user or not.

Step 4: If registered user is sending the request and password is also correct then data owner generates new SR, g.

Step 5: New KEY_USER is calculated using H (ID), SR, g.

$$KEY\_USER = H(H(H(g) \| H(SR)) \| H(ID)).$$

Step 6: Now, the data owner encrypts KEY_USER using ST_{pub} as key shown below:

$$ENC\_ KEY\_USER = Enc(KEY\_USER, ST_{pub})$$

Step 7: After encrypting KEY_USER, data owner selects an image from database and embed ENC_ KEY_USER into the selected image IMG using steganography algorithm as used at the time of registration and generate new STEGO_IMG.

Step 8: Now, Data owners ends new STEGO_IMG to user's registered email-id.

AT USER END:

Step 9: User need to enter password twice.

Step 10: User needs to upload new STEGO_IMG file and click on submit. After clicking on submit button, it is mandatory to use new STEGO_IMG.

AT DATA-OWNER END:

Step 11: Data owner check the validity of entered password. If valid then only next steps are executed else process is terminated.

Step 12: Data owner generates new SN. SN and g is sent to user. SN_PWD_HASH and g is sent to service provider.

AT USER END:

Step 13: After receiving new STEGO_IMG from data owner, user executes extracting algorithm of steganography on STEGO_IMG and extracts ENC_KEY_USER.

Step 14: Now, user decrypts ENC_KEY_USER and get new KEY_USER.

$$ST_{priv} = D\ (D4,\ H\ (H\ (PWD)\ \|\ H\ (ID)))$$

$$KEY\_USER = Dec\ (ENC\_KEY\_USER,\ ST_{priv})$$

Step 15: Using new KEY_USER, user encrypts [SN, g, DS_{priv}] and update in database of local system. If the uploaded STEGO_IMG file is not valid, then user cannot update database and process terminated.

$$D1 = E\ (SN,\ KEY\_USER)$$

$$D2 = E\ (g,\ KEY\_USER)$$

$$D3 = E\ (DS_{priv},\ KEY\_USER)$$

AT SERVICE PROVIDER END:

Step 16: service provider updates database.

5.2.3.4.3 BOTH CREDENTIALS LOST

AT USER END:

Step 1: User needs to enters his/her username "ID" and submit.

AT DATA OWNER END:

Step 2: First of all, data owner checks whether request is coming from a registered user or not.

Step 3: If the user is registered, then sends a verification code to user on his/her registered mobile number.

AT USER END:

Step 4: Now, user enters verification code and clicks on submit button.

Step 5: verification code is sent to data owner.

AT DATA OWNER END:

Step 6: Validity of verification code is checked. If it is valid then new password and new SN, SR, g is generated. New STEGO_IMG is created using same procedure as used previously. These are sent to email id of user.

AT USER END:

Step 7: Now, user has to submit new password and upload new STEGO-IMG.

AT DATA OWNER END:

Step 8: If the entered password is valid then SN, g and old H (PWD) are sent to user and SN_PWD_HASH, g are sent to service provider.

AT USER END:

Step 9: After receiving new STEGO_IMG from data owner, user executes extracting algorithm of steganography on STEGO_IMG and extracts ENC_KEY_USER.

Step 10: Now, user decrypts D4 by calculating H (H (PWD) || H (ID)) with old H (PWD).

After this, ENC_KEY_USER is decrypted and get new KEY_USER.

$$ST_{priv} = D\ (D4,\ H\ (H\ (PWD)\ ||\ H\ (ID)))$$

$$KEY\_USER = D\ (ENC\_KEY\_USER,\ ST_{priv})$$

Step 11: Using new KEY_USER, user encrypts [SN, g] and update new D1 and D2 in database of local system. If the uploaded STEGO_IMG file is not valid, then user cannot update database and process terminated.

$$D1 = E\ (SN,\ KEY\_USER)$$

$$D2 = E\ (g,\ KEY\_USER)$$

$$D3 = E\ (DS_{priv},\ KEY\_USER)$$

AT SERVICE PROVIDER END:

Step 12: service provider updates database.

CHAPTER 6: IMPLEMENTATION & RESULTS

This chapter presents the implementation of proposed scheme. The complete process of each phase is explained in detail taking an example. It also contains snapshots of implemented scheme depicting each and every step of scheme.

6.1 THE GRAPHICAL USER INTERFACE (UI)

A graphical user interface is created to implement the proposed scheme using JAVA. The complete demonstration of proposed algorithm using GUI is provided in subsequent sections. The GUI of proposed scheme is shown below:

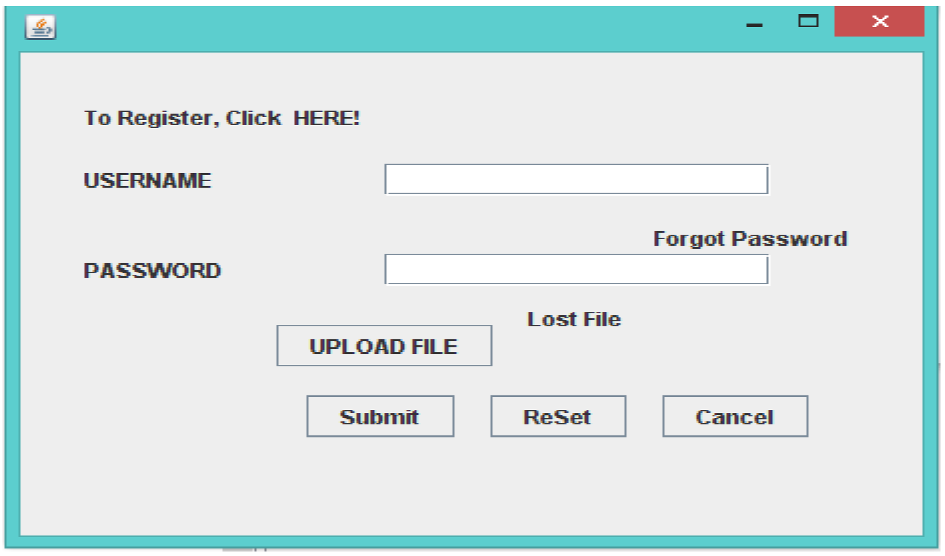


Figure 10: GUI of proposed Scheme

6.1.1 REGISTRATION PHASE

Every user who wants to access cloud services needs to register first. The homepage of GUI is LOG-IN page as shown in Fig. [10]. It consists of a link for registration. New user has to click on that link. After clicking on the link “HERE”, user is navigated to registration page as shown in Fig. [11]. On registration page, there are mandatory fields which need to be filled up. The fields are username, password, registered email-id, and registered Mobile number. Now, we are taking an example of a user who wants to access cloud services to demonstrate the working of GUI and proposed algorithm. Steps and corresponding outputs are as follows:

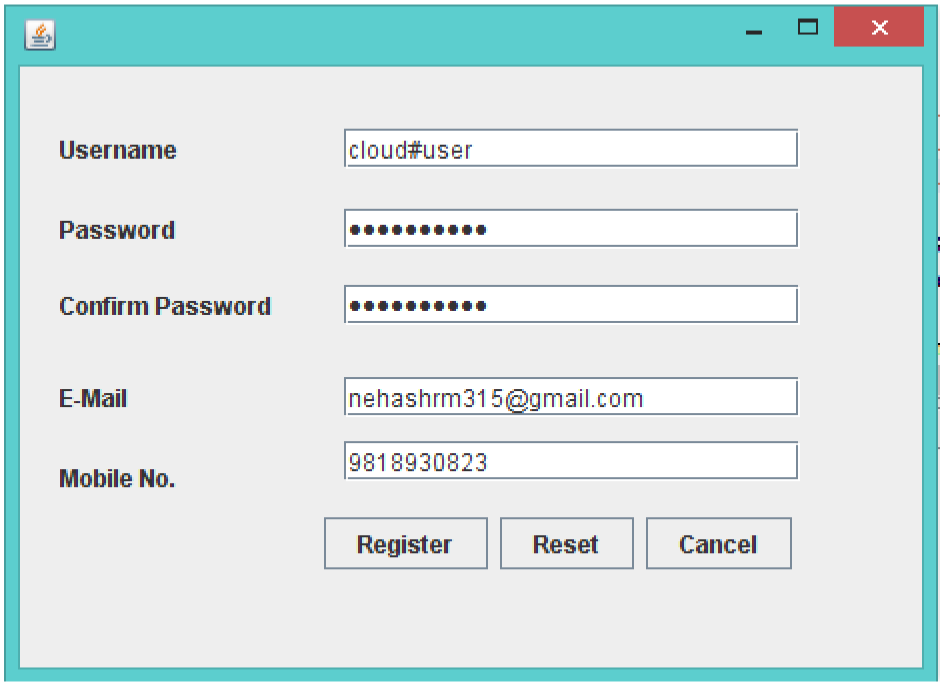
AT USER END:

First of all, the user needs to choose username, password and fill them along with registered email-id and mobile number. At user end, hash of username and password is calculated. The complete steps at user end are as follows:

Step 1: User Inputs his/ her username "cloud#user".

Step 2: User select alphanumeric password i.e. "secret.123".

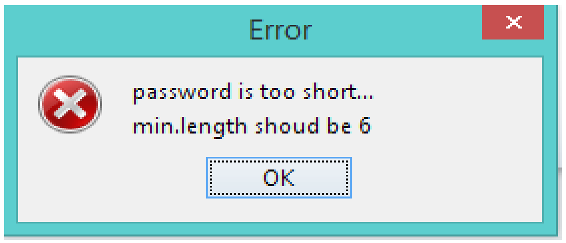
Step3: User enters registered email-id "nehashrm315@gmail.com" and mobile no "9818930823".



Username	cloud#user
Password	●●●●●●●●●●
Confirm Password	●●●●●●●●●●
E-Mail	nehashrm315@gmail.com
Mobile No.	9818930823

Figure 11: UI for Registration

Every field follows some rules. The very first rule is any field could not be left empty. The username chosen must be available. The password should consist of at least one uppercase letter, one lowercase letter, one numeral, one special character. The password length should be minimum 6 characters long and maximum 14 characters long. There are two password fields provided. Both typed password should match to each other. The mail-id and mobile number should be registered. If any of these rules is violated, then error message box is popped up. For example, if typed password length is less than 6 characters then a message box appears as shown above in fig. [12].



password is too short...
min.length shoud be 6

Figure 12: Example of Error Message Box

Step 4: After user presses "Register" button, hash of password and username i.e. H (PWD), H (ID) is calculated. Values of H (ID) and H (PWD) are as shown below:

H (ID) = 0x00301DE72A2A5C1BB3D46BE3D8C14094F9CEEB4F6696AFEA47
6A244D923BA4

H (PWD) = 0x5FF2D08FF8AE00A00D6A2A41D023416A54D9F4F1D6979A57
DA44BD0A6C14DC0E

Step 5: Now, the credentials of users i.e. H (ID), H (PWD), registered mail id and mobile no are sent to data owner.

<u>AT DATA OWNER END:</u>

Step 6: First of all, data owner checks the availability of username selected by user. Username "cloud#user" is available. Send a flag message to user.

<u>AT USER END:</u>

Step 7: Generate public-private key pair ST_{pub} - ST_{pri} using "RSA" algorithm for asymmetric encryption decryption.

ST_{pub} = Sun RSA public key, 2048 bits

modulus:
1618178674112717986279650628589018042599845798508460914829785258067314
3706305968968917136837982083921979972818029568444530172926428208676094
2548027376900222551584941494367009847820040375686101632567632085310868
8296305528139470626917746120501298901688076223145593430497651737238445
9959606315830486096489387337755309520017017471996752801785765759774096
1312362003790781854398609337357561899033153099569725061543466746232621
2980962087455870380693682681305983452653074624326866134656919834986485
2491376247572990893115739069760593158503087546461248098666461660503169
383175683580739619297639189597125697783688419019079753721
Public exponent: 65537

ST$_{pri}$ = Sun RSA private CRT key, 2048 bits
modulus:
1618178674112717986279650628589018042599845798508460914829785258067314
3706305968968917136837982083921979972818029568444530172926428208676094
2548027376900222551584941494367009847820040375686101632567632085310868
8296305528139470626917746120501298901688076223145593430497651737238445
9959606315830486096489387337755309520017017471996752801785765759774096
1312362003790781854398609337357561899033153099569725061543466746232621
2980962087455870380693682681305983452653074624326866134656919834986485
2491376247572990893115739069760593158503087546461248098666461660503169
383175683580739619297639189597125697783688419019079753721
public-exponent: 65537
private-exponent:
3513292392686553279334291895294740029014633850645725369960940298921192
0258331573379727778211948528636625575358613108533686288748363059226413
3469014685736800080336623971702730232484147047566647867586987036618757
7363646697269714443812290698172479984149356329880624520889129281042136
2095493884058163581938064364433889466190990577775182740023182057469535

```
0164055500926551853172904429320653027201099115389978989565440682430364
8283983647051116977599282158122681907062581498227558596354073342671534
3676700765860207397838465553373051336837142226271546506886802688948487
60157900636001371241418810038814074981949101463480152321
  prime-p:
1626410598066242202717982677624270572593581658286664599818923596231900
3423497633848539091835479408414495128808402474103731063693591653286739
4322811402818352079471477603421752370062212375205378620457703516733081
1718074564789925164273665485726222928837460631009941022137582402403532
65562050745486908363067327769
  prime-q:
9949385942496244495250284217641142371523753410406539409564032783586192
1784311854464866873804265560151624238657745409616926997946584555048406
2353572200536406569211202299520770390011538659780367634797618659290543
6002988318724219369149386703922985306733229509272771159762526789398331
9141267946768643791714250209
  prime-exponent-p:
9804324808551357355811120686772671546066287833427111150197020693094540
0041521738934378122456671038990411378890330125641409148014514769733008
0338024336045994721101006405295202844812521828469245335981581829435866
4349294571182015268468148121608540099421938317181094642963435612212393
7051740830400404179313989321
  prime-exponent-q:
7312844970780537670876088175592014099459834929570822640015555475309319
5787880159750562847111577765727813900180716181412749645102567679580721
2468858399069818643497621416419969020352713997308700565607233941407532
9237895346337880083188518814226623162875012061303834273246576978581833
899987605468430919094367201
  crt-coefficient:
1418780276288495209518001512046528529807908240644284424462311422979249
```

1893716169351859017069158732497656459217601276427622182727152943625044370133888865823072121200320801172930957689191865487199242417332981114866734562499

Step 8: Send ST_{pub} to data owner.

AT DATA OWNER END:

Step 9: Data owner store these credentials in database.

Step 10: Data owner generates three unique random numbers i.e. SN, SR, g and public-private key pair i.e. DS_{pub} and DS_{priv} using "DSA" algorithm for digital signature as shown below:

SN = 3119180841198366788, 0x2b498e72edd53044

SR = 514496251439242438, 0x723db9d614e5cc6

g = 6584293170253087411, 0x5b601bc0d5f742b3

DS_{pub} =Sun DSA Public Key

Parameters: DSA

p: fd7f5381 1d751229 52df4a9c 2eece4e7 f611b752 3cef4400 c31e3f80 b6512669455d4022 51fb593d 8d58fabf c5f5ba30 f6cb9b55 6cd7813b 801d346f f26660b7 6b9950a5 a49f9fe8 047b1022 c24fbba9 d7feb7c6 1bf83b57 e7c6a8a6 150f04fb 83f6d3c5 1ec30235 54135a16 9132f675 f3ae2b61 d72aeff2 2203199d d14801c7

q: 9760508f 15230bcc b292b982 a2eb840b f0581cf5

g: f7e1a085 d69b3dde cbbcab5c 36b857b9 7994afbb fa3aea82 f9574c0b 3d07 82675159578e bad4594f e6710710 8180b449 167123e8 4c281613 b7cf0932 8cc8a6e13c167a8b 547c8d28 e0a3ae1e 2bb3a675 916ea37f 0bfa2135 62f1fb62 7a01243b cca4f1be a8519089 a883dfe1 5ae59f06 928b665e 807b5525 64014c3b fecf492a

Y: fad65a46 00575517 64c3071c 5f3974f7 80e31037 fadd5752 b5c392e8 8c2a 91d81bbb166a 0ded3d2e a29b95d3 7d6a61c6 3f6aac68 5493fbdb aeaae6c4 c97601829628d1db dc5a5f2a d8a78472 fd50bbc7 d8c8d3d8 a52e5e61 37b4b893

8befb71ed5974bcf b3fbd8e4 dae84cd0 6b9e3f05 8d1d8d5d 35b7bd16 251abc0a 84017a05

DS_{priv} = Sun DSA Private Key
Parameters: DSA

p: fd7f5381 1d751229 52df4a9c 2eece4e7 f611b752 3cef4400 c31e3f80 d14801c7 b6512669455d4022 51fb593d 8d58fabf c5f5ba30 f6cb9b55 6cd7813b 801d346f f26660b7 6b9950a5 a49f9fe8 047b1022 c24fbba9 d7feb7c6 1bf83b57 e7c6a8a6 150f04fb83f6d3c5 1ec30235 54135a16 9132f675 f3ae2b61 d72aeff2 2203199d

q: 9760508f 15230bcc b292b982 a2eb840b f0581cf5

g: f7e1a085 d69b3dde cbbcab5c 36b857b9 7994afbb fa3aea82 f9574c0b ecf492a2 3d0782675159578e bad4594f e6710710 8180b449 167123e8 4c281613 b7cf093 8cc8a6e13c167a8b 547c8d28 e0a3ae1e 2bb3a675 916ea37f 0bfa2135 62f1fb62 7a01243bcca4f1be a8519089 a883dfe1 5ae59f06 928b665e 807b5525 64014c3b

x: 4782b1f7 c312bd24 efb76dad 0e0ae3b8 214b161c

Step 11: Now, with the help of hash function some key credentials are generated as follows:

SN_PWD_HASH = H (H (SN) || H (PWD))
=0x62EC74CC56573B5A73F285496B9F334E696B60D357 1DFD9E705EB2DFF390DC

KEY_USER = H (H (H (g) || H (SR)) || H (ID))
=0x9E6B3EEFFCDC5B80A892BAF4670BB62AC34DDFFE1079 6FBC411F29D6B953463F

Step 12: Data owner sends a set of credentials i.e. [SN_PWD_HASH, $DS_{pub,}$ H(ID), g, mobile number] to Service provider.

Step 13: Now, the data owner encrypts KEY_USER using ST_{pub} as key. The value of ENC_KEY_USER is as shown below:

ENC_KEY_USER = Enc (KEY_USER, ST_{pub})
=0x3C1BFB72B52DB3373C8B9142B259A44475CA7E129 87EE270FE2A8B8C67CC6CDCA38673AD7A2685B303F4

57984F6AE581C2B0E7DFF66EC24AB58F60617C9DCA29
9F7E3E20E429240BCD1F3979DEBF23BB3C9651593C6F
24274FE097C8D3AA9550C1647C08E582FF7A88B26ED1
5CAACB6DCF8BEE94D8651D2417ADC5C696C4C682EB
6DB3379371C120A0B6896BC1D83C13153D119E497A3F
17A1DAAE696EE374AAA59E1C4B8AFB6929F0ABEA15
4BF5E97FF3491A809330D0E0D14ADAB197D1C230934C
CD0B23A397E269DB3B6A1629C2683F35ECF2CED76D9
00F8EEDBBD0BDE4A8B7A84546E64694FF93018E4A3D
F423F1C324D004340E6266120C77BAF0B00289

Step 14: After encrypting KEY_USER, data owner selects an image from database and embed ENC_ KEY_USER into the selected image IMG using steganography algorithm and generate STEGO_IMG as shown in Fig. [13].

Step 15: Now, Data owner sends a set of credentials to user i.e. [DS_{priv}, g, SN, STEGO_IMG].

Figure 13: STEGO_IMG

<u>AT USER END:</u>

Step 16: After receiving credentials from data owner, user executes extracting algorithm of steganography on STEGO_IMG and extracts ENC_KEY_USER.

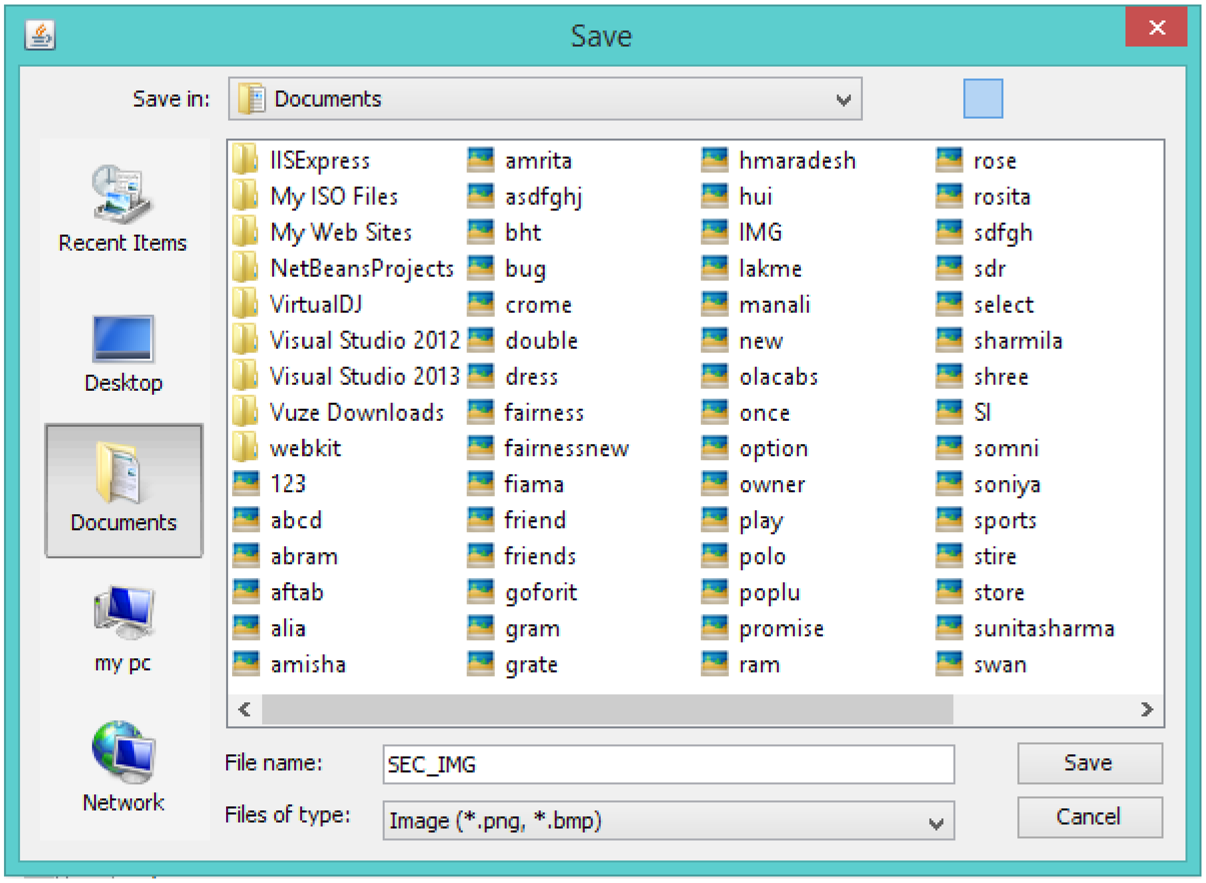
Step 17: Now, user decrypts ENC_KEY_USER and get KEY_USER.

KEY_USER = Dec (ENC_KEY_USER, ST_{priv})

= 0x9E6B3EEFFCDC5B80A892BAF4670BB62AC34DDFFE10796B FC411F29D6B953463F

Step 18: Using KEY_USER, user encrypts [SN, g, DS_{priv}] and stores them in database of local system. It is assumed that user now saves the STEGO_IMG secretly.

D1 = E (SN, KEY_USER)



Save in: Documents

File name: SEC_IMG

Files of type: Image (*.png, *.bmp)

Figure 14: Save Dialog Box for STEGO_IMG

= 0xABA5D0A8ABC364E4E3B003ADA5E6A42852C71064A9922C976B 543277C671F197

D2 = E (g, KEY_USER)

= 0xF7BBD4A6D8C7F00FF287072AD2B062437209F569127DB6856F2A8
2BD7C46E722

D3 = E (DS_{priv}, KEY_USER)
=0x7AAC4690EF5303A061BB3330658FA28935F6B18CDBF4D84C4DDED
455F3199E5533FAA44A7B4E5B2E2245E5AFD38277B619E8ED8112FE86
A7FE5DDC4AA254079F73F2CF067DDEE6F951D829A46A72B43178D715
577D1AF2BD66131BEF1084D1A5F112E961C24744210DCF63F449E5BFB
1B3C5B480F77A0C0D181848A91A57A191B7B4C818D7BCCAD421B0861
49A5BBED12779835538B161D6A064C734056FBA070E817C456E8E87397
2A4128D0219864EB83DD153B5D9A2799D98A15551F56C133756E17F5A4
221D06B39041D51A7CE85897DDCFECFB42BC952E6A9E37486A0683FE
3E6D959161BA3C1BE2D9BD79B65606EECD31EE7D6FB7CA399965F414
FCF4FE0D15BA4A5D3F5A439DDB24C332CDA989E66A16A79F00E8080
457670018C7CCFC40496E233A11AB7C53AB164BB93B2D8276BEDF153
EDD0CC0F6931D32F05AF131C8558A00356D97E5685747BB3E53E57E7F
66F52AB87F295FB812B96562DE9C0DAF62EAAD1765EA338F30483CDD
31AFDA936B96FBC77CB6D6F1EC3460B72007DED29760FDD833C87E4E
E9B4438FF88477BA0A6261F25ED3FD23EBBB9A9197E3316C5E91451D2
2C5376F954991856DB45E1FB676C3332F1CEF753C62A25C10BBA342864
43886319B8053E1A1B8A4835C5

Now, calculate H (H (PWD) || H (ID)) and use it as key to encrypt ST_{pri} as follows:

H (H (PWD) || H (ID)) =0x0D2A6541BBC2E02CC9A1FD1DEE00FF3ED61FE8
F1837D43D26737C873198BFE0E

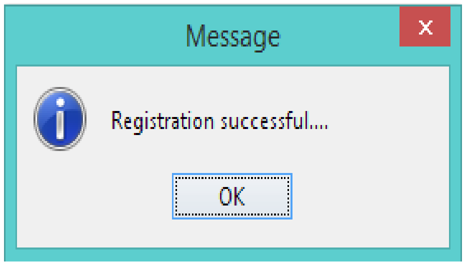
ST_{priv} = E (D4, H (H (PWD) || H (ID)))
=A8BA179933AE70B9F926BB7F6C730FDD3919A01AA3EC73AD0369
8BE0FC1AE74624A05A354896663ED981019826E99987E883421056FA28
456FCE1B6B6C133AB243BBF96DED3021558BFC21341E13B42468F86D
4A6CBA36BBE8CC2E6A9CBAAADD391BEDD736446B9A62F0B6126B3
0398E89F227668DFBDA51A195D00264F065E3BA45855FE7B6F81917CB

```
47A3C0096631E51D6F895F840AAD666F69523309A3ECFFD793E9EBD0
FAC443F1E6D88C3FBF42499E425A532446C5B7CA67FEFBE068FF99897
89BBE13090C781A366BCBF6146B4F97017B273116FFCF42C560BA8659
B10662D396496CC2EBF8A51ABED58ED1B31387307E29AB01E7E9F026
B0C17FD3256E1942496EF30893BAC287F04B531BF779849F2989E32FB1
6AF3538F6FB957334E134C6016AAA9F5E836E33514F45AA84B44FBED
92D40F0C8C553B1E7FD540F9AFA0147D94B753BDEA1595F087B66B5C
7CE15E297FE2958F19C1BB8F9740855EF6BAC56F5BB0ED75FB527A11
00AA3FA17121A154AE52DD20F8E7D1F4750518AC76874D9B0FD99F5E
66F8B22EE551833459C3CAA0E42AF235B0E798A60B8F953445EF73B46
65571349D57D38BD5765AF184106BBBF808032D501DB54EC32066F1DE
3549DD9D883DDDAE29E0E49B8EC4180E53052435D20A157DD6BDAF6
03B4C9FF847E6AC112A97EC842DF74FA790B26CE1F8C5F1F49BC3DD
D647634E279053FAE736A53A43A895AF88C1B5FECC91E60E21758AB5
B682F3582E01678DADB7863E4688599C2194FABF15A185979BAED49A
01BBAACD7F204330262535F81822EDA21D7D0772113B354B3FADECE
73A8E317E5E4BAD5F0A8905677402B186AC60DE73F8F1BEB19AF81FF
79A56F55684F9C5E89E46095457BF8ACC26A0CEDC26EDA4AE0435996
075D739F564FEBC39374D121F02DCBB382D978FA48DDA7B65F9A67E
1A10081DB9AC20D4655FBCBA87D3DAFBAB2644DA5501F1B9E36253
D08896C03BC1EEDE60DEF3D4C1912B8BE387FCF1A1CE2A1A376AA2
F29D1AAA0E0DB19065E00C97484750B0A45E77A6383122617D22B412
AAA8EB840B0089FF2DE9224FC64C26EDF8D6BFAD4F36A00F0FC37C
D5722692C9CF242E0C123CD909BDD4544FE2A09D7C8E7C84FFBDA59
BC20CF3643251852CAB323A82251F556E18B46E16DC29E69343F4E1C6
39728E4DE6FB5B172E56985E95BE75168538282A42509B763DB681EAD
6CC9F49002E5E26B529902A3D4E10A84F6B71DCDAD5A69BF7375B685
BAB208A0AD562DB425B687E27C31C22A8B462A151D68E8DB8E5E72
A4DF847C3C261C64CAE3D86E42CF180FA8D2F1B98FA0A42994D5D23
5AE2E5E0C5BCC66CEA616170DF878C9EDFC889AC11AF6549E5603E9
```

```
568E05E2B42A6CDA00D9F8B72AA960ACE311ECCB25760AD5D75F22
A1A44F1C6BEBDB8727D03B3782613568F727E87B5AED8C1BC0FA63D
523EE802478B415D761B9A584F0BAF12AD65E589309CBAA12C0AAFB
A17CACD971AC63D146A4EE47D3A6478063BCEAADD57B6E31E59867
5C45EBF7D1441B6D861B383747396D56E57D4B660F572454A695FCEC6
27A7FBF85F462E2668E0796FFA4CA22073851A5111DEC3E36764129747
171C2A81195DBF188639CAB27AD8500881FC05C6EB47115E05F1D0E4
F1E847BA25A1533BB4B8C14DD890CB7517E32CCDA9E5C262D0D45B
12775FAF32FE71D0703B3B971BD8AAC7C01BBC4492B4A5A14546D54
E891BBA7189E858A4C0E1401E260DCF35D3E721C724BB23AA636A370
BAE9DDBC4BA8DFA4074FD0672E9F670C31185F37A52E755D5BCFB44
29E016DD92D9BFC0CFE53D241EBB7C8BBC92941CC60F2AC0CA82D7
98B9EF725FA62C4EA3F137FF6C534FDD1FD4A6249AC4ACB29203A4C
B42453C8E006A35B69DE09C11F4BB51CA7A4BC8720E7723A87350E5E
1F9689D87DC49505FEEFEE30BC445603AF63AD6392F5029F26A5EED9
45CA516023793C160CDB6C307059FA57E0267252E26ECEA16C2F1088D
761F12AB275058C23507922FFAD56EF087B7880A93528A71AE69F85031
535AF06348C633BDFACD5DDBC4DEE87FCDECFF545C70A04C07676F
A17F12DBC2637BF6BA578BD4E4D138959B744768E7D4D640B3CF1BC
9A6577FD0CE145B818A5B0492F3D0C7E7817F889880576DA6525F75AF
7768EC258B3BF8CB8FCAFF58B2716B283BFF3DF4C4F927BED94535D1
059915AAD3488548BA053BCA12BEAC7EE05C0C05463C63B37D22C65
0901FC20D00CC54C9F8792D008FFA0A9BD4B596AAA6D424A65CD5D
8A4C31B38CE6EDB8AD14E6D3738C4F8D9F8E3FC
```

AT SERVICE PROVIDER END:

After receiving credentials i.e. [SN_PWD_HASH, $DS_{pub,}$ H (ID), g, Mobile Number] from data owner, service Provider stores them in its database.



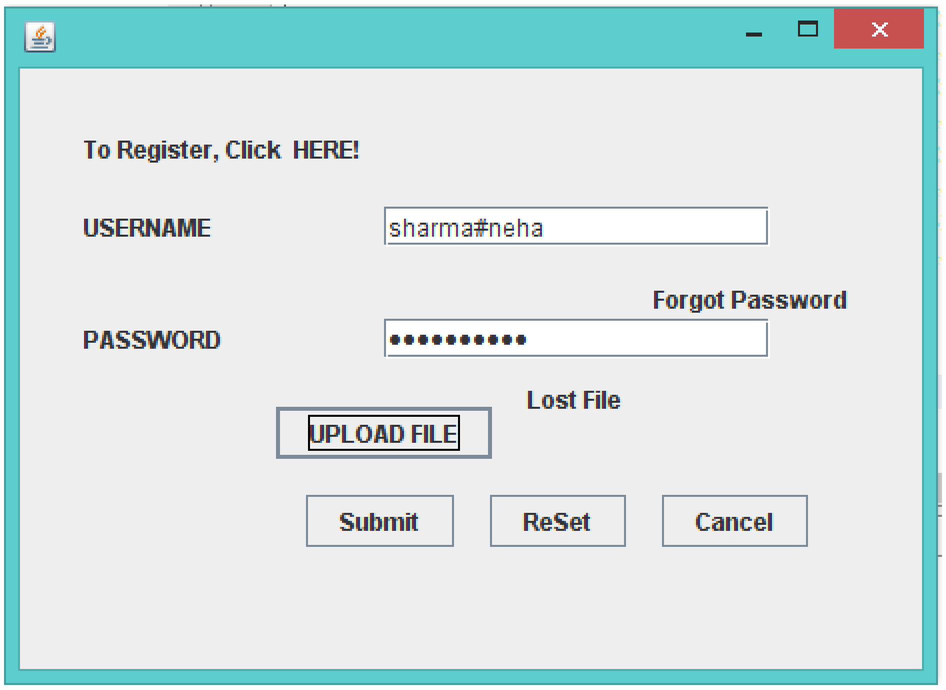
Registration successful....

Figure 15: Message Box on Successful Registration

After successful completion of above steps, a message box appears as shown in Fig. [15]

6.1.2 LOG-IN PHASE

The registered users can access the cloud services through log–in page providing correct credentials. The complete login phase is described with the help of example.



To Register, Click HERE!

USERNAME sharma#neha

Forgot Password

PASSWORD ••••••••••

Lost File

Figure 16: UI for Log-In

<u>PART 1</u>: <u>MUTUAL AUTHENTICATION</u>

Step 1: Registered User enters his username “cloud#user”.

Step 2: User enters password “secret.123” that was chosen during registration phase.

Step 3: User needs to upload image file “SEC_IMG”.

Step 4: After providing password and image file, user needs to submit by clicking on “Submit” button.

Step 5: Now, steganography extraction algorithm is executed on STEGO_IMG at client side.

ENC_KEY_USER is extracted from image.

ENC_KEY_USER=

```
0x3C1BFB72B52DB3373C8B9142B259A44475CA7E1298
7EE270FE2A8B8C67CC6CDCA38673AD7A2685B303F45
7984F6AE581C2B0E7DFF66EC24AB58F60617C9DCA299
F7E3E20E429240BCD1F3979DEBF23BB3C9651593C6F2
4274FE097C8D3AA9550C1647C08E582FF7A88B26ED15
CAACB6DCF8BEE94D8651D2417ADC5C696C4C682EB6
DB3379371C120A0B6896BC1D83C13153D119E497A3F1
7A1DAAE696EE374AAA59E1C4B8AFB6929F0ABEA154
BF5E97FF3491A809330D0E0D14ADAB197D1C230934C
CD0B23A397E269DB3B6A1629C2683F35ECF2CED76D9
00F8EEDBBD0BDE4A8B7A84546E64694FF93018E4A3D
F423F1C324D004340E6266120C77BAF0B00289
```

Now, KEY_USER is extracted from ENC_KEY_USER by applying decryption algorithm using ST_{priv}. To get ST_{priv} , D4 is decrypted using H (H (PWD) || H (ID)).

H (H (PWD) || H (ID)) =0x0D2A6541BBC2E02CC9A1FD1DEE00FF3ED61F
E8F1837D43D26737C873198BFE0E

ST_{priv} = D (D4, H (H (PWD) || H (ID)))
= Sun RSA private CRT key, 2048 bits

modulus:

```
1618178674112717986279650628589018042599845798508460914829785258067314
3706305968968917136837982083921979972818029568444530172926428208676094
```

2548027376900222551584941494367009847820040375686101632567632085310868
8296305528139470626917746120501298901688076223145593430497651737238445
9959606315830486096489387337755309520017017471996752801785765759774096
1312362003790781854398609337357561899033153099569725061543466746232621
2980962087455870380693682681305983452653074624326866134656919834986485
2491376247572990893115739069760593158503087546461248098666461660503169
3831756835807396192976391895971256977836884190190797537 21

public-exponent: 65537

private-exponent:

3513292392686553279334291895294740029014633850645725369960940298921192
0258331573379727778211948528636625575358613108533686288748363059226413
3469014685736800080336623971702730232484147047566647867586987036618757
7363646697269714443812290698172479984149356329880624520889129281042136
2095493884058163581938064364433889466190990577775182740023182057469535
0164055500926551853172904429320653027201099115389978989565440682430364
8283983647051116977599282158122681907062581498227558596354073342671534
3676700765860207397838465553373051336837142226271546506886802688948487
60157900636001371241418810038814074981949101463480152321

prime-p:

1626410598066242202717982677624270572593581658286664599818923596231900
3423497633848539091835479408414495128808402474103731063693591653286739
4322811402818352079471477603421752370062212375205378620457703516733081
1718074564789925164273665485726222928837460631009941022137582402403532
655620507454869083630673 27769

prime-q:

9949385942496244495250284217641142371523753410406539409564032783586192
1784311854464866873804265560151624238657745409616926997946584555048406
2353572200536406569211202299520770390011538659780367634797618659290543
6002988318724219369149386703922985306733229509272771159762526789398331
9141267946768643791714250209

prime-exponent-p:
98043248085513573558111206867726715460662878334271111501970206930945400041521738934378122456671038990411378890330125641409148014514769733008033802433604599472110100640529520284481252182846924533598158182943586643492945711820152684681481216085400994219383171810946429634356122123937051740830400404179313989321

prime-exponent-q:
73128449707805376708760881755920140994598349295708226400155554753093195787880159750562847111577765727813900180716181412749645102567679580721246885839906981864349762141641996902035271399730870056560723394140753292378953463378800831885188142266231628750120613038342732465769785818338999876054684309190943672O1

crt-coefficient:
14187802762884952095180015120465285298079082406442844244623114229792491893716169351859017069158732497656459217601276427622182727152943625044370133888865823072121200320801172930957689191865487199242417332981114866734562499

KEY_USER = Dec (ENC_KEY_USER, ST_{priv})

=0x9E6B3EEFFCDC5B80A892BAF4670BB62AC34DDFFE107 96FBC411F29D6B953463F

Step 6: Using KEY_USER, credentials stored in the database i.e. SN, g and DS_{pri} are decrypted.

SN = 3119180841198366788,   0x2b498e72edd53044

g = 6584293170253087411,   0x5b601bc0d5f742b3

DS_{priv} = Sun DSA Private Key
Parameters: DSA

```
p:  fd7f5381  1d751229  52df4a9c  2eece4e7  f611b752  3cef4400  c31e3f80
    d14801c7  b6512669  455d4022  51fb593d  8d58fabf  c5f5ba30  f6cb9b55
    6cd7813b  801d346f  f26660b7  6b9950a5  a49f9fe8  047b1022  c24fbba9
```

d7feb7c6 1bf83b57 e7c6a8a6 150f04fb 83f6d3c5 1ec30235 54135a16 9132f675 f3ae2b61 d72aeff2 2203199d

q: 9760508f 15230bcc b292b982 a2eb840b f0581cf5

g: f7e1a085 d69b3dde cbbcab5c 36b857b9 7994afbb fa3aea82 f9574c0b ecf492a2 3d0782675159578e bad4594f e6710710 8180b449 167123e8 4c281613 b7cf093 8cc8a6e13c167a8b 547c8d28 e0a3ae1e 2bb3a675 916ea37f 0bfa2135 62f1fb62 7a01243bcca4f1be a8519089 a883dfe1 5ae59f06 928b665e 807b5525 64014c3b

x: 4782b1f7 c312bd24 efb76dad 0e0ae3b8 214b161c

Step 7: Random numbers “r” and “k” are generated. Hash of r, g, k and ID are calculated.

r = 2952809980401868331

k = 4255457457498861758

H (r) = 0xE798531E233759A943D55FB27B4260405CB9B9F442E517BFFAE5 84EC20361CA5

H (k) = 0x38903EDE6B1B453F76741D37364A8C0CCB563C571AC16AE44 B04F348ACFE33CF

H (g) = 0xB55D44D388824D55DB499C842EB2B21E0E5CBD696F9C C9E1274509029254505

H (PWD) =0x5FF2D08FF8AE00A00D6A2A41D023416A54D9F4F1D6979A57 DA44BD0A6C14DC0E

Step 8: The data to be sent to service provider i.e. K and E1 is generated as follows:

E1 =E (H (r), g)

=0x6F7ED00750DBCDF25383E8BE674649775F8D3020EBC6BB45F

0BF9F3B620899D90FF936EFA8C936ADC73E9938A7128075

K = E (H (k), g)

=0xA48B4B3B5BCB0B55AB78032C5B69F51FC96A93B1A8AF

5A8FAD482B03523A12D0401545E8B497D1F3C39F7343DC37D4B1

Step 9: H (ID), K and E1 are sent to Service provider.

<u>AT SERVICE PROVIDER END:</u>

After receiving credentials from user, Service provider performs following steps:

Step 10: Using H (ID), service provider extracts credentials related to user i.e. g, SN_PWD_HASH stored in database i.e.:

g=6584293170253087411, 0x5b601bc0d5f742b3

SN_PWD_HASH =0x62EC74CC56573B5A73F285496B9F334E696B60D
3571DFD9E705EB2DFF390DC

DS_{pub} =Sun DSA Public Key

Parameters: DSA

p: fd7f5381 1d751229 52df4a9c 2eece4e7 f611b752 3cef4400 c31e3f80 b6512669455d4022 51fb593d 8d58fabf c5f5ba30 f6cb9b55 6cd7813b 801d346f f26660b7 6b9950a5 a49f9fe8 047b1022 c24fbba9 d7feb7c6 1bf83b57 e7c6a8a6 150f04fb 83f6d3c5 1ec30235 54135a16 9132f675 f3ae2b61 d72aeff2 2203199d d14801c7

q: 9760508f 15230bcc b292b982 a2eb840b f0581cf5

g: f7e1a085 d69b3dde cbbcab5c 36b857b9 7994afbb fa3aea82 f9574c0b 3d07 82675159578e bad4594f e6710710 8180b449 167123e8 4c281613 b7cf0932 8cc8a6e13c167a8b 547c8d28 e0a3ae1e 2bb3a675 916ea37f 0bfa2135 62f1fb62 7a01243b cca4f1be a8519089 a883dfe1 5ae59f06 928b665e 807b5525 64014c3b fecf492a

Y: fad65a46 00575517 64c3071c 5f3974f7 80e31037 fadd5752 b5c392e8 8c2a 91d81bbb166a 0ded3d2e a29b95d3 7d6a61c6 3f6aac68 5493fbdb aeaae6c4 c97601829628d1db dc5a5f2a d8a78472 fd50bbc7 d8c8d3d8 a52e5e61 37b4b893 8befb71ed5974bcf b3fbd8e4 dae84cd0 6b9e3f05 8d1d8d5d 35b7bd16 251abc0a 84017a05

Step 11: Now, service provider decrypts E1 and K to get R1, R2 respectively as follows:

R1 = D (E1, H (g))

= 0xE798531E233759A943D55FB27B4260405CB9B9F442E517BFFA

E584EC20361CA5

R2 = D (K, H (g))

= 0x38903EDE6B1B453F76741D37364A8C0CCB563C571AC16AE44B0
4F348ACFE33C

Step 12: After decrypting K and E1, Service provider use R2 as a key to encrypt R1 as follows:

R = E (R1, R2)

= 0xB0431C55EF4AA53EE4DE9C03381FF97859B1020095205D3A220
8F2D332CC66C48E5ADCB0D1ACD200B202D105DA740452

R is sent to user.

<u>AT USER END:</u>

After receiving R from service provider, following steps are performed at client side:

Step 13: Decryption of R using already computed H (k) is done to get R' as follows:

R' = D (R, H (k))

= 0xE798531E233759A943D55FB27B4260405CB9B9F442E517B
FFA E584EC20361CA5

Step 14: R' is compared with H(r). If both are equal then next step is executed else log in process is terminated and "Unsuccessful log-in" message on the screen is displayed as shown in Fig. [20].

Step 15: Now, Key_Session, X, E2 and E3 are calculated as shown below:

Key_Session = H (H (ID) || g) EXOR H(r)

=0x324D5102876C66085536A2D8A6B541C1406D617086F
87249B0B07C8A88CFA2FD

X = H (H (H (SN) || H (PWD)) || g)

= 0x4E140031799751EB435313BAB0DC65F6BF47DDE6FBE0060E45551

07CF54F0C81

E2 = E (H (H (ID) || g), g)

=0xF6CBAAD1707DA80103590A566852A7904EA5D0842C1F04879335B F425F557EB95747A509AA269B0C571CFFB61A58B0C7

E3= E(X, Key_Session)

=0x52EF83A488CA4165A6A0CAA5E079017D32FDA84EA98E3176C2 47EC0315A9AF2E1DEECDE851A30D1DB9662259104DBE51

Step 16: After completing step 3, E2 and E3 are sent to service provider.

<u>AT SERVICE PROVIDER END:</u>

Step 17: Service provider decrypts E2 using H (g) as key to get E2'

E2'= D (E2, H (g))

=0x31645543484B526250364557342F31713366636867527A5532495445 48575832536C58345A716A35766C673D

Step 18: Key_Session is computed by applying EXOR operation on E2' and R1.

Key_ Session = E2' EXOR R1

=0x324D5102876C66085536A2D8A6B541C1406D617086F 87249B0B07C8A88CFA2FD

Step 19: Now, X is calculated using SN_PWD_HASH and g as shown below:

X = H (SN_PWD_HASH || g)

=0x4E140031799751EB435313BAB0DC65F6BF47DDE6FBE0060E45551 07CF54F0C81

Step 20: After computing X, E3 is decrypted using Key_Session as key to get X' as shown below:

X' = D (E3, Key_Session)

=0x4E140031799751EB435313BAB0DC65F6BF47DDE6FBE0060E45551 07CF54F0C81

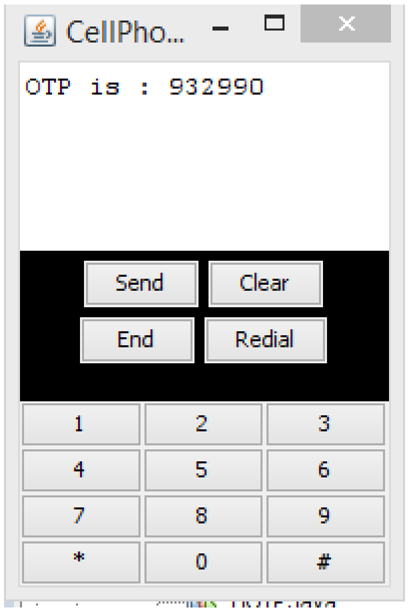

OTP is : 932990

Figure 17: Mobile Phone Receiving OTP

Step 21: Now, X and X' are compared to see if both are equal. If both are equal then only next step is executed else the process is terminated with "unsuccessful login" message displayed on screen as shown in Fig. [20].

Step 22: Now, one time password (**OTP**) is generated and sent to the user's registered mobile phone as shown in Fig. [17].

<u>AT USER END:</u>

Step 23: After entering **OTP** in input box as shown in Fig. [18], data to sign i.e. DTS is computed as follows:

DTS = H (H (H (SN) || H (PWD)) || OTP)

=0x5594B5A3C575060B41399A40E934DD94F74343C1FFB62
A877CCFC6F374B78085

Step 24: After calculating DTS, it is digitally signed using private key of digital signature algorithm DS_{priv} to get signature i.e. SIGN as shown below:

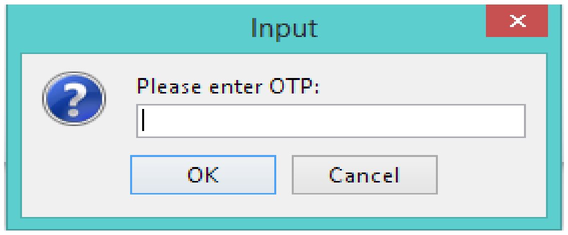


Figure 18: OTP Input Box

SIGN = DS (DTS, DS_{priv})

=0x302C02146C6DBCA340171F080916A360141321F90FB595220
21404903C5D72BD14F541F292870B67E2B3733C35DE

Step 25: Now, SIGN is sent to service provider to get verified.

<u>AT SERVICE PROVIDER END:</u>

Service provider receives signature SIGN from user which needed to be verified.

Step 26: After receiving SIGN, Service Provider first computes SIGN'

SIGN'= H (SN_PWD_HASH || OTP)

=0x5594B5A3C575060B41399A40E934DD94F74343C1FFB62
A877CCFC6F374B78085

Step 27: Service provider checks verify SIGN' using public key of digital signature algorithm DS_{pub}. If verified successfully, next step is executed else the process is terminated with "Unsuccessful login" message displayed on screen.

<u>PART 2: UPDATING DATABASE</u>

After successful completion of mutual authentication part of login phase, next part of updating database is started. SN, SN_PWD_HASH is updated using "α".

AT SERVICE PROVIDER END:

Step 1: A random number **“α”** is generated and sent to user.

α = 4910281114035795437, 0x4424d2a84d69b5ed

AT USER END:

After receiving “α”, following steps are executed:

Step 2: Now, a new random number is generated, replace it with SN.

SN = 7119595438394103241, 0x62cde25479f8d5c9

Step 3: Now, new SN_PWD_HASH is calculated as follows:

SN_PWD_HASH = H (H (SN) || H (PWD))

= 0xFFC2F25EF49EA84650EC2EA21E5C67BD3FB4FBCB
541EF8E845DFD7522797D7B

Step 4: After this, E4 is calculated as shown below:

E4 = E (SN_PWD_HASH, Key_Session)

=0x72199806063261662E57E34B85529AAFEC049BEE4403BA152836
745550C91A38279E30A04FD4F761A5CB695FCF9F759B

And E4 is sent to service provider.

AT SERVICE PROVIDER END:

After receiving E4, following steps are performed:

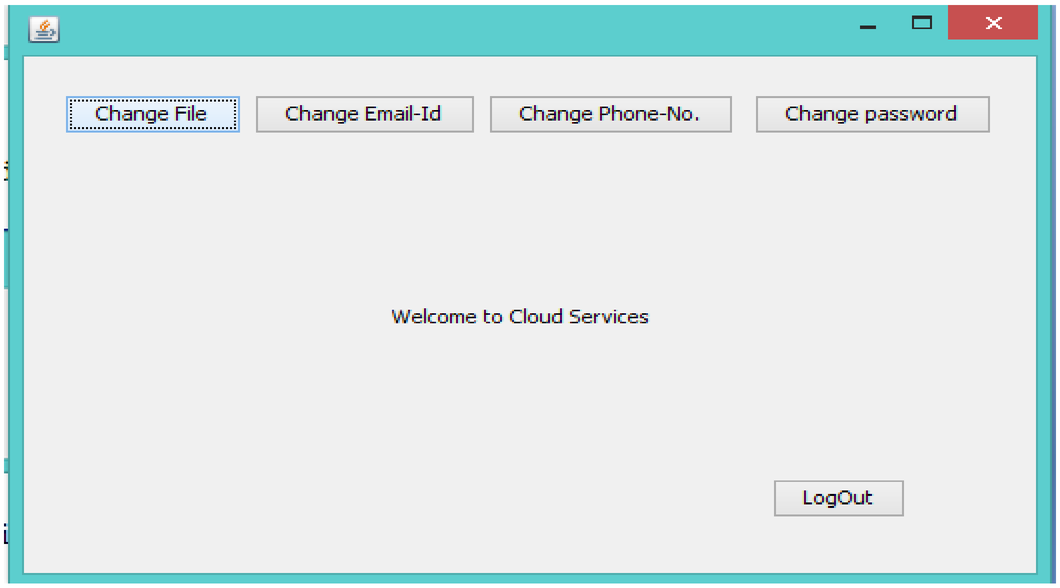
Step 5: Decryption of E4 is performed now using Key_Session as key.

E4’= D (E4, Key_Session)

= 0xFFC2F25EF49EA84650EC2EA21E5C67BD3FB4FBCB
541EF8E845DFD7522797D7B

Step 6: Now, update SN_PWD_HASH with E4’ i.e. new SN_PWD_HASH.

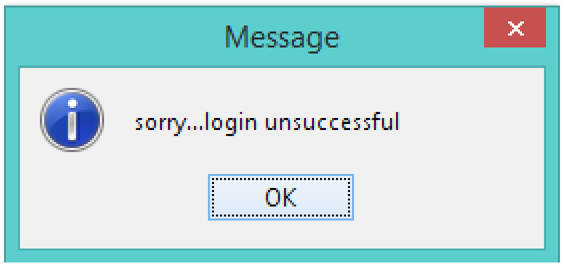
Step 7: After completion of step 2, user is allowed to access Cloud services. User is navigated to service provider's home page where user can access services and facilities to change image file, password, phone number, email-id are provided. The homepage of cloud services having tabs for changing file, password, email-id, phone number is shown below in Fig. [19].



Welcome to Cloud Services

Figure 19: Cloud Services Home Page

During the entire Log-in phase, if user or service provider cannot prove their authenticity,



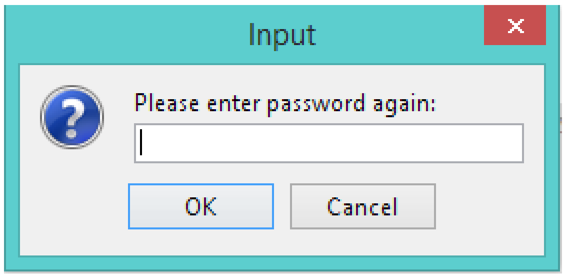
sorry...login unsuccessful

Figure 20: Message Box on Unsuccessful Log-In

process is stopped and message of "unsuccessful Log-in" is displayed as shown in Fig. [20].

6.1.3 CHANGE CREDENTIALS

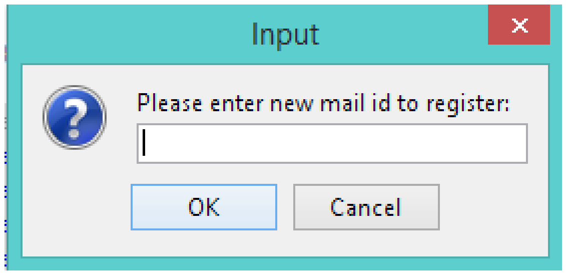
After successful log-in, user is navigated to home page of cloud services where he/she can access these services. The flexibility of resetting credentials is provided by proposed scheme. User can reset his/her password, email-id, phone number, STEGO-IMG. For security purpose, user is asked to enter his/her password again as shown in Fig. [21(1)].



Please enter password again:

Figure 21(1): Input Box for Re-entering Password to Change Mail-id

If password is correct, then user is asked to input new credentials such as password, email-id, mobile number. For example, if user wants to change his/her registered email-id, first user inputs correct password in the input box appeared as shown above in Fig. [21(1)], and then user needs to enter new email-id in the box as shown in Fig. [21(2)]:



Please enter new mail id to register:

Figure 21(2): Input Box for Entering New Mail-id to Change Mail-id

6.1.4 LOST CREDENTIALS PHASE

This phase consists of three possible situations i.e. password is missing or image file is lost or both are missing. In every case, by clicking on the links corresponding to situation, user is asked to enter his/her username as shown in Fig. [22]. Taking the case of forget password, the proposed scheme provides the flexibility to again get the password. When user clicks on the link "Forgot password", an input box appears as shown above to get username. Then user receives a mail containing new password. User needs to enter new password twice and upload STEGO-IMG as shown in Fig. [23]. If everything is fine, password is set and user can use it for successful login.

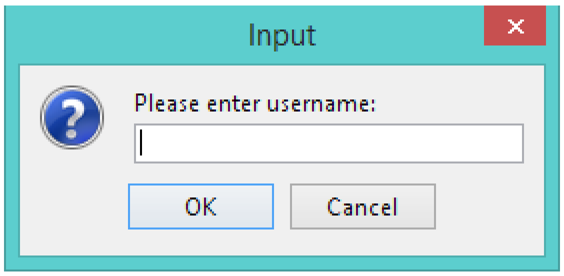


Figure 22: Input Box to Enter username

In the case of lost image file, user needs to click the corresponding link and have to enter username in the input box shown in Fig [22]. After this, a mail is sent to user containing new STEGO_IMG file. To confirm resetting, user needs to enter his/her password and upload new STEGO_IMG in the same box as shown in Fig [23]. In case of losing both credentials, user

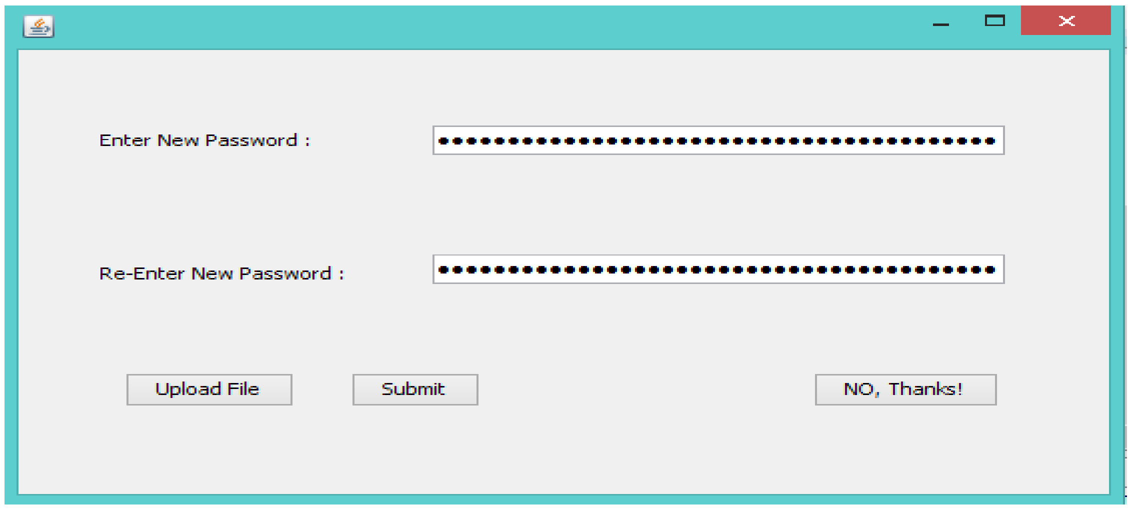


Figure 23: UI for Setting New Password

needs to enter username then after verifying user is registered or not, user is sent OTP on his/her mobile number. User needs to enter the OTP. After entering correct credentials, new credentials are sent to user's mail id.

CHAPTER 7: SECURITY ANALYSIS

This chapter provides security analysis of the proposed scheme. We will analyze the security facilities given by the proposed scheme and resistivity of the proposed scheme to various known attacks. We will see that all the common attacks such as MITM, replay attack etc. are resisted using the proposed scheme.

Proposition 1: Our proposed scheme ensures mutual authentication.

Proof: Mutual authentication is a security feature which implies that the adversary cannot impersonate legitimate parties which are communicating to each other. In our proposed scheme, the legitimate user proves his/her authenticity to the service provider with the factors i.e. password, STEGO_IMG, OTP and digital signature. Only a legitimate user can provide these factors. Also, at the time of registration data owner gives some credentials such as g, SN to user and SN_PWD_HASH, g to service provider which is used in the scheme in combination with other data as part of challenge and response. For example, E1 = E (H (r), g), K = E (H (k), g) are sent by user to service provider. Only legitimate service provider can decrypt E1 and K because these are encrypted using g which is shared credential between service provider and user. Therefore, we can claim that our scheme provides strong mutual authentication.

Proposition 2: Our proposed scheme ensures user's privacy.

Proof: User's privacy is an important feature which must be provided by mutual authentication scheme. Our scheme sends all the data in encrypted form using symmetric encryption techniques or using crypto one way hash function. It is never sent in plaintext over the communication channels. So, it is not easy to decrypt or decode data. Therefore, we can claim that our scheme provides user privacy.

Proposition 3: Our proposed scheme ensures security of the stored data.

Proof: In our scheme, the secret data are stored in client side database in encrypted form using key KEY_USER which is again encrypted by asymmetric encryption scheme using key ST_{pub}. So, to decrypt and access secret data stored in database adversary needs to know STEGO_IMG as well as ST_{priv} to decrypt KEY_USER embedded in STEGO_IMG. Also, use of advanced steganography algorithm makes it difficult to extract KEY_USER. So, if we consider the case of stolen STEGO_IMG, still it is very difficult for adversary to access credentials stored in database because he/she doesn't possess $ST_{priv.}$ Therefore, we can claim that our scheme ensures security of stored data.

Proposition 4: Our proposed scheme ensures identity management.

Proof: In our scheme, both service provider and data owner stores registered user's credentials in the database using H (ID) as primary key. Also, the data owner checks the availability of unique ID in every new registration request.

Proposition 5: Our proposed scheme can forward secrecy.

Proof: To provide confidentiality for entire session, session key is used. In our scheme, we are using session key [H (H (ID) || g) EXOR H(r)] that is agreed between client and service provider. Every key is different because of the use of random number 'r'. Also, we are using OTP in every session which is valid for that session only. So, if any attacker succeeds in getting session key or OTP, he/she can't derive past or future session key or OTP. Therefore we can claim that our scheme can forward secrecy.

Proposition 6: Our proposed scheme resists impersonation attack.

Proof: In our proposed scheme, the legitimate user proves his/her authenticity to the service provider with the factors i.e. password, STEGO_IMG, OTP and digital signature. Only a legitimate user can provide all these factors at same time. Use of private key of public cryptosystem ST_{priv} to encrypt KEY_USER which is further encrypted by hash of password and username makes scheme stronger. OTP which is a high entropy factor is set over different communication channel makes the impersonation attack very difficult. Therefore, we can claim that our scheme can resist impersonation attack.

Proposition 7: Our proposed scheme resists replay attack.

Proof: In our proposed scheme, secret data such as SN, SN_PWD_HASH are updated after every login session. During authentication, random numbers such as r, k which are used with other key credentials are generated newly for every new session. If somehow, attacker obtains data, next session cannot be attacked. Also, OTP is used which is generated newly and randomly for every new session. So, replay attacks are not possible. Therefore, we can claim that our proposed scheme resists replay attack.

Proposition 8: Our proposed scheme resists insider attack.

Proof: Insider attack is one of most dangerous attacks to any inter-networking system. In our proposed scheme, the password PWD, username ID is always used in hashed form along with other key credentials such $H (H (H (g) \| H (SR)) \| H (ID))$, $H (H (H (SN) \| H (PWD)) \| g)$. Also the STEGO_IMG is a factor which conceals the fact that there is anything special with this image file. Only a user can know that the image file is not an ordinary file. Also, other credentials are encrypted using KEY_USER which is further embedded in encrypted form in STEGO_IMG. So, these credentials are safe as well. Along with this, OTP and digital signature private key are needed at the time of login. Only a legitimate user can have all these factors simultaneously. Therefore, our proposed scheme is claimed to resists insider attack.

Proposition 9: Our proposed scheme can withstand password guessing attack.

Proof: Password guessing attack is one of the most common attacks. In our scheme, password is never used openly, it is always used in hashed form such as H (H (H (SN) || H (PWD)) || OTP), H (H (H (SN) || H (PWD)) || g) using one way hash function. Also, we are using STEGO-IMG, digital signature, OTP and other some other credentials which makes the scheme very robust. Therefore we can claim that our scheme cannot be broken by password guessing attack.

Proposition 10: Our proposed scheme resists Man-In-The-Middle (MITM) attack.

Proof: In Man-In-The-Middle attack, adversary intercepts the message between communicating parties and use this message after user logs out. In our proposed scheme, every message is sent always in encrypted form. Every sensitive message consists of random numbers which are generated newly for each login session. So after every sign out, the sensitive messages are of no use any more. Therefore, we can claim that our proposed scheme can resist Man-In-The-Middle (MITM) attack.

Proposition 11: Our proposed scheme resists denial of service attack.

Proof: In proposed scheme, every user needs to enter password, username and upload STEGO_IMG image file. Before generating login request that is to be sent to service provider, STEGO_IMG is verified at the local system. If correct file is provided than only login request is sent. And if we consider the case when attacker uses lost STEGO_IMG, still it cannot be verified because verification involves use of private key ST_{priv} which is also encrypted with hash of PWD and ID. So, attacker cannot send bogus login request to overload service provider. Therefore, we can claim that our scheme can thwart denial of service attack.

Proposition 12: Our proposed scheme resists parallel-session attack.

Proof: In our proposed scheme, it is not possible to impersonate a legal user by creating valid login request because only a legitimate user has access to key factors and credentials are shared between user and service provider. Also, only a valid user has private key ST_{priv} and DS_{priv} which are essential part in order to get authenticated. Also, only legitimate user receives one time password (OTP) on his/her mobile phone which is verified by service provider. Therefore, we can claim that our scheme is strong against parallel session attack.

Proposition 13: Our proposed scheme resists phishing attack.

Proof: In our scheme, service provider and user both are authenticated to each other. Only a genuine service provider can be authenticated because only genuine service provider is able to compute keys to encrypt secret data sent to user and decrypt data sent by user using credentials which are available only to genuine service provider. Therefore, we can claim that our scheme resists phishing attack.

Proposition 14: Our proposed scheme resists forgery attack.

Proof: If an attacker wants to impersonate valid user, he must be able to generate valid login request [K, E1, E2, E3, SIGN, and STEGO_IMG]. Only after entering correct password, username and uploading correct STEGO_IMG along with ST_{priv}, login request is sent to service provider. Further, to compute other parts of login request, credentials such as [g, PWD, ID, OTP, DS_{priv}, SN]are required but adversary has no idea about these. So, Login request cannot be forged. Therefore, we can claim that our scheme is strong against forgery attack.

CHAPTER 8: CONCLUSION AND FUTURE WORK

From last few decades, there have been major changes and advancements in technologies. Now, the traditional IT resources setups are being replaced with CLOUD COMPUTING technology. Cloud computing consists of a large-scale distributed and virtual machine computing infrastructure which offers dynamically scalable resources such computational power, storage, hardware platforms and applications as services over the Internet. Cloud computing suffers from many issues. One of the major issues of cloud computing is security.

Among the security issues of cloud computing, **authentication** is considered as one of the most important key issue. So a strong and secure authentication scheme is needed for cloud computing**. Factors based Authentication** is more appropriate with principles of cloud authentication. Various solutions have been proposed in research papers to implement factor based Authentication.

In our work, we proposed a strong Two Factor (2FA) mutual authentication Scheme for cloud computing. The proposed authentication scheme comprises of key factors which when combined give an unambiguous identification to user. We are using two categories of factors i.e. knowledge factor and possession factor. The proposed scheme uses Password, STEGO-IMG, OTP and Digital signature. In our scheme, a user gets username, password and STEGO_IMG during registration phase. Data owner sends some secret credentials to user and service provider. In order to successfully login, user needs to provide correct username, password, STEGO_IMG, OTP. We are using digital signature as well. Along with the factors, the secret data sent to user and service provider by data owner at the time of registration plays an important role in authenticating both parties.

Our scheme can withstand many common attacks such as forgery attack, Man-In-The-Middle attack, password guessing attack, impersonation attack, phishing attack, parallel session attack, denial of service (DoS) attack, replay attack, insider attack. Our scheme is designed in such a way that only legitimate user and service provider can prove their authenticity.

The proposed scheme provides many facilities along with strong mutual authentication such as it provides identity management, user’s privacy, security of stored data, session key management. Along with many features, the scheme provides feature of forward secrecy as well. Our scheme is easy to implement, user friendly and strong.

Graphical password is a popular research area in the field of authentication. In future, graphical password can be used instead of alphanumeric password in our proposed scheme and both schemes i.e. one with graphical password and present scheme can be compared. Also, new graphical password techniques can be implemented that are better than the previously proposed graphical password techniques.

REFERENCES

[1] Nimmy K. and M. Sethumadhavan, “Novel Mutual Authentication Protocol for Cloud Computing using Secret Sharing and Steganography”, 2014 Fifth International Conference on the Applications of Digital Information and Web Technologies (ICADIWT), 2014.

[2] Yang and chu, “Cloud Password Manager Using Privacy-Preserved Biometrics”, IEEE International Conference on Cloud Engineering (IC2E), 2014.

[3] Yang and Lai, “Protecting Data Privacy and Security for Cloud Computing Based on Secret Sharing”, International Symposium on Biometrics and Security Technologies (ISBAST), 2013.

[4] L. B. Jivanadham, A.K.M.M Islam, Y. Katayama, S. Komaki, and S. Baharun, “Cloud Cognitive Authenticator (CCA): A Public Cloud Computing Authentication Mechanism”, in Proc. International Conference on Informatics, Electronics & Vision (ICIEV), Dhaka, Bangladesh, pp. 1-6, 2013.

[5] Yang, Chang and Huang, “A User Authentication Scheme on Multi-Server Environments for Cloud Computing”, 9th International Conference on Information, Communications and Signal Processing (ICICS), 2013.

[6] Jaidhar C. D., “Enhanced Mutual Authentication Scheme for Cloud Architecture”, IEEE 3rd International Advance Computing Conference (IACC), 2013.

[7] Ali A.Yassin, Hai Jin, Ayad Ibrahim, Deqing Zou, “Anonymous Password Authentication Scheme by Using Digital Signature and Fingerprint in Cloud Computing”, Second International Conference, Cloud and Green Computing (CGC), 2012.

[8] Ali A.Yassin, Hai Jin, Ayad Ibrahim, Weiz hong Qiang, Deqing Zou, “A Practical Privacy-preserving Password Authentication Scheme for Cloud Computing”, IEEE 26th International Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2012 .

[9] Guo and Liaw and Hsiao, “Authentication Using Graphical Password in Cloud”, 15th International Symposium on Wireless Personal Multimedia Communications (WPMC), 2012.

[10] Tsague and Nelwamondo, “An Advanced Mutual-Authentication Algorithm Using 3DES for Smart Card Systems”, Second International Conference on Cloud and Green Computing, 2012.

[11] Z. Hao, S. Zhong. N. Yu, “A Time-Bound Ticket-Based Mutual Authentication Scheme for Cloud Computing”, International Journal Of Computers, Communications & Control, 2011.

[12] Tien-Ho, Chen,Hsiu-lien, Yeh, Wei-KuanShih, “An Advanced ECC Dynamic ID-Based Remote Mutual Authentication Scheme for Cloud Computing", Fifth FTRA International Conference on Multimedia and Ubiquitous Engineering,2011.

[13] A. J. Choudhury, P. Kumar, M. Sain, H. Lim, and H. Jae-Lee, “A Strong User Authentication Framework for Cloud Computing”, in Services Computing Conference (APSCC), IEEE Asia-Pacific, pp. 110–115.,2011.

[14] J.H. Yeh, “A PASS scheme in cloud computing - protecting data privacy by authentication and secret sharing”, Proc. of International Conference on Security and Management, 2011.

[15] S. Lee, I. Ong, H.T. Lim, H.J. Lee, “Two factor authentication for cloud computing”, International Journal of KIMICS, vol. 8, pp. 427-432,2010.

[16] J.H. Yang and C.C.Chang, “An ID-based remote mutual authentication with key agreement scheme for mobile devices on elliptic curve cryptosystem", Computers & Security, Vol. 28, pp. 138-143, 2009.

[17] S. Shin, K. Kobara and H. Imai, “A Secure Construction for Threshold Anonymous Password-Authenticated Key Exchange”, IEICE Transactions on Fundamentals, Vol.E91-A, No.11, pp.3312-3323, 2008.

[18] G. Shailaja, K. P. Kumar and A. Saxena, “Pairing based Mutual Authentication Scheme Using Smart Cards”, in Secure Technology Lab., Institute for Development and research in Banking Technology, Castle Hills, Masab Tank, Hyderabad 500057, India, 2006.

[19] V. Shoup and A. Rubin, “Session key distribution using smartcards”, in: Proc. EUROCRYPT 96, in: LNCS.,vol 1070, Springer-Verlag, , pp 321-333, 1996.

[20] G.E. Blonder, “Graphical Passwords,” in Lucent Technologies, Inc., Murray Hill, NJ, U. S. Patent, Ed. United States, 1996.

[21] M. Bellare, P. Rogaway, “Provably secure session key distribution—The third party case”, in: Proc. 27th ACM Symp. on Theory of Computing, ACM, Las Vegas, pp 57-66, 1995.