International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 8 (2018) pp. 6138-6150

Time-Based Proxy Re-Encryption for User Revocation with Reduced UAKs in Cloud Storage

VenkataVara Prasad, Lokeswari Y. Venkataramana, Pandiyan Muthuraj

Department of Computer Science and Engineering, Sri Sivasubramaniya Nadar College of Engineering, Chennai, India.

Abstract

With rapid development of cloud computing, more and more enterprises will outsource their sensitive data for sharing in cloud. To keep the shared data confidential against un-trusted Cloud Service Providers (CSPs), a natural way is to store only the encrypted data in cloud. The key problems of this approach include establishing access control for the encrypted data and revoking the access rights from users when they are no longer authorized to access the encrypted data. Symmetric key Cryptography provides a single shared key for group of users. If a user leaves the group, secret key needs to changed and data needs to be encrypted again with new secret key. To overcome this, Proxy Re-Encryption (PRE) scheme efficiently handles user revocation by re-encrypting the cipher text again at proxy server. For fine-grained access control Attributed Based Encryption (ABE) uses attributes of the users to provide access to data. Time-based Proxy Re- Encryption specifies time for every attribute of a user which is termed as access time of the attribute. Each user will be provided with set of User Attribute secret Keys (UAKs). Each UAK is associated with user, attributes and access time of the attribute. In this way, each attribute of a user will have a separate UAK. This results in creation of many numbers of UAKs for a user. To reduce the number of UAKs, the proposed system will generate UAKs for group of attributes in the access structure rather than generating UAKs for each of the attribute.

Keywords: Cloud Security, Data Security, Time-based Proxy Re-Encryption, User Revocation, Attribute-Based Encryption, CP-ABE, KP-ABE.

INTRODUCTION

Cloud computing is an emerging technology [8] in which resources of the computing infrastructures are provided as services over internet. Cloud allows user to access application without installation and store their personal data in remote computer. It provides with a way to share distributed recourses and services that belong to different organizations. In this technology users have to entrust their data to cloud providers, there are several security and privacy concerns on outsourced data. As the data is shared over the network, data should be encrypted to maintain confidentiality against untrusted users. There are various encryption schemes that provide security and access control over the network. They are discussed in the following subsections.

A. Health Care Moving into Cloud

Personal Health Record (PHR) is an emerging patient-centric model [9] of health information exchange, which is often outsourced to be stored at a third party, such as cloud providers. However, there have been wide privacy concerns as personal health information could be exposed to those third party servers. To assure the patients control over access to their own PHRs, it is a promising method to encrypt the PHRs before outsourcing. Yet, issues such as risks of privacy exposure, scalability in key management, flexible access and efficient user revocation have remained the most important challenges towards achieving fine-grained, cryptographically enforced data access control. Moving health care into cloud helps in the following ways:

1. Maintaining Electronic Health Record.
2. Remote Monitoring of patients (Patients in Battle field).
3. Sharing of medical data with other health organizations.
4. Collaborative consultation among experts in different health organizations.
5. Efficient treatment given to patients in regular / in emergency basis.
6. To detect serious diseases in initial stage and recover them.
7. Self-caring service by retrieving similar medical data and diagnosing patients themselves. (Home Diagnosis)
8. Medical Research.

B. Symmetric Key Cryptography

Symmetric key encryption [10] involves using a single key to encrypt and decrypt data. For the receiver to decrypt the encrypted data, they must know the secret key. This enables the sender to send secret key along with the message to the receiver. Anyone who might be monitoring the network could steal the encrypted data and the key necessary for decrypting it. The other way is to share the secret key to group of users and use it for decryption. Whenever a user leaves a group, this key needs to be changed and encrypt the data again.

C. Public Key Cryptography

Public key encryption [11] uses a pair of keys: a public key that is sent along with the message and a private key which is always in the possession of the recipient. The private key is based on a derivative of the public key and only these two keys working together can decrypt the data. Because the private key is never sent across the network, it remains secure. The down side of public key encryption is that it tends to be very slow and resource intensive. This makes it difficult to send large amounts of data using public key encryption. Public key cryptography is more suitable only when there is one sender and one receiver.

D. Attribute Based Encryption

Attribute-based encryption [1, 2] is a type of public-key encryption in which the secret key of a user and the cipher text are dependent upon attributes (e.g. the country in which he lives, or the kind of subscription he has). In such a system, the decryption of a cipher text is possible only if the set of attributes of the user key matches the attributes of the cipher text. A crucial security aspect of Attribute-Based Encryption is collusion resistance. It prevents access to two unauthorized users even if they combine their attribute keys.

E. Proxy Re-Encryption

Proxy re-encryption schemes [3] are cryptosystems which allow third parties (proxies) to alter a cipher text which has been encrypted for one party, so that it may be decrypted by another. In which a semi-trusted proxy converts a cipher text for owner into a cipher text for user without seeing the underlying plaintext. The Re-Encryption scheme assists the owner of the data to delegate the role of secure access to the proxy. Proxy manages the set of user public keys in a key storage. In sharing of Personal Health Records (PHRs) in healthcare domain, owner is the patient, users will be many such as doctors, surgeon, nurse, etc.. Thus Symmetric Key Encryption has the downside of re-encrypting the PHR whenever a key is changed due to user leaving the group (hospital). Asymmetric Key Encryption is more suitable for one-one communication and it is not suitable for sharing of PHRs where there is one data owner (patient) and many users such as doctors, pharmacy, nurse, surgeon, etc., So authors exploited Attribute Based Encryption (ABE) for one-to-many communication and user revocation is easily managed with Time-based Proxy Re-Encryption (PRE).

RELATED WORK

A. Securing Personal Health Records in cloud computing: patient-centric and fine-grained data access control in multi-owner settings

A novel framework was proposed [1] for access control to PHRs within cloud computing environment. To enable fine-grained and scalable access control for PHRs, Attribute Based Encryption (ABE) techniques was to encrypt each patients' PHR data. To reduce the key distribution complexity, the system was divided into multiple security domains, where each domain manages only a subset of the users. In this way, each patient has full control over his/ her own privacy, and key management complexity is reduced dramatically. For example, if owner of the data (patient) provides access to a Doctor, DeptAdultCardiology in a health centre, the access structure will be defined by the owner as shown in the Figure 1. Further if the patient allows surgeon to access record, the existing access tree can be extended for surgeon too as shown in Figure 2.

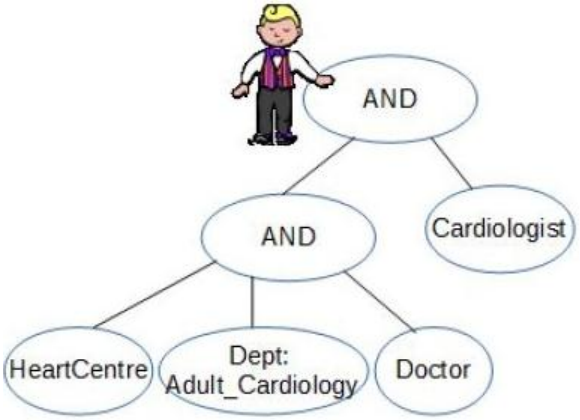

Figure 1. General ABE Structure

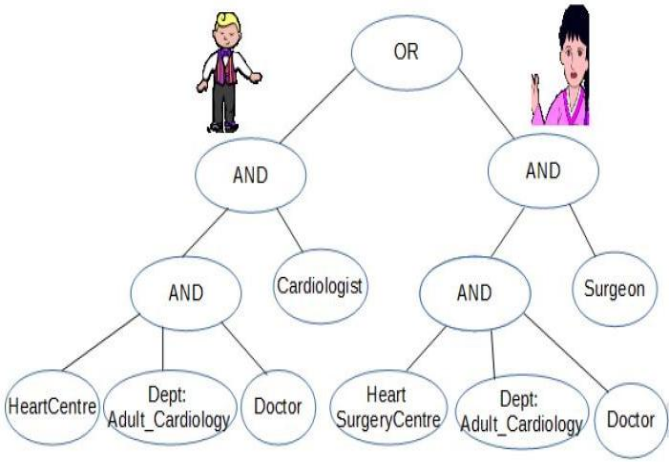

Figure 2. General ABE structure with Scalability

B. Secure Sharing of Personal Health Records in Cloud Computing: Cipher text-Policy Attribute based Signcryption

The storage of personal medical and health information is usually outsourced to some third parties. This may result in the exposure of patient's privacy to unauthorized individuals or organizations. In order to address this security loophole, a promising solution was proposed [2]. New approach was proposed for fine-grained access control and secure sharing of signcrypted (sign-then encrypt) data. This new primitive is called as Ciphertext-Policy Attribute-Based Signcryption (CP-ABSC) which satisfies the requirements of cloud

computing scenarios for PHR. CP-ABSC combines the merits of digital signature and encryption to provide confidentiality, authenticity, Unforgeability, anonymity and collusion resistance. The correctness, security and efficiency of this scheme are also proven. It avoids collusion if attributes of two different users are combined together to access data as shown in Figure 3. The problem with ABE scheme is the encryptor cannot decide who can decrypt the encrypted data. It can only choose descriptive attributes for the data; it is not suitable in some application because a data owner has to trust the key issuer.

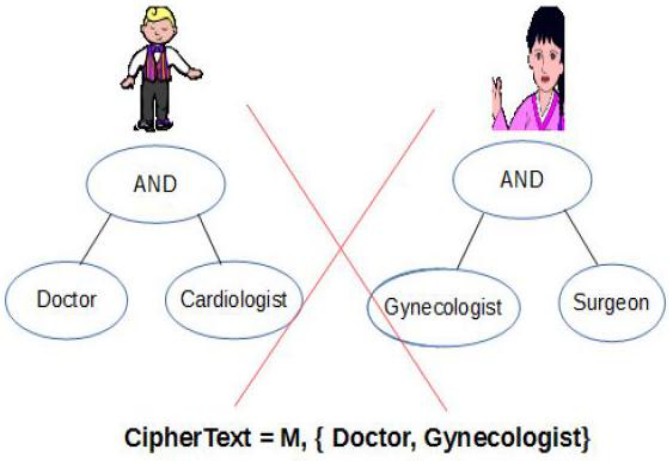

Figure 3. Fine Grained Access Control ABE structure

C. A Platform for Secure Monitoring and Sharing of Generic Health Data in the Cloud

Proxy Re-Encryption is based on the concept of a semi-trusted proxy that uses a re-encryption key to translate a cipher-text under the data owner's public key into another cipher text that can be decrypted by another user's private key [3]. The data is never decrypted before it is re-encrypted hence the proxy will never be able to reveal the plaintext at any time. Many recent works have realised proxy re-encryption as a technique to enable data sharing in the cloud. Although the proxy re-encryption was not explicitly used, the system mimics a proxy re-encryption (PRE) algorithm scheme from the point of view of the data owner and user. Re-Encryption of original cipher text is done with the help of semi-trusted third party (proxy server). Here, encrypted data which is already done by the owner provided to the proxy server, proxy server will re-encrypt that file without knowing the plain text and user can decrypt without sharing his/ her secret key to the proxy server. The drawback of PRE is that each of the user is provided with separate key which may cause burden to the user.

D. Time-Based Proxy Re-Encryption Scheme for Secure Data Sharing in a Cloud Environment

A fundamental approach [4] for secure data sharing in a cloud is to let the data owner encrypt data before out-sourcing. To simultaneously achieve fine-grained access control on encrypted data and scalable user revocation, existing work combines Attribute Based Encryption (ABE) and Proxy Re-Encryption (PRE) to delegate the Cloud Service Provider (CSP) to execute re-encryption. However, the data owner should be online in order to send the PRE keys to the CSP in a timely fashion, to prevent the revoked user from accessing the future data. The delay of issuing the PRE keys may cause potential security risks. *Time-based Proxy Re-Encryption (TimePRE)* scheme was proposed to allow a users' access right to expire automatically after a predetermined period of time. In this case, the data owner can be offline in the process of user revocations. The basic idea is to incorporate the concept of time into the combination of ABE and PRE. Specifically, each data is associated with an attribute-based access structure and an access time, and each user is identified by a set of attributes and a set of eligible time periods which denote the period of validity of the user's access right.

The notations $s_a(y)$, $s_a(y, m)$ *and* $s_a(y, m, d)$ are represented to denote *PRE* keys on attribute '*a*' in time *(y), (y, m), and (y, m, d),* which can be used to update attribute *a's* initial public key PK_a to time-based public keys $PK_a(y)$, $PK_a(y, m)$ and $PK_a(y, m, d)$ respectively. Since the PRE key is derived from a root secret key '*s*' and the current access time '*t*' as represented in Figure 4. For each attribute '*a'*, the CSP can use the root secret key '*s*' and the time tree to hierarchically calculate the *time-based PRE keys*. Each user is granted with a set of Time-based User Attribute Secret Keys *(UAK).* Each time-based UAK is associated with a user, an attribute, and an effective time period. If user '*u*' is eligible for attribute *'a'* in *day (y, m, d),* the data owner first uses the root secret key '*s*' to obtain day-based attribute public key $PK_a(y, m, d)$ from initial attribute public key PK_a and then uses $PKa(y, m, d)$ to generate a day-based $UAK\ SK_{u,a}\ (y, m, d)$ for user *'u'*. The same situation holds for the case that user *'u'* is eligible for attribute '*a*' in a *month (y, m)* or a *year (y).* The drawback of Time-based PRE is that same access time is provided for all the attributes associated with particular user.

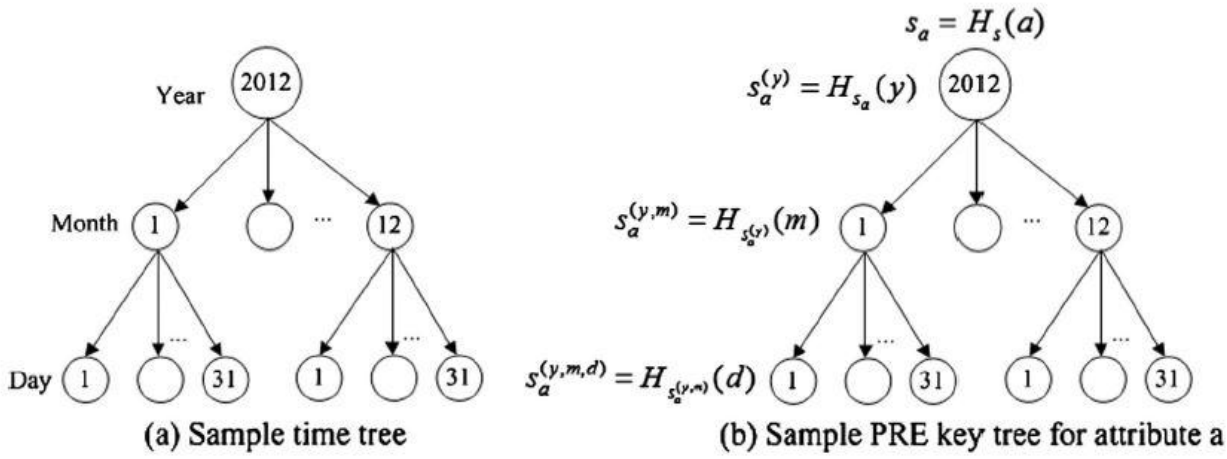

Figure 4. Time-Based PRE Tree Structure

E. Attribute Based Data Sharing with Attribute Revocation

Cipher text-Policy Attribute Based Encryption (CP-ABE) is a promising cryptographic primitive for fine-grained access control of shared data. In CP-ABE, each user is associated with a set of attributes and data are encrypted with access structures on attributes. A user is able to decrypt a cipher text if and only if his/her attributes satisfy the cipher text access structure. They achieve this by uniquely integrating the technique of proxy re-encryption with CP-ABE [5, 6], and enable the authority to delegate most of laborious tasks to proxy servers. This technique can also be applicable to the Key-Policy Attribute Based Encryption (KP-ABE) counterpart. Data owner will generate the root secret key and the user secret key by the use of universal attributes along with security parameters. Then owner encrypts his/her data with public key and access structure and those encrypted data will fed to the CSP. Whenever there is a request for accessing particular data, CSP will re-encrypt those data with effective time period. Meanwhile *UAKs* for that particular user will be generated with attributes and the access time. The user will use *UAKs* to decrypt the re-encrypted data. The downside of this method is generating *UAKs* for every attribute may affect the Key Management System. Having discussed about related research work in securing cloud environment, the motivation and objective is described below.

F. Improved Proxy Re-Encryption schemes with Applications to Secure Distributed Storage

A malicious user or operator can identify and exploit the vulnerabilities of the system. Numerous works are being done in order to reinforce the cloud capacities in term of protecting data and managing access control using cryptography, data fragmentation and access control policies [6]. A new approach was proposed by authors in which the cloud provider is excluded from any involvement in the access management with the aim of minimizing the leaks. Authors developed and tested programs based on a capability-list and using both symmetric and asymmetric cryptography. Protecting and managing access control to outsourced data has been the main issue. The authors proposed a proxy re-encryption; where owner encrypts with symmetric content key blocks of data before sending it to cloud servers. Content keys were encrypted with the owner's master public key. Owner's master private key and user's public keys are then combined to generate proxy re-encryption keys which are used to recover plain text intended to a specific user.

G. Security of Key in Cloud using Cryptography

To construct a secure cloud computing system, security at infrastructure, service platforms and application software levels have to be studied. Information encryption is one of the effective means to achieve cloud computing information security. Traditionally, information encryption focuses on specified stages and operations, such as data encryption. For cloud computing, a system level design has to be implemented. Crypto cloud computing is a new secure cloud computing architecture. It can provide protection of information security at the system level, and allows users to access shared services conveniently and accurately. Crypto cloud computing protects individuals connections with the outside world. It can protect the personal privacy without any delay of information exchange. Crypto cloud computing is based on the *Quantum Direct Key system (QDK).* Quantum Direct Key is a set of advanced asymmetric offline key mechanism [7]. All entities get public and private key pair according to their ID. Each entity only holds its own private key, but has a public key generator to generate any public key. In this system, an entity can produce the public key of any other entities offline, no third-party agency is necessary. Crypto cloud computing architecture is based on *QDK,* it can avoid network traffic congestion, and other drawbacks using current encryption system. In the crypto cloud computing system, each entity encrypts data using their own private key. All elements in the system such as cloud computing infrastructure units, platform, virtualization tools and all involved entities have their own keys. While fulfilling their own functions of information exchange and processing, all these elements will use the public key and private key to perform authentication first. All events

occurring in cloud environment are assigned with a unique key. Thus, crypto cloud system assures the security and credibility of information exchange. Having discussed about related research work in securing cloud environment, the motivation and objective is described below.

The motivation of this research work includes the following. In order to provide data access to the health record stored in the cloud, Symmetric Key Cryptography (SKC) is not appropriate [7]. In SKC single secret key will be shared to group of users. If the user leaves the group, secret key needs to be changed every time and encryption has to be done. In order to provide a scalable user revocation scheme, Proxy Re-Encryption (PRE) was used in literature. This PRE can be combined with Attribute Based Encryption (ABE) for secure data access. Time Based Proxy Re-encryption was used to provide access to users on time basis and also to resolve user revocation issue. In the existing Time based PRE scheme, User Attributes secret Keys (UAKs) were generated for each and every attribute in Access Structure (Access Policy), where here number of UAKs generated for a single user will be many in number. This creates burden to the proxy server.

PROPOSED TIME-BASED PROXY RE-ENCRYPTION WITH REDUCED UAKS

Instead of creating UAKs for each attribute, UAKs will be created for group of attributes in an *Access Structure (AS)*. This reduces the number of UAKs generated by proxy server. Consider the following Access Structure:

AS1 = (Neurologist ^ Surgeon) ∨ (Physiologist)

UAK1 for Neurologist ^ Surgeon

UAK2 for Physiologist

AS2 = (Pulmonologist ∨ Surgeon ∨ Oncologist)

UAK for AS2 is only one.

Similarly, one UAK will be generated for *Conjunctive Normal Form (CNF)*

AS3=(Pulmonologist ^ Surgeon) ∨ (Oncologist ^ Surgeon)

UAK1 for Pulmonologist ^ Surgeon

UAK2 for Oncologist ^ Surgeon

Hence the objective is to provide a scalable user revocation system with Time-based Proxy Re-Encryption in cloud storage and also reduce the number of User Attribute secret Keys (UAKs) in Time-based Proxy Re-encryption.

The main idea of the *TimePRE* scheme is to incorporate the concept of time into the combination of ABE and PRE. Intuitively, each user is identified by a set of attributes and a set of effective time periods that denotes how long the user is eligible for these attributes, i.e., the period of validity of the users' access right. The data accessed by the user is associated with an attribute-based access structure and an access time. The access structure is specified by the data owner, but the access time is updated by the CSP with the time of receiving an access request. The data can be recovered only by the user whose attribute's satisfies the access structure and whose access right's are effective in the access time. To enable the CSP to update the access time automatically, we first express actual time as a time tree. The height of the time tree can be changed as required. For ease of presentation, time is accurate to the day, and the time tree is classified into three layers in order: year, month, and day. Notations used for particular day, month and year are *(y, m, d), (y, m) and (y)* respectively. For example, (2017, 4, 5) denotes April 5, 2017. The access time associated with data corresponds to a leaf node in the time tree and the effective time periods associated with a user correspond to a set of nodes in the time tree as depicted in Figure 4. If there is a node corresponding to an effective time period that is an ancestor of (or the same as) the node corresponding to the access time, then the users' access right is effective in the access time.

The proposed framework for *Time-Based PRE* is depicted in Figure 5.

The following steps describe the sequence of steps to be followed while accessing *PHR* from cloud using Time based PRE with less number of *UIKs* and *UAKs.*

1. Initially key generator will produce the Public key *(PK),* Master key *(MK)* and Root secret key (*s*) by taking input as a security parameters and the user attributes.
2. User will share their Public Key *(PKu)* with the owner of the data.
3. Key generator will again generate the *User Identity Key (UIK)* and *User Attribute Key (UAK)* for the secure data decryption.
4. Data owner will generate the cipher text by encrypting the *original data (F).*
5. *Cipher text CA* and the necessary keys will be sent to proxy server for the re-encryption of cipher text *CA.*
6. *UIK* and *UAK* will be sent from owner to the each of the authorized user for data access.
7. User will make the request for encrypted cipher text to the proxy server.
8. Proxy server provides the encrypted cipher text C^tA to the user and user will decrypt those C^tA with *UIK* and *UAK.*

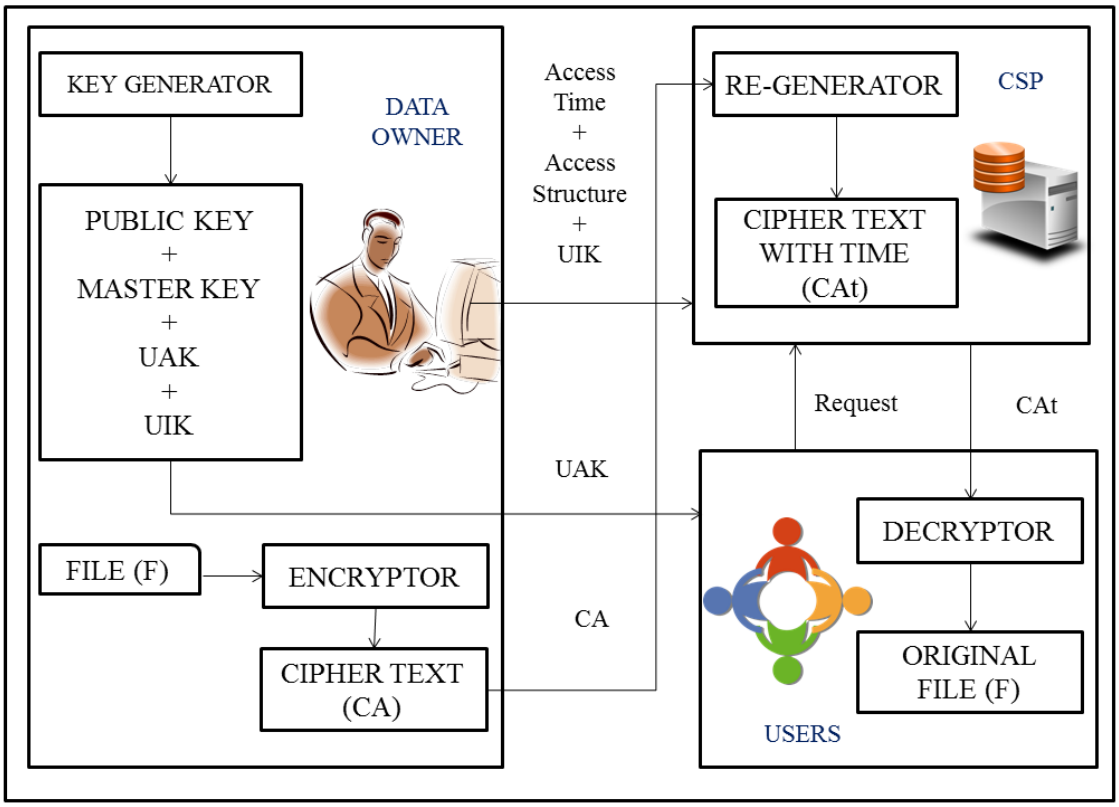

Figure 5. Architecture for Time Based PRE

A. Key Generation

The data owner takes a sufficiently large security parameter K as input to generate the system public key *PK*, the system master key *MK*, and the root secret keys. The system public key will be published, the system master key will be kept secret, and the root secret key will be sent to the CSP.

B. UIK and UAK generation

Suppose that user '*u*' with public key PK_u is eligible for attribute '*a*' and his/her access right is effective in time T_u. The data owner uses the system public key *PK*, the system master key *MK*, the root secret key *s*, user public key PK_u, attribute '*a'*, and effective time period T_u to generates user identity secret key *(UIK)* SK_u and time-based user attribute secret key *(UAK)* SK_u, $a^T{}_u$ for user '*u'*.

C. Proxy Setup

Proxy re-encryption [6] allows a proxy to transform a cipher text computed under owner's public key into one that can be opened by users' secret key. There are many useful applications of this primitive. For instance, owner might wish to temporarily forward encrypted email to his/her colleague, without giving his/her secret key. In this case, owner as a delegator could designate a proxy to re-encrypt his/ her incoming mail into a format that users the delegate can decrypt using his/ her own secret key. Clearly, owner could provide his/ her secret key to the proxy, but this requires an unrealistic level of trust in the proxy. The primary advantage of this schemes is that they are unidirectional (i.e., owner can delegate to users without users having to delegate to owner) and do not require delegators to reveal all of their secret key to anyone or even interact with the delegate in order to allow a proxy to re-encrypt their cipher texts. In this scheme, only a limited amount of trust is placed in the proxy. For example, proxy could not decrypt the cipher text, that is re-encrypted by proxy itself and this scheme is secure even when the proxy publishes all the re-encryption information it knows. This enables number of applications that would not be practical if the proxy needed to be fully trusted.

D. Encryption

Cloud encryption is the transformation of a cloud service customer's data into cipher text. Cloud encryption is almost identical to in-house encryption with one important difference here is, the cloud customer must take time to learn about the provider's policies and procedures for encryption and encryption key management. The cloud encryption capabilities of the service provider need to match the level of sensitivity of the data being hosted.

Encryption by Owner.

The data owner takes a Disjunctive Normal Form (*DNF)* access structure '*A*', a data '*F'*, and system public key *PK*, e.g., initial

public keys of all attributes in the access structure PK_a as inputs to output a cipher text *CA.*

Encryption by Proxy server.

Given a cipher text *CA* with structure *A*, the CSP first uses the system public key '*PK'* and the root secret key '*s'* to generate *PRE* keys on all attributes in the access structure '*A*' based on the access time '*t*' and then uses these *PRE* keys to re-encrypt the original cipher text *CA* to $C'A$.

E. Decryption by Users

User '*u*', whose attributes satisfy the access structure '*A"* and whose effective time period T_u satisfy the access time '*t',* can use SK_u and SK_u, aT_u to recover *F* from $C'A$.

F. Test Bed

The Test Bed consists of two systems. One being Server at owner end and other is client at Users' end. Owner and proxy server will reside in the Server while users will reside in the Client. Key generation and the encryption is done by owner and stores the encrypted file in the Server. Decryption will be done at client machine at users' side by requesting the file from proxy server. NextCloud an open source software, which provides Platform as a Service (PaaS) was exploited to build proxy server.

NextCloud and its Features

With Nextcloud [13], system administrators can control and direct the flow of data between users or between servers. Rule based file tagging and responding to these tags as well as other triggers like physical location, user group, file properties and request type enables administrators to specifically deny access to resources, convert, delete or retain data following business or legal requirements. Nextcloud puts you in control of your data and keeps it safe. Nextcloud is a suite of client-server software for creating file hosting services and using them. The primary functional difference is that Nextcloud is free, open-source and thereby allowing anyone to install and operate it without charge on a private server. In Nextcloud, the open architecture allows adding additional functionality to the server in form of new applications. NextCloud supports ***LAMP*** feature which includes ***L*inux, *A*pache, *M*ySQL and *P*HP.** It also provides other features as mentioned below.

Work Flow Management

Through File Access Control and automatic file tagging, Nextcloud gives administrator's control over data access by enabling them to define strict rules that is need to adhere. If users in certain groups or geographic regions should not be given access to certain file types or if data with a specific tag should not be shared outside the company, administrators can make sure their Nextcloud instance enforces these rules. File Access Control can play a crucial role in enforcing company policy on data sharing.

File Access Control at Home Page

Home users will find that the File Access Control app and other workflow tools that can be used to prevent accidental sharing of sensitive data, adding an additional layer of protection to Nextcloud.

Security and Authentication

Administrators can set permissions on sharing and access to files using groups. Permissions of underlying storage, like Windows Network Drive access rights, are respected by Nextcloud. Nextcloud uses industry-standard (Secure Layer/ Transport Layer Security) SL/TLS encryption for data in transit. Additionally, data at rest, in storage can be encrypted using a default military grade (Advanced Encryption Standard) AES-256 encryption. Keys can be handled with the build in key management or you can opt for a custom key management for integration in existing infrastructure. As keys never leave the Nextcloud server, external storage systems never have access to unencrypted data.

User Privilege and Revocation

The Nextcloud authentication system supports pluggable authentication including Two-factor authentication and device specific passwords, complete with a list of connected browsers and devices on the user's personal page. As extra protection, device specific password tokens can deny access to the file system.

Active user sessions can be invalidated through a list, by removing the user in the admin settings or by changing passwords. Users can manage their own sessions and devices.

Data Administrators can set password quality policies enforced by Nextcloud as well as limit or disable sharing, enforce expiration dates and passwords on shares, disable preview generation and more.

CP-ABE Toolkit and its Features

The *cpabe* toolkit [12] provides a set of programs, implementing a cipher text-policy attribute-based encryption scheme. It uses the *PBC* library for the algebraic operations. It is to be noted that the *cpabe* toolkit might not compile against versions of *PBC* older than 0.5.4. The code is split into two packages, *libbswabe* (a library implementing the core crypto operations) and *cpabe* (higher level functions and user interface). In a cipher text policy attribute-based encryption scheme, each user's private key is associated with a set of attributes representing their capabilities, and a cipher text is encrypted such that only users, whose attributes satisfy a certain policy can decrypt.

For example, we can encrypt a cipher text such that in a company it can only be decrypted by a person with attributes such as *Senior and Human Resources* or has the attribute *Executive*. One interesting application of this tool is that we can do *Role-Based Access Control (RBAC)* without requiring trusted data storage. The toolkit provides four command line tools used to perform the various operations of the scheme. They are designed for straightforward invocation by larger systems in addition to manual usage.

cpabe-setup generates a public key and a master secret key.

cpabe-keygen generates a private key with a given set of attributes.

cpabe-enc encrypts a file according to a policy, which is an expression in terms of attributes.

cpabe-dec decrypts a file using a private key.

CASE STUDY ON DIABETICS PATIENTS

There may be chance of having many numbers of persons involved in treating a patient. Each person is considered as an attribute. Consider a case study on treating Diabetic patients, the possible list of persons (attributes) involved in treating a patient are as listed below.

1. General Physician
2. Endocrinologist
3. Anesthesiologist
4. Emergency Doctors
5. Immunologist
6. Infectious Disease Specialist
7. Microbiologist
8. Nurse
9. Surgeon

In this case study on Diabetic patient, patients can be categorized as In-patient, Out-Patient and Emergency Patient as shown in the Figures 6 to 8. Each of them is provided with different attributes and access structure as discussed in the following sections.

A. In-Patient

In-patient can be admitted for a long time, so the access time for the attributes will slightly vary when compare to the others. In Figure 6 *Surgeon* will do the surgery for patient along with the *Anesthesiologist*, so they are provided with same access time. Access time for general physician is varying from surgeon, anesthesiologist and nurse, since they all have only a specific time treating a patient with respect to the operation done for a patient, rather than general check-up for a whole year as the access time for *general physician*. The access policy for an In-Patient is given below with respect to access structure in Figure 6.

[(General Physician AND Endocrinologist)] OR [(Surgeon AND Anathesian) OR (Immunologist OR Nurse)]

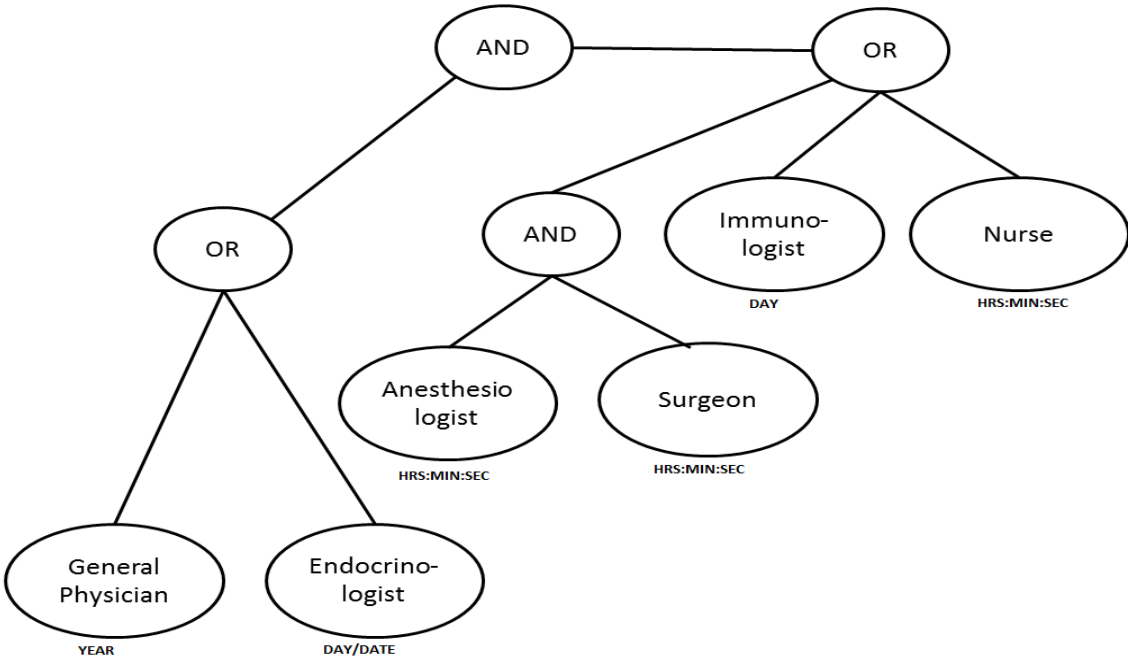

Figure 6. Access Control Tree for In-Patient

B. Out-Patient

The access structure represented in Figure 7, the access time for each of the attribute will be provided depending on the day when the patient is undergoing treatment. Day is the maximum access time limit for the accessing the record of the out-patient in most of the cases.

The access policy for an Out-Patient is given below with respect to access structure in Figure 7.

[(General Physician OR Endocrinologist)] OR [(Immunologist OR Nurse)]

C. Emergency-Patient

The attribute access time represented in the Figure 8, may not necessarily be static always. We can extend the access time for any of the attribute depending on the seriousness of the disease.

The access policy for an Emergency-Patient is given below with respect to access structure in Figure 8.

[(General Physician AND Emergency Doctor)] OR [(Surgeon AND Anathesian AND General Physician) OR (Infectious Disease Specialist OR Microbiologist)]

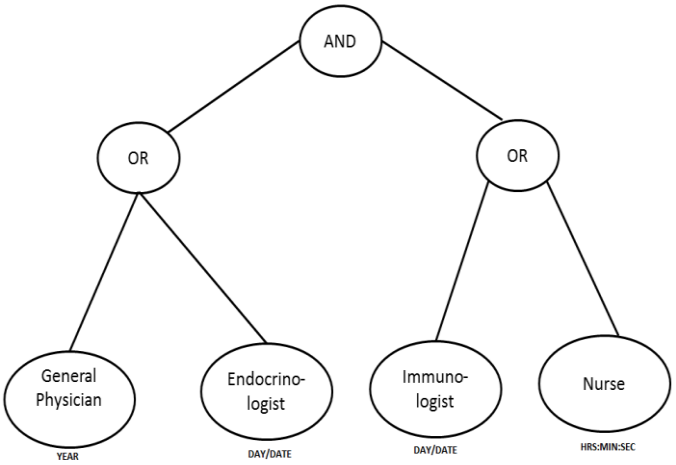

Figure 7. Access Control Tree for Out-Patient

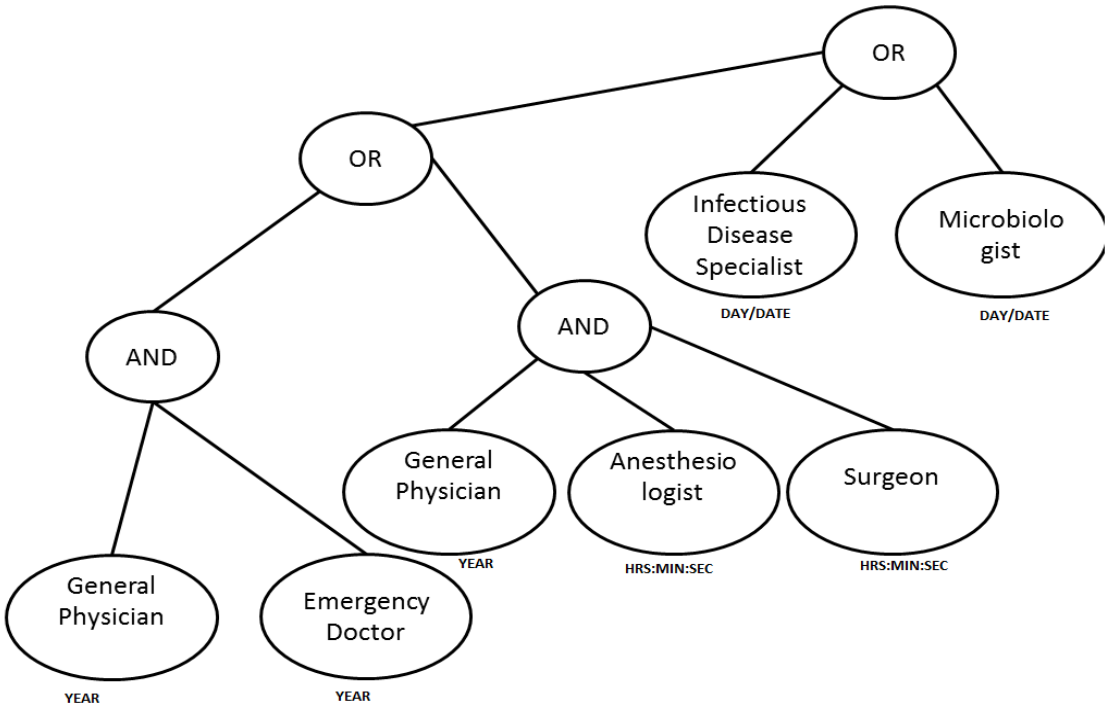

Figure 8. Access Control Tree for Emergency-Patient

WORKING PRINCIPLE OF TIME-BASED PRE FOR USER-REVOCATION

A. Key Generation using CP-ABE Toolkit

Ubuntu is Debian-based Linux operating system that provides a platform environment to built *cpabe*. First it is necessary to download, untar, compile, and install the most recent tarball of *libbswabe*, the support library [12]. Each can be installed with the standard GNU build system commands.

Installation of Source File

To work with *ABE*, we need to install and configure four source files *cpabe, gmp, libbswabe and pbc*. Create the make file and include that to library. The below are commands to install the source file.

$ ./configure

$ make

$ make install

Generating Keys

The method of generating the owner public key and master key was done with the help of *cpabe-setup*. Master key will be kept secret and the public key will be distributed to all the users who are associated with the access structure. The following command is used to generate the keys for the owner.

$ cpabe-setup

Whenever the command ***'cpabe-setup'*** is executed, the new master key and public key will be generated by overwriting the existing keys.

Key Generation for Different attributes

Based on the above discussed three differnt patient type scenarios, keys will be generated for each of the available attributes. *cpabe-key* function is utilized to generate the key based on the access structure. Each attribute will have a separate private key, which will be used for decryption. Private Key can be generated for each attribute using the following commands by specifying a particular attribute as mentioned below.

$ cpabe-keygen -o General-Physician pub-key master-key Att1 Att2

$ cpabe-keygen -o Surgeon pub-key master-key Att3

$ cpabe-keygen -o Endocrinologist pub-key master-key Att3 Att4

$ cpabe-keygen -o Nurse pub-key master-key Att1 Att3

To work with NextCloud, LAMP software bundle has to be installed and configured. While installing it is necessary to create make file and include that into the library. Mysql will be acting as a backend database and it will be available along with LAMP itself. To work well with Mysql, it is necessary to configure it with log and error log file. User interface with the local server can be accessed via web browser by providing IP address of the local machine (eg: https://xxx.xxx.xx.x).

Server admin can login with authenticated username and the password. File can be uploaded and shared with the users. Local server can create the link for all the files and folders and it will also allow users to access those shared link of file or folder. Data owner can add many number of users depending on the complexity of access structure. Authentication is maintained by providing separate password for each user.

B. Encryption Configuration

Patient record will be submitted for encryption. File can be of any format (text / pdf / images file) and it should be placed in source file location where data owner actually resides. The primary purpose of the Nextcloud server-side encryption is to protect users' files on remote storage, such as *Dropbox* and *Google Drive*, and to do it easily and seamlessly with Nextcloud. Nextcloud encrypts owner's local data and stores it in a remote server. Encryption and decryption are performed on the Nextcloud server. All files sent to remote storage, will be encrypted by the Nextcloud server, and during retrieval, Nextcloud server decrypts the file and serves it to authorized users and groups.

Nextcloud encryption consists of two parts. The base encryption system is enabled and disabled on Admin page. First Admin must enable this, and then select an encryption module to load. Currently the only available encryption module is the Nextcloud Default Encryption Module. Data owner have to enable encryption button, *'No encryption module loaded, please load a encryption module in the app menu'* message will be displayed. After which owner needs to redirect the admin's Apps page to enable the Nextcloud Default Encryption Module. Next cloud default encryption module will be added to the module selector, and gets selected automatically once the data owner returns to the admin page. Now admin must log out and then log in to initialize user's encryption keys.

C. Attributes in a Single Group

Each of the attributes associated with the access structure can be added within a single group, where distribution of keys and cipher text will be made easier for the data owner. One user is allowed to present in more than one group.

D. Encrypting File with Full Access Structure at Owner Side

Data owner encrypts medical report by specifying complete access structure in the *cpabe-enc* command as mentioned below. This creates cipher text with the extension <***filename*>.cpabe**.

]$ cpabe-enc pub_key Patient_Report.pdf (((gp or endo) and (imm or neuro)) or surg)

E. Proxy Re-Encryption at NextCloud

Proxy Re-Encryption can be achieved via Nextcloud default encryption module. The uploaded files in the Nextcloud local server will automatically get encrypted. This will generate *BIN* file which is in unreadable format.

F. Owner Sharing Keys and Cipher text to Users via NextCloud

Data owner will generate necessary key for their data users and it will be shared with other users using Nextcloud proxy server along with the cipher text .

G. Sharing Files to Group of Users with Reduced UAKs

Admin can create a link for each file with a password, which can be shared to users or groups. Local server is allowed to set different password for same link which will be shared to users or group as highlighted in Figure 9.

H. Time Privilege for Accessing the File

Time privilege for each of the user can be assigned to access the shared file by selecting expiration date as shown in Figure 10. Here the minimum privilege time is day. If no time is mentioned, then the default expiration time is one day for accessing the shared link.

I. Decryption by Data Users

Once the request is approved, data owner will share the keys and cipher text to their users. Data user can download those file and decrypt the cipher text by using the public key of user with the following command

]$ cpabe-dec pub_key neuro_key Patient_report.pdf.cpabe

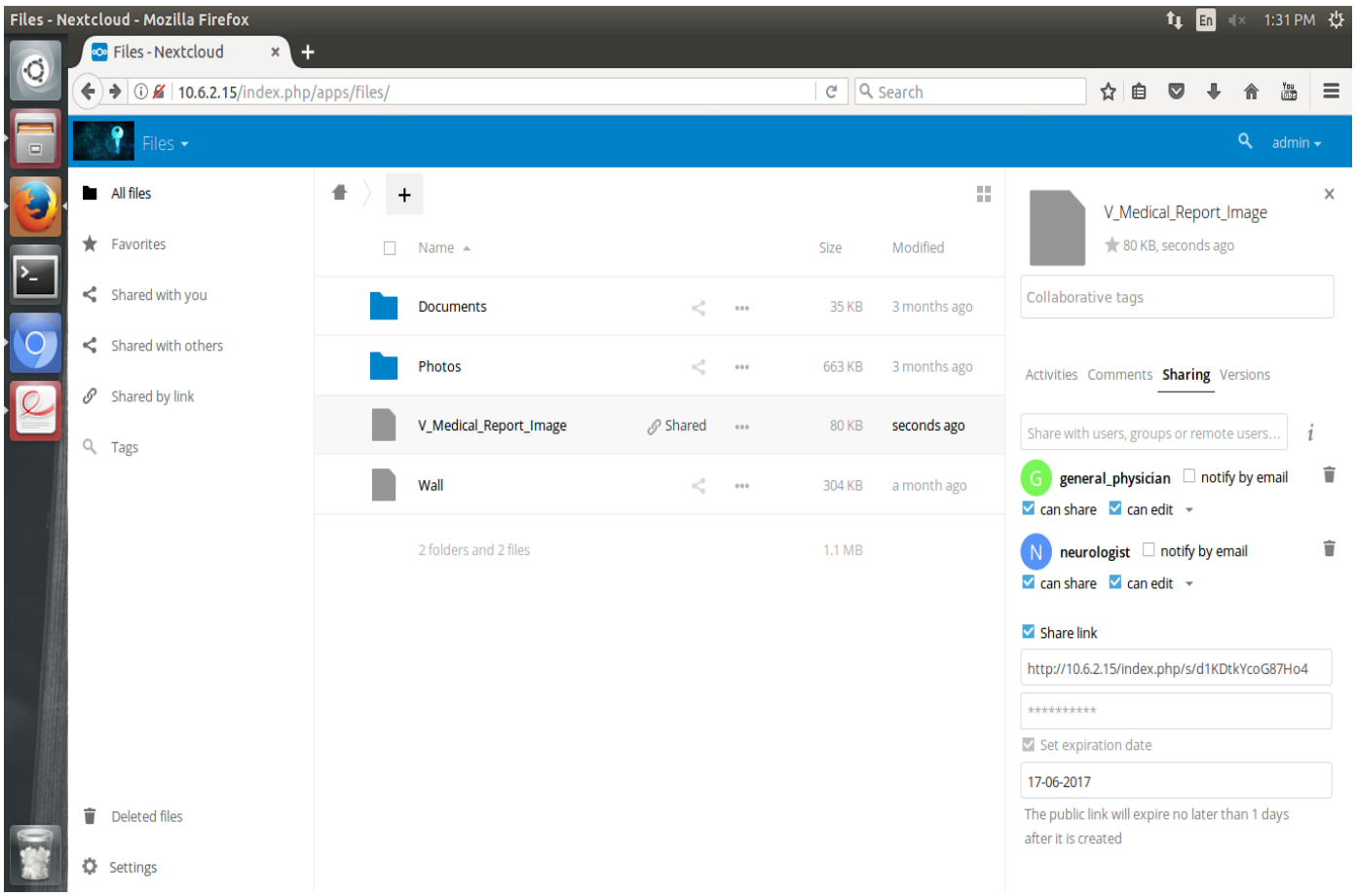

Figure 9. Sharing Files to Group of Users with Reduced UAKs

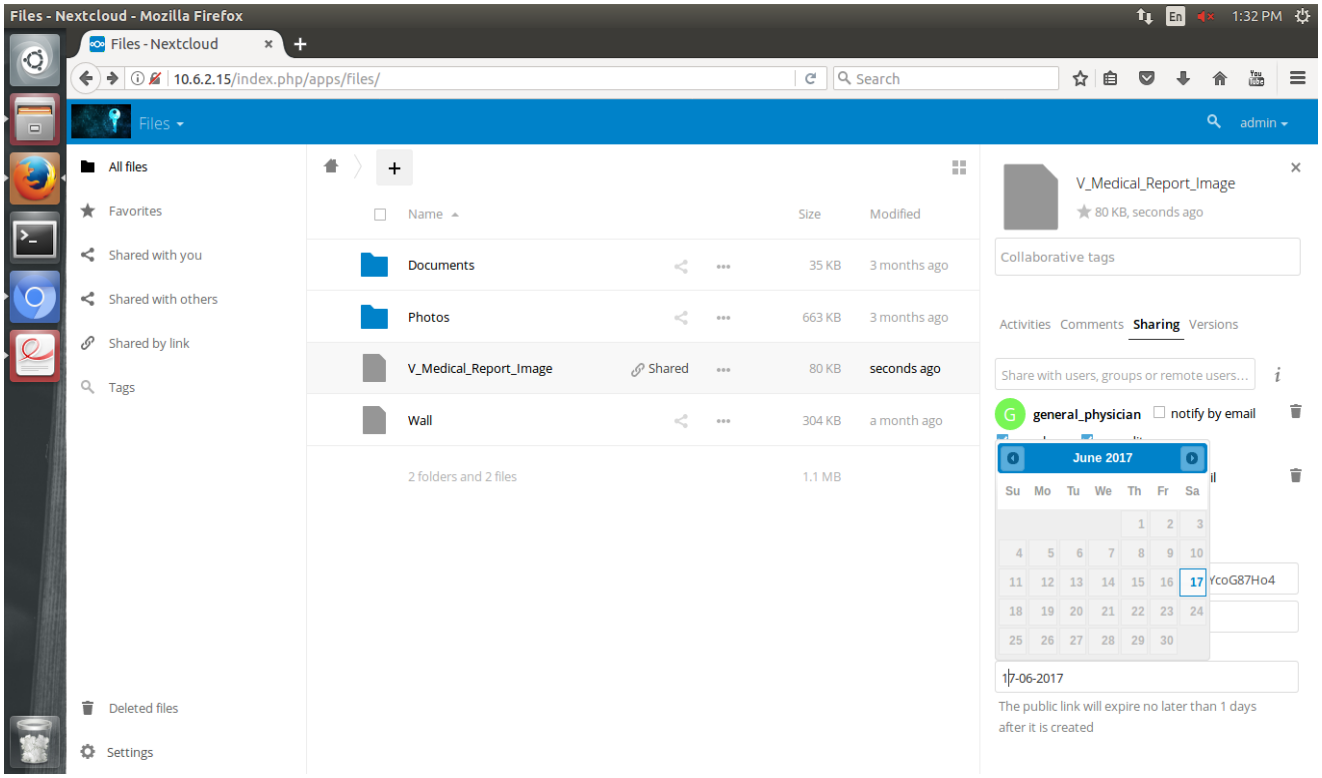

Figure 10. Granting Time Privilege for Accessing the File in NextCloud

CONCLUSION

Personal Health Records (PHRs) stored in cloud storage will enable doctors to view patients' reports and provide necessary treatment. This also helps in remote monitoring and collaborative consultation with doctors across the globe. When PHRs are stored in cloud, security of reports is the well known issue to be handled. This research work discusses about how to securely share medical reports to the professionals working at hospital community. Attribute Based Encryption will help patients to securely share their *PHRs* to a group of users (people) working in a hospital. Next cloud is a local server which act as a CSP, provides the encryption module for the re-encryption and also time privileges for accessing particular file. This will enable each user's access right to be effective in a pre-determined period of time, and enable the CSP to re-encrypt cipher texts automatically, based on its own time. In order to deal with user revocation, Time based PRE was implemented to provide access to PHRs on timely basis by using a proxy server. Our future work is to implement the minimum privilege time in terms of hours, minutes for a particular attribute in an access structure.

REFERENCES

[1] Li, Ming, Shucheng Yu, Kui Ren, and Wenjing Lou , Securing personal health records in cloud computing: Patient-centric and fine-grained data access control in multi-owner settings, In International Conference on Security and Privacy in Communication Systems, Springer Berlin Heidelberg, (2010): (pp. 89-106).

[2] Liu, Jianghua, Xinyi Huang, and Joseph K. Liu,Secure sharing of personal health records in cloud computing: ciphertext-policy attribute-based signcryption, Future Generation Computer Systems 52 (2015): (pp. 67-76).

[3] Thilakanathan, Danan, Shiping Chen, Surya Nepal, Rafael Calvo, and Leila Alem, A platform for secure monitoring and sharing of generic health data in the Cloud , Future Generation Computer Systems 35 (2014): (pp. 102-113).

[4] Liu, Qin, Guojun Wang and Jie Wu, Time-based proxy re-encryption scheme for secure data sharing in a cloud environment, Information Sciences 258 (2014): (pp. 355-370).

[5] Liang, Kaitai, Man Ho Au, Joseph K. Liu, Willy Susilo, Duncan S. Wong, Guomin Yang, Yong Yu, and Anjia Yang, A secure and efficient cipher text-policy attribute-based proxy re-encryption for cloud data sharing, Future Generation Computer Systems 52 (2015): (pp. 95-108).

[6] Ateniese G, Fu K, Green M, Hohenberger S, Improved proxy re-encryption schemes with applications to secure distributed storage, ACM Transactions on Information and System Security (TISSEC) 9(1), (2014): (pp. 1-30) .

[7] Priya Sharma , Security of Key in Cloud Using Cryptography, International Journal of Advanced Research in Computer Science and Software Engineering 5(3), March - (2015): (pp. 823-826).

[8] Xu, Dong, Cloud computing: An emerging technology, In Computer Design and Applications (ICCDA), 2010 International Conference on, vol. 1, pp. V1-100. IEEE, 2010.

[9] Kuo, Mu-Hsing, Opportunities and challenges of cloud computing to improve health care services, Journal of medical Internet research 13, no. 3 (2011): e67.

[10] Delfs, Hans and Helmut Knebl, Symmetric-Key Cryptography, In Introduction to Cryptography, pp. 11-48. Springer Berlin Heidelberg, 2015.

[11] Salomaa, Arto, Public-key cryptography, Springer Science and Business Media, 2013.

[12] JohnBethencourt, AmitSahai, BrentWaters, http://acsc.cs.utexas.edu/cpabe/, University of Texas, (last accessed date 21 August 2017)

[13] Next Cloud: https://nextcloud.com/ (last accessed date 23 August 2017)