**Key-Aggregate Cryptosystem For Scalable Data Sharing In**

**Cloud Storage**

**ABSTRACT:**

Data sharing is an important functionality in cloud storage. In this article, we show how to securely, efficiently, and flexibly share data with others in cloud storage. We describe new public-key cryptosystems which produce constant-size ciphertexts such that efficient delegation of decryption rights for any set of ciphertexts are possible. The novelty is that one can aggregate any set of secret keys and make them as compact as a single key, but encompassing the power of all the keys being aggregated. In other words, the secret key holder can release a constant-size aggregate key for flexible choices of ciphertext set in cloud storage, but the other encrypted files outside the set remain confidential. This compact aggregate key can be conveniently sent to others or be stored in a smart card with very limited secure storage. We provide formal security analysis of our schemes in the standard model. We also describe other application of our schemes. In particular, our schemes give the first public-key patient-controlled encryption for flexible hierarchy, which was yet to be known.

**EXISTING SYSTEM:**

Considering data privacy, a traditional way to ensure it is to rely on the server to enforce the access control after authentication, which means any unexpected privilege escalation will expose all data. In a shared-tenancy cloud computing environment, things become even worse. Data from different clients can be hosted on separate virtual machines (VMs) but reside on a single physical machine. Data in a target VM could be stolen by instantiating another VM co-resident with the target one. Regarding availability of files, there are a series of cryptographic schemes which go as far as allowing a third-party auditor to check the availability of files on behalf of the data owner without leaking anything about the data, or without compromising the data owner’s anonymity. Likewise, cloud users probably will not hold the strong belief that the cloud server is doing a good job in terms of confidentiality. A cryptographic solution, with proven security relied on number-theoretic assumptions is more desirable, whenever the user is not perfectly happy with trusting the security of the VM or the honesty of the technical staff. These users are motivated to encrypt their data with their own keys before uploading them to the server.

**DISADVANTAGES OF EXISTING SYSTEM:**

Unexpected privilege escalation will expose all

It is not efficient.

Shared data will not be secure.

**PROPOSED SYSTEM:**

The best solution for the above problem is that Alice encrypts files with distinct public-keys, but only sends Bob a single (constant-size) decryption key. Since the decryption key should be sent via a secure channel and kept secret, small key size is always desirable. For example, we cannot expect large storage for decryption keys in the resource-constraint devices like smart phones, smart cards or wireless sensor nodes. Especially, these secret keys are usually stored in the tamper-proof memory, which is relatively expensive. The present research efforts mainly focus on minimizing the communication requirements (such as bandwidth, rounds of communication) like aggregate signature. However, not much has been done about the key itself.

**ADVANTAGES OF PROPOSED SYSTEM:**

It is more secure.

Decryption key should be sent via a secure channel and kept secret.

It is an efficient public-key encryption scheme which supports flexible delegation.

**Implementation Modules:**

**1. Searchable encryption**

**2. Data Group sharing,**

**3.data privacy**

**4.cloud storage**

**5.Access control**

**6.Encrypted database model**

**1. Searchable encryption**

Generally speaking, searchable encryption schemes fall into two categories, i.e., searchable symmetric encryption (SSE) and public key encryption with keyword search (PEKS). Both SSE and PEKS can described as the tuple SE= (Setup, Encrypt, TrapdoorTest): Setup(1 ): this algorithm is run by the owner set up the scheme. It takes as input a security parameter 1 , and outputs the necessary keys. Encrypt(k; m): this algorithm is run by the owner to encrypt the data and generate its keyworciphertexts. It takes as input the data m, ownernecessary keys including searchable encryption key k and data encryption key, outputs data ciphertext and keyword ciphertexts C m Trpdr(k; w): this algorithm is run by a user generate a trapdoor Tr for a keyword w using key k. Test(Tr, C ): this algorithm is run by the cloud server to perform a keyword search over encrypted data. It takes as input trapdoor Tr and the keyword ciphertexts C m m . , outputs whether C contains the speciﬁed keyword.

the problem of searching on data that is encrypted using a public key system. Consider user Bob who sends email to user Alice encrypted under Alice's public key. An email gateway wants to test whether the email contains the keyword "urgent" so that it could route the email accordingly. Alice, on the other hand does not wish to give the gateway the ability to decrypt all her messages. We define and construct a mechanism that enables Alice to provide a key to the gateway that enables the gateway to test whether the word "urgent" is a keyword in the email without learning anything else about the email. We refer to this mechanism as Public Key Encryption with keyword Search. As another example, consider a mail server that stores various messages publicly encrypted for Alice by others. Using our mechanism Alice can send the mail server a key that will enable the server to identify all messages containing some specific keyword, but learn nothing else. We define the concept of public key encryption with keyword search and give several constructions.

**Access control :**

Access control a is way of limiting access to a system or to physical or virtual resources. In computing, access control is a process by which users are granted access and certain privileges to systems, resources or information .In access control systems, users must present credentials before they can be granted access. In physical systems, these credentials may come in many forms, but credentials that can't be transferred provide the most security. The management of admission to system and network resources. It grants authenticated users access to specific resources based on access policies and the permission level assigned to the user or user group. Access control often includes authentication, which proves the identity of the user or client machine attempting to access the files. the MuteDB models and schemes for combining encryption and key management to support data confidentiality and isolation in cloud data bases. After the presentation of the models related to access control in plaintext and encrypted databases, we describe how MuteDB transforms an access control matrix for the plaintext model to a matrix suitable for the encrypted database, and how it generates user credentials. Let R be the set of resources that represent plain text tenant data, S the set of plaintext database structures, E the set of encrypted tenant data, U the set of users, and K the set of encryption keys. We define A as the access control matrix where, for each user u P U and for each structure s P S, there exists a binary authorization rule a that defines whether an access to s by u is denied or allowed .

**Encrypted database model:**

Database encryption is the process of converting data, within a database, In plaintext format into meaningless cipher text by the means of a suitable algorithm. Database decryption is converting the meaningless cipher text into the original information using keys generated by the encryption angorithms. Database encryption be provided at the file or column level. Encryption of a database is costly and requires more storage space than the original data. The steps in encrypting a database are: Determine the criticality of the need for encryption, Determine what data needs to be encrypted, Determine which algorithms best suit the encryption standard, Determine how the keys will be managed. Numerous algorithms are used for encryption. These algorithms generate keys related to the encrypted data. These keys set a link between the encryption and decryption procedures. The encrypted data can be decrypted only by using these keys.

Encrypted data are contained in encrypted tables stored in cloud database servers. For each plaintext table, the MuteDB DBA client generates the corresponding encrypted table and a unique encryption key. The name of the encrypted table is computed by encrypting the name of the plaintext table through that key. The encryption algorithm used for encrypting the table names is a standard AES algorithm in a deterministic mode (e.g., CBC with constant initialization vector). In such a way, only the users that know the plaintext table name and the corresponding encryption key are able to compute the name of the encrypted table. The deterministic scheme is preferred because it allows a correspondence between plaintext and encrypted tables and improves the efficiency of the query translation process.

**Data Group sharing,**

server can use this aggregate trapdoor and some public information to perform keyword search and return the result to Bob. Therefore, in KASE, the delegation of keyword search right can be achieved by sharing the single aggregate key. We note that the delegation of decryption rights can be achieved using the key-aggregate encryption approach recently proposed in [4], but it remains an open problem to delegate the keyword search rights together with the decryption rights, which is the subject topic of this paper. To summarize, the problem of constructing a KASE .

**Cloud Data privacy**

Cloud Data privacy issues are among the key concerns for companies moving to the cloud. In most countries and in most industries, data privacy regulations apply whenever personally identifiable information (PII) is collected and stored. When this information resides in the cloud, it presents a unique challenge because cloud computing resources are distributed, making it difficult to know where data is located and who has access at any given time. In addition to the cloud data privacy laws highlighted below, many enterprises need to also adhere to series

**cloud storage**

Cloud storage is a model of data storage where the digital data is stored in logical pools, the physical storage spans multiple servers (and often locations), and the physical environment is typically owned and managed by a [hosting](http://en.wikipedia.org/wiki/Internet_hosting_service) company. These cloud storage providers are responsible for keeping the data available and accessible, and the physical environment protected and running. People and organizations buy or lease storage capacity from the providers to store user, organization, or [application data](http://en.wikipedia.org/wiki/Application_Data).Cloud storage services may be accessed through a co-located cloud computer service, a [web service](http://en.wikipedia.org/wiki/Web_service) [application programming interface](http://en.wikipedia.org/wiki/Application_programming_interface) (API) or by applications that utilize the API, such as cloud desktop storage, a [cloud storage gateway](http://en.wikipedia.org/wiki/Cloud_storage_gateway) or [Web](http://en.wikipedia.org/wiki/World_Wide_Web)-based [content management systems](http://en.wikipedia.org/wiki/Content_management_systems).

# System Configuration:

**HARDWARE REQUIREMENTS:**

Hardware - Pentium

Speed - 1.1 GHz

RAM - 1GB

Hard Disk - 20 GB

Key Board - Standard Windows Keyboard

Mouse - Two or Three Button Mouse

Monitor - SVGA

**SOFTWARE REQUIREMENTS:**

Operating System : Windows

Technology : Java and J2EE

Web Technologies : Html, JavaScript, CSS

IDE : My Eclipse

Web Server : Tomcat

Tool kit : Android Phone

Database : My SQL

Java Version : J2SDK1.5