Secure File Sharing on Cloud

Attribute-Based Access Control and Keyword Search over Encrypted Data

Rachita Gupta, Supriya, Qiuxiang Dong
School of Computing, Informatics and Decision Systems Engineering
Arizona State University, Tempe, AZ, USA
{rgupta36, sashokk2, qiuxiang.dong}@asu.edu

***Abstract*— Cloud computing is a promising new technology where computing resources are provided as a service to users over the internet. Since cloud storage provides greater flexibility and availability at a much lower investment, more and more users are attracted to outsource their data on the cloud servers. As more sensitive data are outsourced on cloud server environment, the security of the data may be compromised. To combat unsolicited accesses from malicious insiders (e.g., cloud server administrator) and outsiders (e.g., hacker), these sensitive data have to be encrypted before outsourcing. Although data confidentiality could be protected by traditional encryption schemes (e.g., AES, RSA), it is difficult to implement data sharing and searches over the encrypted data.**

This project aims to implement secure file sharing in cloud storage through utilizing some recently proposed encryption schemes. Firstly, the data owners encrypt their files by Attribute-Based Encryption (ABE) scheme so that only authorized data users could decrypt these files. Second, Attribute-Based Keyword Searchable (ABKS) encryption scheme is implemented, thus enabling the authorized data users to search over the encrypted files to find their desired files. Finally, some parallel processing approaches will be employed in this project to enhance search efficiency, when there are huge number of files stored on the cloud servers.

Keywords*— *Cloud Computing, Data Leakage, Secure Cloud Storage, Attribute-Based Encryption (ABE), Attribute-Based Keyword Search (ABKS), Access Control, Keyword Search

INTRODUCTION

Cloud computing has emerged as an innovative technology which has changed the way we do computing by providing all services and applications through the internet. It is an ecosystem which enables on-demand access to pool resources with greater flexibility and availability at a much lower investment cost by allowing the users to pay only for the resources and services they use. The main concept of cloud computing is not to own any hardware or software but to avail the computational resources, storage databases or any other services at a minimal cost to the cloud provider. Cloud infrastructure incorporates the five characteristics which are on-demand service, broad network access, resource pooling, rapid elasticity and measured service. Adopting the cloud for enterprises has empowered businesses by ensuring lesser cost, faster deployment, lesser acquisition of physical assets and much greater customer satisfaction.

Since many IT enterprises are hosting their services, applications and data on cloud, there is a growing concern regarding the privacy of the sensitive data being compromised. The most obvious solution to this problem is to apply some encryption techniques before uploading or sharing the data on cloud. But then how the uploaded encrypted data would be used on cloud is a new challenging problem that needs to be addressed. This project involves the search over encrypted data but most of the techniques selectively search the data only at a coarse-grained level i.e., by sharing the private key with the other authorized users. But this is not a feasible solution in the multi-owner and multi-user cloud based file storage system where owner either has to act as an intermediary and decrypt all the files it wants to share with some other parties or give them its private decryption key, giving access to all its data. Hence, to

enable multiple owners to encrypt and upload their data files to the cloud storage and make them searchable by other users, a new cryptosystem "fine grained search authorization" was proposed that only allows the authorized users to search over the encrypted data [1].

"Fine grained" implies that the search authorization is controlled at the granularity of each file and exploits the **attribute-based encryption (ABE)** technique introduced by Sahai and Waters [2]. In attribute based encryption the data owner encrypts the index of each file with a self defined access policy, which dictates that which type of user can search this index [1]. An **access control policy** would define the type of users who would have permission to access the documents. For instance, in an academic institute only the professor of CS department or the TA of that course would have access to the grade sheet for a particular course. This can be expressed in form of a policy as below. Hence only the users satisfying this attribute predicate would be able to access the grade sheet.

$$((Professor \wedge CS\ dept.) \vee (CS\ student \wedge course\ TA))$$

The keywords and attributes are defined separately in the report. **Keywords** are the actual content of the files and **attributes** are the properties of the users[1]. The owner of data creates the index of all the keywords in the data file but encrypts the index with the access control policy structure based on the attributes of the authorized users [1]. **Attribute-Based Keyword Searchable encryption scheme** (ABKS) is a data-owner-enforced search authorization system that only users who meet the owner-defined access policy can retrieve the valid search results [1]. Since the searchable scope is very large in the cloud environment and cryptographic search has to be done with linear time complexity, so some parallel computing approaches are used to enhance search efficiency.

The outcome of this project would be the implementation of attribute-based keyword search, and attribute-based data access control using parallel processing. **Table 1** describes the various tasks taken up by the following group members in the defined time frame to the project completion.

System Models

System Model

There are three kinds of servers in our system (ref. **Figure 1**). The first is key generation server (namely Trusted Authority, **TA** for short), which generates private keys for users. The second is **metadata server**, which is responsible for managing the metadata and also process data uploading, data search requests from data contributors (owners) and data readers (users) respectively. The third is **data storage server**, which stores the files. There are multiple data contributors (or owners) and multiple data readers. The design goals of this system are as follows.

- Data access control: Enable data-owner-enforced data access control, i.e., only users who meet the owner-defined access policy can decrypt the files.
- Authorized Keyword Search: Enable data-owner-enforced search authorization, i.e., only users who meet the owner-defined access policy can obtain the valid search results.
- Multiple Data Owners and Data Readers: accommodate multiple data owners and data readers. Each reader is able to read and search over the encrypted data contributed by multiple data owners.
- Security Goals: Protect both the contents and the keywords privacy.
- Efficiency Goals: Search operations could be implemented by the cloud server efficiently.

The workflow of our system is presented in **Figure 3**. The detailed description is as follows.

- **System Setup**: At the beginning, each user obtains her/his private key from TA.

Task assigned			Timeline	Group Member
Team Formation			25th Aug - 28th Aug 2016	
Idea and Design development Time and task allocation			29th Aug - 8th Sep 2016	Rachita Gupta, Supriya, Qiuxiang Dong
Project Proposal Report Writing			9th Sep - 14th Sep 2016	Rachita Gupta, Supriya, Qiuxiang Dong
Implementation	Data structure design and Environment Setup		15th Sep - 17th Sep 2016	Rachita Gupta, Supriya, Qiuxiang Dong
	Algorithm design and implementation	Client Server Model	18th Sep - 25th Sep 2016	Supriya
		Cryptographic Algorithm	18th Sep - 25th Sep 2016	Qiuxiang Dong
		Parallel processing	18th Sep - 25th Sep 2016	Rachita
		System Integration	26th Sep - 29th Sep 2016	Rachita Gupta, Supriya, Qiuxiang Dong
Testing			30th Sep - 1st Oct 2016	Rachita Gupta, Supriya, Qiuxiang Dong
Second Report writing			2nd Oct - 5th Oct 2016	Rachita Gupta, Supriya, Qiuxiang Dong
Testing and troubleshooting			6th Oct - 15th Oct 2016	Rachita Gupta, Supriya, Qiuxiang Dong
Final Report writing			16th Oct - 26th Oct 2016	Rachita Gupta, Supriya, Qiuxiang Dong

Table 1: Project Task allocation and Timeline

- **File Upload**: When the data contributor uploads a file, the following operations are performed. First, encrypt the file f with a symmetric encryption algorithm (e.g., AES). Second, encrypt the AES key (K) with an Attribute-Based Encryption algorithm (ABE for short). Third, generate the Keyword List (KL) for the file and encrypt the keywords with the keyword encryption algorithm of the Attribute-Based Keyword Searchable Encryption scheme (ABKS for short). Finally, the user sends the ciphertext of the f, K, and KL to the metadata server.
  Upon receiving the file uploading request from the data contributor, the metadata server will store the following item (ref. **Figure 2**) for this request and store the file on the storage server(s).

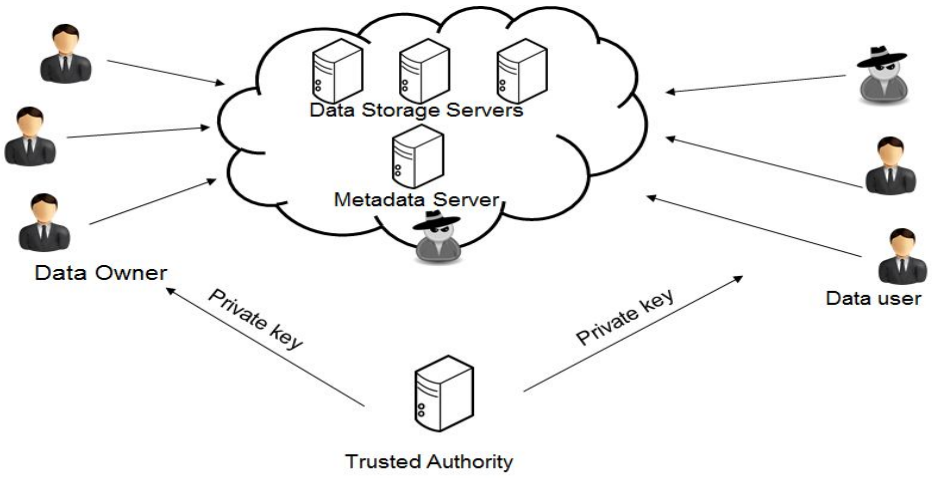

Figure 1: System architecture

- **Token Generation**: When a data reader wants to search his/her interested files, he/she will generate a search ToKen (TK) for his/her interested keyword (e.g., kw) with his/her private key and send the token to the metadata server.

- **Search**: Upon receiving the search request, the metadata server runs the search algorithm of the ABKS scheme to check whether a file contains the keyword (i.e., kw is included in some KL). On the one hand, randomness in included in the ciphertexts, so the complexity of the search will be linear. On the other hand, there are a lot of files. Therefore, it will be time-consuming to perform keyword search over encrypted data. To solve this problem, some parallel computing technology will be used on metadata server. If some file(s) contain the interested keyword, the metadata server will send back the corresponding ABE(K) and encrypted files to the data reader.

- **File Decryption**: Upon receiving the search results, the data reader decrypts ABE(K) and obtains K, which can be used to decrypt the files.

Software

PBC [8] and OpenSSL [9] library will be utilized to implement the cryptographic algorithms. On the file system, to search a particular file, Apache spark will be used. For the communication between the metadata server and the user, the project uses client- server architecture.

Security Model (optional)

The **TA** is assumed to be fully trusted. The cloud servers (including the metadata server and the data storage servers) are semi-honest, which means they will run the pre-defined algorithms or protocols honestly but they will try to find private information based on background knowledge or other information they could get. The data readers are assumed to be malicious. They want to access the files of which they are actually not authorized to read. They might collude with each other.

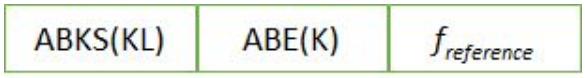

Figure 2: Storage format on metadata server

Key Generation

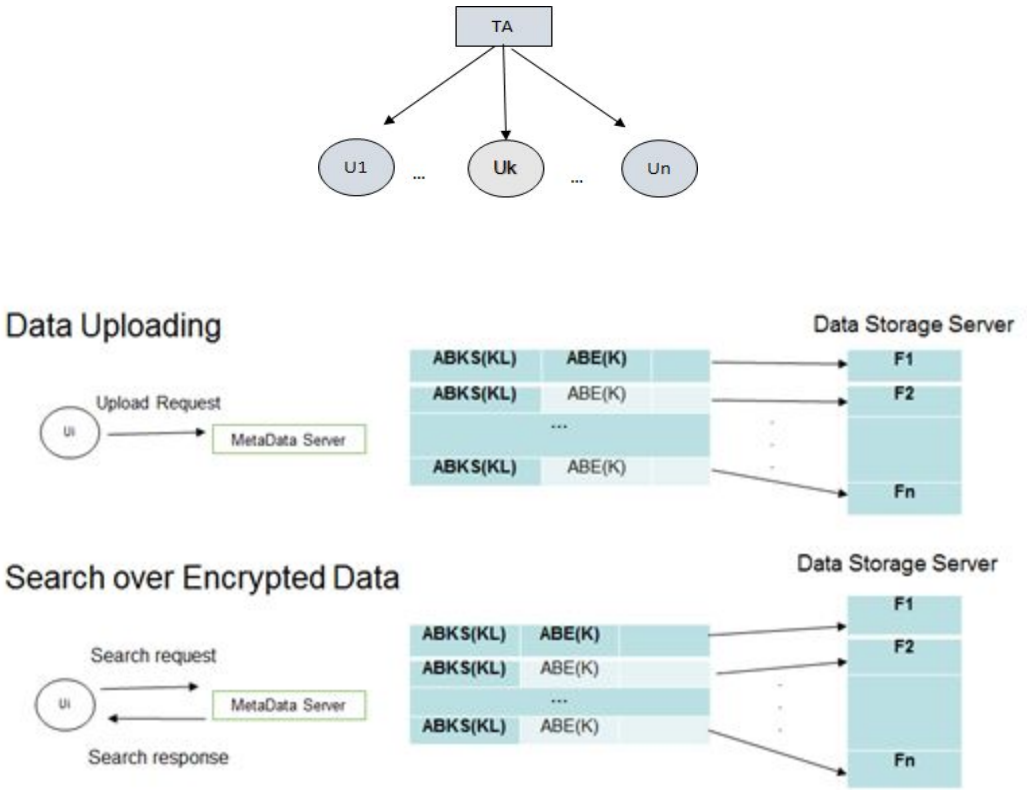

Figure 3: Message sharing between entities

Project Description

With the advent of cloud computing, a new computing paradigm, more and more users and enterprises are shifting their data to cloud storage servers. As more sensitive data are outsourced on cloud server environment the new challenging problem is that confidentiality or privacy of the data may be compromised, and access control over the data becomes difficult to handle, because the commercially operated clouds are likely to be outside the trusted domain of the users. The obvious solution to ensure the confidentiality and desired access control of data is to encrypt the data and share the private key with the authorized users. However this can only implement coarse-grained level access control and is an unscalable solution involving lot of computation for key distribution and management on the client side. This project explores the scenario of multiple owners uploading their encrypted data files to the cloud storage and make them searchable to multiple users by "fined grained search authorization", which is an access control policy method that makes the data files accessible only to the authorized users without revealing the actual data content.

Attribute-Based Keyword Searchable Encryption scheme (ABKS), a data-owner-enforced authorization search system, is used that only allows users who meet the owner-defined access policy to search over the encrypted data [1]. Some parallel processing approaches will be employed to achieve faster keyword search over encrypted data in the huge searchable space of cloud environment.

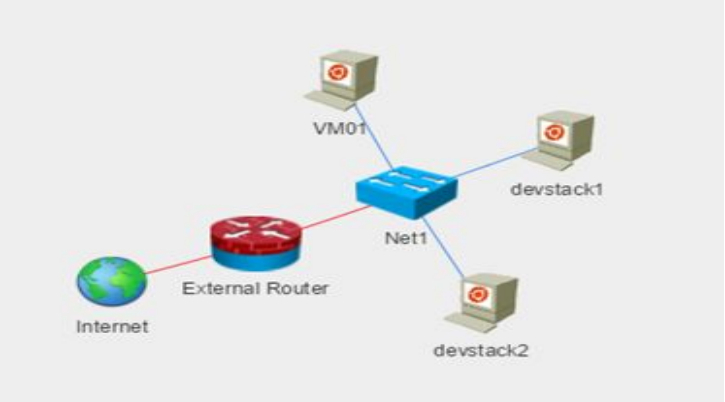

Figure 4: This project will use the above architecture. This architecture is provided by the thoth lab.

RESPONSIBILITY	MACHINE
Trusted Authority(TA)	devstack 1
Metadata Server	devstack2
Data Storage Server	devstack 1/2
user	VM01

Table 2: Server Machines in system architecture and their responsibilities

Project Overview

Task 1 : Environment Setup

Figure 4 shows the environment setup architecture in thoth lab. The different types of servers needed in this project are key generation server (i.e., TA), metadata server, and data storage server. Table 2 shows the various machines in the system and their various responsibilities.

1. Key Generation Server: This server is responsible for generating private keys for all the users. After system setup it could be offline until when new users join the system.

2. *Metadata server:* This server stores metadata of users' files, including searchable keywords ciphertexts, encrypted key as well as the references to the files stored on the data storage server. It processes the data uploading request from data owners and data search requests from data readers.

3. Data storage Server: These server stores all the files on the cloud storage. Number of data storage server may vary depending on the requirement. This project uses two data storage servers. Some cloud storage management platform will be installed there. The physical machine or VM should provide large storage space, or else open cloud computing platform could be used.

Task 2: Data structure design

Design data structure of each file's index stored on the metadata server. Design data structure used in the ABE and ABKS algorithm.

Task 3: Algorithm design and implementation

Design and implement ABE and ABKS algorithms. Design and implement parallel searching algorithm with some parallel processing approach.

TEAM MEMBER	TASKS	PERCENTILE
Rachita Gupta	Parallel Computations on the metadata using apache spark. This will help in faster search by the user on the data stores.	33.3%
Supriya (Team Leader)	Setting up client server communication between the user and the servers.	33.3%
Qiuxiang Dong	Implement file encryption, attribute-based encryption and the attribute-based keyword encryption schemes.	33.3%

Table 3: Task distribution

Task 4: System development

Integrate all the algorithms and components into a whole system. Web service development will be needed in this part.

Task 5: System Testing

Use some real-world dataset to test whether the system could work effectively and efficiently.

Project Task Allocation

Project will be completed by 3 members of the team. Table 3 provides details of the task allocation of each member in the project.Once individual tasks are completed all the team members will integrate the entire project to obtain full functionality.

Deliverables

This project will create a software package for the user. This software helps in interacting with the TA to obtain private key. Also, it will provide encryption algorithms, which will use the user's private key and encrypt the files, symmetric encryption key and keyword list. It will also create service for the cloud system which will provide users secure file sharing on cloud.

Project Timeline

Figure 5 shows the Gantt chart for this project depicting the detailed task-time distribution.

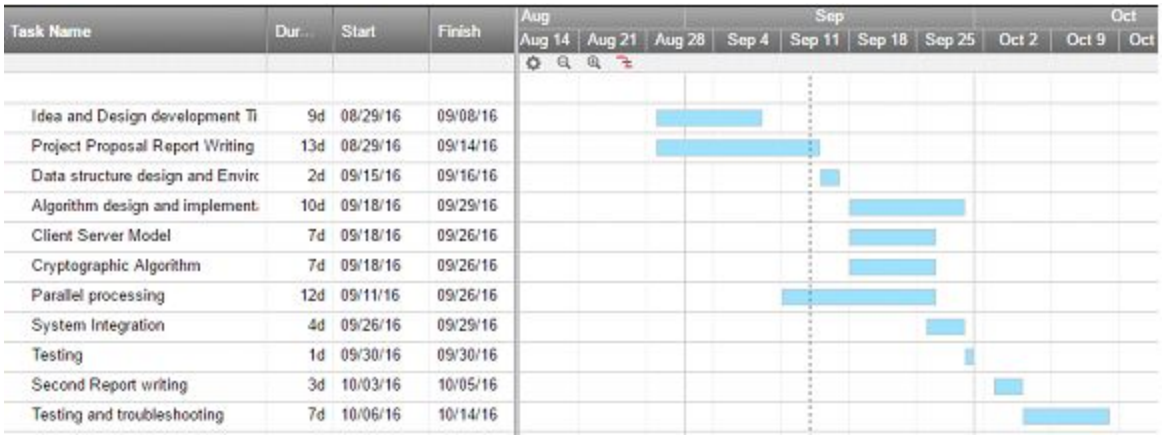

Figure 5: Gantt Chart for the project

Risk Management of the project

Table 4 shows the various probable risks in the project and the possible mitigation risk plans to avoid them.

Probable Risks	Risk Mitigation
Cloud Computing is a new domain for all the members. Complete knowledge of all the tools and technology is difficult. Having time as a constraint, research may consume lots of time.	All the members will sit together and discuss with the professor about each and everything. Devoting proper and efficient time in research in correct direction as guided by the professor.
Technology is changing at a very fast pace. Being aware of everything is not an easy task for anyone. This could be an issue as choice of softwares, tools and technology may not be updates or correct.	In depth knowledge of the domain is required to solve this issue. Lots of research and paper reading reading is to be done for this.
Project planning if not done properly, estimations made without right calculations can lead to technical risks. Main issue could be wrong estimation of schedule and effort.	To solve this issue all the team members will have regular meetings. These will involve discussions on status update and project forecast. This will allow us in early mitigation of the risk.

Table 4: Risks Management Plan

Conclusion

This project aims to implement secure file sharing on cloud. The data owners could implement access control and search capability authorization, so that only authorized users could search or decrypt the encrypted files. This project uses some existing algorithms. These current schemes do not provide complete solution for data sharing in cloud computing, e.g., they do not provide effective and efficient user revocation mechanism, which is very important in real-world applications. Therefore, in the future, based on the established demo system, user revocation functionality might be added. In addition, in this project, the stored data are files, therefore, keywords search will be implemented. With the popularity of internet of things (IoT), more and more private sensor data might be stored on the cloud servers, thus making searches over numerical data, e.g., range search, important. Finally, how to implement heavy cryptographic algorithms on resource-constrained devices, e.g., smartphones, is also an interesting future work. In our developed system each user's attributes are revealed and in the future, work on protecting this information could be done.

Acknowledgment

Thanks Dr. Huang, the instructor of this course for discussions and guiding us through project idea selection. We also thank Ankur, the TA for his support in idea formulation and helping us with the project environment setup.

References

[1] Sun, Wenhai, et al. "Protecting your right: Attribute-based keyword search with fine-grained owner-enforced search authorization in the cloud." *IEEE INFOCOM 2014-IEEE Conference on Computer Communications*. IEEE, 2014

[2] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in Proc. of CCS. ACM, 2006, pp. 89–98.

[3] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in Proc. of S&P. IEEE, 2007, pp. 321–334.

[4] L. Cheung and C. Newport, "Provably secure ciphertext policy abe," in Proc. of CCS. ACM, 2007, pp. 456–465.

[5] S. Yu, C. Wang, K. Ren, and W. Lou, “Attribute based data sharing with attribute revocation,” in Proc. of ASIACCS. ACM, 2010, pp. 261–270.

[6] S. Yu, C. Wang, K. Ren, and W. Lou, “Achieving secure, scalable, and fine-grained data access control in cloud computing,” in Proc. of INFOCOM. IEEE, 2010, pp. 1–9.

[7] http://www.cs.cityu.edu.hk/~congwang/research.html

[8] https://crypto.stanford.edu/pbc/download.html

[9]https://www.openssl.org