# Sprite: Security and Privacy for Cloud Storage

Kyaw Thu Win

***Abstract*—**Some organizations give storage services using cloud computing technology. The users of such services kept their data at third party places. In this case, security and privacy concerns for cloud user data arise. In order to solve such concerns, this paper describes the security and privacy model for cloud storage. The cloud user data are encrypted using a public-key cryptosystem and the decryption key is stored at key manager in secure fashion and the key is only accessible between the key users or owner and key manager via key exchange protocol. Despite this protocol, the cloud user data, both key and data file, are also controlled by access policies. The cryptographic key and processes are done at client location using agent-based technology and Rivest-Shamir-Adleman (RSA) algorithm.

***Keywords*—**Cloud Security, Key exchange protocol, Policy-based access control, RSA algorithm

## I. Introduction

THE cloud computing technology become popular in this decade. Many organization moves from traditional technology to provide better services to their user. The US National Institute of Standards of Technology (NIST) defines cloud computing as "a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g. networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction" [3]. Users outsource their data to the third-party area, cloud storage providers, (e.g., Amazon S3 [8] and OpenStack [9]). The individuals users can also get advantages of storing the data in the cloud in the case of moving their personal data files (e.g., photos, music file, etc.) to the cloud and make effective use of space in their mobile devices (e.g., laptops, smartphones). As data owners store their data on external servers, there have been increasing demands and concerns for data confidentiality, authentication and access control [4]. In order to tackle this, cryptographic approaches can use. However, the maintaining of key becomes another security concerns. This paper described a security and privacy system for cloud storage services. This system, namely Sprite, contains three components: key manager, cloud user, and cloud storage provider. Key manager only keeps the encrypted data at his storage not having any key generation and key sharing with the cloud user to perform cryptographic techniques. Although the key manager only stores encrypted key, the key needs to protect from improperly access. Therefore, this system uses a key exchange protocol and policy-based file access protection scheme. The key exchange protocol is only used between the cloud user and key manager in uploading and downloading from key manager. Policy-based file access protection scheme is used in accessing of both key file and data file. The user data file and key file are encrypted at client location using RSA-based key program. The contribution of this paper as follows: This paper described a policy-based file (both key and data) access protection scheme that guarantees the access rights to key file and data file based on the specified privacy policy. This paper proposes the key exchange protocol that is used between the client and key manager. This paper proposes the key program, which is based on RSA algorithm.

**Roadmap.** The remainder of the paper proceeds as follows. Section II describes the overview of the theoretical background. In Section III, the architecture and overall process of the proposed system is explained. Section IV describes the key program. In Section V, the file protection based on policy is discussed. In Section VI, the key exchange protocol is described. Section VII describes experimental result. Related works are described in Section VIII. This paper concludes in Section IX.

Kyaw Thu Win is with the University of Compute Studies, Mandalay, the Republic of the Union of Myanmar (author's email: ktwcumdy2009@gmail.com).

## II. Theoretical Overview

**Basic cloud consumption**: Cloud computing is an information-processing model in which centrally administered computing capabilities are delivered as services, on an as-needed basis, across the network to a variety of user-facing devices [1]. In cloud computing environment, there are three services provided by cloud namely Infrastructure-as-a-Service (IaaS), Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS). Nowadays many internet applications such as Google, Facebook, and Twitter move to cloud computing environment. The key finite attributes of cloud are on-demand availability, ease of provisioning, dynamic and virtually infinite scalability. So, user does not need to have any worries about the storage space, needed file system.

**Cloud Security:** As cloud computing becomes a more important model for enterprise IT, the cloud will necessarily be a more critical part of the overall security infrastructure [2]**.** Since user data are storing at third party places, the user can think that their data is not secure. So, there may need to have security in cloud computing environment. The biggest hurdle to the adoption of cloud storage (and cloud computing in general) is concern over: confidentiality - the cloud storage provider does not learn any information about customer data

and integrity - any unauthorized modification of customer data by the cloud storage provider can be detected by the customer of data [7]. So, cryptographic techniques can overcome this security issues. Despite protection of outsourced data by applying cryptographic encryption onto sensitive data with a set of encryption keys, the maintaining and protecting such encryption keys will be another security problem. In this paper, RSA public key encryption scheme is used in encryption process of user data and key file at the client side.

**RSA algorithm**: A public-key cryptosystem is a one-way authentication system. If user *A* wishes to send a message *M* to user *B*, he "deciphers" it in his secret deciphering key and sends decrypted message, $D_A(M)$. When user *B* receives it, he can read it, and be assured of its authenticity by "enciphering" it with user *A*'s public enciphering key, $E_A$ [5]. The most successful algorithm that has been proposed for public-key cryptography is Rivest-Shamir-Adleman (RSA) scheme. The RSA scheme is a block cipher in which the plaintext and ciphertext are integers between 0 and n - 1 for some n. A typical size for n is less than $2^{1024}$. RSA algorithm works as follows [6]:

1. Generate two large prime numbers, *p* and *q*.
2. Calculate modulus (n) :

$$n = p \times q \tag{1}$$

3. Calculate:

$$\phi(n) = (p-1) \times (q-1) \tag{2}$$

, where ϕ (n) is Euler totient function.

4. Select *e* such that *e* is relatively prime to ϕ (*n*).
5. Calculate the decryption exponent d:

$$d \equiv e^{-1} \pmod{\phi(n)} \tag{3}$$

6. The public key pair is (e, n)
7. The private key pair is (d, n)
8. The plain text M can now encrypt using:

$$C \equiv M^e \mod n \tag{4}$$

, where C is cipher text.

9. The decrypt message C using:

$$M \equiv C^d \mod n \tag{5}$$

User K encrypts the message with the public key (e, n) of user T using (4). User T is now the only one who can decrypt this message using his own private key (d, n) and (5).

## III. Sprite System

This paper described a security and privacy system for cloud storage. This system is namely as Sprite. This system involves three participants cloud user, key manger and the cloud storage provider as described in Fig. 1. The registered cloud user needs to request the key program from key manager in order to perform the cryptographic process. When the key program runs at client location, the client can encrypt any kind of data file by using generated public key. This system intended to store decryption key more securely, the data key (private key) is also encrypted by the client key program. The decryption key of such private key file is given to cloud user. The encrypted data key is uploaded and stored at key manager via key exchange protocol. The encrypted data file is stored at cloud storage provider. The user must defined some access policies for files before uploading the key file and data file to associated keepers.

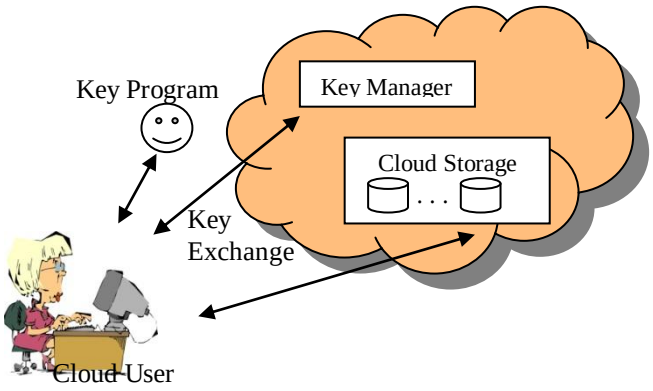


Fig. 1 Sprite system overview

If the cloud user needs to decrypt the outsourced data file at the cloud storage provider, the user must request the encrypted data key from key manager. This can be done by using key exchange protocol. If the user successfully receives the key file from the key manager, she can decrypt it using decryption key of the received key file. Finally, the cloud user has the decryption key of data file and performs the decryption process to have original data file. In the requesting state of data file and key file, both key manager and cloud storage provider protect the file accessible with the defined policies. The file will block if the user revokes a policy. The blocking time may vary depends on revoked policy. Therefore, the key manager does not know the key nature and the decryption process of key cannot do. The cloud storage of Sprite system is private cloud storage to be used in this system.

## IV. Key Program

The key program is used to perform cryptographic processes at client location. This program works two parts such as encryption and decryption of key and data file. Whenever the cloud user needs to encrypt the desired data file, she needs to request this program from key manager. Key manager will sends this key program to the requested location. The program will start automatically when it reaches at the user location. The user can now perform the encryption process by performing the encryption procedure as describes in Fig. 2. The cloud user firstly needs to select desired file to decrypt and then generate the private and public key pair. When the client starts encryption process, the key program will encrypt the data file and finally the decryption key is automatically encrypted by generating another cryptographic key. The key program will output three files. The first one is encrypted data file. The second file is encrypted key file and the last one is decryption key file which to be used to decrypt the encrypted key file. The user

can now upload the encrypted key file by filling ups the required policy value. The key uploading process is performed by using the key exchange protocol as described later in this paper.

RSAEncryption (input_file, output_file, N)
input_file: A File that needed to encrypt
output_file: A File that encrypted data needs to write
N: the max key length (typically 1024bits)

❖ Generate Key:
p← a big prime number generate randomly with N/2bit length
q← a big prime number generate randomly with N/2bit length
$phi_n$←(p-1).(q-1)
n← p.q
e←65537;
d←e InverMod $phi_n$;

❖ Encrypt input file:
keysize←bit length of n
plaintextSize←min((keysize-1)/8,256)
ciphertextSize←1+ (keysize-1)/8
dataSize← plaintextBlockSize;
while(dataSize>0)
M←read inputfile content by plaintextBlockSize
ciphertext←$M^e$ mod n
write ciphertext to ouput_file in byte

❖ Writing Private Key File:
write private key pair (d,n) to a file with extension

Fig. 2 Encryption process of key program

In order to decrypt the desired data file, the cloud user needs to get the encrypted key file and decrypt the downloaded key file. To do so, the user also needs to request this key program from key manager. The user then requests the desired key file by filling up the necessary information. This process will accomplish by using the key exchange protocol. After the user successfully receives the key file, she can now decrypt the key file by inputting the decryption key file that she has. The program will get the private key pair from the input decryption key file and performs the decryption process on downloaded key file. The program will output the data decryption key to the user desired location. Fig. 3 shows the decryption process of this key program.

RSADecryption (input_file, output_file, key_file)
input_file: A File that needed to decrypt
output_file: A File that decrypted data needs to write

❖ Read Key:
n← read key_file
d← read key_file

❖ Decrypt input file:
keysize←bit length of n
plaintextSize←min((keysize-1)/8,256)
CiphertextSize←1+ (keysize-1)/8
dataSize← ciphertextBlockSize;
while (dataSize>0)
M←read inputfile content by cipherextBlockSize
plaintext←$C^d$ mod n
write plaintext to ouput_file in byte

Fig. 3 Decryption process of key program

## V. POLICY-BASED FILE PROTECTION

The cloud user data is encrypted using the key program as described in previous section. This system also protects the file access by defining some access-policy. The defined policies are mixed used of time-based policy and attribute based policy. The cloud user must correct all defined policies. This means that policy conjunctive (AND) rule. If a policy is mismatches or revokes, the cloud user will be blocked. Table I list the policies. The "Owner" policy value is the name of current upload file owner and the permitted user list will be in "Users" policy. 'Access Times' policy is the value of number of access times on each time. This value will be number of times per day, week, month, and year. The "ExpDate" and "ExpTime" policy is the policy for defining the expiration period of upload file. This policy is time-based policy. The policy "Modify" and "Lmodify" are the value of state of file can modify and the list of modification allowance users respectively. The last policy will only need to fill ups if "Modify" policy is set. Whenever user makes a request to a file from key manager and cloud storage provider, these policies must satisfy.

TABLE I
FILE ACCESS POLICIES

| Policies | Description |
|---|---|
| Owner | The name of file owner |
| Users | Permitted user list |
| AccessTimes | Access Time(#day/week/month) |
| ExpDate | File Expired Date |
| ExpTime | File Expired Time |
| Modify | File can overwrite or not |
| Lmodify | List of user who can overwrite file |

## VI. KEY EXCHANGE PROTOCOL

The cloud user data is encrypted using public-key cryptosystem and protected with some access policy. This can say that it is securely enough. Nevertheless, someone can get the key file not knowing the original key access users. If so, the user data can learn in unpredicted way. In order to protect such kind of problem, Sprite system proposes a key exchange protocol. This protocol is only to be used between legitimate key users and key manager in exchanging key. Fig. 4 depicts the proposed key exchange protocol.

The initiator of this protocol is key owner or other legitimate user. The responder is the key manger. When the key file is needed to upload, the initiator initiates this protocol by sending 'Send' message to key manager. The 'Send' message includes the file access policies. The key manager performs some additional tasks when receiving such kind of message. If the user can upload the key file, the key manger sends response message as 'SID' message to initiator. The initiator can now send the key file to key manager by embedding it in 'SEMP' message. The key manager stores the retrieved and stored the embedded key file from the message and informs the user by sending 'SEKEY' message

to initiator.

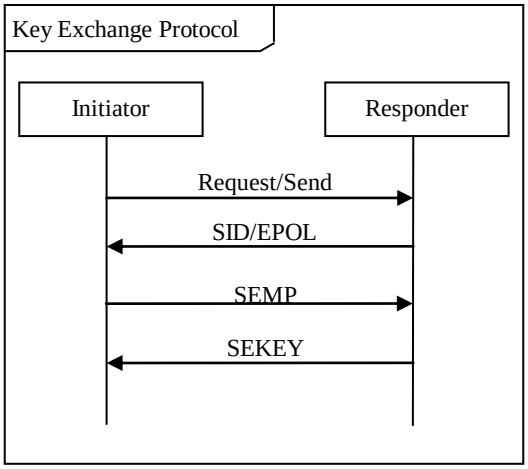


Fig. 4 Sprite key exchange protocol

The initiator needs to send 'Request" message to responder to get the corresponding key file. The 'Request' message will includes the corresponding key file information. The key manager checks the policy of the requested key file for initiator. If the initiator revokes a policy, the key manager will block the initiator for accessing the requested file for a curtained period depending on the revoked policy. If not, the key manager sends the corresponding key file to the initiator by embedding it in 'EPOL' message. The content of message is encoded using some encoding scheme.

## VII. EXPERIMENTAL RESULT

This section describes experimental result of Sprite System. The experimental result will measure on time performance of each process of Sprite system.

Time performance of Sprite system describes the amount of time taken by a piece of operations such as encryption, decryption, uploading, downloading to be finished. The first measurement of time performance is the time of uploading or downloading file ranging size from 1KB to 10MB. Table II shows this kind of measurement.

TABLE II
TIME PERFORMANCE OF UPLOAD/DOWNLOAD PROCESS

| File Size | Upload (File + Policies) | Download (File) |
|---|---|---|
| 1 KB | 0.002s | 0.000s |
| 10 KB | 0.004s | 0.001s |
| 100KB | 0.006s | 0.002s |
| 1 MB | 0.013s | 0.007s |
| 10MB | 0.038s | 0.036s |

Second measurement is the running time of encryption and decryption process of RSA public key cryptographic algorithm. The result is shown in Fig. 5 and 6. The encryption time is a total time of encrypting data file and key file. The decryption time is always slower than encryption time due to the key length. The encryption time and decryption time takes more time upon the file size. The measurement unit is milliseconds.

The third time performance is measured on key exchange process. The running time of key exchange process will divide into two parts such as uploading and downloading key file to and from key manager. The user coordinates with key manager starting from first message of each protocol and ending at final result message arrives. Fig. 7 shows the result of this time performance evaluation.

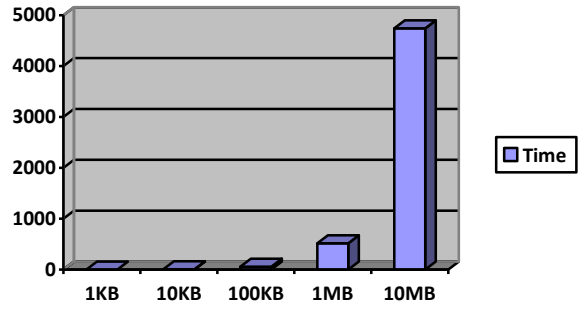


Fig. 5 Time performance on encryption process

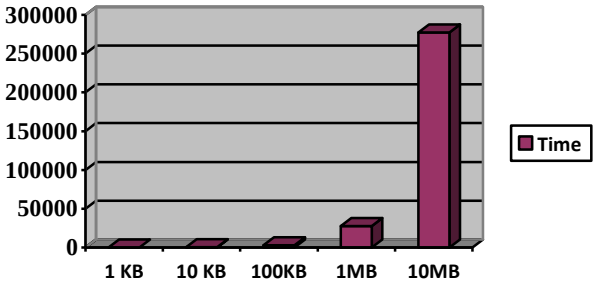


Fig. 6 Time performance on decryption process

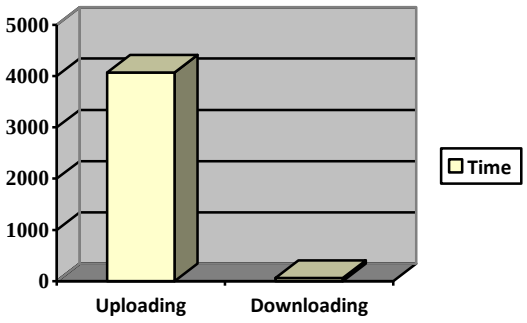


Fig. 7 Time performance on key exchange process

## VIII. RELATED WORK

Y. Tang et al. [3] proposed a cloud storage system called FADE, which aims to provide assured deletion for files that are hosted by today's cloud storage services. In FADE, the files are assuredly deleted when the associated file access policies are revoked and become absolute. FADE does not

guarantee that if the key manager colludes with the storage cloud, then the storage cloud can decrypt the files of data owner.

Cryptographic protection on outsourced data has been described in [2]. S. Kamara et al [2] considered the problem of building a secure cloud storage service on the public clod infrastructure where the service provider is not completely trusted by the customer. They described several architectures that combine recent and non-standard cryptographic primitives in order to achieve such goal. They mentioned in more detail the relevant cryptographic techniques such as searchable encryption, attribute-based encryption.

## IX. CONCLUSION

This paper described a cloud security and privacy model for cloud storage service. In this system, cloud user data was encrypted using a public-key cryptosystem called RSA algorithm to tackle the security and privacy concerns. This system also protected the user files with file access policies. If the user revokes a policy, the user access rights on file will block for specified time period of a revoked policy. The system also proposed a key exchange protocol to better have file access protection. All key uploading or downloading processes are performed via this protocol. The uploading or downloading file duration is smaller. The decryption time is greater than encrypting time.

## REFERENCES

[1] B. Chee, and C. Franklin, Jr., "Cloud Computing Technologies and Strategies of the Ubiquitous Data Center", CRC, United States of America, 2010.
http://dx.doi.org/10.1201/9781439806173
[2] D. Tom, Cryptography, 2000.
[3] P. Mell, and T. Grance, "The NIST Definition of Cloud Computing", National Institute of Standards and Technology, Information Technology Laboratory, Technical Special Publication 800-145, 2011.
[4] S. Jajodia, S. D. C. di Vimercati, S. Foresti, , S. Paraboschi, and P. Samarati, "A Data Outsourcing Architecture Combining Cryptography and Access Control", *Proc. ACM Workshop on Computer Security Architecture (CSAW'07)*, USA, November 2007.
[5] W. Diffie and M. E. Hellman, "New Directions in Cryptography", *IEEE Transactions on Information Theory*, November 1976, .pp. 644-654.
http://dx.doi.org/10.1109/TIT.1976.1055638
[6] W. Stallings, "Cryptography and Network Security: Principles and Practice", Prentice Hall, United States of America, 2011.
[7] Y. Tang, Patrick P.C. Lee, John C.S. Lui, and R. Perlman, "FADE: Secure Overlay Cloud Storage with File Assured Deletion", *in Proc. SecureComm*, Singapore, September 2010, pp. 1-18. ====3
[8] Amazon Simple Storage Service (Amazon S3), http://aws.amazon.com/s3/
[9] OpenStack, http://www.openstack.org.