ISSN (Print) : 0974-6846
Indian Journal of Science and Technology, *Vol 9(48), DOI: 10.17485/ijst/2016/v9i48/89496, December 2016* ISSN (Online) : 0974-5645

Secure File Sharing Mechanism and Key Management for Mobile Cloud Computing Environment

I. Indu, P. M. Rubesh Anand* and Shaicy P. Shaji

Department of Electronics and Communication Engineering, Hindustan University, Chennai - 603103, Tamil Nadu, India; indu.i2043@gmail.com, rubesh.anand@gmail.com, shaicypshaji10@gmail.com

Abstract

Objectives: The use of the mobile devices in cloud computing environment is susceptible to various kinds of attacks like, unauthorized access, account/service hijacking, data breach and malicious insider. These vulnerabilities make the cloud environment unsafe to share and store data for mobile users. **Methods/Analysis:** In this paper, we propose a secure file storing and retrieving mechanism to avoid the limitations in existing systems like, file encryption, access rights and key management. Asymmetric key cryptography is utilized to protect the data and retrieval of the data with minimal access rights. **Findings:** Privacy of the mobile users are protected from the malicious insiders along with the preservation of confidentiality and integrity of the files being accessed. The comparative analysis of different public key infrastructure algorithms utilizing the proposed methodology for key computation, encryption, decryption and resource utilization shows the performance of each algorithm for different file sizes. **Application/Improvement:** The proposed system provides user access management, key management, encryption and decryption of files through trusted third party to make the data secured in mobile cloud environment.

Keywords: Access Control, Asymmetric Key Cryptography, Cloud Computing, Confidentiality, Data Security, Integrity

1. Introduction

Cloud storage systems have been the source of attraction for the online users so as to have easy access anywhere and anytime. Many online service providers have thrived to serve the individual users, industrialists as well as the business people to have their data on cloud with reliability and security. The numbers of mobile users who need to use the resources or services on the go with the help of their mobile devices from cloud based systems are rapidly increasing. The process of utilizing the cloud resources for storage and transition of data by mobile users is a challenging task[1–5]. The cloud environment provided by the online service providers can be in the type of public, private or hybrid cloud. The cloud user selects the type of cloud environment based on the users' decision to privacy or exposure policy. Many IT giants are using the cloud services to reduce the on-premises cost which is greater than they provide for the online service providers[6]. The cloud system provided by different vendors exhibits the heterogeneity with respect to performance and pricing. The design techniques are varied to achieve competitive results in terms of efficient service, reduced cost, secured data storage. The overall benefits of the cloud system are easy sharing, syncing, off-site data storage, better remote accessibility, reduction of internal IT costs, reduced requirement of resources and online data collaboration.

Mobile Cloud Computing (MCC) is the combined approach of the technologies namely, cloud computing, mobile computing, and wireless networks for sharing resources to mobile users. The MCC is used to bring rich computational resources to mobile networks and differentiates from the mobile computing by using the cloud based web apps rather than the native apps. In MCC, mobile devices are used to view the data files as the storage of data and processing of data is done in the cloud infrastructure instead of the mobile device itself. The mobile thin native client devices in cloud computing

* *Author for correspondence*

environment are accessed over the wireless connection. The mobile applications accessed from the thin client devices often move the storage and processing of data into the powerful and centralized cloud computing infrastructure. Moreover, mobile app storage constraint is eliminated as data is stored in cloud rather than the mobile devices. As the data is stored or synced in the cloud storage, the chance of data lost is reduced on the mobile devices which indeed improve the reliability and availability of data while the users are on move. MCC supports multi-tenancy and ease of integration of multiple services provided by different cloud service providers. Cloud based applications are predicted to account for 90% of total mobile data traffic by 2019. Mobile cloud traffic is predicted by CISCO[7] to grow 11-fold from 2014 to 2019, attaining a compound annual growth rate (CAGR) of 60%.

The mobile cloud system reduces the need to send each file every time to different recipients; instead an access link is sent which indeed reduces the bandwidth usage. The annual costs for an organization are reduced largely without the need for employing manpower and resources due to the usage of MCC. Though there are many merits in using the cloud system, the pitfalls of MCC is that the user is charged heavily for every byte of data storage when the limit of certain capacity is crossed andstorage of information in the cloud is vulnerable to external hack attacks and threats.In order to overcome the issues related to cloud storage, multi-cloud data storage has become a key area to explore upon the solutions for the above said problems[8–10].

The MCC has several issues such as limited resources, network related issues, security, availability and privacy.

- **Limited Resources**

The mobile devices which make use of the cloud computing environment have limited resources for utilization. The limited resources include the limited computation power, low quality display and limited battery power. The bandwidth of the network is the big constraint due to the scarcity of frequencies compared to the traditional wired network.Computation offloading is considered to be one of the main features of MCC which deals with the transfer of computation parts of the application to cloud infrastructure. It is critical to determine whether to offload the work or not and to decide on the portions of the service codes to offload[11].

- **Network Related Issues**

In MCC, the mobile user side processing like, connection with the internet provider and cloud server is performed on the wireless network provider side.Apart from the connection issues, the network related problems like, latency, signal strength and heterogeneity also affects the mobile users in accessing cloud services[12].

- **Security**

Though the mobile devices in the cloud computing environment has the functionalities similar to the desktop computers, the issues related to security and privacy are more prone to the mobile devices. As the threat detection services are performed on the cloud, the other security issues related to mobile devices poses great challenges. MCC security issues are categorized as security for mobile users and securing data on clouds. Mobile user security involves the device security and privacy of mobile user. Securing the data on the cloud involves the cryptographic suite, confidentiality and integrity[13–17].

- **Availability**

Availability of the cloud means which services are to be able to access remotely. In simple terms, availability refers that complete resources are accessible and usable at all time by authorized persons. It is the most critical security requirements in mobile cloud computing. The main advantage of availability for cloud systems is to ensure the users use them at any time and at any place. The important features of availability include, continuity, quality, incident management, functionality, and security[18,19]. Continuity means it ensures that the services are available without any interruptions. The Quality of services means it confirms the access time, a number of supported users, and amount of data processed. System availability is the ability to continue operations even in the possibility of any security breach, traffic congestion, network failures and out-of-signal[20,21].

- **Privacy**

The trust of the mobile users in MCC platform is established by preserving the user privacy information like, location of the mobile device and protecting data or application secrecy from adversaries. Location Based Services (LBS) and Global Positioning System (GPS) are responsible for the privacy issues on mobile users which

provide private information such as the current location and history of locations of the mobile user. This problem is worse when any user's information like, travel plan, business schedules and length of stay at a particular location is known to an adversary[22–25].

The paper proposes an efficient and secure file sharing mechanism through Trusted Third Party (TTP) system which is responsible for key management and user access management. The privacy of the mobile user is preserved by TTP while accessing the cloud data storage. The security of the data stored and accessing rights are determined by the proposed methodology in order to overcome the setbacks of the existing symmetric key cryptography[26–28].

The rest of the paper is organized in five sections. The earlier works associated with cloud storage systems are discussed and the methodologies used to overcome the problems are presented in Section 2. The descriptive details of the proposed data hosting technique are given in the Section 3. The experimental results based on the simulation and their comparative analyses are provided in Section 4. Lastly, Section 5 summarizes and gives the conclusion to the work.

2. Materials and Methods

In mobile cloud computing infrastructure, a secure file sharing mechanism utilizing public key cryptography is proposed for cloud data storage and retrieval. The proposed file sharing methodology consists of four entities as shown in Figure 1. The entities in the proposed system are data owner, data users, trusted third party system and cloud storage. The Trusted Third Party (TTP) system also acts as the cryptographic server and key management system. TTP has multiple responsibilities such as, owner management, user management, encryption, decryption and access control. TTP is the centralized authority for any file access related activities by the users from the cloud storage. Data owner has the full control over the file it owns and no other user is allowed to modify the file. The owner or TTP controls the users at any point of time. The data owner registers as an owner in the TTP system for the file upload. Once owner registration is successfully completed, the TTP system generates private key and public key for the file that the owner uploads. The uploaded file is then encrypted by TTP system using private key and the encrypted file is uploaded it to the cloud storage system.

2.1 Key Generation and Encryption

Initially, data owner has to register in trusted third party system for keeping the files in cloud environment. Data owners create the login credentials for uploading files and those credentials are also used to upload the user lists and their permissions. After receiving a particular file (F) from the data owner, the TTP generates keys by using asymmetric key encryption. Asymmetric key generation is not discussed in this paper and it is assumed that any standard asymmetric key generation algorithm (A_{PKI}) is utilized for this purpose. The flow diagram of the data owner submitting a file to the cloud repository through TTP is shown in Figure 2.

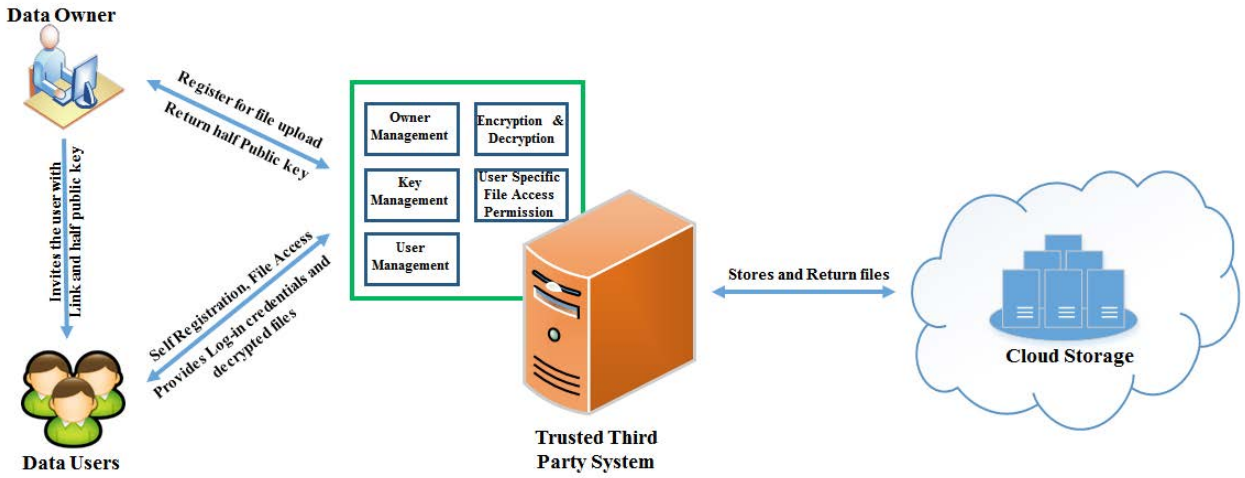


Figure 1. Block diagram of the proposed file sharing methodology.

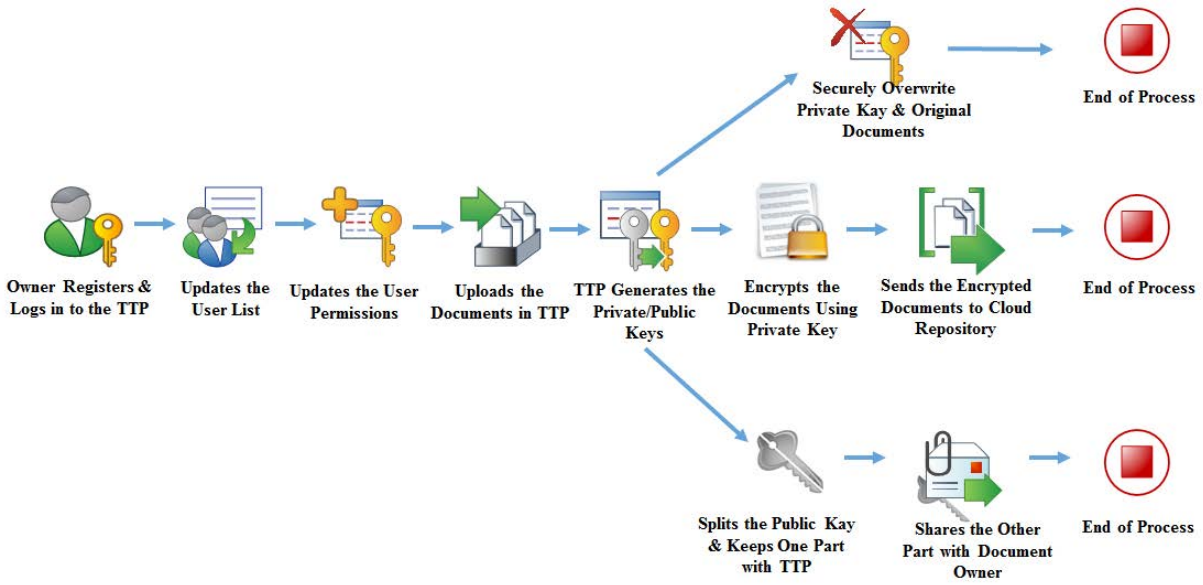


Figure 2. Flow diagram of the data owner process in the proposed method.

In public-private key pair, the private key (K_{pr}) generated by TTP during the asymmetric key generation is used for encryption of the file. The public key (K_{pu}) is divided into two halves or parts by the TTP and is used as a whole for decryption of the uploaded file. One half or part of the public key is transmitted to the data owner (K_o) and the other half or part is kept by TTP (K_{TTP}) itself. The TTP system which acts as a cryptographic server deletes the private key through secure overwriting of one half of the public keyover it after the file is uploaded in the cloud system. The encrypted file (F_{en}) is then sending to the cloud repository for storage. Table 1 shows the notations and its definitions used in the proposed methodology.

Table 1. Notations and definitions used in the proposed model

Notations	Definitions
F	Data file
K_{length}	Key length
A_{PKI}	Public Key Infrastructure Algorithm
K_{pr}	Private key
K_{pu}	Public key
K_{TTP}	TTP's part of Public Key
K_o	Owner's part of Public Key
F_{en}	Encrypted file
K_{half}	Value of half of the key length
F_{ID}	File Identification
U_{list}	User list
U_{APL}	User Access Permission List

Algorithm for Key Generation and Encryption

Input: F, K_{length}, A_{PKI};
Output: K_{pr}, K_{pu}, K_{TTP}, K_o, F_{en};
// Process at TTP //
01: **if** *(Owner = TRUE)* **then** //Owner authentication is successful//
02: $A_{PKI} \leftarrow ENABLE$; // Public Key Infrastructure algorithm is initialized //
03: $\{K_{pr}, K_{pu}\} \leftarrow A_{PKI}$; // Generate public and private keys //
04: $F_{en} \leftarrow en\{K_{pr}, F\}$ // Encryption of File using private key //
05: $K_{half} \leftarrow \{K_{length} / 2\}$;
06: Let $K_o[1...K_{half}]$ and $K_{TTP}[1...K_{half}]$ be new arrays
07: **for** i =1 to K_{half} **do**
08: $K_o[i] \leftarrow K_{pu}[i]$; // Public key is divided in half and copied //
09: **end for**
10: $i \leftarrow 1$;
11: **for** j = (K_{half}+1) to K_{length} **do**
12: $K_{TTP}[i] \leftarrow K_{pu}[j]$; // Second half of the public key is copied //
13: $i \leftarrow i+1$;
14: **end for**
15: *DELETE* (K_{pr}) // Private key is securely deleted after encryption //
16: *DELETE* (K_{pu}) // Public key is securely deleted after encryption //
17: **end if**

2.2 Decryption of File

The data owner provides user list/access permission list (U_{APL}), user specific security questions and answers, number of file access permission to the TTP server. The data owner invites all the users given in the access control list with the link to the TTP and half or part of owner's public key for self-registration. When a user wants to access the uploaded file in the cloud, the user initially registers with the TTP. Once user registration is completed, the user gets login credentials from the TTP. After successful authentication between TTP and requesting user, the user request any particular file through File Identification (F_{ID}) along with the part of the

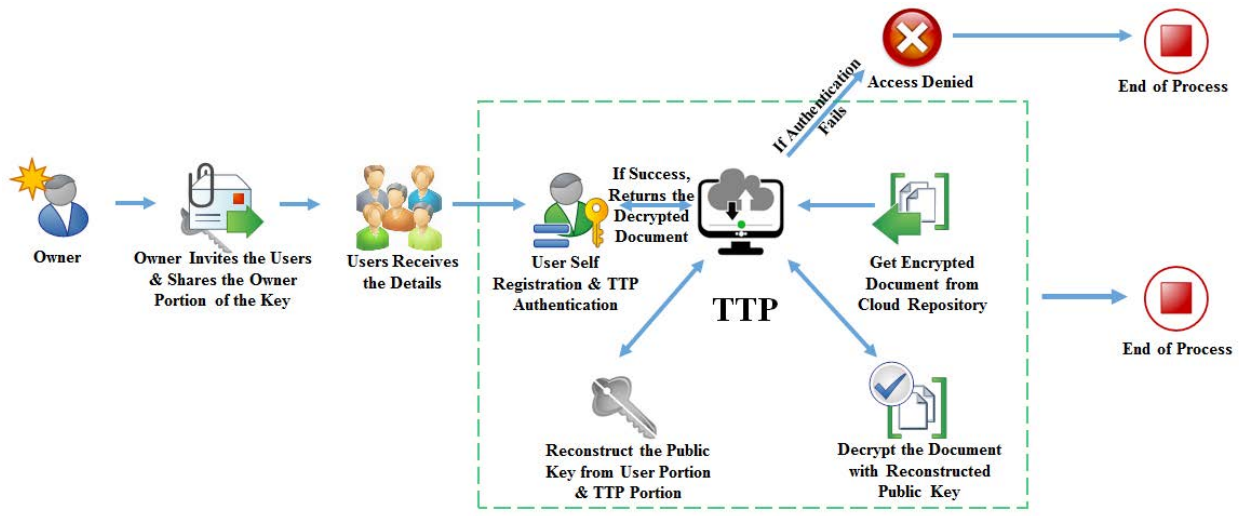


Figure 3. Flow diagram of the data user process in the proposed method.

public key provided to it by the data owner. TTP validates the user access permission for that particular requested file. The TTP regenerates the public key by combining its own half or part of the public key and the received half or part of the public key from the user. In the meanwhile, the TTP downloads the requested file in the encrypted form from the cloud data storage. The downloaded file is then decrypted using the regenerated public key. The decrypted data file is sent back to the corresponding user from TTP as shown in Figure 3.

Algorithm for Decryption of File

Input: F_{ID}, F_{en}, K_{TTP}, K_o, U_{list}, U_{APL}, A_{PKI};
Output: F_{ID}, F;

// Process at TTP for every user request //
01: **if** *(U_{list} = TRUE)* **then** // User is verified with user list//
02: **if** *(U_{APL} = TRUE)* **then** // User is verified with user access permission list //
03: **if** *(F_{ID} = TRUE)* **then** // File is retrieved from the cloud repository //
04: *A_{PKI} ← ENABLE;* // Public Key Infrastructure algorithm is initialized //
05: *K_{pu} ← CONCAT {K_{TTP}, K_o};* // Parts of public key are joined together //
06: *F ← de{K_{pu},F_{en}}* // Decryption of File using recomputed public key //
07: *DELETE(K_{pu});* // Public key is securely deleted after decryption //
08: **else**
09: *NOT FOUND (F_{ID});* // File ID not matches with the existing file list //
10: **end if**
11: **else**
12: *DELETE (K_o)* //Owner key is securely deleted if user is not in ACL//
13: **end if**
14: **end if**

2.3 File Access and Data Owner

When a new user wishes to access the file, it sends joining request message to the data owner. The data owner adds the details of the newly arrived user in the Access Permission List (APL) and sends the updated APL to TTP. The TTP after verifying the credentials of the data owner updates the APL by securely overwriting the access control list for the particular data owner's file. Concurrently, the data owner provides the half of owner's public key and link for accessing the file through TTP to the newly joined user. The authenticated user who wishes to modify and upload the accessed file needs to become the data owner or high privileged user of that modified file. The user who needs to modify the file registers as owner in TTP and upload the modified file in the cloud as another name through TTP. For each file stored in the cloud, one user in the group is data owner others are data accessing users.

2.4 Implementation Details

The proposed mechanism is implemented in cloud environment with the help of JAVA technology. As a part of the implementation, self-registration User Interface (UI) for file owners is developed. In the UI, a file owner registers in the Trusted Third Party (TTP) system and setup the credentials. After the successful registration, the owner is able to setup user lists, user details and their access permissions in TTP through UI. A feature is developed in TTP to perform the setup as a bulk upload as well as individual user creation. Once the users are added to TTP, they get an invite from the TTP/Owner. The TTP logon credentials are setup by following the instructions in the invite that is send to the user. TTP uses the proposed

asymmetric key encryption method for securing the files in cloud environment. In the implementation of the proposed mechanism, RSA algorithm is used for key generation and encryption. TTP uses one of the key (private key) for encryption of the file and the key is securely deleted (over write). The public key is split into two parts with the help of secret sharing algorithm. The TTP keeps half of the public key and the other half shared to the owner. After sharing the owner's portion of the public key, TTP securely deletes (over writes) it from repository. The owner shares the public key portion to the required users. The users access the TTP and request for a particular file along with the corresponding key portion. TTP validates the user permissions and reconstructs the public key. TTP downloads the encrypted file from the cloud storage and decrypt it with the reconstructed public key. The decrypted file is shared to the requested user. Once the process is completed, TTP deletes the reconstructed public key and the decrypted file.

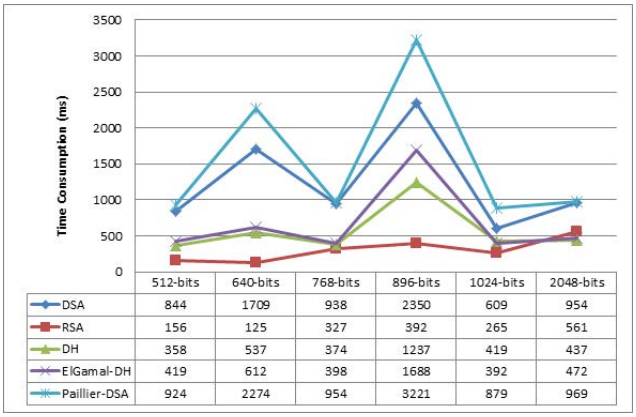
In Public Key Infrastructure (PKI), the commonly used key generation algorithms for generating public key and private key are Rivest, Shamir and Adleman (RSA), Diffie-Hellman (DH) and Digital Signature Algorithm (DSA). The proposed key management methodology including key generation, encryption and decryption is tested in Intel i3 processor of 1.4 GHz with 4 GB of RAM. The performance of the different PKI algorithms like, RSA, ElGamal and Paillier Algorithms[29,30] are compared for the process of encryption, decryption, CPU usage and memory usage.

3. Results and Discussion

3.1 Key Computation Time Consumption

The time consumption is the major factor for the key generation. The optimized system design must ensure that the time needed for the key generation process is less. The Time consumption parameter is used in the proposed key computation scheme to analyse the different standard PKI key generation algorithms such as, DSA, DH, RSA, ElGamal with DH and Paillier with DSA. As ElGamal utilizes Diffie-Hellman for key generation process and Paillier utilizes DSA for asymmetric key generation, the comparison of all PKI algorithms is done for key generation time alone.

The time consumption for key computation is expressed in milliseconds and the comparison shows that RSA performs better than other key generation algorithms as shown in Figure 4. It is also observed that the key length of 512 bits, 768 bits, 1024 bits and 2048 bits are computed faster than the other values like, 640 bits or 896 bits. This is due to the factors like, selection of prime numbers, performing modular exponentiation and other computations for the bit length which are not exactly the powers of 2.



	512-bits	640-bits	768-bits	896-bits	1024-bits	2048-bits
DSA	844	1709	938	2350	609	954
RSA	156	125	327	392	265	561
DH	358	537	374	1237	419	437
ElGamal-DH	419	612	398	1688	392	472
Paillier-DSA	924	2274	954	3221	879	969

Figure 4. Time consumption for different key sizes by key generation algorithms using the proposed methodology.

3.2 Encryption and Decryption Process

Encryption and Decryption in cryptography mechanism are the vital elements for establishing security in cloud computing environment. The encryption and decryption process is performed in PKI through RSA, ElGamal and Paillier algorithms. The comparison of the PKI algorithms in terms of time consumption during the process of encryption and decryption when 10 KB file is used highlights that RSA performs better as shown in Figure 5. However, the RSA algorithm degrades in its performance during the encryption of large files in the order of hundreds of MB size. But ElGamal and Paillier are proved for its usage in encrypting large size files. The comparison of both ElGamal and Paillier exhibits their performance equally when the proposed methodology of key management is utilized.

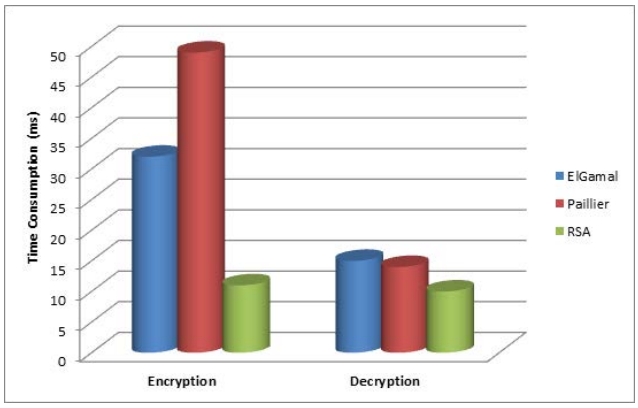


Figure 5. Time consumption for encryption and decryption process by key generation algorithms using the proposed methodology.

3.3 Resource Utilization

The proposed methodology of key management, encryption and decryption of files which is performed by trusted third party is considered for the calculation of resources utilizations. The parameters to calculate resource utilization includes CPU usage, RAM or memory usage, and power usage. The comparative results of the CPU usage and memory usage for different PKI algorithms utilizing the proposed methodology are shown in Figure 6 and Figure 7. In case of small file sizes, RSA consumes fewer amounts of processor cycles and RAM capacity. Both ElGamal and Paillier algorithms consumes equal amount of resources for small file sizes. But, RSA has limitations in handling large file sizes, whereas, ElGamal and Paillier algorithms are suitable for encryption and decryption of large file sizes.

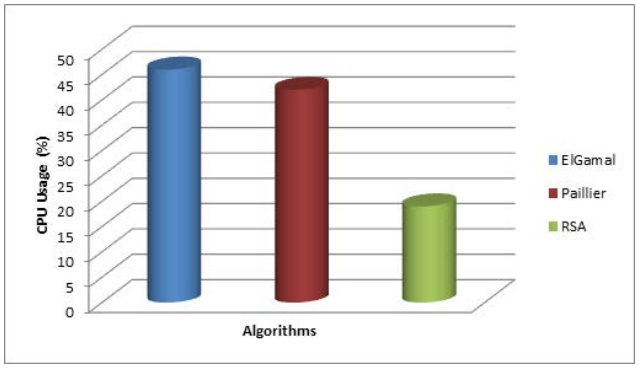


Figure 6. CPU usage by different PKI algorithms using the proposed methodology.

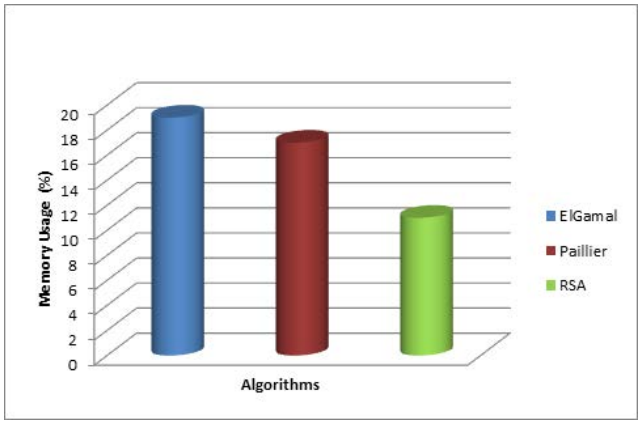


Figure 7. Memory usage by different PKI algorithms using the proposed methodology.

4. Conclusion

Mobile cloud computing has become an inevitable part in the recent business and administrative environment. The proposed secure file sharing mechanism for accessing cloud data storage ensures security and privacy for group user access. Trusted third party plays the main role in protecting the security for the file access and privacy for the users from malicious insiders in cloud environment. The confidentiality and integrity of the stored and retrieved file is preserved through the proposed file sharing mechanism. The group user access management, key management, encryption and decryption of files are performed through trusted third party to make the data secured in mobile cloud environment. The comparative analysis of different PKI algorithms using the proposed methodology for key computation, encryption, decryption and resource utilization shows that RSA algorithm performs well in handling small file sizes whereas, ElGamal and Paillier algorithms are more suitable for larger files to be stored in cloud storage.

5. References

1. Khan AN, Mat Kiah ML, Khan SU, Madani SA. Towards secure mobile cloud computing: A survey. Futur Gener Comput Syst. 2013; 29(5):1278–99.
2. Kumar R, Rajalakshmi S. Mobile cloud computing: Standard approach to protecting and securing of mobile cloud ecosystems. Proceedings of International Conference on Computer Sciences and Applications; 2013. p. 663–9.

3. Uddin M, Memon J, Alsaqour R, Shah A, Rozan MZA. Mobile agent based multi-layer security framework for cloud data centers. Indian Journal of Science and Technology. 2015 Jun; 8(12):171–8.
4. Rajathi A, Saravanan N. A survey on secure storage in cloud computing. Indian Journal of Science and Technology. 2013 Apr; 6(4):1–6.
5. Lee JY. A study on the use of secure data in cloud storage for collaboration. Indian Journal of Science and Technology. 2015 Mar; 8(S5):33–6.
6. Grobauer B, Walloschek T, Stocker E. Understanding cloud computing vulnerabilities. IEEE Secur Priv. 2011; 9(2):50–7.
7. Cisco Visual Networking Index. Available from: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.pdf
8. Sanaei Z, Abolfazli S, Gani A, Shiraz M. SAMI: Service-based arbitrated multi-tier infrastructure for mobile cloud computing. Proceedings of 1st IEEE International Conference on Communications in China Workshops; 2012. p. 14–9.
9. Kalpana V, Meena V. Study on data storage correctness methods in mobile cloud computing. Indian Journal of Science and Technology. 2015 Mar; 8(6):495–500.
10. Mishra A, Jain R, Durresi A. Cloud computing: Networking and communication challenges. IEEE Communications Magazine. 2012; 50(9):24–5.
11. Fernando N, Loke SW, Rahayu W. Mobile cloud computing: A survey. Future Generator Computer System. 2013; 29(1):84–106.
12. Dinh HT, Lee C, Niyato D, Wang P. A survey of mobile cloud computing: architecture, applications, and approaches. Wireless Communications and Mobile Computing. 2011; 13(8):1587–611.
13. Tsai JL, Lo NW. A Privacy-Aware Authentication Scheme for Distributed Mobile Cloud Computing Services. IEEE System Journal. 2015 May; 9(3):805–15.
14. Neela TJ, Saravanan N. Privacy preserving approaches in cloud: A survey. Indian Journal of Science and Technology. 2013 May; 6(5):1–5.
15. Sen J. Security and privacy issues in cloud computing. Architecture Protocol Security Information Technology. 2013; (4):1–42.
16. Xiao Z, Xiao Y. Security and privacy in cloud computing. IEEE Communication Survey Tutorials. 2013; 15(2):843–59.
17. Nagaraju S, Parthiban L. Sec Authn: Provably secure multi-factor authentication for the cloud computing systems. Indian Journal of Science and Technology. 2016 Mar; 9(9):1–18.
18. Jasmine R, Nishibha GM. Public cloud secure group sharing and accessing in cloud computing. Indian Journal of Science and Technology. 2015 Jul; 8(15):1–7.
19. Manjusha R, Ramachandran R. Secure authentication and access system for cloud computing auditing services using associated digital certificate. Indian Journal of Science and Technology. 2015 Apr; 8(S7):220–7.
20. Sun H, Wen Q, Zhang H, Jin Z. A novel remote user authentication and key agreement scheme for mobile client-server environment. Application Mathematical Information Science. 2013; 7(4):1365–74.
21. Xie Y, Wen H, Wu B, Jiang Y, Meng J. A modified hierarchical attribute-based encryption access control method for mobile cloud computing. IEEE Trans Cloud Computing. 2015; (99):1–1.
22. Yang X, Huang X, Liu JK. Efficient handover authentication with user anonymity and untraceability for Mobile Cloud Computing. Future Generation Computer Systems. 2015 Sep; 62:190–5.
23. Armando A, Carbone R, Compagna L, Cuellar J, Pellegrino G, Sorniotti A. An authentication flaw in browser-based single sign-on protocols: Impact and remediations. Computer Security. 2013; 33:41–58.
24. Zhang Y, Chen Q, Zhong S. Privacy-preserving data aggregation in mobile phone sensing. IEEE Transaction on Information Forensics Security. 2016; 11(5):980–92.
25. Suo H, Liu Z, Wan J, Zhou K. Security and privacy in mobile cloud computing. Proceedings of 9th International Wireless Communications and Mobile Computing Conference (IWCMC); Sardinia. 2013. p. 655–9.
26. Rajarajeswari S, Somasundaram K. Data confidentiality and privacy in cloud computing. Indian Journal of Science and Technology. 2016 Jan; 9(4):1–8.
27. Sugumar R, Imam SBS. Symmetric encryption algorithm to secure outsourced data in public cloud storage. Indian Journal of Science and Technology. 2015 Sep; 8(23):1–5.
28. Saikeerthana R, Umamakeswari A. Secure data storage and data retrieval in cloud storage using cipher policy attribute based encryption. Indian Journal of Science and Technology. 2015 May; 8(S9):318–25.
29. Paillier P. Public-key cryptosystems based on composite degree residuosity classes. Proceedings of Advances in Cryptology (Eurocrypt '99); Prague, Czech Republic. 1999. p. 223–38.
30. Dawahdeh ZE, Yaakob SN, Sagheer AM. Modified ElGamal elliptic curve cryptosystem using hexadecimal representation. Indian Journal of Science and Technology. 2015 Jul; 8(15):1–8.