A Lightweight Hybrid Key Management Scheme using Third Party Auditor for Data Security in Cloud

V. Vasanthi #1, Dr. M. Chidambaram*2
# Research Scholar, Research and Development Centre,
Bharathiar University, Coimbatore, Tamilnadu, India
1vasanthi_78@rediffmail.com
* Assistant Professor, Department of Computer Science,
Rajah Serfoji Govt. Arts College, Thanjavur, Tamilnadu, India
2 chidsuba@gmail.com

Abstract—**Authentication and key management are the key challenges in the cloud environment while exchanging the confidential information. It requires a lightweight key management protocol for communication in the cloud environment. This paper presents a lightweight hybrid key management scheme for improved data security in the cloud computing environment. A Third Party Auditor (TPA) ensures secure data communication between the data owner and cloud service provider using the proposed key management scheme. The hybrid scheme is developed by combining identity-oriented key management and pairwise probabilistic key pre-distribution schemes. The lightweight two-level session key is generated using the Hash Message Authentication Code (HMAC) and Exclusive OR (XOR) operations. It involves two level of session key establishment to reinforce the key against the traffic analysis attack. The Advanced Encryption Standard (AES) key and session key are required to download and decrypt the file. The cloud server schedules the tasks to the Virtual Machines (VMs) by applying the Genetic Algorithm (GA). The experimental analysis shows that the proposed key management scheme requires lower minimum key size, energy consumption, file uploading time, file downloading time and encryption time than the existing schemes.**

Keyword-Advanced Encryption Standard (AES) algorithm, Cloud Computing, Genetic Algorithm (GA), Lightweight Key Management, Third Party Auditor, Two-level Session Key

I. INTRODUCTION

With the rapid increase in the data sharing across the Internet, the cloud computing system is frequently used in the multiple data owner scenario. The cloud computing system offers various types of services. Platform as a Service (PaaS) is a cloud-based service that provides more choices to the subscriber for choosing the computing platform. Infrastructure as a Service (IaaS) provides the same features as the PaaS, but the customer is completely responsible for controlling the rented infrastructure. Software as a Service (SaaS) allows the business enterprises to access the functionality at a lower cost than the cost of licensed applications, as the SaaS pricing is based on a monthly fee. Due to the remote hosting of the software, the users do not need to invest in the additional hardware. The SaaS reduces the effort of installation, setup and maintenance by the business enterprises. Hence, it is referred as simply hosted applications.

The data owner can upload the data to the cloud service provider and access the stored data using the software provided by the service provider. As the data received from the data owner is not enough to fill the storage space of the server, it leads to a lot of storage waste. The data received from different data owners is stored in the same server to increase the utilization rate of server storage space. There is a need to protect the confidential data for preventing the theft of confidential data by unauthorized person or other data owners. Hence, the confidential data is encrypted and stored. The data owner and cloud user should obtain the security keys from the service provider. If the data owner and user access a huge amount of data, there is a need to manage a large number of keys. The lightweight key management approach is the optimal solution for the key management issue.

In multiple data owner scenario, it is assumed that the number of keys increases with the increase in the number of data owners and users. In Single-layer Derivation Encryption [1-5] schemes, there is a need to store a single key corresponding to the data owner. In Double-layer Derivation Encryption [6-8] schemes, the number of cryptographic keys used for the data encryption process is double than the Single-layer encryption scheme. The Attribute-based Encryption (ABE) [9-13] scheme accepts an extensive key related to a set of attributes relevant to the data owner and cloud user. The encryption schemes [14-16] are used to manage the authorization, in which each user should manage a set of decryption keys corresponding to the set of data owners. The Ciphertext Policy-ABE (CP-ABE) [17] is an effective cryptographic based access control technique with better data security in the cloud environment. The attribute authority issues a set of attributes to generate an access policy for encrypting the data file. Only the users having attributes that satisfy the access policy can decrypt the file. The unauthenticated or third-party person cannot access the file.

The CP-ABE based access control schemes [18-21] are flexible and scalable. The main problem is key revocation management. The attributes are shared by multiple users. In the case of user or attribute revocation, the non-revoked users should update their keys. If the revoked attribute appears in the access policy, the files should be re-encrypted under the new policy to prevent the unauthorized decryption of the keys comprising revoked attributes. It requires revocation cost andcomputational complexity. In most of the existing methods, the user has to manage a set of keys.

Cryptographic key management plays a main role to provide secure communication between end-users. The symmetric key pre-distribution techniques are found to be more appropriate, due to the resource-constrained property. With Elliptic Curve Cryptography (ECC), many researchers explored the application of asymmetric key for authentication and secure key distribution. ECC consumes less energy and requires minimum key sizes while achieving same security strengths. Encryption guarantees the data security and privacy [22].

To overcome the existing key management issue, this paper proposes a lightweight hybrid key management scheme using the TPA to enhance the data security in the cloud environment. It involves two level of session key establishment to reinforce the key against the traffic analysis attack. Both, the AES key and session key are required to download and decrypt the file. The GA is applied for scheduling the tasks to the VMs.

The following sections in the manuscript are organized in the way: Section II describes a brief overview of the existing key management schemes in cloud. Section III explains the proposed work including AES, two-level session key establishment protocol and GA-based task scheduling. Section IV shows the performance evaluation analysis of the proposed key management scheme. The proposed work is concluded in Section V.

II. RELATED ART

Li et al. [23] proposed a new structure for the secure distribution of the convergent key sharing across multiple shares. The Ramp secret sharing scheme is used to implement the proposed structure. The proposed structure incurs minimum overhead in the real-time environment. Wu et al. [24] developed the time-based hierarchical key management scheme in the cloud environment. From the security analysis, it is observed that the proposed scheme is secure against both the outsider and insider attacks. Chu et al. [25] defined public key cryptosystems for efficient allocation of the decryption rights to a set of ciphertexts. The set of secret keys can be aggregated to form a single compact key that can be transmitted conveniently or stored in a smart card without requiring more secure storage. Tysowski and Hasan[26] proposed novel modifications to the ABE scheme for enabling authorized access of the cloud data. Data access control is achieved based on the satisfaction of required attributes. The higher computational load from the cryptographic operations is assigned to the cloud service provider and the total communication cost is reduced for the mobile user. Li et al. [27] presented ABE techniques for efficient data access control of the Personal Health Record (PHR). The ABE technique achieved high scalability and reduction in the key management complexity for the data owners and users. Beak et al. [28] proposed a secure big data management framework for the smart grids. A security solution is presented to address the issues in the proposed framework.

Kao et al. [29] developed a user-centric key management scheme called as uCloud for efficient data protection in the cloud. Zhou et al. [30] proposed a key management scheme depending on the collaboration of the patients in the social group. From the security analysis and simulation results, it is observed that the proposed scheme is robust to the time and location-based mobile attacks. Xie et al. [31] presented a hierarchical key management system for improving security in cloud-based smart grid. Zhao et al. [32] introduced a completely homomorphic encryption algorithm for enhancing the security and storage capacity in the cloud system. Effective processing and retrieval of the encrypted data can be achieved and data transmission security can be improved without requiring much storage. Cui et al. [33] proposed a lightweight key management technique for ensuring high data security in the cloud system. All authorized data can be decrypted using a single key and a set of public information stored on the server. Ren et al. [34] presented a lightweight key management scheme that guaranteed both forward and backward secrecy. This scheme does not depend on the trusted third party. Cui et al. [35] presented a security model for Internet of Things (IOT) and proposed an access control method and an authorization update method to reduce the key management cost. The IOT owner can easily manage the sensitive data and authorization irrespective of the need to change the authorization policy. Yao et al. [36] presented a lightweight ciphertext access control device for the mobile cloud computing environment. The proposed scheme is based on the authorization certificates and secret sharing. Efficient and fine-grained ciphertext access control can be achieved without requiring much cost than the ABE scheme. Belguith et al. [37] proposed a novel lightweight encryption algorithm comprising the combination of symmetric and asymmetric algorithms for data encryption and key distribution. The proposed algorithm requires minimum processing time than the existing cryptographic algorithms.

This requires the need for the management of hundreds of keys, while storing a huge amount of data in the cloud and using multiple key management servers in a possible way. If the key management function is carried out using a Hardware Security Module (HSM), it requires the creation and maintenance of multiple HSM partitions. Hence, this paper proposes a lightweight key management scheme for the secure processing of the data in the cloud environment. .

III.PROPOSED LIGHTWEIGHT HYBRID KEY MANAGEMENT SCHEME

The data owner employs the AES algorithm to encrypt and decrypt the input data file. The data owner sends the session key generation request to the cloud service provider through the TPA. The service provider checks whether the data owner is valid based on the identity (ID) of the owner. If the data owner is found to be valid, the provider sends nonce to the owner. Then, Message authentication code (MAC) value of the ID and nonce is computed. The session key is generated. The AES key for the data owner is generated to upload the data to the service provider. To download the data file, the owner has to enter the session key. If the session key is found to be valid, the file name is displayed. The AES key should be entered to decrypt the data file. The GA algorithm is applied for scheduling the tasks to the appropriate VMs. Fig.1 depicts the system architecture of the proposed lightweight hybrid key management scheme.

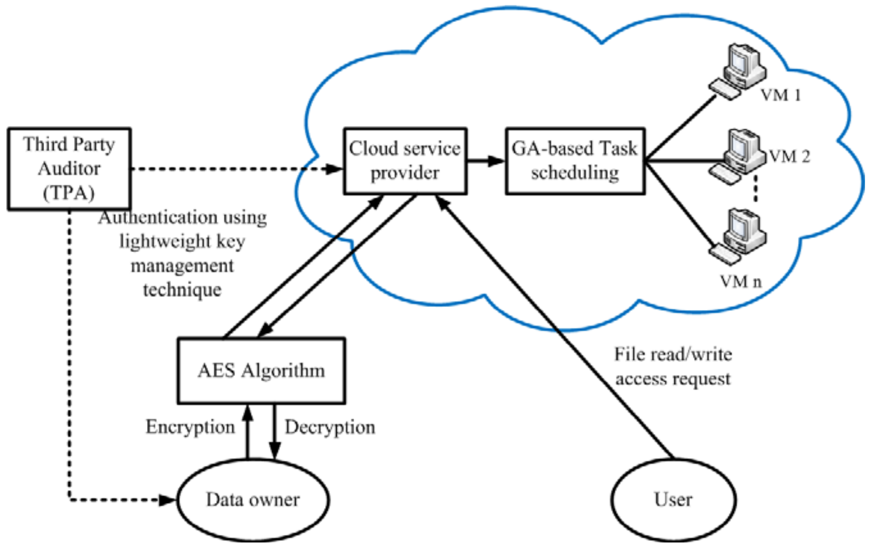
Third Party Auditor (TPA)
Cloud service provider
GA-based Task scheduling
VM 1
VM 2
VM n
Authentication using lightweight key management technique
AES Algorithm
Encryption
Decryption
File read/write access request
Data owner
User

Fig.1 System architecture of the proposed key management scheme

A. *AES algorithm*

The AES algorithm is used for encrypting the plaintext into ciphertext and decrypting it back into the plaintext. It is a symmetric block cipher with the fixed block size of 128 bits. The size of the cryptographic keys is 128, 192 and 256 bits. The size of the cryptographic key mentions the number of rounds required to convert the input plaintext into a ciphertext. The mathematical calculations in AES are performed in a special finite field. Table I shows the length of the key and number of rounds. A 128-bit round key is required separately for each encryption and decryption round. Fig.2 shows the operation of the AES algorithm.

TABLE I. Key length and Number of rounds

Key length	Number of rounds
128-bit	10
192-bit	12
256-bit	14

The steps for encryption and decryption are

Sub-bytes: The substitution box is used for replacing each byte in the array by another byte.

Shift rows: Every row is shifted left to about ‘k’ bytes in a cyclical manner. The ‘k’ value depends on the key and number of the rows.

Mix columns: The linear mixing operation mixes the four bytes in each column.

Add round key: The XOR operation is applied for adding a unique round key to each byte.

The transformation sequence of the decryption process is different from the encryption process. The key expansion is similar for the encryption and decryption processes.

The decryption operations are described below

Inverse sub-bytes: The inverse of the substitution box is used.

Inverse shift rows: The row operation is inversed by shifting the elements in the row to the right side.

Add round key: This process is same as the encryption process.

Inverse mix columns: In this process, linear mixing is performed but with different matrix.

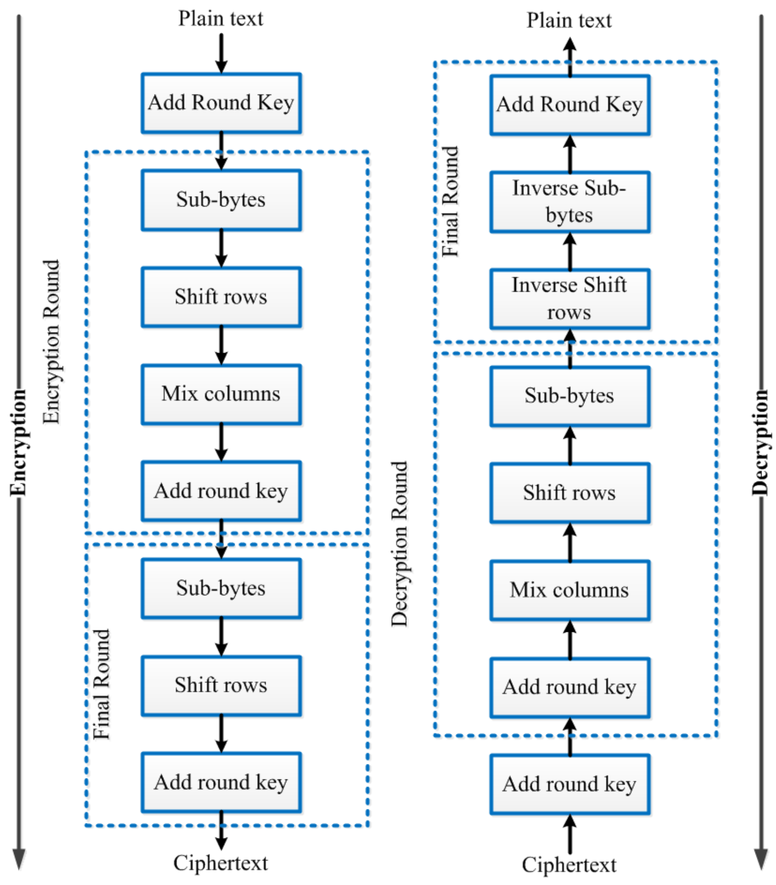

Fig.2 Operation of the AES algorithm

B. Two level session key establishment

The TPA checks the data integrity based on the requirements of the user and outputs the audit reports to calculate the risk of the services [38]. The TPA ensures secure data communication between the data owner and service provider. The TPA can reveal a secret session key to the service provider and request for a fresh MAC for comparison during each audit. In the hybrid scheme, an identity-predicated key management scheme is combined with a pairwise probabilistic key pre-distribution scheme. Fig.3 depicts the cloud storage service.

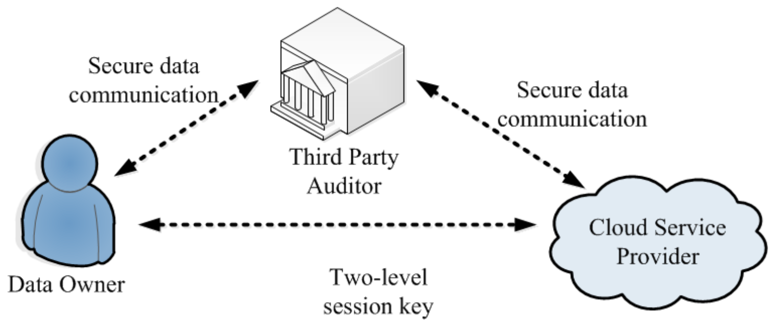

Fig.3 Cloud storage service

A session key is proposed to ensure secure communication between the data owner and service provider for a particular session. A new key is established for each new communication session. For next communication session, data remains in the encrypted form. The data owner forwards the encrypted data to the service provider. The next datacommunication starts when there is a need to upload data to the service provider.

Two-level Session key establishment protocol

Step 1: $N_D \rightarrow N_S: K_{N_D-N_S}\{ID_{N_D}, SK_{REQ}, V_{nN_D}, MAC(ID_{N_D} \| V_{nN_D})\}$

Step 2: $N_S: SK1_{N_D-N_S} = ID_{N_D} \otimes K_{N_D-N_S} \otimes V_{nN_D} \otimes ID_{N_S}$

Step 3: $N_S \rightarrow N_D: SK1_{N_D-N_S}\{ID_{N_D}, V_{nN_S}, MAC(ID_{N_D} \| V_{nN_S})\}$

Step 4: $N_D, N_S: SK_{N_D-N_S} = ID_{N_S} \otimes SK1_{N_D-N_S} \otimes V_{nN_S} \otimes ID_{N_D}$

The data owner N_D sends data uploading request to the cloud service provider N_S and send the identity (ID) of the data owner, session key request, nonce for the data owner and MAC value of the data owner ID and nonce. The Hash-MAC (HMAC) is used to integrate the message. The service provider and data owner generate primary session key $SK1_{N_D-N_S}$ using the nonce value V_{nN_D} and performing XOR with the pairwise symmetric key $K_{N_D-N_S}$ between the data owner and service provider. The service provider uses the primary session key for data encryption, including the identity and nonce value for the service provider and HMAC of the ID and nonce value. Finally, the session key is obtained $SK_{N_D-N_S}$ by performing XOR of the primary session key and nonce value along with the ID of the data owner and service provider [22]. Fig.4 illustrates the two-level session key establishment process.

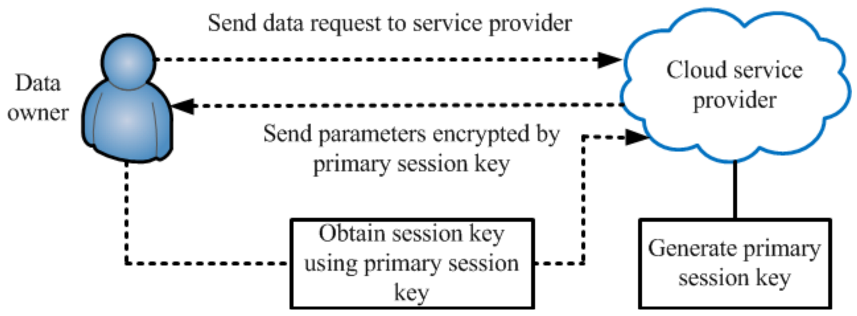
Send data request to service provider
Data owner
Cloud service provider
Send parameters encrypted by primary session key
Obtain session key using primary session key
Generate primary session key

Fig.4 Two-level session key establishment

C. GA-based task scheduling

The main aim of the scheduling process is to minimize the makespan and overall execution time by assigning the tasks to a set of optimal VMs. Let us assume a group of 'm' tasks $T = \{t_1, t_2, t_3, \dots, t_m\}$ and 'q' number of VMs $VM = \{VM_1, VM_2, VM_3, \dots, VM_q\}$ are required to complete the tasks. The Directed Acyclic Graph (DAG) $D = (V, E)$ denotes the dependence among the tasks. 'V' and 'E' denote the set of vertices and edges respectively. The vertices of the graph indicate the tasks and edges represent the priority between the tasks. An edge E_{ij} between the vertices V_i and V_j is represented by $E_{ij} \in E$ and $V_i \rightarrow V_j$ denotes that V_i should be executed before V_j. The size of the dependency matrix 'S' is represented using the following equation

$$S = [\sum_{i=1}^{m} t_i]^2 \qquad (1)$$

Initially, 'l' number of chromosomes is generated in the form of $1-d$ arrays whose size is equal to the number of tasks. The assignment of jth task in the ith chromosome to the kth VM is represented by $pop[i][j] = k$, where$1 \le i \le l, 1 \le j \le number\ of\ tasks\ and\ number\ of\ VMs\ k \ll j$. The operation of the GA algorithm is described below

1) Initial population

The initial population represents the set of randomly generated chromosomes. The number of tasks defines the length of the chromosomes. Fig.5 shows the chromosome representation. Let us consider there are 15 tasks and 5 VMs. As there are 15 tasks, the length of the chromosome is 15. Each task is allocated randomly to the VMs.

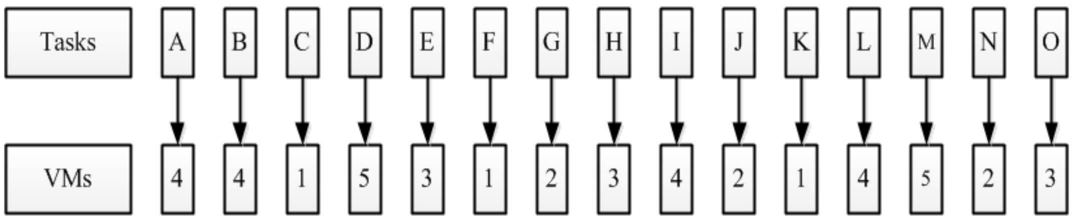

Fig.5 Chromosome representation

2) Derivation of fitness function

The makespan of each chromosome is determined as the fitness function of the GA algorithm. The completion time of the task is the addition of execution time and waiting time. The completion time for each task should be calculated before computing the makespan for a chromosome. Let $TC_{i,j}$ denotes the execution time of task T_i on the VM V_j. If the task T_i depends on the task T_k, then the task T_i cannot start its execution before the completion of task T_k. But, the task T_k should wait for the completion of the parent task T_p. The task T_i may depend on multiple parent tasks. The waiting time for the task $T_i w(T_i)$ is calculated as

$$w(T_i) = \begin{cases} 0 \; if \; T_i \; is \; independent \\ w(T_k) + TC_{k,j} \; such \; that \; T_k \rightarrow T_i, \forall T_i, T_k \in T, V_j \in V \end{cases} \quad (2)$$

The completion time for each task T_i denoted by $\sigma(T_i)$ in the chromosome is given as follows

$$\sigma(T_i) = w(T_i) + TC_{i,j} \quad (3)$$

Where $V_j \in V$ is the assigned VM for the task T_i for the chromosome. Hence, the makespan is the sum of the task completion time in the chromosome.

$$Makespan = \sum_{i=1}^{n} \sigma(T_i) \quad (4)$$

3) Selection

Best chromosomes are selected from the population to generate better solutions using the crossover and mutation operation. Best chromosome is selected by using the roulette wheel selection policy.

4) Crossover

In crossover, some information of two chromosomes is exchanged to produce two better child chromosomes.

5) Mutation

In the mutation process, the quality of the child chromosomes is improved by reducing the makespan and improving the resource utilization rate. The mutation is performed on the selected gene rather than performing random mutation. Initially, the busy VM is found out and a task T_i is selected randomly. If the selected task T_i is independent, the task T_i is removed and assigned to the least utilized VM. If the task is dependent, all the tasks are assigned to the same VM [39].

IV. PERFORMANCE ANALYSIS

A. Security Analysis

In the proposed scheme, a session key is established to secure the communication between the data owner and cloud service provider. The data freshness is provided by using the concept of the nonce and two-way session key. As the session key is changed after every data transaction, it is difficult for the brute force attacker to find the correct session key. Also, the replay attack can be prevented as the session keys can be used once. The data integrity is guaranteed by using a hash function. The proposed scheme can determine party authentication built on the HMAC with a shared session key between the data owner and cloud service provider to determine whether the parameters are changed or not. The attacker cannot be able to analyze the transmitted data as the session keys are changed using the proposed scheme. Mutual authentication is achieved using the ID of both the data owner and service provider. High scalability and robust security can be achieved due to the two-level session key establishment.

B. Storage Cost Analysis

The efficiency of the proposed scheme is analyzed by comparing the key size and energy consumption of the proposed Lightweight Hybrid Key Management Scheme (LHKMS) and existing AES, ECC, Two-level Session Key (TSK), Elliptic Curve Digital Signature Algorithm (ECDSA) and Secure Hash Algorithm 1 (SHA-1) schemes. Fig.6 shows the impact of the key sizes and Fig.7 shows the energy consumption for the proposed and existing schemes. The key size of our proposed scheme is 55 bits and consumes 41 microJoules. The TSK scheme uses a 64 bit key and consumes 46.08 microJoules. The AES scheme uses a 128 bit key and consumes 92.16 microJoules. The ECC scheme uses a 108 bit key and consumes 77.76 microJoules and ECDSA scheme uses a 160 bit key and consumes 115.2 microJoules. The SHA-1 algorithm uses a 128 bit key and consumes 92.16 microJoules. From the graph, it is observed that the key size and energy consumption of the proposed LHKMS scheme is lower than the TSK, AES, ECC, ECDSA and SHA-1 algorithms.

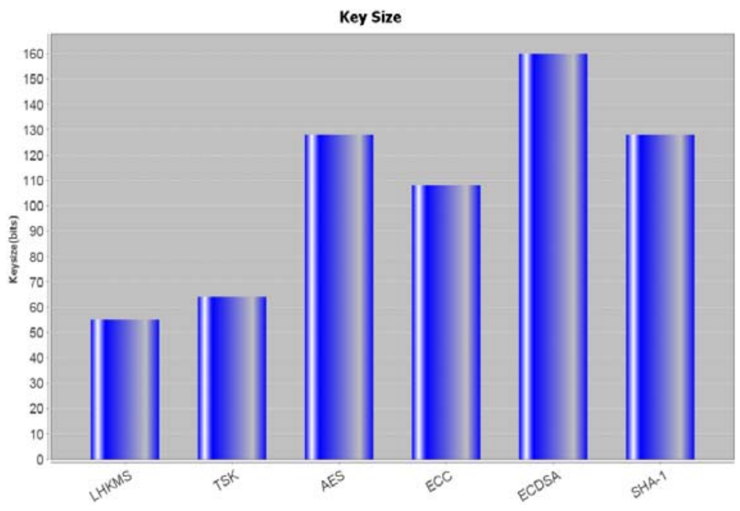

Fig.6 Key size analysis of the proposed LHKMS and existing AES, TSK, ECC, ECDSA and SHA-1 schemes

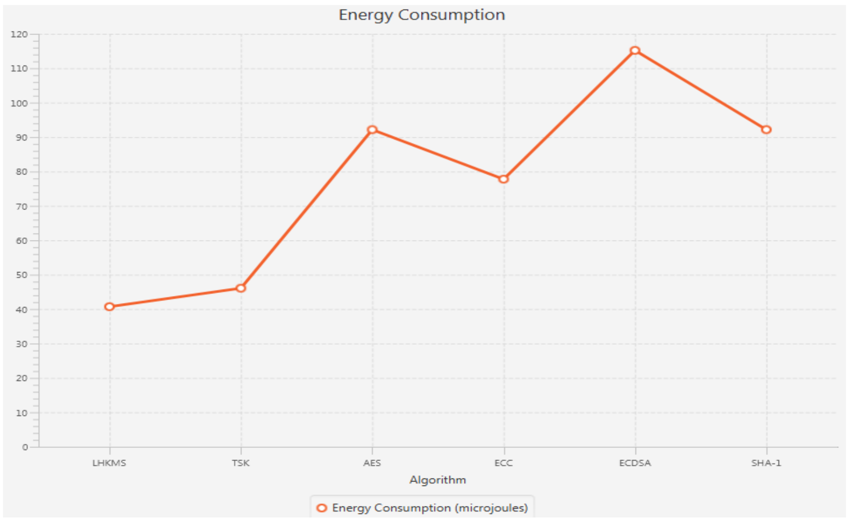

Fig.7 Energy consumption analysis of the proposed LHKMS and existing AES, TSK, ECC, ECDSA and SHA-1 schemes

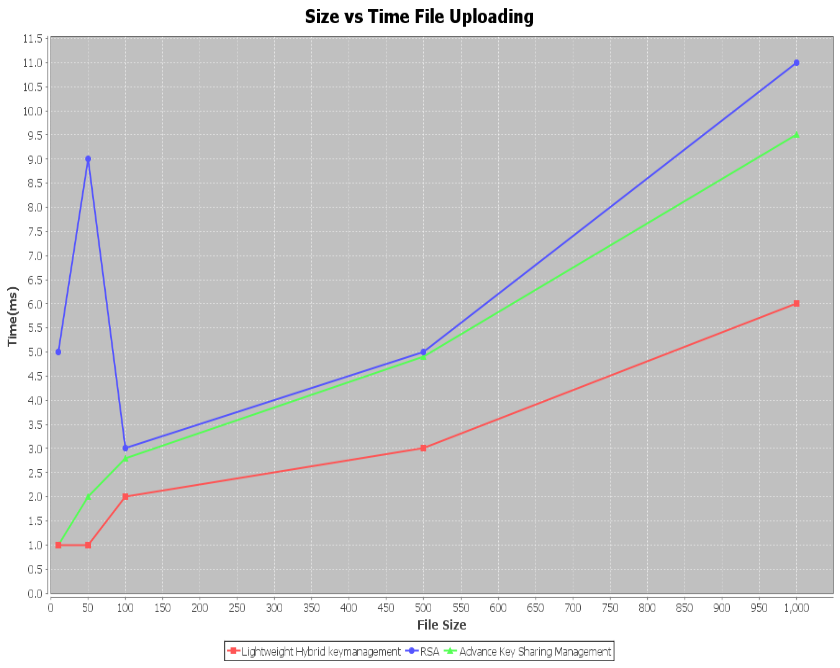

Fig.8 File uploading time of the proposed and existing key management schemes

Our proposed scheme is compared with the RSA and advanced secret sharing key management scheme [40]. File uploading time includes the time required to encrypt the file as requested by the client. It is the time between the points when the data owner requests the cloud service provider to upload the file, the finishing time of the encryption and key generation tasks and the encrypted file is actually stored in the cloud data storage. Fig.8 shows the time taken for uploading the file of different sizes. File downloading time is the time required to collect the shares, generate the secondary key, merge the master key and the secondary key and time needed to decrypt the input file. It is the time between the two points when the user makes a request to download a file and user actually receives his file. Fig.9 shows the time taken for file downloading for different file sizes. Fig.10 shows the comparative analysis of the encryption time of the proposed scheme and existing Blowfish, AES, RC4 and lightweight key [41]. Our proposed LHKMS scheme requires minimum encryption time than the existing cryptographic schemes.

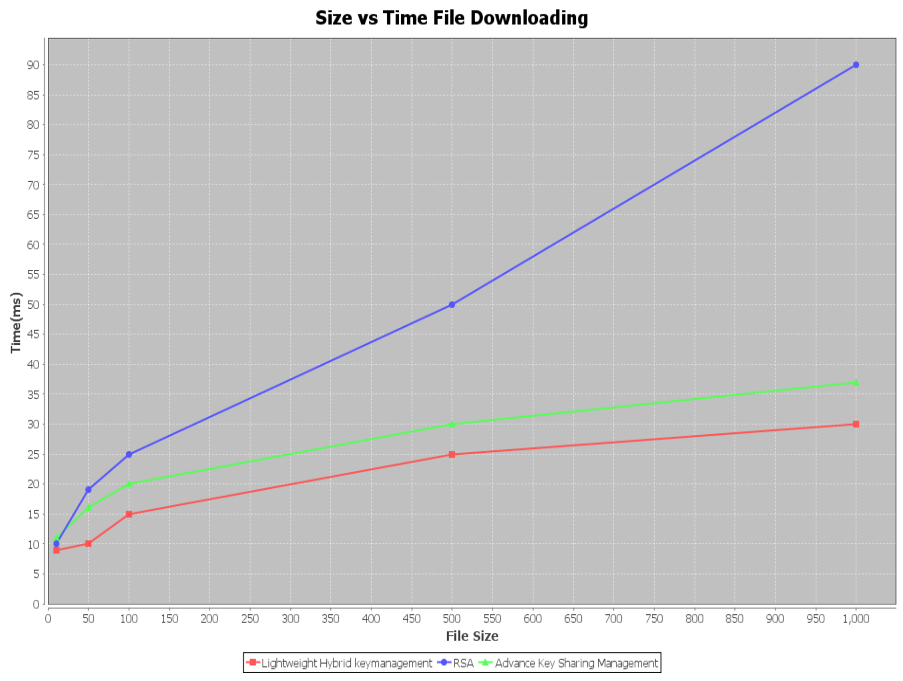

Fig.9 File downloading time analysis of the proposed and existing key management schemes

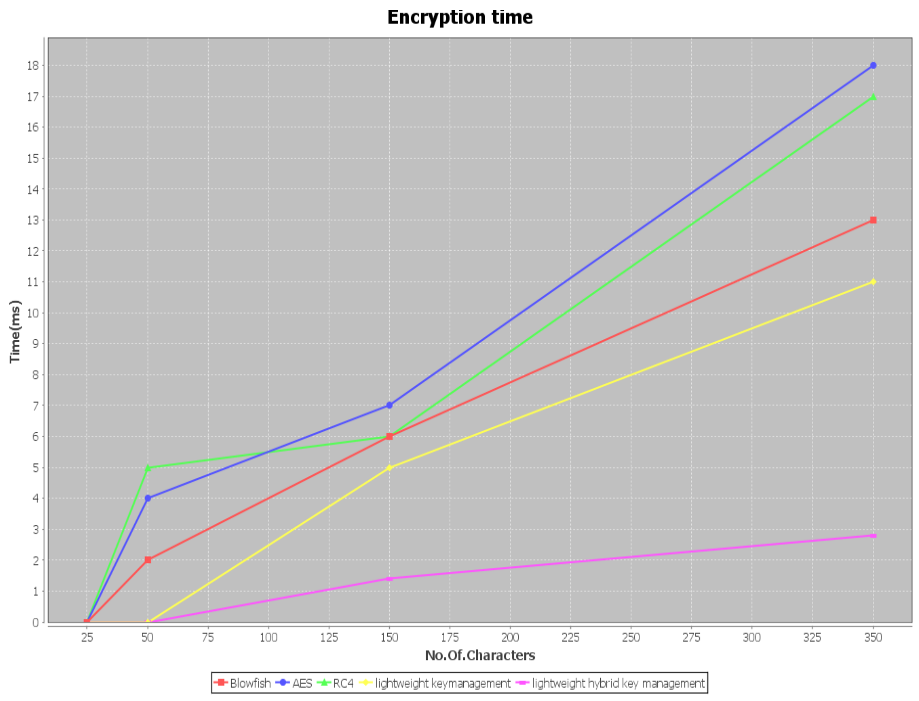

Fig.10 Encryption time of the proposed and existing key management schemes

V. CONCLUSION

The proposed key management scheme is a two level authentication mechanism in which the data owner and cloud service provider independently depend on the two-way session key. The TPA ensures secure data communication between the data owner and cloud service provider using the two-level session key. The aim of the proposed scheme is to resist against the reply, channel, forward and key regeneration attack and ensure secure communication with the data owner and cloud service provider. A lightweight two-level session key is generated from the ID of the data owner and cloud service provider, nonce using the HMAC and XOR operations. The security analysis states the proposed key management scheme yields high security and scalability due to the two-level session key establishment. The proposed key management scheme is compared with the existing cryptographic schemes such as TSK, AES, ECC, ECDSA, SHA-1, RSA, Blowfish, RC4 and lightweight key management scheme. From the comparative analysis, it is observed that the proposed key management scheme requires minimum key size, energy consumption, file uploading time, file downloading time and lower encryption time than the existing schemes.

REFERENCES

[1] C. Blundo, S. Cimato, S. De Capitani di Vimercati, A. De Santis, S. Foresti, S. Paraboschi, et al., "Efficient key management for enforcing access control in outsourced scenarios," Emerging Challenges for Security, Privacy and Trust, pp. 364-375, 2009.
[2] E. Damiani, S. Di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati, "Key management for multi-user encrypted databases," in Proceedings of the 2005 ACM workshop on Storage security and survivability, 2005, pp. 74-83.
[3] E. Damiani, S. Di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati, "Meta resource Management in Outsourced Encrypted Resource bases," Lecture Notes in Computer Science, pp. 1-17, 2005.
[4] E. Damiani, S. Di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati, "Selective resource encryption in outsourced dynamic environments," Electronic Notes in Theoretical Computer Science, pp. 1-15, 2006.
[5] A. Singh, M. Srivatsa, and L. Liu, "Search-as-a-service: Outsourced search over outsourced storage," ACM Transactions on the Web (TWEB), vol. 3, p. 13, 2009.
[6] S. D. C. Di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati, "A data outsourcing architecture combining cryptography and access control," in Proceedings of the 2007 ACM workshop on Computer security architecture, 2007, pp. 63-69.
[7] S. D. C. Di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati, "Over-encryption: Management of access control evolution on outsourced data," in Proceedings of the 33rd international conference on Very large data bases, 2007, pp. 123-134.
[8] S. D. C. D. Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati, "Encryption policies for regulating access to outsourced data," ACM Transactions on Database Systems (TODS), vol. 35, p. 12, 2010.
[9] B. Parno, M. Raykova, and V. Vaikuntanathan, "How to Delegate and Verify in Public: Verifiable Computation from Attribute-Based Encryption," in TCC, 2012, pp. 422-439.
[10] Y. Tang, P. P. Lee, J. C. Lui, and R. Perlman, "Secure overlay cloud storage with access control and assured deletion," IEEE Transactions on dependable and secure computing, vol. 9, pp. 903-916, 2012.
[11] G. Wang, Q. Liu, and J. Wu, "Hierarchical attribute-based encryption for fine-grained access control in cloud storage services," in Proceedings of the 17th ACM conference on Computer and communications security, 2010, pp. 735-737.
[12] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable, and fine-grained data access control in cloud computing," in Infocom, 2010 proceedings IEEE, 2010, pp. 1-9.
[13] Y. Zhu, H. Hu, G.-J. Ahn, M. Yu, and H. Zhao, "Comparison-based encryption for fine-grained access control in clouds," in Proceedings of the second ACM conference on Data and Application Security and Privacy, 2012, pp. 105-116.
[14] H. Pang, J. Zhang, and K. Mouratidis, "Scalable verification for outsourced dynamic databases," Proceedings of the VLDB Endowment, vol. 2, pp. 802-813, 2009.
[15] A. Singh and L. Liu, "Sharoes: A data sharing platform for outsourced enterprise storage environments," in IEEE 24th International Conference on Data Engineering, 2008. ICDE 2008., 2008, pp. 993-1002.
[16] H. Wang and L. V. Lakshmanan, "Efficient secure query evaluation over encrypted XML databases," in Proceedings of the 32nd international conference on Very large data bases, 2006, pp. 127-138.
[17] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in IEEE Symposium on Security and Privacy, 2007. SP'07., 2007, pp. 321-334.
[18] S. Fugkeaw and H. Sato, "An extended CP-ABE based Access control model for data outsourced in the cloud," in IEEE 39th Annual Computer Software and Applications Conference (COMPSAC), 2015, pp. 73-78.
[19] Y. Kawai, "Outsourcing the Re-encryption Key Generation: Flexible Ciphertext-Policy Attribute-Based Proxy Re-encryption," in ISPEC, 2015, pp. 301-315.
[20] J. Lai, R. H. Deng, Y. Yang, and J. Weng, "Adaptable ciphertext-policy attribute-based encryption," in International Conference on Pairing-Based Cryptography, 2013, pp. 199-214.
[21] K. Yang, X. Jia, K. Ren, B. Zhang, and R. Xie, "DAC-MACS: Effective data access control for multiauthority cloud storage systems," IEEE Transactions on Information Forensics and Security, vol. 8, pp. 1790-1801, 2013.
[22] Z. Mahmood, H. Ning, and A. Ghafoor, "Lightweight Two-Level Session Key Management for End User Authentication in Internet of Things," in IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), 2016, pp. 323-327.
[23] J. Li, X. Chen, M. Li, J. Li, P. P. Lee, and W. Lou, "Secure deduplication with efficient and reliable convergent key management," IEEE transactions on parallel and distributed systems, vol. 25, pp. 1615-1625, 2014.
[24] T.-Y. Wu, C. Zhou, E. K. Wang, J.-S. Pan, and C.-M. Chen, "Towards time-bound hierarchical key management in cloud computing," in Intelligent Data analysis and its Applications, Volume I, ed: Springer, 2014, pp. 31-38.
[25] C.-K. Chu, S. S. Chow, W.-G. Tzeng, J. Zhou, and R. H. Deng, "Key-aggregate cryptosystem for scalable data sharing in cloud storage," IEEE transactions on parallel and distributed systems, vol. 25, pp. 468-477, 2014.
[26] P. K. Tysowski and M. A. Hasan, "Hybrid attribute-and re-encryption-based key management for secure and scalable mobile applications in clouds," IEEE Transactions on Cloud Computing, vol. 1, pp. 172-186, 2013.
[27] M. Li, S. Yu, Y. Zheng, K. Ren, and W. Lou, "Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption," IEEE transactions on parallel and distributed systems, vol. 24, pp. 131-143, 2013.
[28] J. Baek, Q. H. Vu, J. K. Liu, X. Huang, and Y. Xiang, "A secure cloud computing based framework for big data information management of smart grid," IEEE transactions on cloud computing, vol. 3, pp. 233-244, 2015.

[29] Y.-W. Kao, K.-Y. Huang, H.-Z. Gu, and S.-M. Yuan, "uCloud: a user-centric key management scheme for cloud data protection," IET Information Security, vol. 7, pp. 144-154, 2013.
[30] J. Zhou, Z. Cao, X. Dong, N. Xiong, and A. V. Vasilakos, "4S: A secure and privacy-preserving key management scheme for cloud-assisted wireless body area network in m-healthcare social networks," Information Sciences, vol. 314, pp. 255-276, 2015.
[31] Y. Xie, Y. Jiang, R. Liao, H. Wen, J. Meng, X. Guo, et al., "A hierarchical key management system applied in cloud-based smart grid," in IEEE/CIC International Conference on Communications in China-Workshops (CIC/ICCC),, 2015, pp. 22-26.
[32] F. Zhao, C. Li, and C. F. Liu, "A cloud computing security solution based on fully homomorphic encryption," in 16th International Conference on Advanced Communication Technology (ICACT),, 2014, pp. 485-488.
[33] Z. Cui, H. Zhu, and L. Chi, "Lightweight key management on sensitive data in the cloud," Security and communication networks, vol. 6, pp. 1290-1299, 2013.
[34] W. Ren, J. Lin, Q. Cao, and L. Ma, "LIFE: a lightweight and flexible key management scheme for securely and pervasively file editing in mobile cloud computing," International Journal of Internet Protocol Technology, vol. 8, pp. 122-129, 2014.
[35] Z. Cui, H. Lv, C. Yin, G. Gao, and C. Zhou, "Efficient key management for IOT owner in the cloud," in Big Data and Cloud Computing (BDCloud), 2015 IEEE Fifth International Conference on, 2015, pp. 56-61.
[36] X. Yao, X. Han, and X. Du, "A lightweight access control mechanism for mobile cloud computing," in IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS),, 2014, pp. 380-385.
[37] S. Belguith, A. Jemai, and R. Attia, "Enhancing Data Security in Cloud Computing Using a Lightweight Cryptographic Algorithm," in The Eleventh International Conference On Autonomic and Systems, 2015, pp. 98-103.
[38] R. Goyal and N. Sidhu, "Third Party Auditor: An Integrity Checking Technique for Client Data Security in Cloud Computing," International Journal of Computer Science & Information Technologies, vol. 5, 2014.
[39] T. T. Tejaswi, M. Azharuddin, and P. K. Jana, "A GA based approach for task scheduling in multi-cloud environment," arXiv preprint arXiv:1511.08707, 2015.
[40] A. Mishra, "Data Security in Cloud Computing Based on Advanced Secret Sharing Key Management Scheme," 2014.
[41] D. C. Verma, A. Mohapatra, and K. Usmani, "Light weight encryption technique for group communication in cloud computing environment," International Journal of Computer Applications, vol. 49, 2012.