JES
Journal of Engineering Sciences

Vol 10, Issue 11, Nov /2019
ISSN NO:0377-9254

IMPLEMENTATION OF PRIVACY PROTECTION ACCESS CONTROL AND KEYWORD SEARCH SCHEME FOR CLOUD STORAGE SYSTEMS

J. VAMSINATH[1], B. TEJASWINI[2]

[1]Assoc. Professor, Dept. of CSE, PBR VITS, Kavali, A.P, India

[2]M.Tech, Dept. of CSE, PBR VITS, Kavali, A.P, India

Abstract – Secure search over encrypted remote data is crucial in cloud computing to guarantee the data privacy and usability. To prevent unauthorized data usage, fine-grained access control is necessary in multi-user system. However, authorized user may intentionally leak the secret key for financial benefit. Thus, tracing and revoking the malicious user who abuses secret key needs to be solved imminently. In this paper, we propose a Privacy Protection Access Control and Keyword Search Scheme (PP-ACKS). The key escrow free mechanism could effectively prevent the key generation centre (KGC) from unscrupulously searching and decrypting all encrypted files of users. Also, the decryption process only requires ultra lightweight computation, which is a desirable feature for energy-limited devices. In addition, efficient user revocation is enabled after the malicious user is figured out. Moreover, the proposed system is able to support flexible number of attributes rather than polynomial bounded. Flexible multiple keyword subset search pattern is realized, and the change of the query keywords order does not affect the search result. Security analysis indicates that PP-ACKS is provably secure. Efficiency analysis and experimental results show that PP-ACKS improves the efficiency and greatly reduces the computation overhead of users' terminals.

Keywords – Privacy protection, Access Control, Cloud Computing, Key Generation.

I. INTRODUCTION

With the development of new computing paradigm, cloud computing [1] becomes the most notable one, which provides convenient, on-demand services from a shared pool of configurable computing resources. Therefore, an increasing number of companies and individuals prefer to outsource their data storage to cloud server. Despite the tremendous economic and technical advantages, unpredictable security and privacy concerns [2], [3] become the most prominent problem that hinders the widespread adoption of data storage in public cloud infrastructure.

Encryption is a fundamental method to protect data privacy in remote storage [4]. However, how to effectively execute keyword search for plaintext becomes difficult for encrypted data due to the unreadability of ciphertext. Searchable encryption provides mechanism to enable keyword search over encrypted data [5], [6].

For the file sharing system, such as multi-owner multiuser scenario, fine-grained search authorization is a desirable function for the data owners to share their private data with other authorized user. However, most of the available systems [7], [8] require the user to perform a large amount of

complex bilinear pairing operations. These overwhelmed computations become a heavy burden for user's terminal, which is especially serious for energy constrained devices.

The authorized entities may illegally leak their secret key to a third party for profits [13]. Suppose that a patient someday suddenly finds out that a secret key corresponding his electronic medical data is sold on e-Bay. Such despicable behavior seriously threatens the patient's data privacy. Even worse, if the private electronic health data that contain serious health disease is abused by the insurance company or the patient's employment corporation, the patient would be declined to renew the medical insurance or labor contracts. The intentional secret key leakage seriously undermines the foundation of authorized access control and data privacy protection. In attribute based access control system, the secret key of user is associated with a set of attributes rather than individual's identity. As the search and decryption authority can be shared by a set of users who own the same set of attributes, it is hard to trace the original key owner. Providing traceability to a fine-grained search authorization system is critical and not considered in previous searchable encryption systems.

In this paper, we propose a novel primitive: a Privacy Protection Access Control and Keyword Search Scheme (PP-ACKS).

II. RESEARCH METHOD

In order to provide an easier way to implement PP-ACKS, we design a traceable attribute based multiple keywords subset search system with verifiable outsourced decryption (TAMKS-VOD), where KGC is responsible to generate user's public/secret key pair like in traditional PEKS schemes.

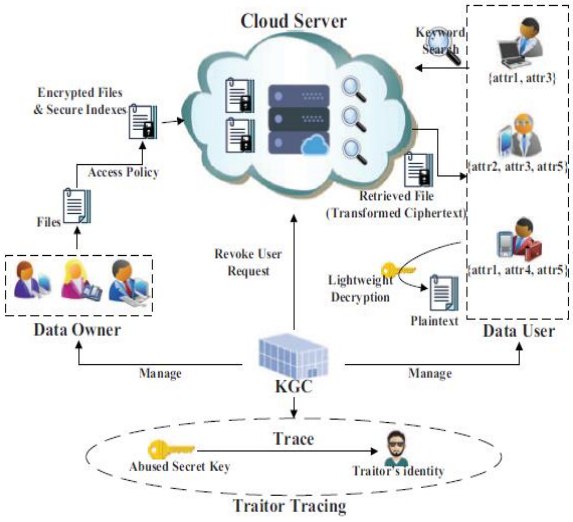


Fig. 1: System Model

A) System Model

The system model of TAMKS-VOD is presented in Fig. 1, and the formal definition is provided in Section A in the Supplemental Materials. The system comprises of four entities, whose responsibilities and interactions are described below.

(1) Key generation centre (KGC). KGC is responsible to generate the public parameter for the system and the public/secret key pairs for the users. Once the user's secret key is leaked for profits or other purposes, KGC runs trace algorithm to find the malicious user. After the traitor is traced, KGC sends user revocation request to cloud server to revoke the user's search privilege.

(2) Cloud server (CS). Cloud server has tremendous storage space and powerful computing capability, which provides on-demand service to the system. Cloud server is responsible to store the data owner's encrypted files and respond on data user's search query.

(3) Data owner. Data owner utilizes the cloud storage service to store the files. Before the data outsourcing, the data owner extracts keyword set from the file and encrypts it into secure index. The document is also encrypted to ciphertext. During the encryption process, the access policy is specified and embedded into the ciphertext to realize fine-grained access control.

(4) Data user. Each data user has attribute set to describe his characteristics, such as professor, computer Science College, dean, etc. The attribute set is embedded into user's secret key. Using the secret key, data user is able to search on the encrypted files stored in the cloud, i.e., chooses a keyword set that he wants to search. Then, the keyword is encrypted to a trapdoor using user's secret key. If the user's attribute set satisfies the access policy defined in the encrypted files, the cloud server responds on user's search query and finds the match files. Otherwise, the search query is rejected. After the match files are returned, the user runs decryption algorithm to recover the plaintext.

B) TAMKS-VOD Workflow

TAMKS-VOD has eight algorithms Setup, KeyGen, CreteUL, Enc, Trapdoor, Test & Transform, Dec, Key Sanity Check &Trace, and the system workflow is shown in Fig. 2.

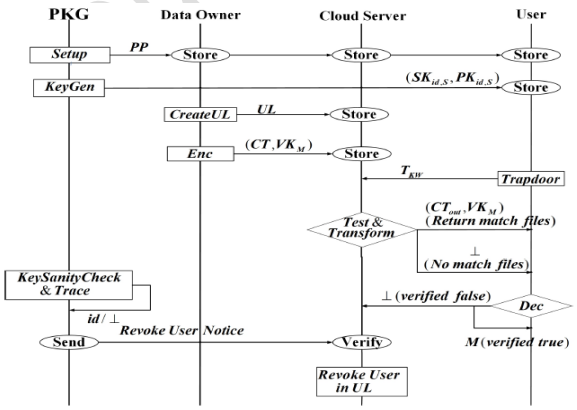


Fig. 2: System Workflow

(1) In of the system establishment phase, KGC runs Setup algorithm to generate the public parameter PP and master secret key MSK of the system. The master secret key MSK is kept secret by KGC. The system public parameter PP is disseminated to cloud server, data owners and users.

(2) For a system user (including data owner and data user) with attribute set S and identity id, KGC runs $KeyGen$ algorithm to generate an attribute public key $PK_{id,S}$ and secret key $SK_{id,S}$, in which the users' identity id and attribute set S are implicitly embedded. The attribute set S describes the characteristic of the user's identity id.

(3) A data user list UL is stored by the cloud server. The data owner runs $CreateUL$ algorithm to generate a pseudonym ζ_{id} and a parameter $\widetilde{D}_{id}$ for each authorized user with identity id. The tuple $(\zeta_{id}, \widetilde{D}_{id})$ is inserted into UL, which is used in the following $Test\&Transform$ algorithm and user revocation phase.

(4) The data owner runs Enc algorithm to encrypt the file M and the extracted keyword set KW. In this process, an access policy (A, ρ) is specified to define the set of authorized users, which is embedded into the ciphertext. Meanwhile, a verification key VK_M is generated in the Enc algorithm, which is used to verify the correctness of the transformed ciphertext CT_{out} that is generated by the cloud server in the following $Test\&Transform$ algorithm. The encrypted files, secure keyword set indexes and verification key are outsourced to cloud server.

(5) In the query phase, data user specifies a query keyword set KW' and runs $Trapdoor$ algorithm to generate a trapdoor $T_{KW'}$ using

his attribute secret key $SK_{id,S}$. Data user's attribute set *S* is implicitly embedded into the trapdoor. Then, the data user submits $T_{KW'}$ to the cloud server.

(6) Receiving the search query from the data user, the cloud server runs $Test\&Transform$ algorithm to search on the data owner's encrypted files. The $Test\&Transform$ algorithm is divided into two algorithms, i.e., Test algorithm and Transform algorithm. In the Test algorithm, CS tests whether the query keyword set KW' (implicitly embedded in $T_{KW'}$) is a subset of *KW* (implicitly embedded in *CT*) and whether the attribute set S (implicitly embedded in $T_{KW'}$) satisfies the access policy (A, ρ) (implicitly embedded in *CT*). If one of the two conditions does not satisfy, the Test algorithm outputs "0" and the $Trasform$ algorithm outputs a symbol $\perp$ indicating that they do not match. If both of the two conditions satisfy, the Test algorithm outputs "1" indicating that the ciphertext *CT* matches with the trapdoor $T_{KW'}$. Then, the $Trasform$ algorithm outputs a transformed ciphertext CT_{out}, so that the data user can recover the plaintext *M* using a lightweight calculation in the following Dec algorithm. The transformed ciphertext CT_{out} and the corresponding verification key VK_M are returned to the data user.

(7) In Dec algorithm, the data user verifies whether the transformed ciphertext CT_{out} is correct using the verification key VK_M. If invalid, a symbol $\perp$ is returned to cloud server. Otherwise, the data user executes lightweight computation to recover the message *M*.

(8) If a secret key is sold for beneficial gain, $KeySanityCheck\&Trace$ algorithm is run by *KGC* to check the validity of the key. If the secret key is not well-formed, $KeySanityCheck$ algorithm outputs 0, and Trace algorithm outputs a symbol $\perp$ Otherwise, $KeySanityCheck$ algorithm outputs 1, and $Trace$ algorithm recovers the real identity of the sold secret key's owner.

(9) After the traitor is traced, *KGC* sends a revocation request to *CS* to revoke the user.

C) PP-ACKS

In TAMKS-VOD, all users' secret keys are generated by the fully trusted KGC, which brings another security concern. Knowledge of all the secret keys, KGC can search on all data owners' encrypted files. Besides, when a secret key is discovered to be sold, the traced secret key owner may argue that the sold key is possible to be leaked by KGC. The above security risk is the key escrow problem.

In order to resolve the key escrow problem, we design an escrow free TAMKS-VOD system, which is denoted as PP-ACKS. In this system, user's secret key is generated by the interaction between KGC and CS. Assumed that KGC and CS do not collude with each other.

i) Concrete Construction

The improved protocol generates master secret keys for KGC and CS utilizing $KGC.Setup$ and $CS.Setup$ algorithms, respectively. A secure two-party computation between KGC and CS is introduced to generate user's secret key. None of them is capable to generate the whole secret key of user independently.

PP-ACKS has different "system initialization" and "new user authorization"

mechanisms from TAMKSVOD, and the other operations are the same.

System Initialization

$KGC.Setup(\kappa) \rightarrow (PP_1, MSK_1)$. Taken a security parameter as input, KGC chooses random elements $\alpha_1, \beta, \lambda \in R\, Z_p^*, f \in R\, G, k_1, k_2 \in R\, \mathcal{K}$ and computes $Y_1 = e(g,g)^{\alpha_1}$. The public parameter and master secret key of KGC are denoted as $PP_1 = \left(f, g, g^{\beta}, g^{\lambda}, Y_1\right)$ and $MSK_1 = (\alpha_1, \beta, \lambda, k_1, k_2)$.

$CS.Setup(\kappa) \rightarrow (PP_2, MSK_2)$ Taken a security parameter as input, CS chooses random elements $\alpha_2 \in_R Z_p^*$ and computes $Y_2 = e(g,g)^{\alpha_2}$. The public parameter and master secret key of CS are denoted as $PP_2 = Y_2$ and$MSK_2 = \alpha_2$.

New User Authorization

When a user applies to join EF-TAMKS-VOD, KGC assigns an attribute set S for the user according to his identity. Then, KGC and CS interactive with each other to generate the public/secret key for user.

- $KeyGen(MSK_1, MSK_2, id, S) \rightarrow \left(PK_{id,S}, SK_{id,S}\right)$.

(1) CS selects a homomorphic public/secret key pair (hpk, hsk) according to the requirement of fully homomorphic encryption scheme in [45]. hpk is made public and hsk is kept secret by CS. Then, CS sends $W_1 = HEnc_{hpk}(\alpha_2)$ to KGC.

(2) KGC computes $W_2 = \left(W_1 \oplus HEnc_{hpk}(\alpha_1)\right) \otimes HEnc_{hpk}(\beta)$, which is sent to CS.

(3) CS recovers $W_3 = HDec_{hsk}(W_2) = (\alpha_1 + \alpha_2)\beta = \alpha\beta$. Then, CS chooses a random number $\varpi \in_R Z_p^*$ and computes $W_4 = g^{W_3/\varpi}$, which is sent to KGC.

(4) KGC selects random elements $t, \theta \in_R Z_p^*$ and computes $\zeta_{id} = SEnc_{k_1}(id), \delta = SEnc_{k_2}(\zeta_{id} || \theta)$. Then, KGC computes $W_5 = W_4^{\frac{1}{(\lambda+\delta)\beta}} = g^{\frac{\alpha}{(\lambda+\delta)\varpi}}, W_6 = g^{\beta t}$ which is sent to CS.

(5) CS constructs $D_1 = W_5^{\varpi} W_6 = g^{\frac{\alpha}{(\lambda+\delta)\beta}} g^{\beta t}$ and sends $D_1$to user.

(6) KGC selects random elements $x_{id}, D_4 \in_R Z_p^*$ and computes $D_1' = \delta, D_2 = g^t, D_2' = g^{\lambda t}, \forall\, x \in S, D_{3,x} = H(x)^{(\lambda+\delta)t}, Y_{id} = Y^{x_{id}}$, which are sent to user. Then, the secret/public keys of user are $SK_{id,S} = \left(D_1, D_1', D_2, D_2', \{D_{3,x}\}_{x \in S}, D_4, x_{id}\right)$ and $PK_{id,S} = Y_{id}$. The value id is also sent to user and functions as user's pseudonym.

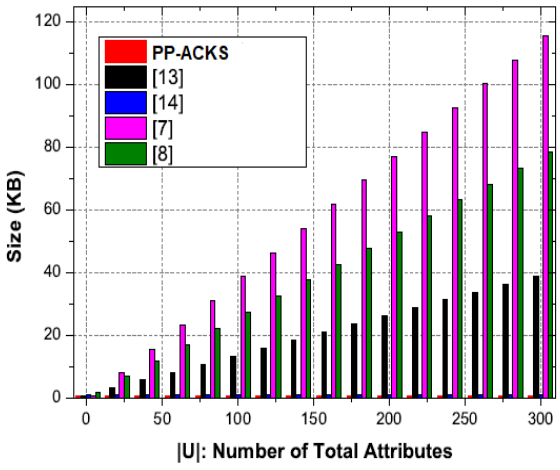
III. RESULT ANALYSIS

We analyze the transmission and computation performance of PP-ACKS scheme from the following two parts. (1) The storage and computation overheads of PP-ACKS are analyzed and compared with other searchable encryption schemes and ABE schemes. (2) We also evaluate the performance of PP-ACKS and other schemes on an experimental workbench.

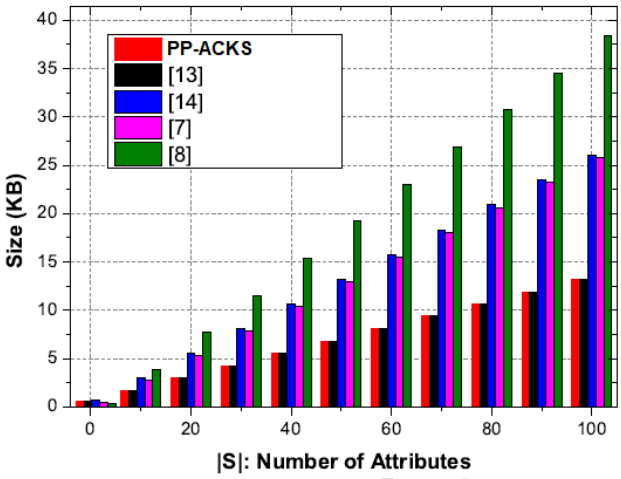
To evaluate the performance, the schemes in [7], [8], [13], and PP-ACKS are simulated using the Stanford Pairing-Based Crypto (PBC) library. The experiments on these schemes are conducted on a laptop running Windows 7 operation system with the following settings: CPU: Intel core i5 CPU at 2.5GHz; physical memory: DDR3 4GB 1333MHz.

The type A elliptic curve parameter is selected for test. It provides 1024-bit discrete log security strength equivalently with the group order of 160-bit. Type A parings are constructed on the curve y2 = x3 + x over the field Zp for some prime p = 3 (mod 4). In the experiment, we select the parameter p =8780710799663312522437781984754049815806883199414208211028653399266475630802229570786251794226622214231558587695823174592777133673174813249251299982247 91, which is provided in PBC library. The core algorithms are executed on the experimental workbench to test the transmission and computation overheads of the schemes in [7], [8], [13], [14] and PP-ACKS. According to the selected parameter in the experiment, we have jZpj = 160 bits, jGj = 1024 bits and jGT j = 1024 bits. The number l1 of the keyword set is fixed to be 5 to do the tests.

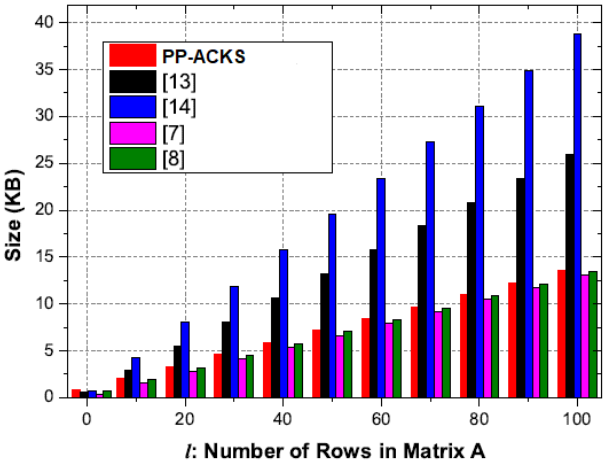
Fig. 3 presents the test results of the transmission overheads of public parameter, secret key, ciphertext and trapdoor.



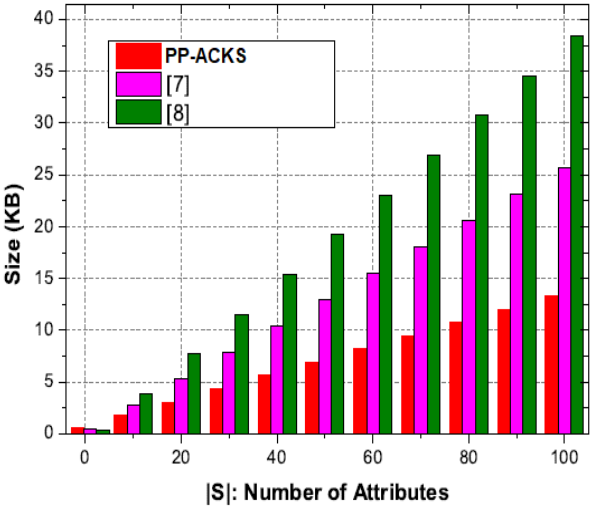
(a) Public Parameter Size



(b) Secret Key Size



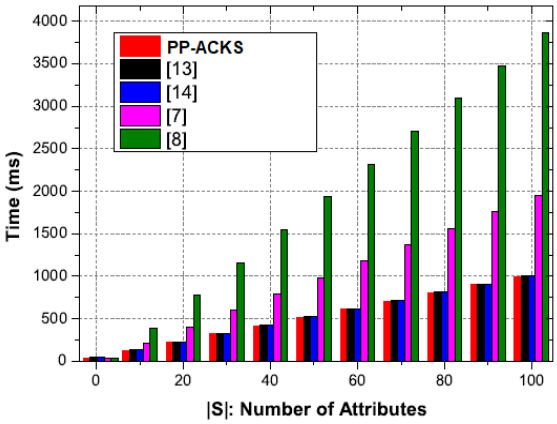
(c) Ciphertext Size



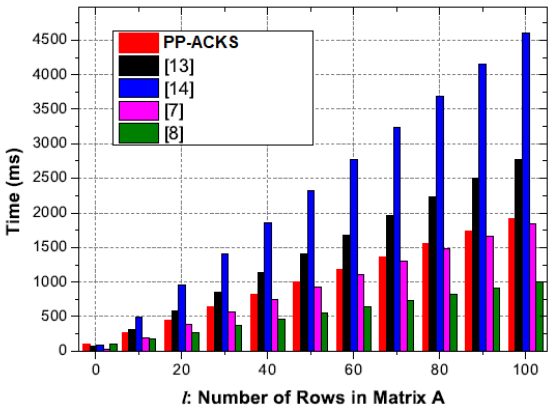
(d) Trapdoor Size

Fig. 3: Transmission and Storage Overheads

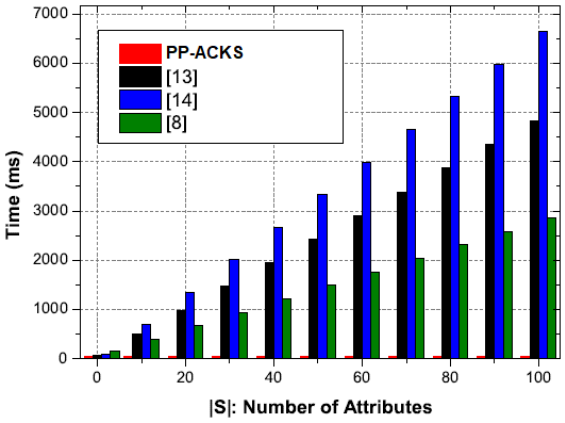
Fig. 4 shows the computation overheads of key generation, encryption, decryption, trapdoor generation, test and transform, and key sanity check and trace algorithms.



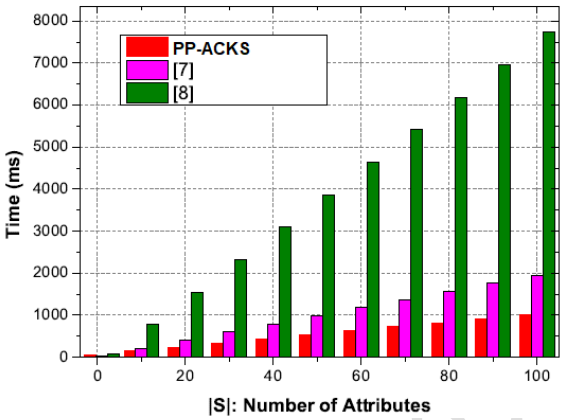
(a) Key Generation Time



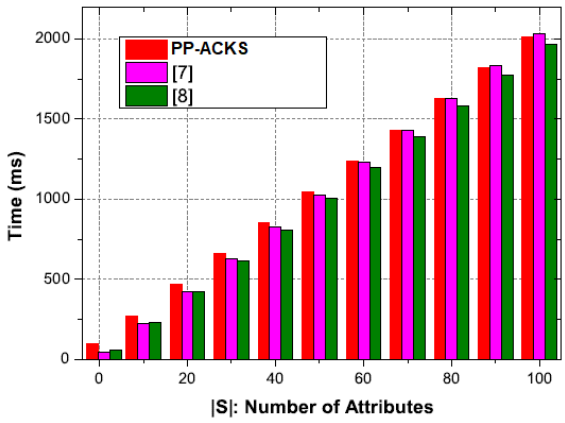
(b) Encryption Time



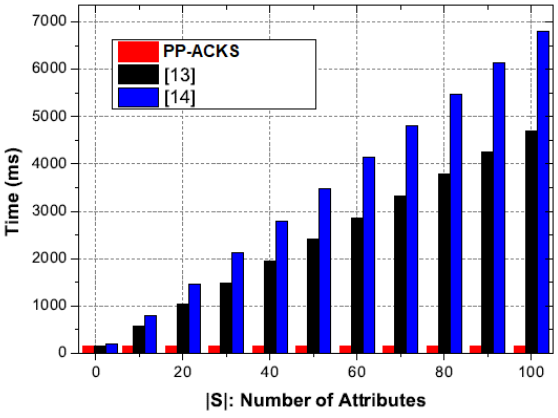
(c) Decryption Time



(d) Trapdoor Generation Time



(e) Test and Transform Time



(f) Sanity Check and Trace Time

Fig. 4: Computation Overheads

Compared with the schemes in [7], [8], [13], PP-ACKS has better efficiency. In Fig. 4(a), the key generation time of the schemes [13] and PP-ACKS is much lower than that of the

schemes [7], [8]. In Fig. 4(b), the encryption time of PP-ACKS is at the same level as the scheme [7], higher than the scheme [8] and lower than the schemes [13], [14]. In Fig. 4(c) and Fig. 4(d), the decryption and trapdoor generation time of PP-ACKS is much lower than that in [7], [8]. Fig. 4(e), the test and transform time of the schemes in[7], [8] and PP-ACKS is on the same level. Fig. 4(f), the key sanity check and trace time of PP-ACKS is much lower than that in [13].

We should concentrate on the algorithms (such as trapdoor generation and decryption algorithms) that are frequently executed by user's terminal, because user's portable device has very limited storage and computation power compared with the cloud server. PP-ACKS is much more efficient in those algorithms, and the experimental result further demonstrates its high efficiency.

IV. CONCLUSION

The implementation of access control and the support of keyword search are important issues in secure cloud storage system. In this work, we defined a new paradigm of searchable encryption system, and proposed a concrete construction. It supports flexible multiple keywords subset search, and solves the key escrow problem during the key generation procedure. Malicious user who sells secret key for benefit can be traced. The decryption operation is partly outsourced to cloud server and the correctness of half-decrypted result can be verified by data user. The performance analysis and simulation show its efficiency in computation and storage overhead.

REFERENCES

1. C. Wang, N. Cao, J. Li, K. Ren, W. Lou. "Secure ranked keyword search over encrypted cloud data"[C]//IEEE 30th International Conference on Distributed Computing Systems (ICDCS), IEEE, 2010: 253-262.

2. Q. Zhang, L. T. Yang, Z. Chen, P. Li, M. J. Deen. "Privacy-preserving Double-Projection Deep Computation Model with Crowdsourcing on Cloud for Big Data Feature Learning," IEEE Internet of Things Journal, 2017, DOI: 10.1109/JIOT.2017.2732735.

3. R. Chen, Y. Mu, G. Yang, F. Guo and X. Wang, "Dual-Server Public- Key Encryption with Keyword Search for Secure Cloud Storage," IEEE Transactions on Information Forensics and Security, 2016, vol. 11, no. 4, 789-798.

4. X. Liu, R.H. Deng, K.K.R. Choo, J. Weng. "An efficient privacypreserving outsourced calculation toolkit with multiple keys." IEEE Transactions on Information Forensics and Security 11.11 (2016): 2401-2414.

5. B. R. Waters, D. Balfanz, G. Durfee, and D. K. Smetters, "Building an encrypted and searchable audit log," in NDSS, 2004.

6. Y. Yang, X. Liu, R.H. Deng, "Multi-user Multi-Keyword Rank Search over Encrypted Data in Arbitrary Language". IEEE Transactions on Dependable and Secure Computing, 2018, publish online, DOI: 10.1109/TDSC.2017.2787588.

7. W. Sun, S. Yu, W. Lou, Y. Hou and H. Li, "Protecting Your Right: Verifiable

Attribute-based Keyword Search with Finegrained Owner-enforced Search Authorization in the Cloud," IEEE Transactions on Parallel and Distributed Systems, 2016, vol. 27, no. 4, pp. 1187-1198.

8. K. Liang, W. Susilo, "Searchable Attribute-Based Mechanism with Efficient Data Sharing for Secure Cloud Storage," IEEE Transactions on Information Forensics and Security, 2015, vol. 10, no. 9, pp. 1981- 1992.

9. M. Green, S. Hohenberger, and B.Waters, "Outsourcing the decryption of ABE ciphertexts," in USENIX Security Symposium, ACM, 2011, pp. 34-34.

10. J. Lai, R. H. Deng, C. Guan, and J. Weng, "Attribute-based encryption with verifiable outsourced decryption," IEEE Transactions on Information Forensics and Security, 2013, vol. 8, no. 8, pp. 1343- 1354.

11. B. Qin, R. H. Deng, S. Liu, and S. Ma, "Attribute-Based Encryption with Efficient Verifiable Outsourced Decryption," IEEE Transactions on Information Forensics and Security, 2015, vol. 10, no. 7, pp. 1384-1394.

12. D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in: EUROCRYPT, 2004, pp. 506-522.

13. Z. Liu, Z. Cao, D.S. Wong, "White-box traceable ciphertext-policy attribute-based encryption supporting any monotone access structures," IEEE Transactions on Information Forensics and Security, 2013, vol. 8, no. 1, pp. 76-88.

AUTHORS

J. VAMSINATH has received his B.Tech in Information Technology and M.Tech degree in Computer science from JNTU, Hyderabad in 2005 and JNTU, Hyderabad in 2009 respectively. He is pursuing Ph. D in JNTU, Kakinada. He is dedicated to teaching field from the last 14 years. He has guided 15 P.G and 120 U.G student batches. His research areas included Data Mining and Machine Learning. At present he is working as Associate Professor in PBR VITS, Kavali, and Andhra Pradesh, India.

B. Tejaswini has received her B.Tech degree in CSE from PBR VITS, Kavali affiliated to JNTU, Anantapur in 2017 and pursuing M.Tech degree in CSE from PBR VITS, Kavali affiliated to JNTU, Anantapur in 2019.