Implementing a Secure Data Storage on the Cloud under Key Exposure State

NADIPALLI UMA SATYA VENKATESH[#1] & K.RAMBABU [#2] & D.D.D.SURIBABU [#3]
[#1] M.Sc Student, Master of Computer Science, D.N.R. College, P.G.Courses & Research Center, Bhimavaram, AP, India.
[#2] Assistant Professor, Master of Computer Science, D.N.R. College, P.G.Courses & Research Center, Bhimavaram, AP, India.
[#3] Head & Associate Professor, Dept of CSE, D.N.R. College of Engineering, Bhimavaram, AP, India.

Received: January 11, 2019 Accepted: February 14, 2019

ABSTRACT: *Now a day's cloud computing has become one of the fascinating domain which was accessed by almost all users in order to store ,retrieve and access the data from remote systems rather than from the local machines. We propose a novel and efficient scheme that guarantees data confidentiality even if the encryption key is leaked inside the network level and the adversary has access to almost all cipher text blocks. In this proposed thesis we try to analyze the security of our proposed application and try to evaluate the performance by means of a key exposure prototype implementation. Here we try to show the performance of our proposed application by integrating the current application with a real time cloud real time cloud server like drivehq cloud for storing the data in a secure manner.*

Key Words*: Drivehq.com, Cipher Text, Key Exposure, Efficient Schemes, Intruder, Confidentiality.*

I. Introduction

As we all know that in recent days there was a lot of user's attention towards the cloud data storage for storing and retrieving the data to and from the cloud server. As the data is been increasing day by day almost all the companies are unable to store their valuable data on their own individual devices, so in this situation they opt for a new data storage area known as Cloud Data Storage [1], [2]. Generally cloud service providers allow the users to access their services for a low economical and ascendable marginal cost compared with primitive data storage services. Generally the data which is stored in the cloud server is mainly used for sharing within the users of same group or between the users of different group with a valid authentication. Some of the best cloud data storage services are as follows: Google Drive, DriveHq Server, DropBox and ICloud. As these all are the best among various types of cloud service providers in which the data can be stored either in public cloud or private cloud, sometimes can be stored in both combine known as Hybrid Cloud. In our current application we try to use the DRIVEHQ hybrid cloud for data storage and accessing in a real time environment manner[3].

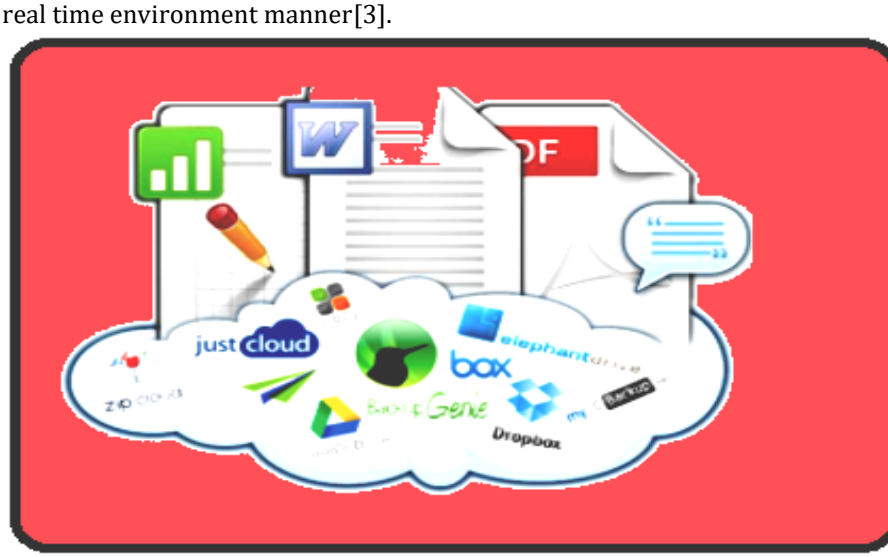

Figure.1. Denotes the Architecture of Various Cloud Service Providers

From the figure 1, we can clearly find out that there are various cloud service providers that are available in the real time environment that are used for storing various applications like word documents,pdf,excel and many more files. If you look at the above figure you can find out the various cloud service providers like Drivehq.com Zip Cloud, Just Cloud, BOX, Google Drive, DROP BOX and a lot more.

II. Related Work

In this section we mainly discuss about the related work that is carried out in order to design the proposed application secure data sharing under key exposure state.

Motivation:

All the existing cloud servers try to store the data in a plain text manner rather than in a encrypted manner. If the encryption key is exposed, the only viable means to guarantee confidentiality[4] is to limit the adversary's access to the cipher text, e.g., by spreading it across multiple administrative domains, in the hope that the adversary cannot compromise all of them. However, even if the data is encrypted and dispersed across different administrative domains, an adversary equipped with the appropriate keying material can compromise a server in one domain and decrypt cipher text blocks stored therein[5].

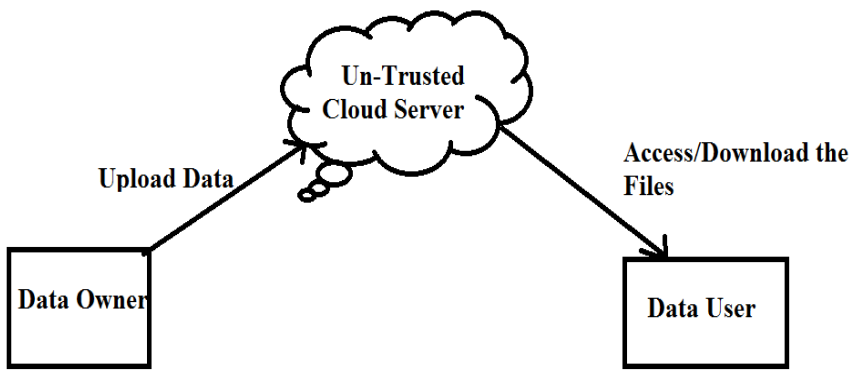

Figure.2. Denotes the Architecture of Primitive Cloud Data Sharing

Limitations with the Current Cloud Data Storage

All the primitive cloud data storage schemes, however, require *at least* two rounds of block cipher encryptions on the data: one preprocessing round to create the AONT, followed by another round for the actual encryption. Notice that these rounds are sequential, and cannot be parallelized[6]. This results in considerable—often unacceptable—overhead to encrypt and decrypt large files. On the other hand, there is no technique in the primitive clouds which takes only one round of encryption for making the data well-suited to be integrated in existing primitive storage systems.From the above figure 2 ,we can clearly show that all the primitive cloud servers try to store their valuable data inside the cloud and there is no concept like encryption inside the primitive cloud servers.Hence any data user can able to access the files despite of restricting the un-authorized users not to access all the files from the cloud server[7].

Data Correctness Verification

All the traditional approach for checking data correctness is to retrieve the whole data from the cloud, and then verify data integrity by checking the correctness of signatures (e.g., RSA [7]) or hash values (e.g., MD5 [8]) of the entire data. Certainly, this conventional approach is able to successfully check the correctness of cloud data. However, the efficiency of using this traditional approach on cloud data is not accurate [9]. The reason behind the in accuracy of the current methods is, the audit can be done on the cloud files based on the individual keys that are granted for the cloud user and no cloud server is able to restrict the keys of one user not to be accessed by the other user. Hence it is a tedicious job for the cloud audit department to identify which file is been attacked by the un-authorized users and which one is in normal state. And one more reason is that the size of cloud data is large in general. Downloading the entire cloud data to verify data integrity will cost or even waste user's amounts of computation and communication resources, especially when data have been corrupted in the cloud. Besides, many uses of cloud data (e.g., data mining and machine learning) do not necessarily need users to download the entire cloud data to local devices [10]. It is because cloud providers, such as Amazon, can offer users computation services directly on large-scale data that already existed in the cloud.

III. Proposed Cloud Data Sharing Under Key Exposure State

In this section we mainly try to discuss about our proposed secure data sharing in cloud under key exposure state.

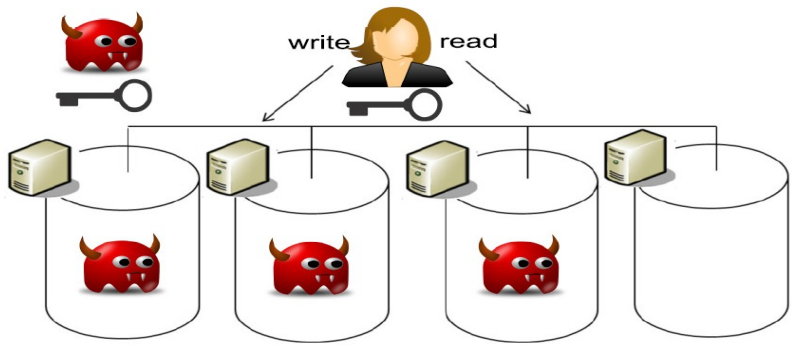

Figure. 3. Denotes the Proposed Secure Data Sharing Under Key Exposure State

Here in the proposed system we study data confidentiality against an adversary which knows the encryption key and has access to a large fraction of the ciphertext blocks. The adversary can acquire the key either by exploiting flaws or backdoors in the key-generation software, or by compromising the devices that store the keys (e.g., at the user-side or in the cloud). To counter such an adversary, we propose a novel and efficient scheme which ensures that plaintext data cannot be recovered as long as the adversary has access to at most all but *two* ciphertext blocks[11], even when the encryption key is exposed. We evaluate the performance of our proposed approach in comparison with a number of existing encryption schemes. Our results show that our proposed method only incurs a negligible performance deterioration (less than 5%) when compared to symmetric encryption schemes, and considerably improves the performance of existing various encryption schemes. We propose an efficient scheme which ensures data confidentiality against an adversary that knows the encryption key and has access to a large fraction of the cipher text blocks. Here in our proposed system we use a polynomial time based key generation method in which single round is required to convert the data into encrypted manner rather than multiple rounds. Also if any adversary try to gain illegal access over files ,he can't able to gain the files without proper access. From the above figure 3 ,we can clearly find out if any un-authorized user try to access the files without proper access permission, he/she cannot able to view the files which he don't have access rights. The files which he is having the permissions can only be accessed and all the remaining files he can't able to access even if the keys of that files are exposed.

IV. Implementation Phase

Implementation is a stage where the theoretical design is converted into programmatically manner. Here in this stage the application is mainly divided into several modules and each and every module differs from one another. In this proposed thesis, we try to divide the application into following 4 modules. Now let us discuss about these modules in detail as follows:

1) System Construction Module

In this module we try to construct an system with following attributes like : Single admin with multiple data owners and multiple data users. Here the data owner try to upload the files into the cloud server and before he upload he try to encrypt the data with a polynomial based encryption and then try to add those files into cloud server. Here the data users are those who try to enter into their account and search for the files which are uploaded by the various data owners. Once the files are found they will request for the keys and try to download the files.Here we try to use a live public cloud account from DRIVEHQ.com site for showing that data is stored in a secure manner .

2) Data Owner Module

In Data Owner module, Initially Data Owner must have to register their detail and admin will approve the registration by sending signature key and private key through email. After successful login he/she have to verify their login by entering signature and private key. Then data Owner can upload files into cloud server with Polynomial key generation. He/she can view the files that are uploaded in cloud by entering the secret file key.

3) Data User Module

In Data User module, Initially Data Users must have to register their detail and admin will approve the registration by sending signature key and private key through email. After successful login he/she have to verify their login by entering signature and private key. Data Users can search all the files upload by data owners. He/she can send search request to admin then admin will send the search key. After entering the search key he/she can view the file.

4) Admin Module

In Admin module, Admin can view all the Data owners and data user's details. Admin will approve the users and send the signature key and private key to the data owners and data users. Also admin will send the search request key to the users. Admin can able see the files in cloud uploaded by the data owners.

V.Conclusion

In this proposed thesis we finally implemented a novel and efficient scheme that guarantees data confidentiality even if the encryption key is leaked inside the network level and the adversary has access to almost all cipher text blocks. In this proposed thesis we try to analyze the security of our proposed application and try to evaluate the performance by means of a key exposure prototype implementation. By conducting various experiments on our proposed model we finally came to an conclusion that our proposed approach is best in providing security for the data sharing under key exposure state.

VI. References

1. D. Boneh, B. Lynn, and H. Shacham, "Short Signatures from the Weil Pairing," Proc. Seventh Int'l Conf. Theory and Application of Cryptology and Information Security: Advances in Cryptology (ASIACRYPT'01), pp. 514-532, 2001.
2. A. Beimel, "Secret-sharing schemes: A survey," in International Workshop on Coding and Cryptology (IWCC), 2011, pp. 11–46.
3. R.L. Rivest, A. Shamir, and Y. Tauman, "How to Leak a Secret," Proc. Seventh Int'l Conf. Theory and Application of Cryptology and Information Security: Advances in Cryptology (ASIACRYPT'01),
4. B. Wang, B. Li, and H. Li, "Oruta: Privacy-Preserving Public Auditing for Shared Data in the Cloud," Proc. IEEE Fifth Int'l Conf.Cloud Computing, pp. 295-302, 2012.
5. M. Abd-El-Malek, G. R. Ganger, G. R. Goodson, M. K. Reiter, and J. J. Wylie, "Fault-Scalable Byzantine Fault-Tolerant Services," in ACM Symposium on Operating Systems Principles (SOSP), 2005, pp. 59–74.
6. M. K. Aguilera, R. Janakiraman, and L. Xu, "Using Erasure Codes Efficiently for Storage in a Distributed System," in International Conference on Dependable Systems and Networks (DSN), 2005, pp. 336–345.
7. W. Aiello, M. Bellare, G. D. Crescenzo, and R. Venkatesan, "Security amplification by composition: The case of doublyiterated, ideal ciphers," in Advances in Cryptology (CRYPTO), 1998, pp. 390–407.
8. C. Basescu, C. Cachin, I. Eyal, R. Haas, and M. Vukolic, "Robust Data Sharing with Key-value Stores," in ACM SIGACTSIGOPS Symposium on Principles of Distributed Computing (PODC), 2011, pp. 221–222
9. A. Bessani, M. Correia, B. Quaresma, F. André, and P. Sousa, "DepSky: Dependable and Secure Storage in a Cloud-ofclouds," in Sixth Conference on Computer Systems (EuroSys), 2011, pp. 31–46.
10. G. R. Blakley and C. Meadows, "Security of ramp schemes," in Advances in Cryptology (CRYPTO), 1984, pp. 242–268.
11. V. Boyko, "On the Security Properties of OAEP as an Allor-nothing Transform," in Advances in Cryptology (CRYPTO), 1999, pp. 503–518.