ISSN:2348-2079

Volume-6 Issue-2

International Journal of Intellectual Advancements and Research in Engineering Computations

TPA Authentication Based Security and Verifying Setting in Multicloud Environment

[1]Mr. S. Sambasivam,MCA., Associate Professor,
[2]Mr.J.Meighanam, Final MCA.,
Department of MCA, Nandha Engineering College,Erode 52.
Email ID: sammy2173@gmail.com, meighanamjaganathan@gmail.com

***Abstract*-Cloud computing is provide a dynamically scalable resources provisioned as a service over the webpage. The third-party, on-demand, self-service, pay-per-use, and seamlessly scalable computing resources and services offered by the cloud environment promise to reduce capital as well as operational expenditures for hardware and software. Various distinct architectures are introduced and discussed according to their security and privacy capabilities and prospects. It provides four distinct models in form of abstracted multicloud architectures. These developed multi cloud architectures allow to categorize the available schemes and to analyze them according to their security benefits. To maintain a data in a scalable and security manner key storage goes high modularity. To reduce the key storage, in this dissertation mainly compromise to maintain data in a secure manner and provide with less key management cost and bandwidth of a cloud server. The process of convergent key management, original data copy is first encrypted with a convergent key derived by the data copy itself, and the convergent key is then encrypted by a master key that will be kept locally and securely by each user manner. The encrypted convergent keys are then stored, along with the parallel encrypted data records, in public cloud storage area. Likewise, session based de-duplication is considered. Here if the user provides the session duration, from date and to date, then only with the data range, proof of ownership can be allowed in server on those dates. This increases the security if the secure data storage needs to be safely accessed on the given duration.**

***Keywords*— Cloud Computing, Muticloud, Integrity, Isolation Preserving Auditing, TPA, Secure De duplication Model**

I. INTRODUCTION

Cloud computing creates a large number of security issues and challenges. A list of security threats to cloud computing is presented in . These issues range from the required trust in the cloud provider and attacks on cloud interfaces to misusing the cloud services for attacks on other systems. The main problem that the cloud computing paradigm implicitly contains is that of secure outsourcing of sensitive as well as critical data and processes. When considering using a cloud service, the user must be aware of the fact that all data given to the cloud provider leave the own control and protection sphere. Hence, a strong trust relationship between the cloud provider and the cloud user is considered a general prerequisite in cloud computing. Depending on the political context this trust may touch legal obligations. An attacker that has access to the cloud storage component is able to take snapshots or alter data in the storage. This might be done once, multiple times, or continuously. An attacker that also has access to the processing logic of the cloud can also modify the functions and their input and output data.

Replication of applicationsallows to receive multiple results from one operation performed in distinct clouds and tocompare them within the own premise. This enables the user to get evidence on the integrity of the result. Partition of application System into tiersallows separating the logic from the data. This gives additional protection againstdata leakage due to flaws in the application logicPartition of application logic into fragmentsallows distributing the application logic to distinct clouds. This has twobenefits. First, no

cloud provider learns the complete application logic. Second, no cloud provider learns the overall calculated result of the application. Thus, this leads to data and application confidentiality.

Partition of application data into fragmentsallows distributing fine-grained fragments of the data to distinct clouds. Noneof the involved cloud providers gains access to all the data, which safeguards the data's confidentiality

II. RELATED WORKS

The cloud computing paradigm has been hailed for its promise of enormous cost-saving potential. In spite of this euphoria, the consequences regarding a migration to the cloud need to be thoroughly considered. Amongst many obstacles present, the highest weight is assigned to the issues arising within security. Cloud security is discussions to date mostly focus on the fact that customers must completely trust their cloud providers with respect to the confidentiality and integrity of their data, as well as computation faultlessness. However, another important area is often overlooked: if the Cloud control interface is compromised, the attacker gains immense potency over the customer's data. This attack vector is a novelty as the result of the control interface (alongside with virtualization techniques) being a new feature of the Cloud Computing paradigm, as NIST lists On-demand self-service and Broad network access as essential characteristics of Cloud Computing systems . The main goal of this paper is the investigation and evaluation of security and privacy threats caused by the unawareness of users in the cloud. Although the methods and techniques described in this paper are applicable to arbitrary IaaS providers, they focused on one of the major cloud providers

However, to actually agree on a specific SLA a user first has to assess his organizational risks related to security and resilience . Current solutions that restrict the provision of sensible services to dedicated private, hybrid or so called national clouds do not go far enough as they reduce the user's flexibility when scaling in or out and still force him to trust the cloud provider. Furthermore, private clouds intensify the vendor lock-in problem. Last but not least, there is no support for deciding which services and data could be safely migrated to which cloud. Instead they demanded new methods and technical support to put the user in a position to benefit from the advantages of cloud computing without giving up the sovereignty over his data and applications. In their current work, they followed a system oriented approach focusing on technical means to achieve this goal.

They identified security as a major obstacle that prevents someone to transfer his resources into the cloud. In order to make sound business decisions and to maintain or obtain security certifications, cloud customers need assurance that providers are following sound security practices and behave according to agreed SL As . Thus, their overall goal is the development of a flexible open source cloud platform that integrates all necessary components for the development of user-controlled and -monitored secure cloud environments.

ACCESS CONTROL

Tahoe uses the capability access control model to manage access to files and directories. In Tahoe, a capability is a short string of bits which uniquely identifies one file or directory. Knowledge of that identifier is necessary and sufficient to gain access to the object that it identifies. The strings must be short enough to be convenient to store and transmit, but must be long enough that they are unguessable (this requires them to be at least 96 bits). Such an access scheme is known as \capabilities as keys" or \cryptographic capabilities". This approach allows fine-grained and dynamic sharing of files or directories.

The Tahoe filesystem consists of files and directories. Files can be mutable, such that their contents can change, including their size, or immutable, such that they are writable only once. Each immutable file has two capabilities associated with it, a read capability or read-cap for short, which identifies the immutable file and grants the ability to read its content, and a verify capability or verify-cap, which identifies the immutable file and grants the ability to check its integrity but not to read its contents.For mutable files, there are three capabilities, the read-write-cap, the read-only-cap, and the verify-cap.

Users who have access to a file or directory can delegate that access to other users simply by sharing the capability. Users can also produce a verify-cap from a read-cap, or produce a read-only-cap from a read-write-cap. This is called diminishing a capability. Tahoe is designed to run in software without requiring "Trusted Computing" (also called \Treacherous Computing") hardware. Therefore, the only robust way to constrain users or administrators from certain behavior such as unauthorizedly reading or altering files is to make such behavior require secrets, and to withhold those secrets from people who are not authorized to perform those behaviors.

III. MAIN CONTRIBUTIONS

General Symmetric Encryption process includes
KeyGenSE: is the key generation algorithm that generates K using security parameter.

EncryptSE(K, M) C is the symmetric encryption algorithm that takes the secret and message M and then outputs the ciphertext C.
DecryptSE(K, C): M is the symmetric decryption algorithm that takes the secret K and ciphertext C and then outputs the original message M.

In the existing system, a user (or data owner) derives a convergent key from each original data copy and encrypts the data copy with the convergent key. In addition, the user derives a tag for the data copy, such that the tag will be used to detect duplicates.

Apart from step 1, 2 and 3 it also contains

TagGenCE(M):T(M) is the tag generation algorithm that maps the original data copy M and outputs a tag T(M). It allows TagGenCE to generate a tag from the corresponding ciphertext, by using T(M)=TagGenCE(C), where C=EncryptCE(K, M).

This tag will be used to reuse the key after some time by the same user if the proof of ownership is satisfied. Instead of encrypting the convergent keys on a per-user basis, Dekey constructs secret shares on the original convergent keys (that are in plain) and distributes the shares across multiple KM-CSPs (Key Management-Cloud Service Provider). If multiple users share the same block, they can access the same corresponding convergent key.

All the existing system methods are implemented in proposed system. Using the proposed algorithms, the KM-CSP can efficiently manage the group of users.

In addition, the project also considers the revocation of users in the given group. If the original (first) user of the group intimates the server with a user's (B) revocation, then the server rejects the proof of ownership submitted by that user (B).

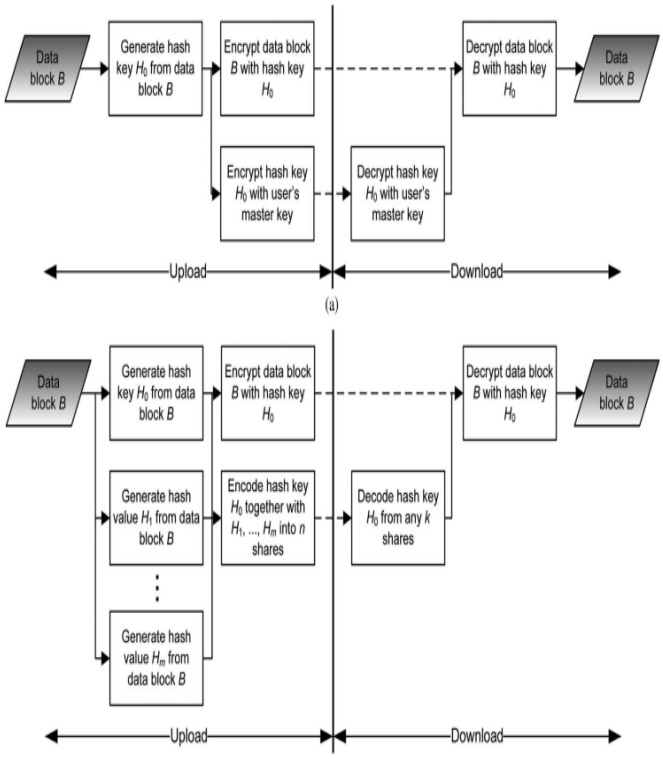
Likewise, session based deduplication is considered. Here if the user provides the session duration i.e, front date and to date, then only with the data range, proof of ownership can be allowed in server on those dates. This increases the security if the outsourced data need to be safely accessed on the given duration.

IV. PROPOSED PROTOCOL

This paper study makes new construction Dekey to provide efficient and reliable convergent key management through convergent key deduplication and secret sharing. Dekey supports both file-level and block-level deduplications. Security analysis is demonstrates that Dekey is secure in terms of the definitions specified in the proposed security model.

In particular, Dekey remains secure even the adversary controls a limited number of key servers. They implement Dekey using the Ramp secret sharing scheme that enables the key management to adapt to different reliability and confidentiality levels. Their evaluation demonstrates that Dekey incurs limited overhead in normal upload/download operations in realistic cloud environments.

Symmetric encryption uses a common secret key to encrypt and decrypt information. Since the key used for this project are very weak, the existing system is less secure. User revocation management is not implemented. The key can be management only within the group members.



PROTOCOL PROCESS

A. MULTI-CLOUD SECURITY

In this first step process, the cloud node id and the cloud provider name is added. There are more cloud nodes for single cloud provider. From the trusted authority, the cloud node receives secret tags for file blocks so that the blocks can be processed/ verified by the cloud nodes. The next step, files are added to cloud nodes and executed based on a) Replication of applications from the random cloud node, b) Partition of application System into tiers such that even the web server does not know the location of record in database server, c) Partition of application logic into fragments such that half of the application login in one file stored in one cloud node and other half of the application

logic in other file stored in other cloud node and d) Partition of application data into fragments such that partial records in one cloud database and remaining records in other cloud database.

B. SYMMETRIC ENCRYPTION

In this module, general symmetric encryption/decryption technique is implemented.Symmetric encryption uses a common secret key *K* to encrypt and decrypt information. A symmetric encryption scheme consists of three primitive functions:

KeyGenSE (1λ): K is the key generation algorithm that generates *K* using security parameter 1λ;

EncryptSE (*K*, M): C is the symmetric encryption algorithm that takes the secret *K* and message M and then outputs the ciphertext C;

DecryptSE (*K*, C): M is the symmetric decryption algorithm that takes the secret *K* and ciphertext C and then outputs the original message M.

C. CONVERGENTENCRYPTION

Convergent encryption provides data confidentiality in deduplication. A user (or data owner) derives a convergent key from each original data copy and encrypts the data copy with the convergent key. In addition, the user derives a tag for the data copy, such that the tag will be used to detect duplicates.

To detect duplicates, the user first sends the tag to the server side to check if the identical copy has been already stored. Note that both the convergent key and the tag are independently derived and the tag cannot be used to deduce the convergent key and compromise data confidentiality. Both the encrypted data copy and its corresponding tag will be stored on the server side.

In this module, four primitive functions are implemented to achieve the convergent encryption mechanism.

$KeyGen_{CE}$ (M): K is the key generation algorithm that maps a data copy M to a convergent key K;

$Encrypt_{CE}$(K, M): C is the symmetric encryption algorithm that takes both the convergent key K and the data copy M as inputs and then outputs a ciphertext C;

$TagGen_{CE}$(M): T(M) is the tag generation algorithm that maps the original data copy M and outputs a tag T(M). We allow TagGenCE to generate a tag from the corresponding ciphertext, by using $T(M)=TagGen_{CE}(C)$,where$C=Encrypt_{CE}(K,M)$.

D. PROOF OF OWNERSHIP

The verifier derives a short value ∅(M) from a data copy M. To prove the ownership of the data copy M, the prover needs to send **∅'** and run a proof algorithm with the verifier. It is passed if and only if **∅' = ∅(M)** and the proof is correct.

E. PRROF OF OWNERSHIP BASED ON SESSION

In this module, session based deduplication is considered. Here if the user provides the session duration i.e, front date and to date, then only with the data range, proof of ownership can be allowed in server on those dates. This increases the security if the outsourced data need to be safely accessed on the given duration.

F. REVOCATION OF USERS

In this module, we consider the revocation of users in the given group. If the original (first) user of the group intimates the server with a user's (B) revocation, then the server rejects the proof of ownership submitted by that user (B).

V.CONCLUSION

Data Integrity is a technique for eliminating duplicate copies of data, and has been widely used in cloud storage to reduce storage space and upload bandwidth. This project attempts to formally address the problem of achieving efficient and reliable key management in secure deduplication. It introduces a baseline approach in which each user holds an independent master key for encrypting the convergent keys and outsourcing them to the cloud.

However, such a baseline key management scheme generates an enormous number of keys with the increasing number of users and requires users to dedicatedly protect the master keys. It proposes Dekey, a new construction in which users do not need to manage any keys on their own but instead securely distribute the convergent key shares across multiple servers. Security analysis demonstrates that Dekey is secure in terms of the definitions specified in the proposed security model. In addition, the users can revocate from the given group at any time. To do so, if the original (first) user of the group intimates the server with a user's (B) revocation, then the server rejects the proof of ownership submitted by that user (B). Likewise, session based deduplication is considered. Here if the user provides the session duration i.e, front date and to date, then only with the data range, proof of ownership can be allowed in server on those dates. This increases the security if the outsourced data need to be safely accessed on the given duration.

REFERENCES

[1] P. Anderson and L. Zhang, ''Fast and Secure Laptop Backups with Encrypted De-Duplication,'' in Proc. USENIX LISA, 2010, pp. 1-8.

[2] S. Kamara and K. Lauter, ''Cryptographic Cloud Storage,'' in Proc. Financial Cryptography: Workshop Real-Life Cryptograph. Protocols Standardization, 2010, pp. 136-149.

[3] Mac OS X Time Machine [cited 2nd April 2010]. http://www.apple.com/macosx/what-is-macosx/ time-machine.html.

[4] W. J. Bolosky, S. Corbin, D. Goebel, and J. R. Douceur.Single instance storage in Windows 2000.In Proceedings of 4th USENIX Windows Systems Symposium.Usenix, 2000. http://research.microsoft.com/apps/pubs/ default.aspx?id=74261.

[5] C. Dubnicki, L. Gryz, L. Heldt, M. Kaczmarczyk, W. Kilian, P. Strzelczak, J. Szczepkowski, C. Ungureanu, and M. Welnicki. Hydrastor: a scalable secondary storage. In Proc. 7th USENIX Conference on File and Storage Technologies, 2009.

[6] W. Bolosky, S. Corbin, D. Goebel and J. Douceur.Single instance storage in Windows 2000.In Proc. 4th USENIX Windows Systems Symposium, 2000.

[7] B. Zhu, K. Li, and H. Patterson.Avoiding the disk bottleneck in the Data Domain deduplication file system.In Proc. 6th USENIX Conference on File and Storage Technologies, 2008, pp. 1-14.

[8] N. Agrawal, W. Bolosky, J. Douceur and J. Lorch.A five-year study of file-system metadata.In Proc. 5th USENIX Conference on File and Storage Technologies, 2007.

[9] Microsoft Corporation. Volume Shadow Copy Service.MSDN.[Online] 2010.[Cited August 31, 2010.] http://msdn.microsoft.com/en us/library/bb968832(VS.85).aspx

[10] M. Rabin. Fingerprinting by Random Polynomials. Harvard University Center for Research In Computing Technology Technical Report TR-CSE-03-01,1981.Boston, MA.

[11] ShaiHalevi, Danny Harnik, Benny Pinkas and Alexandra Shulman-Peleg: Proof of ownership in remote storage systems. eprint, IACR, 2011/207.

[12] M. Vrable, S. Savage, and G. Voelker. Cumulus: Filesystem backup to the cloud. In Proc. of USENIX FAST, 2009.

[13] Amazon. SmugMug Case Study: Amazon Web Services. http://aws.amazon.com/ solutions/case-studies/smugmug/, 2006.

[14] A. Shamir. How to Share a Secret. CACM, 22(11):612–613, Nov 1979.