International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

IJIEMR Transactions, online available on 3rd Apr 2019. Link :http://www.ijiemr.org/downloads.php?vol=Volume-08&issue=ISSUE-04

Title: **SECURE DISTRIBUTED DEDUPLICATION SYSTEMS WITH IMPROVED RELIABILITY**

Volume 08, Issue 04, Pages: 42–46.

Paper Authors

VISHNU PRASAD GANABOINA, DR.S. SRINIVASA RAO

St. Martin's Engineering College, Hyderabad, TS, India

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

SECURE DISTRIBUTED DEDUPLICATION SYSTEMS WITH IMPROVED RELIABILITY

VISHNU PRASAD GANABOINA [1], DR.S. SRINIVASA RAO[2]

[1]PG Scholar, Dept of CSE, St. Martin's Engineering College, Hyderabad, TS, India.

[2]Professor, Dept of CSE, St. Martin's Engineering College, Hyderabad, TS, India.

ABSTRACT: Makes the first attempt to address the problem of achieving effective and reliable key management securely in secure duplication. We first provide a basic approach in which each user maintains a separate key to encrypt converged keys and outsource them in the cloud. However, such a basic key management scheme generates a large number of keys while increasing the number of users and requires users to protect key keys customarily. To this end, we propose the implementation of Dekey, a new creation where users do not need to manage any keys on their own, but rather distribute secure key shares safely across multiple servers. The security analysis demonstrates that Dekey is safe in terms of the definitions specified in the proposed security model. As proof of the concept, we apply Dekey using the Ramp Confidential Participation Scheme and prove that Dekey bears limited costs in realistic environments. Server-side data-deduplication schema for encrypted data. Allows the cloud server to control access to external data even when the ownership is dynamically changed by exploiting close random encryption and distributing a secure security group key. This prevents data leakage not only to cancel users although they previously owned this data, but also to the express but curious cloud storage server. In addition, the proposed schema ensures data integrity against any attack that conflicts with tags. Therefore, the security in the proposed schema is improved. The results of the efficiency analysis show that the proposed scheme is almost as efficient as the previous ones, while additional incremental accounting costs are negligible.

Keywords Deduplication, proof of ownership, convergent encryption, key management

I. INTRODUCTION

The emergence of cloud storage drives companies and organizations to outsource data storage to third-party cloud providers, as evidenced by many case-studies. [1] One of the critical challenges facing cloud storage today is managing ever-increasing data volumes. According to the IDC analysis report, data in the wild is expected to reach 40 trillion gigabytes in 2020 [2]. To make data management scalable, deduplication was a known technique to reduce storage space and load bandwidth in cloud storage. Instead of keeping multiple copies of the same content, canceling duplicates eliminates redundant data by keeping only one physical copy and forwarding other duplicates to that copy. Each copy can be defined based on exact details: it may refer to a full file (for example, file-level deduplication), a larger data block, or a variable size (such as mass-level deduplication). Today's cloud storage services, such as Dropbox, Mozy and Memopal, apply deduplication to user data to save the cost of maintenance [3]. From the user's perspective, data outsourcing raises security and privacy concerns. We must trust third-party cloud providers to implement confidentiality, access control and properly control mechanisms against attacks from within and without. However, the elimination of duplicate data, while improving storage efficiency and bandwidth, does not conform to traditional encryption. Specifically, traditional encryption requires different users to encrypt their data using their own keys. Copying identical data from different users will therefore lead to different coded texts, making duplicate data impossible. However, the basic approach suffers from two important issues for dissemination. First, it is inefficient, because it will generate a huge number of keys while increasing the number of users. Specifically, each user must associate a converged encrypted key with each block of encoded data that has been outsourced in order to restore data copies later. Although different users may share the same data copies, they must have their own combination of keys so that no other users can access their files. As a result, the number of converged keys entered is linearly measured with the number of blocks that are stored and the number of users. This principal load of management becomes more prominent if we take advantage of deduplication at the cluster level. For example, assume that a user stores 1 terabyte of data with all 4K unique blocks each, and that each key is the SHA-256 tick value that Dropbox uses to eliminate duplicate data. [4] The total size of the keys will be 8 GB. The number of keys multiplied by the number of users. Heavy administrative expenses resulting from key management lead to a huge storage cost, as users must be held accountable for storing a large number of keys in the cloud under the pay-as-you-go model. Second, the primary approach is not trusted, because it requires each user to protect his master key in a custom manner. If the master key is accidentally lost, user data can not be recovered; if hacked by the attacker, user data will be leaked. This motivates us to explore how to manage massive convergence keys efficiently and reliably, while continuing to achieve secure data deduplication. To this end, we propose a new build called Dekey, which provides guarantees of efficiency and reliability to manage convergent key on both sides of the user and cloud storage. Our idea is to apply duplicate data on converged keys and covert covert techniques. Specifically, we build secret shares of converged keys and distribute them across multiple separate key servers. Only the first user who uploads the data must account for and distribute these confidential posts, and not all users with the same data copy need to account for and store these shares again. To restore data copies, the user must reach the minimum number of key servers through authentication and confidential shares to rebuild the converged keys. In other words, the secret shares of the converged key can only be accessed by authorized users who have the corresponding data copy. This greatly reduces load storage for converged keys and makes key management reliable against failures and attacks. To our knowledge, none of the current studies formally address the problem of convergent key management. Of all these file systems, Farsite only combined duplicate deduplication with security. In its original design, its goal was to take advantage of unused disk space in a network of computers from the desktop class, and present it as a central file server [5]. In the original application, security was provided by encrypting files where each user used a set of symmetric and asymmetric keys. The extension of the work was an attempt to achieve better efficiency in space by copying

II. RELATED WORK

Current systems that use single instance storage rely on one of three basic redundant strategies: full file, fixed-sized pieces, and variable-size segments. The first full file typically uses the hash value of the file as its identifier. Therefore, if two or more files have the same value, they should contain identical contents and be stored only once (not including duplicate copies). This type of Content Oriented Storage (CAS) is used in the EMC Centera system. Individual sites and the Windows Single Instance store also de-duplicate data on a per-file basis, although both use traditional identifiers and handle redundancy using a separate data structure. The second type of deduplication, defragmenting data for each block, is represented by the Venti archiving system. In Venti, files are split into fixed size blocks before eliminating duplicate data, so files that share some (but not all) identical content may still achieve savings in storage. The third and more flexible model divides the files into variable-length "parts" using a hash value in a sliding window; using techniques such as Rabin's fingerprints, cutting can be done with high efficiency. Variable length segments are used in Shark LBFS and Deep Store. Many distributed file systems, such as OceanStore SNAD Plutus and e-Vault address file confidentiality, use cryptographic encryption. The use of encryption techniques in these systems ranges from the assumption that all incoming data is already encrypted, to the central structure elements that determine the system. However, none of these systems attempt to achieve the storage efficiency possible by eliminating duplicate data. High-performance distributed file systems such as the Panasas and Luster parallel file system are usually less secure than standard "distributed" file systems, where higher performance is traded to ensure less security. While there is an attempt to add greater security to this effort involves only authentication, not encryption

III. PRAPOSED WORK

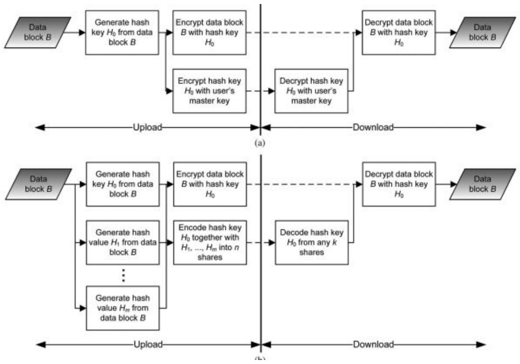
Data block B
Generate hash key H0 from data block B
Encrypt data block B with hash key H0
Decrypt data block B with hash key H0
Data block B
Encrypt hash key H0 with user's master key
Decrypt hash key H0 with user's master key
Upload
Download
(a)
Data block B
Generate hash key H0 from data block B
Encrypt data block B with hash key H0
Decrypt data block B with hash key H0
Data block B
Generate hash value H1 from data block B
Encode hash key H0 together with H1, ..., Hm into n shares
Decode hash key H0 from any k shares
Generate hash value Hm from data block B
Upload
Download
(b)

Fig 1: The system Architecture

Model takes into account the threat of two types of attackers: 1) may get external attacker on some knowledge (for example, a hash value) a copy of important data through public channels. It plays the role of user who interacts with S-CSP. This type of attacker includes the opponent who uses S-CSP as a content distribution network; 2) The internal attacker is honest but curious, and can refer to S-CSP or any KM-CSP. Its goal is to extract useful information from user data or converged keys. We ask the attacker to follow the protocol correctly. Here, we allow collusion between S-CSP and KM-CSPs. However, we ask that the number of KM-CSPs collusion is not more than a predetermined threshold in advance if; n; k; is used rÞ-RSSS (see section 2), so you can not guess the converging key to a message can not be predicted by attack Brute force of KM-CSPs complicit.

CONCLUSION

An efficient and reliable convergent key management system to ensure the elimination of duplicate data. Dekey deletes duplicate data between converged keys and distributes the associated key posts across multiple master servers, while maintaining the semantic security of converged keys and external data confidentiality. We implement Dekey using Ramp's covert sharing scheme and make it clear that it bears small coding / decoding costs compared to network transfer expenses in regular download / download operations. Finally, we checked the information leakage resulting from the major reconciliations and found that the most serious security breaches resulted from losing the client key. However, the damage that occurs if this key is lost is limited to user files. Moreover, the violation of client keys is a serious threat to most secure systems

IV. REFERENCES

[1] AmazonCase Studies. [Online]. Available: https://aws.amazon.com/solutions/case-studies/#backup.

[2] J. Gantz and D. Reinsel, The Digital Universe in 2020: Big Data, Bigger Digital Shadows, Biggest Growth in the Far East, Dec. 2012. [Online]. Available: http://www.emc.com/collateral/analystreports/idc-the-digital-universe-in-2020.pdf

[3] D. Harnik, B. Pinkas, and A. Shulman-Peleg, ‘‘Side Channels in Cloud Services: Deduplication in Cloud Storage,’’ IEEE Security Privacy, vol. 8, no. 6, pp. 40-47, Nov./Dec. 2010.

[4] M. Mulazzani, S. Schrittwieser, M. Leithner, M. Huber, and E. Weippl, ‘‘Dark Clouds on the Horizon: Using Cloud Storage as Attack Vector and Online Slack Space,’’ in Proc. USENIX Security, 2011, p. 5.

[5] A. Adya, W. J. Bolosky, M. Castro, R. Chaiken, G. Cermak, J. R. Douceur, J. Howell, J. R. Lorch, M. Theimer, and R. Wattenhofer. FARSITE: Federated, available, and reliable storage for an incompletely trusted environment. In Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI), Boston, MA, Dec. 2002. USENIX.

[6] J. R. Douceur, A. Adya, W. J. Bolosky, D. Simon, and M. Theimer. Reclaiming space from duplicate files in a serverless distributed file system. In Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS

’02), pages 617–624, Vienna, Austria, July 2002.