2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing

Secure cloud storage service for detection of security violations

Carlos André Batista de Carvalho*†, Miguel Franklin de Castro† and
Rossana Maria de Castro Andrade†
*Computer Science Department, Federal University of Piauí (UFPI), Brazil. Email: candrebc@ufpi.edu.br
†Group of Computer Networks, Software Engineering, and Systems (GREat),
Graduate Program in Computer Science (MDCC), Federal University of Ceará (UFC), Brazil.

***Abstract*—A cloud storage service implements security mechanisms to protect users' data. Moreover, due to the loss of control over the cloud infrastructure, it is essential to demonstrate its security, increasing the trust and transparency in cloud services. However, an analysis of the literature reveals flaws in existing solutions that do not identify all violations. Then, a secure storage service for cloud computing is proposed to address these issues, combining different security mechanisms. The proposed solution includes auditing and monitoring mechanisms to detect and prove violations of security properties. Moreover, Colored Petri Nets (CPNs) are used for evaluation. As results, the violation detection is improved, and the provider cannot deny the identified violations.**

I. Introduction

Cloud computing is a distributed computing paradigm that enables the sharing of computational resources between many clients. It is possible to reduce the infrastructure costs by contracting a public cloud provider and paying only for the consumed resources. Besides, the services' scalability allows the dynamic allocation of resources, in accordance with customers' needs. However, this technology comes with the main drawback of losing control over the cloud infrastructure. Therefore, there is some resistance in adopting public clouds, due to concerns about security and privacy [1].

The cloud providers develop security mechanisms, based on frameworks and security guidelines elaborated by standardization bodies, such as ISO (International Organization for Standardization), NIST (National Institute of Standards and Technology) and CSA (Cloud Security Alliance) [1]. However, customers have a limited view of the security and can request more transparency with mechanisms that provide service security guarantees [1].

In this context, scientific studies have been realized to develop solutions that improve the trust in cloud providers and assure security properties. It is important to highlight the audit and monitoring mechanisms to demonstrate the security and allow the violation detection [2], [3]. It is more efficient to monitor the security properties because it is possible to anticipate the detection of a violation or even allow blocking an attack, reducing the damage [4]. However, the audit is essential to solve any dispute or identify some violations [5].

Among security concerns (*e.g.*, data loss and leakage, resource location, service disruption and multi-tenancy issues), Rong *et al.* [6] stress that the big concern is the assurance of security properties in cloud storage. Usually, confidentiality, integrity and availability are the required security properties, but the literature highlights other properties related to secure cloud storage, such as retrievability [7], freshness [2], write-serializability [2]. Besides, cloud storage services must comply with the customers' requirements (*e.g.*, file sharing [7]). Cryptography is used to provide confidentiality, preventing data leakage. However, some attacks, performed by malicious insiders (e.g. users and provider), can be not detected, resulting in security violations. Then, it is necessary to monitor and audit a cloud service to identify these violations. A malicious provider can, for example, ignore the access control and execute an illegal transaction. A literature review exposes limitations and flaws of existing solutions. In our preliminary study [8], we identify scenarios in which security violations were not detected by Cloudproof [2]. Recent solutions are also limited and, therefore, the development of a secure storage service is still challenging.

In a secure cloud storage service, security mechanisms are combined to treat different security issues in a complete solution. In this research work, we propose a secure cloud storage service that combines security mechanisms to ensure security properties and to allow the data sharing. This doctoral proposal does not aim to address all security aspects and focuses on mechanisms to verify security properties, ensuring the confidentiality, integrity, retrievability, freshness and write-serializability of the data stored in the cloud.

In this context, it is essential to identify the attacks that violate these security properties and to analyze how the violations can be detected. This analysis is used to design an improved secure storage service, fixing identified flaws and detecting security violations in real-time whenever is possible. It is important to emphasize that a provider cannot deny a detected violation. In addition, the modeling using CPNs and the development of a prototype will be used to validate and demonstrate the security of the proposed cloud storage service.

In our scenario, a cloud customer can purchase this storage service, and define permissions for cloud users to perform cloud transactions. The cloud transactions are performed by a cloud provider and managed by a broker, and the security properties monitored by the users and audited by a Third-Party Auditor (TPA). Some solutions do not include a broker as a stakeholder, but it is indispensable to detect freshness

978-1-5090-6611-7/17 $31.00 © 2017 IEEE
DOI 10.1109/CCGRID.2017.19

TABLE I
COMPARISON OF RELATED WORK

Paper	Security Properties				Real-time detection	Collusion attack	Access control verification
	Integrity	Freshness	Write-serializabilty	Retrievability			
[2]	Yes	Yes	Partially	No	Only integrity	Unfeasible	No
[9]	Yes	Yes	Partially	No	Only integrity	Unfeasible	No
[4]	Yes	Yes	Partially	No	Yes	No	No
[5]	Yes	Yes	Yes (inefficient)	Yes	Only integrity	Unfeasible	No
[7]	Yes	Yes	Yes (inefficient)	Yes	Only integrity	Unfeasible	No
[3]	Yes	Yes	Yes (inefficient)	Yes	Yes*	Unfeasible	No

* The freshness verification depends on the knowledge of the latest root signing key. However, the authors do not prove that the users keep this key.

violations in real-time.

The rest of this paper is organized as follows. We present the background related to secure cloud storage and the related work in Section II. Sections III and IV describe, respectively, the proposed mechanism and the current state of this research. Lastly, final remarks are presented in Section V.

II. BACKGROUND AND RELATED WORK

Security mechanisms must be designed to protect a user against existing threats, holding the requested security properties. In this research, we analyze the confidentiality, integrity, retrievability, freshness and write-serializability of the data stored in the cloud. The confidentiality and integrity are essential to avoid the data access or data modification by unauthorized users. The retrievability is related to the data loss verification, and the freshness indicates the reading of the updated file. The write-serializability controls the writing order, ensuring that the new version of a file overwrites the last version of it.

The monitoring and auditing increase the transparency of cloud storage services, demonstrating their security and enabling the detection and proof of violations. Thus, it is necessary to identify the attacks, which can affect security properties, and evaluate if they are detected. The attacks result from malicious acts of the external attackers and stakeholders, including end users.

In accordance with the Dolev-Yao model [10], an external adversary can impersonate any entity (user, broker or provider), creating or modifying messages. Although an attacker cannot understand the content of a message, he/she can store it and send this message during a replay attack. Thus, all entities must include verifications to detect corruption or replay attacks.

Besides the attackers, the broker and provider can have malicious behaviors, and the users must identify them. A malicious provider can: i) bypass the retrievability verification [11]; ii) send an outdated data; iii) write files out of order; iv) confirm a write transaction, without committing it; and v) perform a transaction of an unauthorized user.

The last malicious behavior is addressed by an access control mechanism because unauthorized users do not have the credentials to read or write a file. However, the customer can revoke a user's write permission, and this user can use an older encryption key and try to write a new file [12]. Thus, it is necessary to verify if each writing is authorized. A replay attack can result in writing files out of order. In a rollback attack, the provider restores the system to a previous state [9]. As result, a user receives an outdated file, or the files are written out of order.

In order to enable the real-time violation detection, the broker must inform to users the current state of the stored files. However, a malicious broker can notify an old state. It is important also to highlight the collusion attacks, where the broker helps the provider to deceive the violation detection mechanism. For example, during a rollback attack, the provider restores the system to a previous state, and the broker informs an old state to avoid the violation detection.

The security properties are verified in monitoring and auditing based on logs of cloud transactions. In auditing, the violation undetected in real-time is identified (*e.g.*, retrievability of the scarcely accessed files can be checked) and any contestation is solved.

The existing solutions combine, mainly, access control mechanisms with violation detection mechanisms to offer a secure storage solution. Table I presents a comparison of related work, focusing on violation detection mechanisms. The papers were selected in a non systematic review based on results from Google Scholar and citations on selected studies. On queries, we use terms such as violation detection, auditing, monitoring or verification of security properties. Although there are other security properties and others publications about data retrievability, we selected here only papers that include freshness and write-serializability as verified properties.

The secure storage service, proposed by Popa *et al.* [2], is based on Access Control Lists (ACLs) and audited for integrity, freshness and write-serializability verification. Jin *et al.* [3] add a scheme for verifying the data retrievability. Hwang *et al.* [4] specify a trusted third party, called synchronization server, to enable real-time violation detection.

In order to verify each security property, different procedures are performed. For example, the integrity of files and messages exchange between the stakeholders is checked using hash functions or Message Authentication Codes (MACs) [2], [3]. The Proof of Retrievability (PoR) schemes are based on a challenge-response approach and used, in auditing, to ensure that no data was lost [11]. The security of these schemes is discussed in related work, and no flaws

were found.

The logs of cloud transactions are analyzed to check the freshness and write-serializability [2], [9]. However, we detected a scenario of write-serializability violation that is not identified in [2], [9], [4]. The undetected violation occurs when overwriting an older version of a file, if the logs are not modified and no reading is performed previously [8]. In other studies, the protocol to write a file is inefficient because it always requires a reading before each writing.

Normally, the collusion attacks are unfeasible because an unauthorized user or external attacker cannot obtain, from the provider, credentials to read/write a file. However, the users do not have enough information to detect freshness attacks in real-time. On the other hand, Hwang *et al.* [4] assume that the synchronization server is honest and malicious behaviors of this server are not addressed. If this server is untrustworthy, it is possible to send old information, in collusion with the provider, deceiving the real-time violation detection. The auditing is indispensable to detect this violation, and the assumption of a trusted server is valid only if a private cloud is used.

The access control and key management mechanisms provide the confidentiality and file sharing. In the security evaluation of these mechanisms, it is demonstrated that unauthorized users cannot obtain reading keys, ensuring the confidentiality. On the other hand, the integrity verification performed by existing solutions is not enough to detect violations of access control because a revoked user can write new files using old credentials, which may be considered valid. Therefore, it is necessary the access control verification although no found solution verify these violations.

III. Proposal

The proposed storage service works in a similar way to existing storage services, where the users read and write files in a cloud provider, and the customer defines permissions for each file. This service is managed by a broker to improve the violation detection, control concurrent transactions, and, in the future, allow the use of multiple providers (Vide Figure 1). Access control, monitor and audit mechanisms are combined in this robust solution that protects the users' data, ensuring security properties.

The Access Control Lists (ACLs) has been used to specify users' permissions and is suitable to work together with the others security mechanisms. Besides, Broadcast Encryption and Key Rotation are efficient schemes to, respectively, distribute and update the reading and writing keys [2]. Thus, the broker stores the metadata of each file to enable only authorized users to extract the reading or writing credentials. Besides, metadata includes the tag used for retrievability verification [3].

The logs of cloud transactions, called attestations, are stored by the provider and send to the TPA for auditing purposes. The verification of security properties is performed during reading and writing transactions that are managed by a broker. The broker is responsible for controlling concurrent transactions

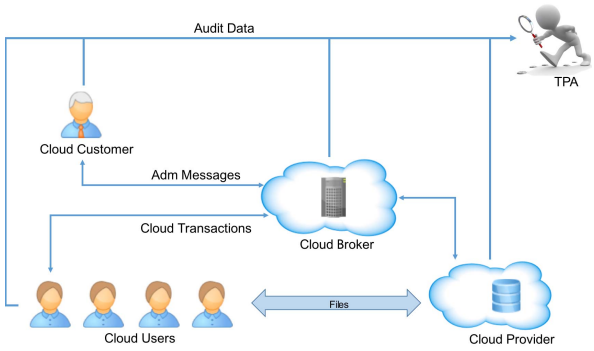

Fig. 1. Service overview

and sending the last attestation when a transaction is requested. Using this attestation, a user verifies the integrity and freshness of the read file, or prepare a write request that complies with the write-serializability. Due to space constraint, only the protocol to perform read transactions is detailed in Figure 2. Besides the attestations, the exchanged messages are signed and verified. Thus, any entity can detect external attacks when an old message or invalid signatures are used.

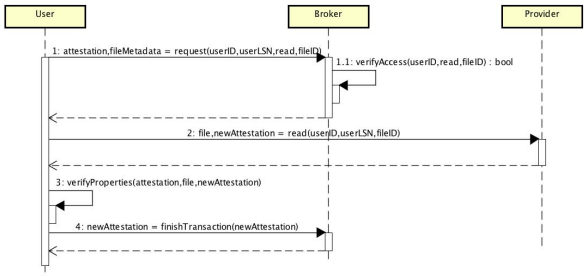

Fig. 2. Reading a file

The elements of an attestation are: **UserID**, **UserLSN**, **FileID**, **FileVersion**, **FileHash**, **TransactionType**, **KeyVersion**, **ChainHash** and **Signatures**. The UserLSN represents the last sequence number used by each user, enabling to detect replay and rollback attacks. The FileVersion is essential to verify the freshness and write-serializability, and the FileHash to check the integrity. The type of the transaction (*i.e.*, reading or writing) indicate whether the expect FileVersion must be replaced during an auditing. The KeyVersion allows a user to derive the correct key using a Key Rotation scheme. The ChainHash is used to build the chain of attestations, sorting the cloud transactions. Lastly, all involved entities (*i.e.*, broker, user and provider) sign the attestation for non-repudiation purposes.

An audit is still necessary to identify some cases of violations and prove any detected violation. During an audit, the TPA receives the attestations from the provider, building and checking the chain of attestations. Thus, a violation occurs when it is not possible to sort the attestations or an unexpected file version is found. The attestations sent by the users and

the broker are used to verify whether the provider hid some transaction. The attestations can be discarded after an auditing.

Besides, there are functions to grant and revoke permissions. Their execution generates attestations that will be used to verify the correctness of the access control. It is possible to observe the holding of the users' privacy because the metadata and attestations do not have any sensible information available to malicious entities.

IV. Partial Results and Next Steps

The use of formal methods is a good approach to model and validate the violation detection mechanism. At this moment, we modeled the protocols for auditing, and for reading and writing files, using CPN Tools[1]. There is a simplified CPN available at https://sites.google.com/site/candrebc/scss.zip[2]. The initial focus is the analysis of the violations of integrity, freshness, without considering the access control and PoR mechanisms.

With the modeling, it is possible to identify important details related to the development of this service. For example, the creation of one chain of attestation per file improves the violation detection. Besides, the broker can allow concurrent access to different files and simultaneous readings of the same file, only blocking when requesting a writing.

In order to validate our proposal, we modeled the attack scenarios and proved, in simulations, that the violations are detected. In our analysis, we conclude that is necessary to check the current version of a file before each writing to properly verify the write-serializability. A PoR scheme can be used to check it efficiently. Besides, we prove the users do not always have enough information to detect collusion attacks and that the audit is mandatory to identify them.

The next step is the inclusion of the detection of access control violations. The attestations of the requests for updates in file permissions must also be stored and verified in the auditing. Due to the diversity of access control and PoR approaches, we need to analyze them, identifying what is most suitable to be used in the complete storage service. It is important to define the interfaces of this service, enabling the replacement of security mechanisms. Lastly, we will develop a prototype to evaluate our solution in a real infrastructure.

V. Final Remarks

This proposal describes a secure cloud storage service, focusing on assurance security properties. During this research, security mechanisms have been studied to ensure confidentiality, integrity, retrievability, freshness and write-serializability of data stored and shared in cloud. These mechanisms protect the customers against data leakage and enable the violation detection, demonstrating the security of an storage service. Thus, it is possible to improve the transparency and trust of security storage services, specifying security guarantees in SLAs [13].

[1]http://cpntools.org

[2]Our simplifications facilitate the analysis of the modeling due to the reduction of its complexity, without interfering with the detection of violations

During this research, we observed that existing solutions do not identify all security violations, and our proposal improves the violation detection. We also used CPNs to validate the violation detection mechanism and proved the necessity of auditing to identify attacks when the broker is untrustworthy.

It is important to highlight that we do not aim to address all security issues, but it is possible to improve our storage service, including new features in future. We also suggest to deploy the broker in a private cloud, protecting against collusion attacks that are detected only in auditing.

Acknowledgment

Carlos André Batista de Carvalho was partially supported by CAPES/FAPEPI Doctoral Scholarship, and Rossana Maria de Castro Andrade has a researcher scholarship (DT Level 2), sponsored by CNPq (Brazil).

References

[1] J. Luna, N. Suri, M. Iorga, and A. Karmel, "Leveraging the potential of cloud security service-level agreements through standards," *IEEE Cloud Computing Magazine*, vol. 2, no. 3, pp. 32 – 40, 2015.

[2] R. A. Popa, J. R. Lorch, D. Molnar, H. J. Wang, and L. Zhuang, "Enabling security in cloud storage slas with cloudproof," in *Proceedings of the 2011 USENIX Conference on USENIX Annual Technical Conference, USENIXATC'11*, 2011.

[3] H. Jin, K. Zhou, H. Jiang, D. Lei, R. Wei, and C. Li, "Full integrity and freshness for cloud data," *Future Generation Computer Systems*, 2016.

[4] G.-H. Hwang, W.-S. Huang, and J.-Z. Peng, "Real-time proof of violation for cloud storage," in *Cloud Computing Technology and Science (CloudCom), 2014 IEEE 6th International Conference on*, 2014, pp. 394–399.

[5] A. Albeshri, C. Boyd, and J. G. Nieto, "A security architecture for cloud storage combining proofs of retrievability and fairness," in *Proceedings of Cloud Computing 2012: The Third International Conference on Cloud Computing, GRIDS and Virtualization*, 2012, pp. 30–35.

[6] C. Rong, S. T. Nguyen, and M. G. Jaatun, "Beyond lightning: a survey on security challenges in cloud computing," *Computers and Electrical Engineering*, vol. 39, no. 1, pp. 47–54, 2013.

[7] D. Tiwari and G. Gangadharan, "A novel secure cloud storage architecture combining proof of retrievability and revocation," in *Advances in Computing, Communications and Informatics (ICACCI), 2015 International Conference on*, 2015, pp. 438–445.

[8] C. A. B. de Carvalho, R. M. de Castro Andrade, M. F. de Castro, and N. Agoulmine, "Modelagem e detecção de falhas em soluções para armazenamento seguro em nuvens usando redes de petri coloridas: um estudo de caso," in *Proceedings of the 34th Brazilian Symposium on Computer Networks and Distributed Systems (SBRC) - 14th Cloud Computing and Applications Workshop (WCGA)*, 2016, pp. 17–30, *Original in Portuguese*.

[9] G.-H. Hwang, W.-S. Huang, J.-Z. Peng, and Y.-W. Lin, "Fulfilling mutual nonrepudiation for cloud storage," *Concurrency and Computation: Practice and Experience*, 2014.

[10] D. Dolev and A. C. Yao, "On the security of public key protocols," *Information Theory, IEEE Transactions on*, vol. 29, no. 2, pp. 198–208, 1983.

[11] K. Yang and X. Jia, "Data storage auditing service in cloud computing: challenges, methods and opportunities," *World Wide Web*, vol. 15, no. 4, pp. 409–428, 2012.

[12] M. Kallahalla, E. Riedel, R. Swaminathan, Q. Wang, and K. Fu, "Plutus: Scalable secure file sharing on untrusted storage," in *Proceedings of the 2nd USENIX Conference on File and Storage Technologies*, 2003, pp. 29–42.

[13] C. A. B. de Carvalho, R. M. de Castro Andrade, M. F. de Castro, E. F. Coutinho, and N. Agoulmine, "State of the art and challenges of security sla for cloud computing," *Computers and Electrical Engineering*, 2017.