A CLUSTER-BASED KEY GENERATION MODEL FOR INITIATING AN EFFICIENT CLOUD AUDITING SYSTEM

Shalini A [1], P. Showmiya* [2]

ABSTRACT

A cloud providers transparency and accessibility to its tenants are ensured by cloud security audits. Due to the intricate operational demands associated with multi-tenancy and self-service features of cloud computing, along with the substantial scale of cloud environments, security audits in the cloud can be prohibitively costly and lack scalability. Consequently, one possible approach to achieving a satisfactory response time is to adopt a proactive auditing technique that initiates the auditing process prior to critical events. Nonetheless, a significant drawback of these methods is their dependence on manual labor to derive the interdependencies between occurrences, thereby limiting their applicability and flexibility. Our research introduces a clustering-based key generation model, an entirely automated system that builds dependency models from runtime events using learning-based techniques, enabling proactive security auditing of cloud activities. Our findings highlight notable benefits, such as a roughly 60% improvement in speed compared to current proactive methods, and a feasible response time of 6 ms for auditing a cloud. We conduct thorough testing in real and virtual cloud scenarios, and we integrate clustering with the popular cloud platform.

Keywords: Component, Formatting, Style, Styling, Insert (Key Words)

Department of Computer Science and Engineering[1],
Karpagam Academy of Higher Education Coimbatore, India[1]
shaliniasokan04@gmail.com[1]
Department of Computer Science and Engineering[2],
Nehru Institute of Technology, Coimbatore, India[2]
nitpshowmiya@nehrucolleges.com [2]

* Corresponding Author

I. INTRODUCTION

Although cloud multi-tenancy has made a significant contribution to resource optimization, the use of this technology presents both advantages and disadvantages, as it may result in a range of security weaknesses. Numerous documented incidents in academic and industrial literature, which include [1] serve to exemplify these security vulnerabilities. Tenants of cloud computing environments consequently express concerns about the transparency and accountability of cloud service providers [2]. As a way to protect against security risks and boost user confidence, formal verification methods for policy verification are also referred to as security auditing have long been employed in the industry and are a preferred solution for cloud environments [3]. Nature of clouds, audit results could quickly become outdated [4]. Furthermore, due to the extensive scope and intricate operational nature of cloud computing, it is imperative for a runtime solution to exhibit high efficiency and scalability in order to provide users with a satisfactory response time [5]. Additionally, given the coexistence of numerous tenants and users with diverse requirements, an auditing system cannot rely on the assumption that user behaviors conform to predetermined or well-established patterns. These issues cannot be resolved by the security auditing techniques now in use. The first type of technique is retroactive, such as seen in [6], which validate cloud states (such configurations and logs) to detect policy breaches after the fact. Because of this, they are unable to stop security breaches from spreading or from perhaps causing permanent damage (such as denial of service attacks or the leakage of private data). Furthermore, a significant lag in responding to user requests occurs due to the interception and verification techniques [7] which assess the impact of each modification request on the cloud before authorization. Additionally, proactive methodologies outlined by assuming a preset order of requests or modification plan, [8] anticipate possible user requests. This presumption about set change plans, however, could not always be realistic given the wide

range and rapidly changing requirements of cloud occupants and users. We further encourage ourselves to work toward our solution with this example [9].

As seen in the first timeline, a standard retroactive auditing is carried out on a regular basis within a predetermined range. After the occurrence, this vulnerability enables malicious actors to take advantage of the vulnerable systems for an extended period, ranging from minutes to seconds in the case of a medium-to-large-scale cloud environment [10]. This could potentially result in irreversible consequences, including data corruption, denial of service, and unauthorized access to information.

The traditional method of intercepting and verifying (illustrated in the second timeline) aims to overcome the drawback of retrospective auditing mentioned earlier. However, this approach may lead to a substantial delay, such as four minutes, as it waits to initiate the verification process until the update port event occurs.

As the preceding timeline shows, our clustering-based key generation model guarantees a significantly accelerated response time, conducts security audits in a proactive manner, meaning it anticipates the critical event (updating the port) before it actually takes place. Furthermore, as will be shown in later sections, we do not assume that events are going to occur in accordance with a preset future modification plan (e.g., the fixed sequence of creating a port, creating a virtual machine, and updating the port), in contrast to current proactive approaches (which are not shown in the timeline).

The work is organized as: the literature is provided in section 2, the methodology is provided in section 3 with outcomes in section 4. The work summary is provided in section 5.

II. RELATED WORKS

Data secrecy has been accomplished against the semi-trusted cloud by Fu et al. [11]. They created a flexible, sufficient, and safe data coordination system. Information consistency is not accomplished by the mechanism. A distributed data sharing utility auditing technique was created by [12]. Examples of this include modifications by multiple users, user repudiation, practical assessment of auditing performance and public auditing. The system defeats the consumer impersonation attack. Its inability to achieve reliability and error detection is a drawback.

The most recent techniques for information audits and cloud computing security have been thoroughly reviewed by Chen et al. [13]. An efficient public verification convention has been proposed. Lazy updating, block less verification, and batch verification are all supported under the proposed convention. The scheme's drawback is that during the verification stage, transmission costs are higher. A secure information exchange technique against conspiracies has been introduced by [14] for cloud-based dynamic clusters. Even when the repudiated customer plots with the CSP, he is unable to obtain the original document. The proposed approach improves fine-grained access control, secure customer repudiation and assured key allocation. A formal explanation and security model for the CP−ABE technique with successful repudiation have been published by Patra et al. [15]. A formal explanation and security model for the CP−ABE technique with successful repudiation have been published. The suggested system is protected from conspiratorial attempts by both renounced and prevailing customers. The scheme's drawback is that the setup phase takes longer than expected.

A technique for carrying out open audits of shared data in distributed repositories has been developed by [16] while maintaining the confidentiality of identities and the ability to track the data. The blind signature method is employed by the mechanism to ensure data privacy. The mechanism's little overhead in achieving identity trackability is its drawback [17]. A technique that meets the cryptographic criterion of security when the regression coefficient estimates are the only output has been described by Nazir et al. [18]. The input data's confidentiality is ensured by the protocol. The regression analysis protocol is built using homomorphic encryption. Two standards that can provide safe and efficient cloud outsourcing of linear regression tasks were suggested. The conventions are effective and safeguard the privacy of the client's data. The mechanism's shortcoming is that it is unable to identify real-world issues pertaining to cloud

computing outsourcing.

In a dynamic, multi-user framework, verifiable data proprietorship method offers dependability and originality. The server's reliable hardware is used to exploit forking and rollback incursions. The suggested architecture does not take server load stabilization into account. A group signature system that supports the managed connection has been described by Sherubha et al. [19]. Reliability attributes like connection and confidentiality are supported by the convention. Global link ability does not protect privacy. For mobile cloud repositories, an identity privacy-conserving convention for distributed data integrity auditing has been proposed. The plan offers trustworthy label updates and anonymity to Third Party Auditors (TPA). The mechanism incurs minimal overhead in terms of calculation, communication, and storage. Sensitive information masking is used to achieve safe information delivery in the remote information auditing system described by [20]. The sensitive data in the paper has been sanitized by the authors using a sanitizer. The system facilitates the interchange of information while hiding sensitive details. The mechanism's drawback is that proof verification requires a higher computation cost for TPA.

III. METHODOLOGY

With a framework for cloud storage that encompasses cloud service providers (CSP), multiple clusters with distinct data owners, and third-party auditors (TPA), the primary goals are:

A. Key generation

Each cluster's IP manager may safely calculate the key by utilizing the clustering approach.

B. Repudiation

The IP can effectively re-sign the chunks that the repudiated customer signed after the malicious customer has been removed from cluster. The CSP uses key that IP has provided to verify the authenticity of shared information signature and prevent the disputed customer from challenging their validity on behalf of the current customers.

C. Privacy-preserving

Because the client whose access has been revoked helped, using the key that was obtained from IP, CSP is unable to acquire secret keys of current users. As a result, the plan prevents collusion and protects consumer privacy.

D. Public auditing

Known as individual auditing, TPA examines each IP's request issued by each cluster separately. Additionally, the TPA conducts batch auditing for many information proprietors for all IP requests at the same time.

E. Scalability

Data in the cloud is efficiently shared across the current users of several clusters.

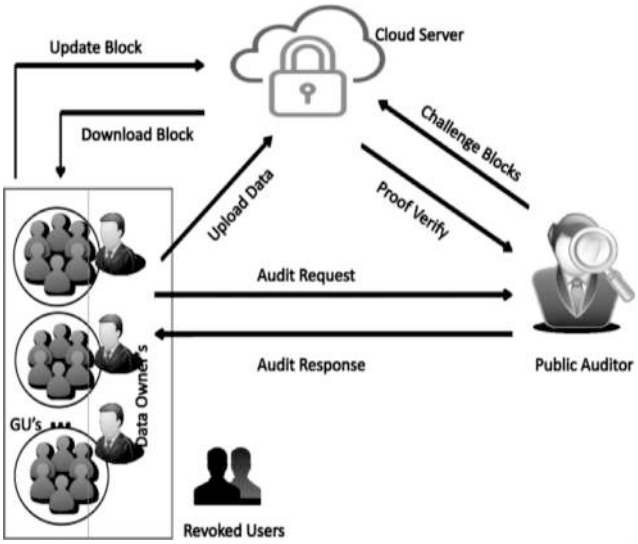

Figure 1. Adversary model

F. System model

Three components make up the system framework: the CSP, TPA and several clusters with individual IP addresses. Customers can obtain information distribution and repository services from the CSP. The TPA and the CSP will use a challenge-and-response mechanism to audit the accuracy of transmitted information. An IP address and several customers make up each cluster. The cluster's manager or head is the IP (group of customers). For each client in the cluster, the IP produces both the public and private keys (Generate Key). The Customer List (CL) is likewise generated by the IP. Information is created and shared by the IP of the relevant cluster with other cluster

members via the cloud.

G. Auditing based on key generation

The authors of the current technique have given trusted CSP permission to use current cluster users secret key to compute key. As a result, the CSP can easily discover and access the private information that is cached on the server. Additionally, if the terminated client collaborates with the CSP, the information that is cached on the cloud server can be vulnerable to their manipulation or hacking. As a result, the existing approach is weak and susceptible to collusion.

The trusted CSP has not been facilitated to calculate the key using the recommended methods. We have given the necessary clusters' IP addresses permission to use the regression technique to compute the key:

$$\tau_{key} = 2[\Delta + 4) \,/(2\sqrt{\delta X^2 \delta Y^2} + 4) \qquad (1)$$

This computed Re-sign key (τ_{key}) comprises of an implicit and secret public key. A revoked customer or a CSP with limited trust can face significant difficulties in deciphering the key and gaining access to the secret keys of the cluster's current $\{(i,\xi_i)\}i \in E$ clients because of its strong security measures. According to the suggested system paradigm, the auditing request is created by the IPs of the corresponding clusters and sent to the TPA. The TPA creates a set of challenges in reaction to each auditing request from the Information Provider (IP). Subsequently, the TPA transmits these challenges to the CSP using the Cluster Chal function. The CSP's response to the TPA, following its acceptance of the challenge, is the storage proof $\{\rho,\{\backslash d\}1 \le d \le D,\{id_i,e_i\}i \in E\}$ for each IP, $d\ (d \in \{1,.....D\}$. Utilizing the Cluster Verify mechanism where the integrity of the storage proof that the cloud provides is verified by the public verifier

Ultimately, the auditing proof is sent to the appropriate IP by the verifier, so completing multi-IP auditing. Both server's transmission cost and the public verifier's computation cost are greatly decreased when multi-information proprietor auditing is employed. In reply to the challenge request from the public verifier, the CSP provides a single group element ρ instead of D individual elements using the bilinear aggregate signature which leads to a substantial decrease in communication costs on the server side. Moreover, while individual verification necessitates $D + 1$ pairing actions, the overall number of costly pairing methods from 2D is reduced through the integration of D auditing equations into one. The public verifier thus saves a reasonable amount of time during the verification process.

IV. NUMERICAL RESULTS AND DISCUSSION

After outlining the settings of the experiment, this part provides a summary of the outcomes using both synthetic and actual data. Eighty compute nodes with two gigabytes of RAM, and an Intel i7 dual core CPU comprise the controller node. We simulated a system with a maximum of 10,000 tenants, 100,000 users, 100,000 VMs, 500 domains, 20,000 routers, 100,000 ports and 40,000 subnets based on a recent OpenStack. For instance, the watchlist for our sample security policy contains information about ports assigned to various tenants. We can change the total tenants (from 1,000 to 10,000) and the total ports (from 10,000 to 100,000) in the no bypasses policy. We conduct tests on 10 different datasets, adjusting the key variables while keeping the remaining values at their maximum levels. Tenants under the common ownership principle might number anywhere between 1,000 and 10,000, and each renter is assigned five tasks every 1,000 years. Every experiment is carried out 100 times. Data gathered from an actual community cloud hosted by a major telecom carrier is used to further assess. To do this, relevant log entries from a period of 100 days are extracted by going through the management logs, which are text-based logs larger than 1.6 GB. We utilize OpenStack's Ceilometer telemetry service because our cloud isn't set up for it, which increases the effort required to process logs.

Measuring the impact of proposed auditing result use on response time is the aim of the first set of tests. Fig 2 shows the outcomes of utilization diverse caching methods, namely IBKE and LB-PPFS by varying cache size and hit ratio which is the number of hits over the total tries. From Fig 2, the hit ratio increases with cache size for caches. According to expectations, the hit ratio rises with cache size and peaks at 0.95 for the 45,000 cache items.

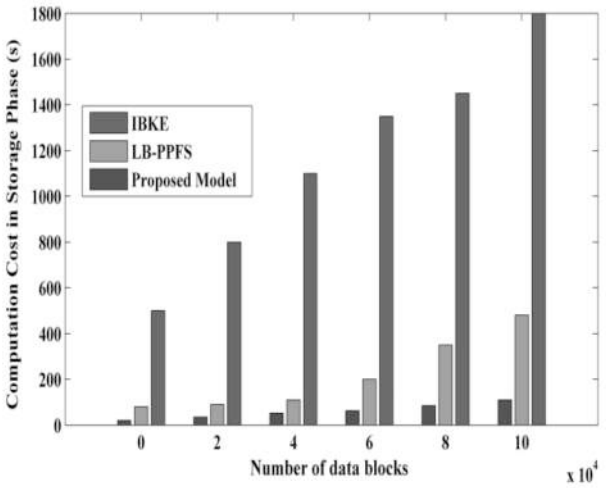

Figure 2 : Graphical representation of the number of blocks Vs computational cost

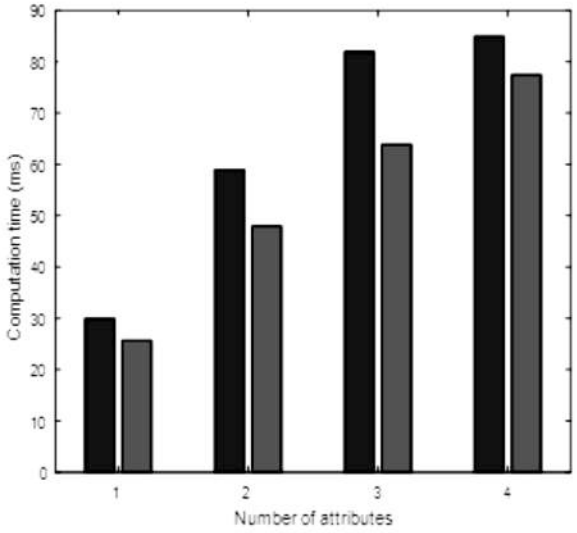

Figure 3: Reaction time of proposed vs. Existing

Fig 3 displays the average reaction time (in nanoseconds) needed in the event of a hit when an intercepted event is discovered in the cache. The anticipated model responds in a maximum of 4,000 nanoseconds with the shortest cache capacity in these conditions. With the notable exception of the MRU cache, which exhibits a noticeable decrease in response time for the two cache sizes (10K/25K), the response times of the two cache types are otherwise rather identical. The proposed model experienced a delay (measured in nanoseconds) as a result of a miss, which indicates that the kind of intercepted event was not located in the cache, as shown in Fig 3. 2,000 nanoseconds is the maximum delay for the largest cache size, and this delay is relatively constant for all cache sizes. The cache with 25K items produces the least amount of delay.

V. CONCLUSION

For cloud service providers and their tenants, scalability and reliability in ongoing auditing are essential. Through the implementation of auditing findings on the cloud prior to any breach occurring, the proposed auditing solution drastically decreases reaction time, as demonstrated in this study. Specifically, the proposed model developed its models (dependency models, important events, etc.) initially. Second, it used its models to perform proactive security verification. Ultimately, it made use of such verification outcomes to impose runtime security on the cloud. One of the most widely used cloud management platforms was integrated with the auditing scheme and we also offered instructions for porting it to other significant cloud platforms. Additionally, we assessed the accuracy and efficiency of our approach and demonstrated a practically usable reduction in response time. The proposed model does have certain restrictions, though, which we will address in later work. In order to choose the best option, machine learning techniques could be used to further automate the manual examination that is now required for the existing way of learning crucial events. Secondly, the proposed model does not currently handle a single-step violation in an effective manner. An effective runtime strategy could assist in resolving this issue.

REFERENCES

[1]. Xia, J. Zhang, T. Q. S. Quek, S. Jin, and H. Zhu, "Energy-efficient task scheduling and resource allocation in downlink C-RAN," in 2018 IEEE Wirel. Commun. Netw. Conf., Apr 2018, pp. 1–6.

[2]. Liu, T. Han, N. Ansari, and G. Wu, "On designing energy-efficient heterogeneous cloud radio access networks," IEEE Trans. Green Commun. Netw., pp. 1–13, 2018.

[3]. Al-Dhabi, F. Paraiso, N. Djarallah, and P. Merle, “Elasticity in cloud computing: State of the art and research challenges,” IEEE Trans. Serv. Comput., vol. 11, no. 2, pp. 430–447, Mar 2018.

[4]. Botta, W. de Donato, V. Persico, and A. Pescapé, “Integration of cloud computing and internet of things: A survey,” Futur. Gener. Comput. Syst., vol. 56, pp. 684–700, Mar 2016.

[5]. Sharafeddine, K. Jahed, O. Farhat, and Z. Day, “Failure recovery in wireless content distribution networks with device-to-device cooperation,” Comput. Networks, vol. 128, pp. 108–122, Dec 2017.

[6]. Su, Q. Xu, J. Luo, H. Pu, Y. Peng, and R. Lu, “A secure content caching scheme for disaster backup in fog computing enabled mobile social networks,” IEEE Trans. Ind. Informatics, pp. 1–11, 2018.

[7]. Yadav, O. A. Dobre, and N. Ansari, “Energy and traffic aware fullduplex communications for 5G systems,” IEEE Access, vol. 5, pp. 11 278– 11 290, 2017.

[8] Fu, X. Sun, Q. Liu, L. Zhou, and J. Shu, ``Achieving ef_cient cloud search services: Multi-keyword ranked search over encrypted cloud data supporting parallel computing," IEICE Trans. Commun., vol. E98-B, no. 1, pp. 190_200, 2015.

[9] Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou, ``Fuzzy keyword search over encrypted data in cloud computing," in Proc. IEEE INFO- COM, San Diego, CA, USA, Mar. 2010, pp. 1_5.

[10] Wei, H. Zhu, Z. Cao, X. Dong, W. Jia, Y. Chen, and A. V. Vasilakos, “Security and privacy for storage and computation in cloud computing,” Information Sciences, vol. 258, pp. 371–386, 2014.

[11] Fu Z, Ren K, Shu J, et al. Enabling personalized search over encrypted outsourced data with efficiency improvement[J]. IEEE Transactions on Parallel and Distributed Systems, 2016, 27(9): 2546-2559.

[12] Zheng Z, Zhou T C, Lyu M R, et al. Component ranking for fault-tolerant cloud applications [J]. IEEE Transactions on Services Computing, 2012, 5(4): 540-550.

[13] Chen W, Lee Y C, Fekete A, et al. Adaptive multiple-workflow scheduling with task rearrangement [J]. The Journal of Supercomputing, 2015, 71(4): 1297-1317.

[14] Jing W, Liu Y. Multiple DAGs reliability model and fault-tolerant scheduling algorithm in cloud computing system[J]. Computer Modeling and New Technologies, 2014, 18(8): 22-30

[15] Patra PK, Singh H, Singh R, et al. Replication and Re-submission Based Adaptive Decision for Fault Tolerance in Real-Time Cloud Computing: A New Approach [J]. International Journal of Service Science, Management, Engineering, and Technology, 2016, 7(2): 46-60.

[16] Duan, C. Chen, G. Min, and Y. Wu, “Energy-aware scheduling of virtual machines in heterogeneous cloud computing systems,” Future Generation Computer Systems, vol. 74, pp. 142–150, 2017.

[17] Ghribi, M. Hadji, and D. Zeghlache, “Energy efficient VM scheduling for cloud data centers: Exact allocation and migration algorithms,” 2013 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing, 2013.

[18] Nazir, K. Qureshi, and P. Manuel, “Adaptive check pointing strategy to tolerate faults in economy based grid,” Journal of Supercomputing, vol. 50, no. 1, pp. 1–18, 2009.

[19] P. Sherubha, “Semi-supervised Learning approach for detecting abnormalities in cloud computing," "International Virtual Conference on Smart Advanced Material Science & Engineering Applications," 2020.

[20] P. Sherubha, “Graph Based Event Measurement for Analyzing Distributed Anomalies in Sensor Networks," Sådhanå (2020) 45:212, https://doi.org/10.1007/s12046-020-01451-w