Operational-Behavior Auditing in Cloud Storage

Zhaoyi Chen[1], Hui Tian[1(✉)], Jing Lu[2], Yiqiao Cai[1], Tian Wang[1], and Yonghong Chen[1]

[1] College of Computer Science, National Huaqiao University, Xiamen 361021, China
cshtian@gmail.com
[2] Network Technology Center, National Huaqiao University, Xiamen 361021, China

Abstract. As an indispensable branch of cloud computing, cloud storage enables individuals and organizations to enjoy large-scale and distributed storage capability in a multi-tenant service pattern. However, there is still a serious lack of mutual trust between the users and cloud service providers, since both of them can perform dishonest and malicious operational behaviors on cloud data. Secure audit for operational behaviors is vital for cloud forensic investigation, which collects and offers essential audit logs for a forensic investigator to track security incidents and accountability determination. Such an auditing service can help to achieve better security assurances within the whole life cycle of cloud data. In this paper, we present an auditing mode for operational behaviors in cloud storage, introduce the open issues in two main phases, log audit and forensic investigation, and discuss the future trends.

Keywords: Cloud security auditing · Operational-behaviors · Secure logging · Forensic analysis

1 Introduction

As an indispensable branch of cloud computing, cloud storage enables individuals and enterprises to enjoy large-scale and distributed storage service, due to a series of advantages, such as on-demand self-service, ubiquitous network access, location-independent resource pooling, rapid resource elasticity, usage-based pricing, and transference of risk [1, 2]. However, as a promising technology, many new security challenges come along with this trend, which impede the development and application of cloud storage [3]. One of most serious issues is a lack of mutual trust between the CSP and users, and this problem has been considered as a non-negligible obstacle for the widespread application of CSS [4]. This issue can be described from two aspects.

From the perspective of users, a tricky problem is how to evaluate if the CSP meets their legal expectations for data security. First, since a CSP actually can be viewed as a separate administrative entity, storing local data in the cloud means abandoning the users' ultimate control over their data. As a result, the correctness and privacy of cloud data cannot be well protected and some security breaches of noteworthy appear in

J. Wan et al. (Eds.): CloudComp 2016, SPNCE 2016, LNICST 197, pp. 162–172, 2018.
https://doi.org/10.1007/978-3-319-69605-8_15

cloud storage frequently. For example, the Amazon's S3 cloud storage service experienced an alarming downtime resulted by a significant system outage in 2008; the Apple's iCloud service suffered a serious privacy issue in 2014 that resulted in worrying leakage of users' personal information. To overcome this problem, devising appropriate cloud data auditing (CDA) mechanisms that offer remote integrity verification service on cloud data is definitely essential. In recent years, there have already been lots of researches on CDA schemes, and we can briefly classify these schemes into two categories [5]: Provable Data Possession (PDP), which provides remote data integrity verification without any retrievals of the outsourced cloud data [6, 7]. Proof of Retrievability (POR), which provides high probability for data recovery capability in addition to integrity verification [8, 9].

From the perspective of the CSP, another problem should be taken into account is how to determine the legality of cloud storage users' behaviors. For example, since the multi-tenant characteristic of cloud storage, the cloud should ensure that the data of users are kept confidential to adversaries from the internal or external, such as malicious users and potential attackers. To avoid this problem, some security access control (SAC) schemes are proposed in recent years to limit unauthorized accessing in cloud environment [10, 11]. Furthermore, to provide cost-efficient storage, deduplication is a necessary requirement in cloud storage, which enables removing of data redundancy. In this scene, one file might be shared by multi-users in cloud, so the CSP has to verify the data ownership of a given user without transferring the file to the cloud. To this end, proof of ownership (POW) strategies are presented to solve this problem [12, 13].

As mentioned above, since users and the CSP are usually not in the same trusted domain in cloud computing, they lack of confidence in cloud data operational behaviors of each other, including data-management behaviors of the CSP and data operations of the users. Thus, there is an urgent need to develop cloud auditing technology to ensure the security of cloud storage. However, current cloud auditing technologies do have certain limitations. The most obvious point is that these CDA schemes (i.e. PDP and POR) are originally designed to ensure the correctness of cloud data, they can only verify the existence of cloud data security incidents, but cannot provide auditing information or evidence to track the operational behavior histories about the disputed data. In this case, it is apparently unfair exclusive put the responsibility for the failures of CDA strategies to the CSP, because some error data operations of users may also cause verification failed. What makes things worse is that the CSS also provides a better platform for ill-disposed users to store and propagate criminal information (e.g. child abuse and terrorism-related materials) [14]. This kind of seemingly "legal" users may also conduct illegal operational behaviors. Thus, cloud auditing system should not only focus on security of data properties, but also the operational behavior legality of cloud data. Recently, as cloud crimes emerge in endlessly, the concept of cloud forensic is put forward to address the problem of forensic investigation [15], but there are still some crucial issues need further research. Two of the most important issues are how to ensure the validity of evidence (i.e. operational behavior logs) [16] and how to perform efficient forensic analysis on massive amounts of cloud logs [29]. Therefore, within the scope of this article, we

focus on these two essential issues, which is intended as a call for action, aiming to motivate further attention to the problem of operational behavior legitimacy on cloud data, and thereby achieve accountability determination in CSS.

2 The Architecture of Cloud Auditing

To illustrate the specific problem mentioned above, we begin with a high-level cloud architecture which integrates CDA and OBA mechanisms for cloud storage security, as shown in Fig. 1. Particularly, CDA mechanism is based on a three-party model, in which an external third party auditor (TPA) is usually introduced to perform remote public verification on outsourced data, which aims to provide more transparent and reliable auditing results [2]. In this paper, to achieve the requirement of operational behavior auditing, we introduce an integrated auditing model which involves four different entities: users, TPA, CSP and forensic investigator (FI). And we also consider both CSP and users can be malicious potentially.

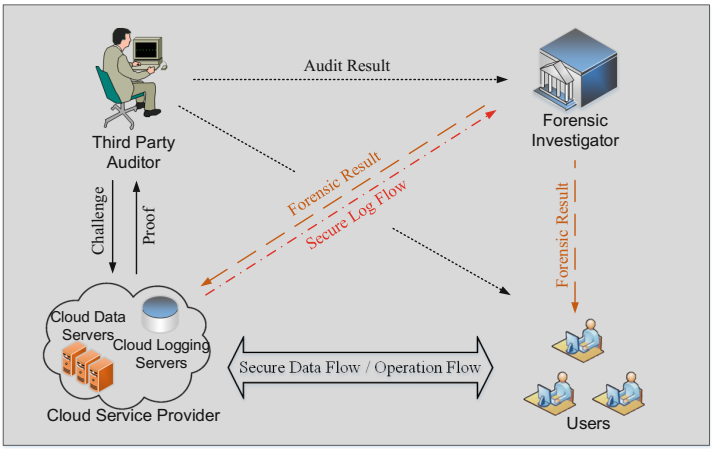

Fig. 1. The architecture cloud auditing

Generally, OBA is consisted of two key components, validity verification and forensic analysis of operational behavior logs [16], which are not considered in traditional CDA schemes. In this model, FI is a party that is responsible for analyzing the content of logs independently and providing convincing forensic report about security incidents on cloud data to the CSP and users for responsibility determination. Although analyzing logs plays a vital role in forensic investigation that is really useful to identify illegal operation behaviors, to determine the authenticity of logs is also an indispensable step prior to the forensic analysis process, since corrupted logs may lead to incorrect or meaningless forensic results. Therefore, the TPA in this model is also equipped with the capability to verify the authenticity of logs, so as to enhance the credibility of forensic results and reduce the heavy burdens of FI.

3 Log Security in Cloud

The process of OBA starts with acquiring the operation logs from the cloud. However, establishing appropriate OBA mechanism to ensure the log security in cloud storage still faces challenges and security threats [15–17]: **(1) Creditability of operation logs.** As a kind of digital resource stored in cloud, the integrity of logs is facing threats of corruption from external or internal adversaries. **(2) Lack of control in cloud.** In cloud systems, to collect audit logs from the cloud, we inevitably need to rely on the CSP, but for the reason of privacy preserving, which in turn brings the honesty problem of CSP, and current CSPs are not obligated to provide relevant logs. **(3) Privacy leakage.** A dishonest CSP or malicious users may try to get access to the log content during the storage or auditing phase, which may lead to serious privacy leakage. This problem can be worse if the logs are written in plaintext.

Although logs are prime evidences for forensic investigation, there has been little concrete work that shows how to provide cloud logs for forensic while preserving privacy and integrity of the logs. What ensures the security of logs is the secure logging methods. From view of technique, to fulfill the requirements of log security, we can rely on existing works of secure logging protocols that listed in Table 1. Generally, the secure logging protocols can be divided into two classes according to the cryptographic techniques they build on, i.e. MAC-based approaches and signature-based approaches.

Table 1. Secure logging protocols and security requirements.

Schemes	Forward security	Append-only	Selective verification	Privacy preserving
Bellare [18]	✓	✗	✗	✗
Schneier [19]	✓	✗	✗	✓
FssAgg-MAC [20]	✓	✓	✗	✓
BBox [23]	✓	✓	✗	✓
SecLaaS [17]	✓	✗	✗	✓
Logcrypt [22]	✓	✗	✗	✓
Stathopoulos [24]	✓	✗	✗	✓
FssAgg-BLS [20]	✓	✓	✓	✓
LogFAS [21]	✓	✓	✓	✓

MAC-based approaches. To ensure the integrity of logs, Bellare and Yee [18] first formally defined the forward security (i.e. forward integrity) property, which prevents attackers from modifying the previous log data even if they know the current key. In this work, they used block ciphers and standard message authentication codes (MACs) to achieve forward security via a chaining process.

Based on the work of Bellare and Yee [18], Schneier and Kelsey [19] presented a classic secure logging protocol which can ensure the security of logs on an untrusted

machine. In this work, they used hash chain for key evolvement, in which the authentication key of each log entry is hashed using a one-way hash function to ensure the forward security of logs. In addition, they presented a secure logging structure, in which the log entry consists of five fields illustrated in Fig. 2. Field D_j is the current log entry generated in time j, and the current authentication key of D_j is A_{j+1} that can be computed as $A_{j+1} = H(A_j)$. W_j is the permission mask that is used to control who can gain access to the contents of entry j. For privacy preserving, D_j will be encrypted with the encryption key of time j K_j, which can be computed as $K_j = H(W_j \| A_j)$. Field Y_j is the j-th element of the hash chain that can be generated as $Y_j = H(Y_{j-1} || E_{K_j}(D_j) || W_j)$.

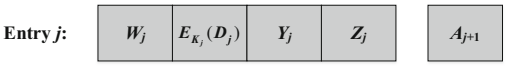

Fig. 2. Secure logging structure of [19]

The forward security can prevent the log entries before the compromise from modification, but cannot detect selective deletion or truncation of log entries, which is a kind of deletion attack that the attacker deletes continuous entries from the tail-end of log data. Thus, Ma and Tsudik [20] presented a secure logging scheme based on FssAgg authentication technique, where forward-secure (MACs) of log entries are sequentially combined into a single aggregate one, so as to achieve the append-only property defined in [22], with which attacker cannot change (i.e. deletion and truncation) logs entries generated before the compromise. In this scheme, a log file involves two parts: log entries $[L_1, \ldots, L_i]$ and FssAgg authentication MACs $\mu_{v,i}$. For each log entry in the log file, there is a unique authentication key A_i for MAC computation, and A_i is generated from the initial key A_1 with a one-way hash function. The construction of log file can be represented as:

$$\mu_{v,i} = H(H(\ldots H(\mu_{v,i} \| mac_{A_1}(L_1) \| \ldots \| mac_{A_1}(L_i))) \quad (1)$$

Another secure logging protocol BBox [23] based on Schneier and Kelsey's work is presented by Accorsi. To avoid the truncation attack, Accorsi applied trusted computing module to sign the hash chain fields of log entries, which is the core of security of this scheme.

There is also attempt to ensure security of audit log in cloud. Zawoad et al. presented forensic framework SecLaaS (secure logging-as-a-service) that implemented in Openstack [15, 17], which stores virtual machines' logs and provides access to FI while ensuring the confidentiality of the audit logs. In this system, a read only API is provided for log acquisition by forensic investigator. Furthermore, a log chain was generated by using the one-way hash function to ensure the order of log entries. To protect the membership of log entrier, *PPL* (Proof of Past Log) is generated by using BloomFilter in the end of every day and published to the public (e.g. Web or RSS). Thus, the integrity of log can be verified with the log chain and *PPL*.

Signature-based approaches. To overcome the short comings of the MAC-based schemes, which cannot provide public verification of logs. Secure logging schemes based on public key cryptography are proposed. To this end, the first attempt is that Holt [22] presented an improved scheme Logcrypt based on work of Schneier and Kelsey [19] by directly substitute the MACs for digital signatures to achieve public verification. However, Logcrypt cannot ensure the append-only property, which is inherited from the disadvantage of [19].

Stathopoulos et al. [24] presented a secure logging that can be implemented in public communication networks. In this scheme, the authors exploited a log structure similar to the Schneier and Kelsey' scheme [19]. To prevent attacks from internal adversary, who can reconstruct parts of the log entries without being detected, this kind of attack is possible if the adversary gains the authentication key A_j. They introduced an independent Regulatory Authority (RA) to store the integrity proofs of log files and verify the log integrity. Periodically, the log collector sends a signature over the log entries generated during this period to RA. In verification phase, the RA recomputes the signature of the corresponding log files and compares with the one stored before. This kind of manual off-line signature protects the logs from modifications after the log file has been signed and the signature have been sent to the RA. However, the security of this approach depends on RA, the compromise of RA may lead to single point of failure.

Ma and Tsudik [20] presented a secure logging scheme based on FssAgg authentication signature. Particularly, by using a collision resistant one-way hash function H for signature key update, this scheme ensures the forward security of log entries. Since the aggregation property of BLS signature [25], signatures of the log entries can be sequentially aggregated into a single signature. However, to remove a signature of a given entry from the aggregation signature is impossible, which ensures the append-only property. In key generation, the system works in bilinear map group G with generator of g. First, it generates a series of key pair for each log entry as $(sk_i = x_i, pk_i = v_i)$ $i \in [1, n]$, where the $x_i = H(x_{i-1})$ and $vi = g^{xi}$. Then, the signature of the i-th log entry M_i can be represented as $\sigma_i = H(i||M_i)^{xi}$. The aggregate of signatures on n log entries can be computed as $\sigma_{1,n} = \sigma_1 \cdot \sigma_2 \cdot \ldots \cdot \sigma_n$. The successful integrity verification of the log file is equivalent to the verification of aggregation signature $\sigma_{i,n}$ as:

$$e(\sigma_{1,n}, g) = \prod_{i=1}^{n} e(H(i \,||\, M_i), v_i) \tag{2}$$

However, the FssAgg-BLS scheme suffers from the problem of high overhead because the computational inefficiency of the signature generation and verification on bilinear map. To achieve more efficiency, Yavuz et al. [21] presented a PKC-based secure logging scheme LogFAS with forward security and append-only property, which supports public verification, selective verification of any subset of log entries and fast detection of corrupted log entries. Moreover, LogFAS outperforms FssAgg-based scheme on computation and storage overhead.

In summary, despite the secure logging schemes listed above are commonly used for forensic investigation in traditional computer and network systems. However,

deploying secure logging mechanisms in untrusted cloud environment still face challenges, which is discussed as the first paragraph of this section, and these problems need further research in the future.

4 Forensic Analysis with Cloud Logs

In this section, we will describe the existing challenges and problems of conducting forensic analysis on huge amounts of log data. Then we make a review on cloud forensic analysis techniques. Operational behavior log contains activities that happen in cloud storage. Forensic analysis for log file plays an important role for OBA, which aims to extract knowledge about abnormal data behaviors in cloud from various type of log information. However, performing log forensic investigation with high-efficiency in cloud also faces challenges as follows:

(1) Massive volume of logs. One of the biggest concerns in cloud forensic investigation is that the scale of log data is much larger than traditional computer and network forensic. Moreover, formats of log data in cloud can be diversified, which increases the difficulty in log analysis. **(2) Encrypted Log data.** In cloud storage, to ensure the security of the cloud data and privacy of users, the CSP usually encrypts the log data during transmission and storage phase. However, current data analysis algorithms can only process data in plaintext. Thus, how to perform analysis on encrypted data needs for further research.

There is an urgent need to develop scalable forensic analysis solutions that can match the explosive growth on the size of log data in cloud. Differing from the traditional digital forensic solutions, which usually implement data analysis algorithms in single workstation. In recent years, an attractive long-term solution is to perform forensic processing in distributed and parallel systems, because only cluster computing environment will offer enough processing resource and power [26, 27]. Thus, within the scope of this paper, we focus on how to process on massive log data with distributed and parallel computing (i.e. MapReduce) to achieve high-efficient forensic analysis. Several efforts have been presented to address this problem showed in Table 2. Generally, from the view of technique, there are two approaches can be exploited to increase the data analysis performance: algorithm improvement and using additional hardware resources.

Table 2. Methods for cloud forensic analysis

Schemes	Algorithm improve	Additional hardware	Mapreduce-based
Marziale et al. [27]	✗	✓	✗
Francois et al. [28]	✓	✗	✓
Roussev et al. [29]	✓	✓	✓
Therdphapiyanak et al. [30]	✓	✗	✓
Lin et al. [31]	✗	✗	✓

In traditional digital forensic solutions that implement data analysis algorithms in a single workstation, algorithm improvement is the only way to increase analysis efficiency. However, in cloud computing, the analysis efficiency can be further improved by using the parallel computing paradigm. Currently, MapReduce (MR) is widely applied in the field of forensic analysis for its high-efficiency and scalability. MR is a programming model and an associated implementation developed by Google for processing large-scale data set with a parallel, distributed cluster. Thus, programs written by MR can be automatically dispatched and executed in parallel cluster.

Roussev et al. [29] first attempted to utilize MR for large-scale log forensic analysis in cloud computing, and they presented an improved MR algorithm MPI MapReduce (MMR), which outperforms the traditional forensic analysis computing techniques. In this scheme, a single data file is first split into N equal blocks, where N is the number of available computing nodes. Then, each block is split into M chunks according to the mapper thread number created at each node. Differing from the original MR implemented in Hadoop, a core improvement of MMR is that by using the Message Passing Interface (MPI) distributed communication standard for management of communication and synchronization in different tasks, the number of nodes can be increased dynamically. Therefore, the MMR is much more efficient than MP.

Another successful usage of MR for forensic analysis is Francois et al. [28], who combined the MR and PageRank (PR) algorithm. The PR algorithm is a link analysis algorithm used by Google Search to weight the relative importance of websites in search engine results. PR works by counting the number and quality of hyperlinks to a page to determine the importance of the website. The first step is to gather netflow from routers to a collector. Then, analyze the interactions between different hosts so as to generate a dependency graph, which will be the input of PR. Next, the PR will be executed by MR by distributing the adjacency matrix of the dependency graph to all computing nodes.

In addition to the PR algorithm, as a popular cluster analysis, K-Means (KM) algorithm can also be used to detect abnormal activities in cloud storage. Therdphapiyanak et al. [30] presented a novel forensic analysis method based on KM, with which malicious activities can be detected by inspecting which cluster has deviated from the others.

Lin et al. [31] presented a comprehensive framework for batch log data analysis with the combination of Hadoop and Spark. In this system, Hadoop is treated as the stable file storage system, and by leveraging the MR and spark, the framework can provide efficient batch data processing in memory. In addition, a special improvement of this work is that there is a parallel data mining (DM) module in this system. With the DM, classification and cluster algorithms (i.e. KM, PR, Bayes) based on MR and Spark engine are implemented in this framework.

In addition to the algorithm improvement, another straightforward approach to enhance the processing performance is to provide additional physical hardware, such as GPU and CPU cluster. As an example, Marziale et al. [27] leveraged the hardware and software capabilities of GPU for high-efficient digital forensic. They evaluate the effectiveness with direct experiments and the result show that using CPU and GPU resource on multiple machines is feasible and more efficient.

5 Concluding Remarks

Cloud computing is envisioned as the next-generation IT architecture. As an important branch of cloud computing, one of most serious issues in cloud storage is a lack of mutual trust between CSP and users. In recent years, to address this problem, many cloud auditing techniques, such as PDP, POR and POW, have been presented. However, auditing for operational behaviors in cloud, which is significant for the detection of potential crimes in the cloud and equitable accountability determination in the cloud forensic, was almost neglected in previous studies. In this paper, we outline the existing challenges and problems of conducting operational behavior auditing in cloud, including the log verification and forensic analysis phase. We also describe the existing approaches for secure logging and forensic analysis with distributed and parallel computing, which would be a reference to solve the crucial problems of OBA in the future research. We would like to motivate more researchers to focus on the how to determine the legitimacy of operational behavior, and to achieve accountability determination in cloud.

Acknowledgments. This work was supported in part by Natural Science Foundation of China under Grant Nos. U1405254, U1536115 and 61302094, Program of China Scholarships Council under Grant No. 201507540001, Natural Science Foundation of Fujian Province of China under Grant No. 2014J01238, Program for New Century Excellent Talents in Fujian Province University under Grant No. MJK2016-23, Program for Outstanding Youth Scientific and Technological Talents in Fujian Province University under Grant No. MJK2015-54, Promotion Program for Young and Middle-aged Teacher in Science & Technology Research of Huaqiao University under Grant No. ZQN-PY115, and Program for Science & Technology Innovation Teams and Leading Talents of Huaqiao University under Grant No. 2014KJTD13.

References

1. Mell, P., Grance, T.: Draft NIST working definition of cloud computing. Technical report (2009) http://csrc.nist.gov/groups/SNS/cloud-computing/index.html
2. Wang, C., Ren, K., Lou, W., Jin, L.: Toward publicly auditable secure cloud data storage services. IEEE netw. **24**, 19–24 (2010)
3. Ren, K., Wang, C., Wang, Q.: Security challenges for the public cloud. IEEE Internet Comput. **16**, 69–73 (2012)
4. Ko, Ryan K.L., Lee, B.S., Pearson, S.: Towards achieving accountability, auditability and trust in cloud computing. In: Abraham, A., Mauri, J.L., Buford, John F., Suzuki, J., Thampi, Sabu M. (eds.) ACC 2011. CCIS, vol. 193, pp. 432–444. Springer, Heidelberg (2011). doi:10.1007/978-3-642-22726-4_45
5. Yang, K., Jia, X.: Data storage auditing service in cloud computing: challenges, methods and opportunities. World Wide Web **15**, 409–428 (2012)
6. Tian, H., Chen, Y., Chang, C.C., Jiang, H., Huang, Y., Chen, Y.H., Liu, J.: Dynamic-hash-table based public auditing for secure cloud storage. IEEE Trans. Serv. Comput. (2015). doi:10.1109/TSC.2015.2512589
7. Wang, Q., Wang, C., Ren, K., Lou, W., Li, J.: Enabling public auditability and data dynamics for storage security in cloud computing. IEEE Trans. Parallel Distrib. Syst. **22**, 847–859 (2011)

8. Juels, A., Kaliski, B.S.: PoRs: proofs of retrievability for large files. In: 14th ACM Conference on Computer and Communications Security, pp. 584–597 (2007)
9. Shacham, H., Waters, B.: Compact proofs of retrievability. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 90–107. Springer, Heidelberg (2008). doi:10.1007/978-3-540-89255-7_7
10. Wang, G., Liu, Q., Wu, J.: A hierarchical attribute-based encryption for fine-grained access control in cloud storage services. In: 17th ACM Conference on Computer and Communications Security, pp. 735–737 (2010)
11. Yang, K., Jia, X., Ren, K., Zhang, B., Xie, R.: DAC-MACS: effective data access control for multiauthority cloud storage systems. IEEE Trans. Inf. Forensics Secur. **8**, 1790–1801 (2013)
12. Halevi, S., Harnik, D., Pinkas, B., Peleg, A.S.: Proofs of ownership in remote storage systems. In: 18th ACM Conference on Computer and Communications Security, pp. 49–500 (2011)
13. Zheng, Q., Xu, S.: Secure and efficient proof of storage with deduplication. In: 2nd ACM Conference on Data and Application Security and privacy, pp. 1–12 (2012)
14. Martini, B., Choo, K.K.R.: An integrated conceptual digital forensic framework for cloud computing. Digit. Invest. **9**, 71–80 (2012)
15. Dykstra, J., Sherman, A.T.: Acquiring forensic evidence from infrastructure-as-a-service cloud computing: exploring and evaluating tools, trust, and techniques. Digit. Invest. **9**, S90–S98 (2012)
16. Zawoad, S., Dutta, A.K., Hasan, R.: Towards building forensics enabled cloud through secure logging-as-a-service. IEEE Trans. Dependable Secure Comput. **13**, 148–162 (2016)
17. Zawoad, S., Dutta, A.K., Hasan, R.: SecLaaS: secure logging-as-a-service for cloud forensics. In: 8th ACM SIGSAC Symposium Information, Computer and Communications Security, pp. 219–230 (2013)
18. Bellare, M., Yee, B.: Forward integrity for secure audit logs. Technical report, Computer Science and Engineering Department (1997)
19. Schneier, B., Kelsey, J.: Secure audit logs to support computer forensics. ACM Trans. Inf. Syst. Secur. **2**, 159–176 (1999)
20. Ma, D., Tsudik, G.: A new approach to secure logging. ACM Trans. Storage **5**, 1–21 (2009)
21. Yavuz, A.A., Ning, P., Reiter, M.K.: Efficient, compromise resilient and append-only cryptographic schemes for Secure audit logging. In: Keromytis, A.D. (ed.) FC 2012. LNCS, vol. 7397, pp. 148–163. Springer, Heidelberg (2012). doi:10.1007/978-3-642-32946-3_12
22. Holt, J.E.: Logcrypt: forward security and public verification for secure audit logs. In: 4th Australasian Workshops on Grid Computing and E-research, pp. 203–211 (2006)
23. Accorsi, R.: BBox: a distributed secure log architecture. In: Camenisch, J., Lambrinoudakis, C. (eds.) EuroPKI 2010. LNCS, vol. 6711, pp. 109–124. Springer, Heidelberg (2011). doi:10.1007/978-3-642-22633-5_8
24. Stathopoulos, V., Kotzanikolaou, P., Magkos, E.: A framework for secure and verifiable logging in public communication networks. In: Lopez, J. (ed.) CRITIS 2006. LNCS, vol. 4347, pp. 273–284. Springer, Heidelberg (2006). doi:10.1007/11962977_22
25. Boneh, D., Lynn, B., Shacham, H.: Short signatures from the Weil pairing. J Cryptol. **17**, 297–319 (2004)
26. Roussev, V., Richard, L., G.G.: Breaking the performance wall: the case for distributed digital forensics. In: 2004 Digital Forensics Research Workshop, vol. 94 (2004)
27. Marziale, L., Richard, G.G., Roussev, V.: Massive threading: using GPUs to increase the performance of digital forensics tools. Digit. Invest. **4**, 73–81 (2007)
28. Francois, J., Wang, S., Bronzi, W.: Botcloud: Detecting botnets using mapreduce. In: IEEE International Workshop on Information Forensics and Security, pp. 1–6 (2011)

29. Roussev, V., Wang, L., Richard, G., Marziale, L.: A cloud computing platform for large-scale forensic computing. In: Peterson, G., Shenoi, S. (eds.) DigitalForensics 2009. IAICT, vol. 306, pp. 201–214. Springer, Heidelberg (2009). doi:10.1007/978-3-642-04155-6_15
30. Therdphapiyanak, J., Piromsopa, K.: Applying Hadoop for log analysis toward distributed IDS. In: 7th ACM International Conference on Ubiquitous Information Management and Communication, vol. 3 (2013)
31. Lin, X., Wang, P., Wu, B.: Log analysis in cloud computing environment with Hadoop and Spark. In: 5th IEEE International Conference on Broadband Network and Multimedia Technology, pp. 273–276 (2013)