

## RESEARCH

# Dizang: A solution for collecting forensic evidences in cloud environments

Hamilton J S Fonte II\* and Marcos A. Simplicio Jr.

\*Correspondence:  
hamiltonii@gmail.com  
Escola Politécnica, Universidade  
de São Paulo (USP), Av. Prof.  
Luciano Gualberto, 380,  
05508-010 São Paulo, SP, BR  
Full list of author information is  
available at the end of the article

### Abstract

Cloud architectures are increasingly more common, as is the number of security issues involving this technology. Unfortunately, due to the volatile nature of virtualized resources in the cloud, the task of gathering evidences for forensic analysis currently faces practical and legal challenges. In this work, we address this issue by analyzing proposals aimed at meeting such challenges, discussing their limitations and then presenting a solution to overcome them. The proposal specifically focuses on the reproducibility of the collection process in virtualized environments, without violating jurisdictions or the privacy of those not involved in the investigation. As such, it should be a useful tool for analyzing the causes of security incidents in cloud computing systems.

**Keywords:** Cloud computing; containers; digital forensics; cloud forensics; evidence acquisition; chain of custody

### Introduction

Virtualization techniques, replication of services and resource sharing among multiple users (multitenancy) are key enablers for the high scalability of computational clouds [1]. At the same time, however, these mechanisms also lead to the volatility of the virtual resources executing cloud-based applications. After all, when submitted to a high load, a cloud application may create clones of the virtual machines (VMs) or containers hosting it, and then balance the load among those copies; this auto-scaling behavior is expected to avoid any degradation of the quality provided by the service. After the load subsides, the cloned instances are usually deactivated, their resources are released and the system returns to the previous capacity, thus avoiding unnecessary costs.

Despite interesting from the efficiency and cost viewpoints, such volatility of the cloud is likely to cause problems from the perspective of attack response teams and forensic experts. For example, suppose that a temporary virtual processing instance undergoes an attack that directly affects its memory, without leaving traces in permanent storage (e.g., log files). In this case, the evidences of this event may be completely lost after such instances are deactivated and their resources are released. This issue is further aggravated by aspects such as multitenancy and multi-jurisdiction, typical of cloud solutions [2]. Particularly, the multitenancy aspect makes it harder to isolate the hardware executing the applications of interest: since each piece of hardware is shared by a number of users, physically removing them for analysis could lead to privacy violation of users not related to the investigation. Moreover, due to the distributed nature of the cloud, data relevant to the investiga-

tion may be allocated in different countries. In practice, this encumbers the acquisition of the corresponding information, especially when there are no cooperation agreements among the entities involved [3]. Combined, these characteristics hinder the implementation of an evidence collection process. As a result, the response to memory-oriented attacks may be delayed, and evidences collected afterward may not have the necessary credibility so that they can be accepted in legal processes. In particular, it becomes harder for forensic experts to comply with privacy, jurisdiction and chain of custody requirements, as well as to ensure the reproducibility of the collection process [4].

Even though the literature include solutions aimed at collecting information in the cloud for forensic analysis, most of them handle collection, transport and storage in an isolated manner. For example, [5] and [6] deal with factors such as multitenancy and multi-jurisdiction, discussing forms of collecting and preserving evidence outside the cloud. In comparison, [7] concentrates on forensic analysis to collect evidence from VMs while they are being executed, whereas [8] deals with ensuring the chain of custody when transporting evidences. However, none of the proposals identified in the literature (1) describe how data are collected and stored observing the chain of custody, and (2) create the required conditions for reproducing the evidence collection process even if a virtualized resource is decommissioned.

Aiming to overcome these limitations, this work presents Dizang, a proposal focusing on: (1) the reproducibility of the collection process; (2) establishing a link between the evidence collected and its origin (assuming the cloud resource is univocally identifiable); and (3) preserving the jurisdiction and the privacy of those not involved in the investigation. As such, the proposed solution can be used as tool for incident management, safeguarding evidences for analysis during a post-incident stage. In addition, these characteristics contribute to the credibility of collected evidences and, thus, to their acceptability in a possible legal process. When compared with similar-purpose solutions, Dizang is particularly useful against code injection attacks [9], which would normally leave no traces when cloud-based virtual instances are deactivated and their memory is released [10, 9].

The rest of this paper is organized as follows. Section ?? briefly discusses cloud solutions and their characteristics. Section ?? analyzes the related works, in particular those focused on forensic analysis of (virtualized) memory information. Section ?? details the proposed solution and its features. Section ?? presents our final considerations.

### **Problem statement: virtualization vs. forensics in the cloud**

Cloud computing refers to a model that provides network access to a configurable amount of computing resources, in such a manner that users can allocate and release computational resources on demand and with minimal management effort [11]. Depending on which resources are provided to clients, three main cloud service models can be defined [11]: software as a service (SaaS), in which the software to be used by the clients is provided; platform as a service (PaaS), in which an environment is provided for clients to develop, test and execute their software; and infrastructure as a service (IaaS), in which basic computational resources are provided (e.g., processing, memory and storage), usually by means of virtualization. In this work,

we focus on the IaaS service model, where clients have further control over the underlying cloud resources.

The traditional way of virtualizing IaaS resources in the cloud is to rely on virtual machines (VMs) [12]. More recently, however, there has been an increasing interest in using containers for this purpose. Indeed, according to a study conducted in 2016 with 235 companies involved with software development [13], 76% of the respondents used containers to improve the efficiency of their development process and of their cloud micro-services architecture. However, whereas VMs involve instantiating a virtual hardware and also an operating system (OS) on top of the native system, virtualization with containers is conducted at the level of the native OS. According to [12], containers allow for a simpler implementation and better resource usage, eliminating layers between the application being executed and the physical hardware. They also have a lower total cost and a more predictable performance.

Whichever the virtualization technology employed, the result is a highly volatile memory environment. This happens because the cloud's on-demand nature implies that computational resources are allocated and released following the actual system's load. Therefore, it is not uncommon that attack response teams are unable to access all possible evidences of a breach because the pieces of volatile memory containing those evidences have already been released as part of the cloud's automatic scaling policies. From a forensic perspective, this is specially troublesome when dealing with injection attacks that operate directly on the target's volatile memory, without affecting the long term storage of VMs or containers [9]. Examples include:

- *Remote library injection*: A malicious process forces the target process to load a library into its memory space [14]. As a result, that code inside that library is executed with the same privileges as the target process. This strategy, commonly used for enabling a malware to be installed in a victim's machine, may also involve storing the malicious library in the system so it can be loaded by different processes.
- *Inline Hooking*: A malicious process writes a piece of code directly into the target process's memory space, as a sequence of bytes, so the victim that code as if it was part of its own design [15]. This is commonly employed to force the execution of shell scripts, giving the attacker remote control over the target machine.
- *Reflective library injection*: a malicious process directly access the target process's memory and writes into it the bytecode corresponding to a library, forcing the victim to execute the injected instructions [16]. In this case, the malicious library is not stored in the system and its loading is not registered by the operation system's logs, making this kind of attack harder to detect.

The ability to analyze the state of volatile memory is, thus, an important requirement for enabling an effective incident response procedure, as well as post-mortem analyses of such attacks. However, it is challenging task to balance the conflicts between (1) this security need and (2) the cloud's performance requirements, usually met via rapid elasticity. In the next section, we discuss some of the proposals in the literature aimed at addressing this issue.

Sub-heading for section

Text for this sub-heading ...

*Sub-sub heading for section*

Text for this sub-sub-heading ...

*Sub-sub-sub heading for section* Text for this sub-sub-sub-heading ... In this section we examine the growth rate of the mean of  $Z_0$ ,  $Z_1$  and  $Z_2$ . In addition, we examine a common modeling assumption and note the importance of considering the tails of the extinction time  $T_x$  in studies of escape dynamics. We will first consider the expected resistant population at  $vT_x$  for some  $v > 0$ , (and temporarily assume  $\alpha = 0$ )

$$E[Z_1(vT_x)] = E\left[\mu T_x \int_0^{v \wedge 1} Z_0(uT_x) \exp(\lambda_1 T_x(v-u)) du\right].$$

If we assume that sensitive cells follow a deterministic decay  $Z_0(t) = xe^{\lambda_0 t}$  and approximate their extinction time as  $T_x \approx -\frac{1}{\lambda_0} \log x$ , then we can heuristically estimate the expected value as

$$\begin{aligned} E[Z_1(vT_x)] &= \frac{\mu}{r} \log x \int_0^{v \wedge 1} x^{1-u} x^{(\lambda_1/r)(v-u)} du \\ &= \frac{\mu}{r} x^{1-\lambda_1/\lambda_0 v} \log x \int_0^{v \wedge 1} x^{-u(1+\lambda_1/r)} du \\ &= \frac{\mu}{\lambda_1 - \lambda_0} x^{1+\lambda_1/rv} \left(1 - \exp\left[-(v \wedge 1) \left(1 + \frac{\lambda_1}{r}\right) \log x\right]\right). \quad (1) \end{aligned}$$

Thus we observe that this expected value is finite for all  $v > 0$  (also see [?, ?, ?, ?, ?]).

#### Competing interests

The authors declare that they have no competing interests.

#### Author's contributions

Text for this section ...

#### Acknowledgements

Text for this section ...

#### References

1. Morsy, A.M., Grundy, J., Muller, I.: An Analysis of the Cloud Computing Security Problem (2010)
2. Keyun, R., Joe, C., Tahar, K., Mark, C.: Advances in Digital Forensics VII. In: Gilbert, P., Sujeet, S. (eds.) 7th IFIP WG 11.9 International Conference on Digital Forensics vol. 1, 7th edn. Orlando (2011). Chap. 3. arXiv:1011.1669v3. doi:10.1017/CBO9781107415324.004
3. Dykstra, J., Sherman, A.: Acquiring forensic evidence from infrastructure-as-a-service cloud computing: Exploring and evaluating tools, trust, and techniques, vol. 9, pp. 90–98. Elsevier Ltd, ??? (2012). doi:10.1016/j.diin.2012.05.001. (Proc. of the 12th Annual DFRWS Conference). dx.doi.org/10.1016/j.diin.2012.05.001
4. Rahman, S., Khan, M.N.A.: Review of live forensic analysis techniques. International Journal of Hybrid Information Technology 8(2), 379–388 (2015). doi:10.14257/ijhit.2015.8.2.35
5. Dykstra, J., Sherman, A.T.: Design and implementation of FROST: Digital forensic tools for the OpenStack cloud computing platform, vol. 10, pp. 87–95. Elsevier Ltd, ??? (2013). doi:10.1016/j.diin.2013.06.010. (Proc. of 13th Annual DFRWS Conference). dx.doi.org/10.1016/j.diin.2013.06.010
6. Reichert, Z., Richards, K., Yoshigoe, K.: Automated forensic data acquisition in the cloud. IEEE Int. Conf. on Mobile Ad Hoc and Sensor Systems, 725–730 (2015). doi:10.1109/MASS.2014.135
7. George, S., Venter, H., Thomas, F.: Digital Forensic Framework for a Cloud Environment. In: IST Africa Conference Report, pp. 1–8. IIMC, Tanzania (2012)
8. Sang, T.: A log-based approach to make digital forensics easier on cloud computing. 2013 3rd International Conference on Intelligent System Design and Engineering Applications, ISDEA 2013, 91–94 (2013). doi:10.1109/ISDEA.2012.29
9. Case, A., Ligh, M., Jamie, L., Walters, A.: The Art of Memory Forensics: Detecting Malware and Threats in Windows, Linux and Mac Memory. Wiley, Hoboken, NJ (2014)

10. Vömel, S., Stüttgen, J.: An evaluation platform for forensic memory acquisition software. In: The Digital Forensic Research Conference, vol. 10. Monterey, Ca, pp. 30–40 (2013). doi:10.1016/j.diin.2013.06.004

11. Mell, P., Grance, T.: The NIST definition of cloud computing. NIST SP 800-145 (2011). 2305-0543. doi:10.1136/emj.2010.096966. csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf

12. Diamanti: Five reasons you should run containers on bare metal, not vms. Technical report, Diamanti (2018). Available: [https://diamanti.com/wp-content/uploads/2018/07/Diamanti\\_WP\\_Five\\_Reasons\\_You\\_Should\\_Run\\_Containers\\_on\\_Bare\\_Metal\\_071918.pdf](https://diamanti.com/wp-content/uploads/2018/07/Diamanti_WP_Five_Reasons_You_Should_Run_Containers_on_Bare_Metal_071918.pdf)

13. DevOps and ClusterHQ: Container market adoption survey 2016. Technical report, DevOps and ClusterHQ (2016). Available: <https://clusterhq.com/assets/pdfs/state-of-container-usage-june-2016.pdf>

14. Miller, M., Turkulainen, J.: Remote library injection. Technical report, NoLogin.org (2004). Available: <http://www.hick.org/code/skape/papers/remote-library-injection.pdf>

15. Yin, H., Liang, Z., Song, D.: HookFinder: Identifying and Understanding Malware Hooking Behaviors

16. Fewer, S.: Reflective DLL injection – v1. Technical report, Harmony Security (Oct 2008)

17. Poisel, R., Malzer, E., Tjoa, S.: Evidence and cloud computing: The virtual machine introspection approach. Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications **4**(1), 135–152 (2013)

18. Dolan-Gavitt, B., Leek, T., Zhivich, M., Giffin, J., Lee, W.: Virtuoso: Narrowing the semantic gap in virtual machine introspection. In: IEEE Symposium on Security and Privacy, pp. 297–312. IEEE, Plymouth, UK (2011). doi:10.1109/SP.2011.11

19. Aljaedi, A., Lindskog, D., Zavarisky, P., Ruhl, R., Almari, F.: Comparative analysis of volatile memory forensics: Live response vs. memory imaging. In: IEEE 3rd Int. Conf. on Privacy, Security, Risk and Trust, pp. 1253–1258 (2011). doi:10.1109/PASSAT/SocialCom.2011.68

20. Dezfouli, F., Dehghantanha, A., Mahmoud, R., Sani, N., Shamsuddin, S.: Volatile memory acquisition using backup for forensic investigation. In: Int. Conf. on Cyber Security, Cyber Warfare and Digital Forensic (CyberSec), pp. 186–189. IEEE, Plymouth, UK (2012). doi:10.1109/CyberSec.2012.6246108

21. van Baar, R.B., van Beek, H.M.A., van Eijk, E.J.: Digital Forensics as a Service: A game changer. Digital Investigation **11**, 54–62 (2014). doi:10.1016/j.diin.2014.03.007

Figures

**Figure 1** Sample figure title. A short description of the figure content should go here.

**Figure 2** Sample figure title. Figure legend text.





































Tables

Additional Files

Additional file 1 — Sample additional file title  
Additional file descriptions text (including details of how to view the file, if it is in a non-standard format or the file extension). This might refer to a multi-page table or a figure.

Additional file 2 — Sample additional file title  
Additional file descriptions text.

**Table 1** Solutions for collecting memory information from cloud machines for forensic analysis

	Continuous collection of relevant data	Chain of custody guarantees	Ability to reproduce the evidence collection process	Jurisdiction and privacy preservation
Dizang (this proposal)				
[7]				
[17]				
[5]				
[6]				
[8]				
[18]				
[19]				
[20]				
[21]	