INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 10(2):222-229, April, 2015.

Logging for Cloud Computing Forensic Systems

A. Pătraşcu, V.V. Patriciu

Alecsandru Pătraşcu*
1. Military Technical Academy, Computer Science Department
39-40 George Coşbuc Street, Bucharest, Romania
alecsandru.patrascu@gmail.com
2. Advanced Technologies Institute
10 Dinu Vintila, District 2, 021102, Bucharest, Romania
ati@dcti.ro
*Corresponding author: ati@dcti.ro

Victor Valeriu Patriciu
Military Technical Academy, Computer Science Department
39-40 George Coşbuc Street, Bucharest, Romania
victorpatriciu@yahoo.com

Abstract: Cloud computing represents a different paradigm in the field of distributed computing that involves more and more researchers. We can see in this context the need to know exactly where, when and how a piece of data is processed or stored. Compared with classic digital forensic, the field of cloud forensic has a lot of difficulties because data is not stored on a single place and furthermore it implies the use of virtualization technologies.
In this paper we present a new method of monitoring activity in cloud computing environments and datacenters by running a secure cloud forensic framework. We talk in detail about the capabilities that such system must have and we propose an architecture for it. For testing and results we have implemented this solution to our previous developed cloud computing system.
Keywords: cloud computing; data forensics; logging framework; distributed computing; binary diff

1 Introduction

Cloud Computing to put it simply, means Internet Computing. It is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

The cloud computing model offers the promise of massive cost savings combined with increased IT agility. It is considered critical that government and industry begin adoption of this technology in response to difficult economic constraints. However, cloud computing technology challenges many traditional approaches to datacenter and enterprise application design and management. Cloud computing is currently being used. However, security, interoperability, and portability are cited as major barriers to broader adoption.

In this context, a new need for IT experts is increasing: the need to know exactly how, where and in what condition is the data from the cloud stored, processed and delivered to the clients. We can say with great confidence that cloud computing forensics has become more and more a need in todays distributed digital world.

In this paper we are going to present a new way in which we can integrate a full forensics framework on top of a new or existing cloud infrastructure. We will talk about the architecture behind it and we will present its advantages for the entire cloud computing community. We will present also the impact that our technology proposal will have on existing cloud infrastructures and as a proof of concept we will present some particular implementation details over our own cloud computing framework that we have already developed in [6].

The rest of the document is structured as follows. In section 2 we present some of the related work in this field, that is linked with our topic and in section 3 we present in detail our proposed cloud forensics logging framework. Section 4 is dedicated to presenting our results from our implementation made so far, and in section 5 we conclude our document.

2 Related work

The integration of cloud computing logging in the field of forensics is not new and we find thesis in this directions, such as the one of Zawoad et al [1] which present an architecture for a secure cloud logging service that collects information from various sources around the datacenter, both software (hypervisors) and hardware (network equipments)in order to create a complete image of the operations done in a datacenter.

The same challenges are evidenced by Marty [2] and Sibiya et al [3]. In their papers they present a perspective over a custom logging framework and talk about the way in which forensics investigators can be provided with reliable and secure data using a standardized way. They propose using a single centralized log collector and processor, in order to save business's and users time.

In order to face the many challenges involving digital forensics in general, but also to take benefits from the opportunities cloud computing is offering, we have to rethink most of the classic network established principles and re-organize well-known workflows, even include and use tools not previously considered viable for forensic use, such as machine learning or large scale computing. Furthermore we must submit to the classic digital forensic main rule and keep all digital evidence intact. All of our investigation is done on a digital copy of the original data.

In our previous research [10] we have also focused on choosing a proper data representation format that will be used between the modules of our framework and between the modules and the central forensic processing core. In the next paragraphs we present a brief comparison of two existing proposals in this directions, that are applicable in our context.

The first one is the "Management metalanguage" [12] proposed by the UnixWare community. Its advantage is that it can be used as a transparent API in the kernel modules as it provides an interface for an external host. The downside is that it needs a lot of auxiliary binary data to be sent in order to re-create the entire picture at the other end, and using it we get quickly a traffic larger than the one that can be obtained by sending only the basic snapshots.

On the other side, the CEE (Common Event Expression) organization [11] proposes a set of specifications using the JSON and XML markup languages for event logging on disk or in transit over a network. These requirements are designed for maximum interoperability with existing event and interchange standards to minimize adoption costs. The advantage of this approach is that CEE expresses its interfaces and does not promote an actual implementation.

After thoroughly analyzing these two proposals we have chosen to use a combination between them, meaning that we want to full details that the management metalanguage encapsulates, under the form of JSON data representation.

3 Logging framework architecture

In the following section we will talk about the top view architecture of our cloud computing enabled forensic system. We will present the main building blocks and modules and then we focus on the logging sub-system. The entire architecture will follow also the perspective from the forensic investigator part.

3.1 General forensics architecture

The framework presented in this paper has a modular architecture and each of the modules is presented in the following paragraphs in detail. It is also easy to see that the entire framework can be extended with other modules or plugins to support various workloads and even processing elements. In order to have a working platform, we must first introduce the concept of a cloud computing framework. In Figure 1 we can see that the top view of a cloud computing framework contains two main layers: a virtualization layer and a management layer.

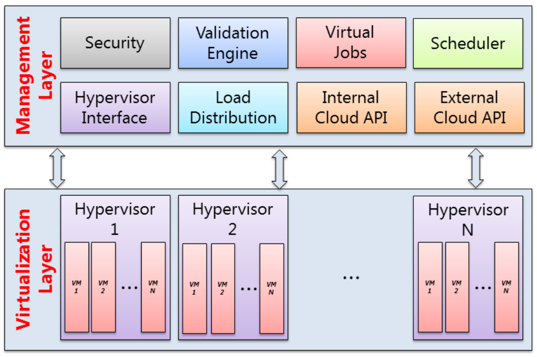

Figure 1: Basic cloud computing architecture

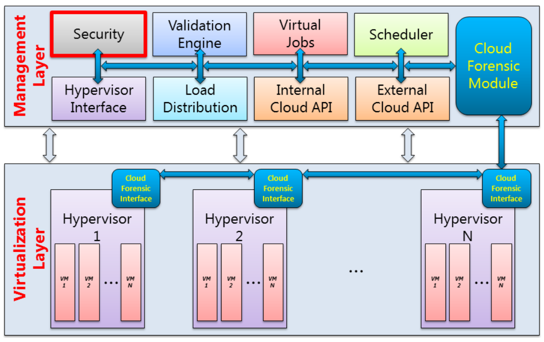

Figure 2: Forensic enabled cloud architecture

In the *Virtualization layer* we find the actual platforms/servers that host the virtual machines and have virtualization enabled hardware. In the *Management layer* we find the modules responsible for enabling the entire operations specific to the cloud. These modules are, in order: **Security** (responsible with all security concerns related to the cloud system - intrusion detection and alarming module), **Validation engine** (receives requests to add new jobs to be processed), **Virtual jobs** (creates an abstraction between the data requested by the user and the payload that must be delivered to the cloud system), **Scheduler** (schedules the jobs to the virtualization layer), **Hypervisor interface** (acts like a translation layer that is specific to a virtualization software vendor), **Load distribution** (responsible with horizontal and vertical scaling of the requests received from the scheduler), **Internal cloud API** (intended as a link between the virtualization layer and the cloud system), **External cloud API** (offers a way to the user for interacting with the system).

Now that the notion of a cloud computing framework was presented, we will talk about the modifications that must be made to it in order to create an forensic enabled cloud computing architecture. As can be seen in Figure 2 the modification affects all the existing modules and includes two new modules, the *Cloud Forensic Module* and the *Cloud Forensic Interface*. Their main goal is to gather all forensic and log data from the virtual machines that are running inside the virtualization layer and it represents the interface between the legal forensic investigator and the monitored virtual machines. The investigator has the possibility to monitor one or more virtual machine for a targeted user for a specific amount of time.

3.2 Cloud logging architecture

In this section we present how our framework is working and how it is created in order to run on top of new or existing cloud computing infrastructures. As example for it we will present the integration with our previously implemented Cloud Computing framework. The cloud architecture presented in our previous work makes use of the concept of leases, in which we can specify the amount of time the job must run, or specify between what hours in a day it is running.

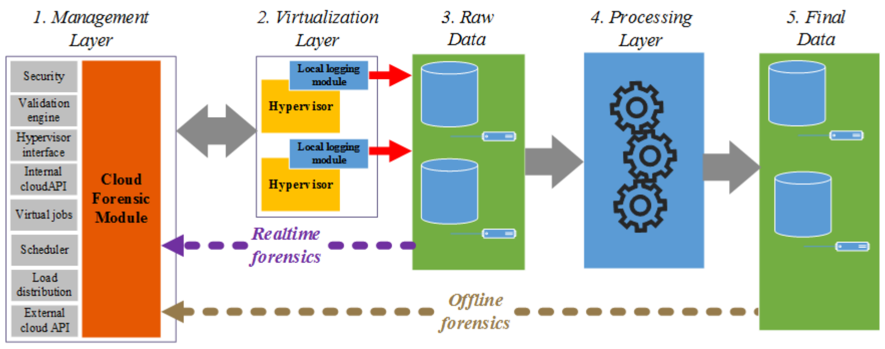

Figure 3: Cloud Forensics Logging Framework

The cloud logging system architecture is a layered one, containing five layers, each with its own purpose. We will present each of them in detail in the following paragraphs. We also used a graphical representation, in Figure 3, where the whole layers and the relationship between them can be seen. The layers are all implemented using the distributed computing paradigm and they represent jobs in our cloud computing environment.

The first layer, as presented in our previous work [4], represents the *management layer* in a cloud computing deployment. All the modules that are responsible with all cloud specific operations can be found at this level, together with the forensic targeted ones, such as "Cloud Forensic Module".

The second layer represents the *virtualization layer* in a cloud computing implementation. At this level we can find the workstations and servers that host the virtual machines. The fact that the main building blocks are represented by the virtual machines, the hardware must also have virtualization enabled. Inside the *Cloud Forensic Interface*, a dedicated "Local logging module" must be installed into the existing physical machine. It is responsible with the RAW data gathering from the monitored virtual machines. The data quantity can be adjusted by the investigator and he can choose to monitor a particular virtual machine or monitor the entire activity existing inside that machine.

In order to gather data reliably from the virtual machines the local logging module must be integrated fully with the running hypervisor inside the physical machine. In this paper we focus on the integration with the "KVM" virtualization technology that exists in modern Linux kernel releases. We have chosen it because it is a full open-source virtualization solution, integrated with the Linux kernel since 2007 and it is actively used by many companies across the world.

An important thing that must be taken in consideration is *what* data are we intercepting from the virtual machine and send it to further processing. Since all the activity can be intercepted, there is the risk of severe time penalties and processing speed. In order to solve this problem, at this point we will offer the possibility for an investigator to choose the logging level for a certain virtual machine. This is helpful considering that, for example, an investigator only wants to analyze the virtual memory for its contents, and it is not interested in virtual disk images or virtual network activity. Also at this step we must consider the problem of network transmission overhead.

The third layer represents a storage layer for the RAW data sent from the local logging modules existing in the virtualization layer. The logging modules will send RAW data, in the form they are gathered from the hypervisor. Thus, this layer has the function of a distributed storage and it contains a series of nodes, each running a database. We have chosen this approach in order to create a flexible and scalable layer architecture that can face the data traffic coming from the upper layer.

Since the data that is going to be sent from the physical virtualization host to the central forensic management unit can reach important size, we will implement a mechanism of *"diff"* between two pieces of data. For example, if an investigator will want to analyze a virtual machine memory over a period, the local forensic module will sent only one initial memory snapshot and after that only what has been changed will be sent. Of course we can use the full potential of the host and provide a local aggregation module that will pre-process the data collected before sending it to the central forensic module. This approach is new to the field of cloud computing forensics and we consider it a great way to reduce the impact over the network.

The process will run in the following manner. Initially the logging modules will send a reference file and then, at an user defined time period, the modules will send a delta file, that represents the difference between the previous reference file and the current state. Thus, it will implement a snapshot mechanism at the hypervisor level. We have chosen this approach because we want to offer to the forensic investigator the possibility to have an image of what is happening inside a virtual machine between two snapshots. This feature is currently not available in other hypervisors, such as VMware's; in their case we can have a snapshot at time t_0 and one at time t_i, but we cannot know the state of the virtual machine between the *0* and i step.

This layer has also another purpose. In case of extreme emergency, the forensic investigator can "see" a real-time evolution of the monitored virtual machine by issuing a direct connection to this layer. This feature is made available through the Cloud Forensic Module, which has the ability to by-pass normal RAW data processing.

The fourth layer has the purpose of analyzing, ordering, processing and aggregating the data stored in the previous layer. Since all these steps are computing intensive, the entire analysis process will be made in an offline manner and will be available to the investigators as soon as the job is ready. After this entire process the investigator will have a full image of what happened over the monitored remote virtual machine in a manner such as the one encountered in software source code version tools, thus permitting him to navigate back and forth into the history of the virtual machine.

This layer is implemented also as a distributed computing applications. We have chosen this approach due to the processing power needs that our framework demands, more exactly it needs to do correlations between different snapshots in a fair amount of time.

Finally, **the fifth layer** represents the storage of the results published by the previous layer. An forensic investigator will interact with the monitored virtual machine snapshots at this layer, by using the Cloud Forensic Module from the Management layer.

4 Results

In this section we are going to present details regarding the results collected after the implementation of our Cloud Logging modules.

4.1 Network configuration

For testing, the modules have been implemented and split across multiple workstations, as can be seen in Figure 4.

They are represented as a cluster of servers, each having the functionality presented in detail in the architecture section. As it can be seen, the entire modules found in the dotted perimeter,

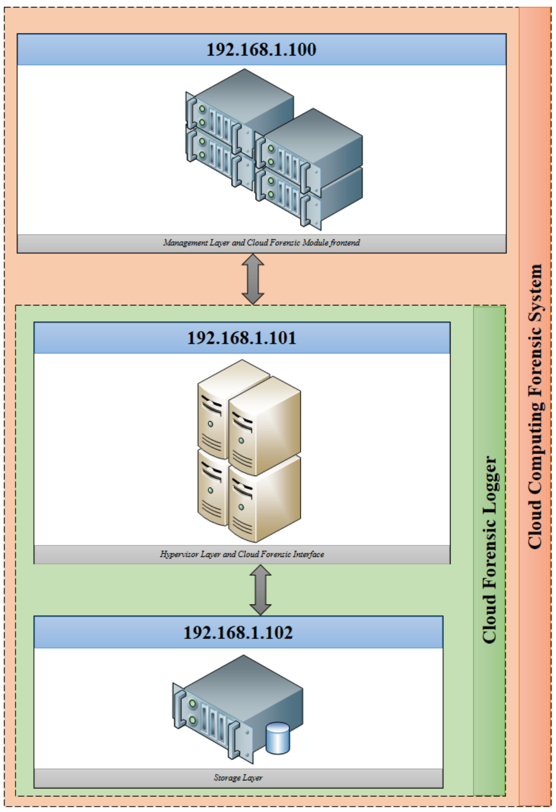
192.168.1.100
Management Layer and Cloud Forensic Module frontend
192.168.1.101
Hypervisor Layer and Cloud Forensic Interface
192.168.1.102
Storage Layer
Cloud Forensic Logger
Cloud Computing Forensic System

Figure 4: Mapping modules to workstations.

called “Cloud Computing Forensic System”, can also be ran all on one workstation. Elements like network switches are not represented in order not to burden the graphic, but the IP address of the hosts are kept. In our configuration we have used three distinct workstation, each having the functionalities and network addresses presented in the figure.

The hardware platform used was composed from an AMD Phenom II X6, 6 cores, 8GB RAM, RAID0 configured hard-disks running KVM as hypervisor and QEMU as a hypervisor interface, an Intel DualCore, 4GB RAM as the storage layer and an AMD C-60 DualCore, 4GB RAM as the management layer. The network used is 10/100 MB.

4.2 Experimental results

The experiments were made using KVM as a hypervisor and QEMU and libvirt as drivers for the hypervisor. The tests had the target set on the virtual machine used memory (RAM snapshot) and the virtual machine storage (DISK snapshot).

The process of recording the virtual machine activity was made over a period of several hours, at a time step of 10 minutes. The CPU load when conducting records using all the 6 cores was about 20%. The results are interesting, if we take in consideration the technologies used internally by KVM. For example, RAM snapshots are made entirely from host machine RAM and do not contain necessarily consecutive RAM location. Nevertheless, in our experiments the RAM snapshots were the largest, reaching even gigabytes in size.

Bellow you can see the actual RAM tests that were made. We have split the tests in two distinct zones, one up to 100 MB and one after this barrier. Table 1 and Figure 5 presents the data collected from our modules and the time needed to process it. The transfer time between the Cloud Forensic Interface module and the Storage module is not taken in consideration, as being a constant time, of about 82 seconds for a 800 MB file. Table 2 and Figure 6 presents the data collected from our modules and the time needed to process it.

Table 1: Tests up to 100 MB in size

Size (KB)	Time (ms)
4	296
454	517
1227	1136
5505	4929
10813	8000

Table 2: Tests over 100 MB in size

Size (KB)	Time (ms)
108036	58982
740032	401156
4251346	2277855

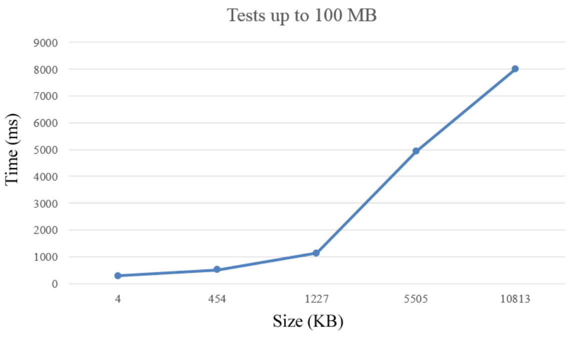

Figure 5: Tests up to 100 MB in size.

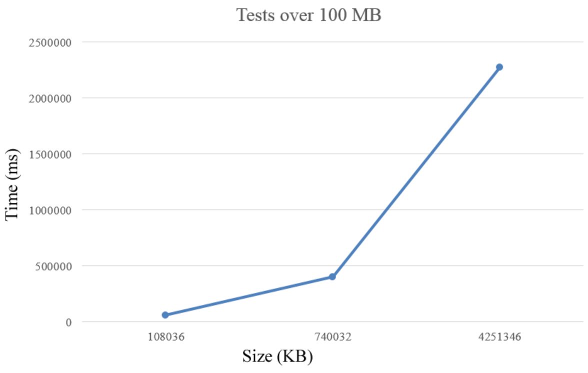

Figure 6: Tests over 100 MB in size.

5 Conclusion

As we have seen in this paper, the topic of cloud computing forensics is very large and poses great challenges in the field of logging. Due to the fact that together with incident response they represent a new field for research, more and more scientists are trying to develop new methods for assuring security in cloud systems. Furthermore, as the environment is purely distributed offers new fields of development much larger than a regular workstation.

In this paper we presented a novel solution that provides to the digital forensic investigators a reliable and secure method in which they can monitor user activity over a Cloud infrastructure. Our approach takes the form of a complete framework on top of an existing Cloud infrastructure and we have described each of its layers and characteristics. Furthermore, the experimental results prove its efficiency and performance.

Acknowledgment

This paper has been financially supported within the project entitled "Horizon 2020 - Doctoral and Postdoctoral Studies: Promoting the National interest through Excellence, Competitiveness and Responsibility in the Field of Romanian Fundamental and Applied Scientific Research", contract number POSDRU/159/1.5/S/140106. This project is co-financed by European Social Fund through the Sectoral Operational Programme for Human Resources Development 2007 - 2013. Investing in people!

Bibliography

[1] S. Zawoad, A.K. Dutta and R. Hasan (2013); SecLaaS: Secure Logging-as-a-Service for Cloud Forensics, in *ACM Symposium on Information, Computer and Communications Security*, DOI: 10.1145/2484313.2484342, 219-230.

[2] R. Marty (2011); Cloud Application Logging for Forensics, *Proceedings of the 2011 ACM Symposium on Applied Computing*, 178-184.

[3] G. Sibiya, H. Venter, T. Fogwill (2012); Digital forensic framework for a cloud environment, *Proceedings of the 2012 Africa Conference*, 1-8.

[4] A. Pătraşcu and V. Patriciu (2013); Beyond Digital Forensics. A Cloud Computing Perspective Over Incident Response and Reporting, *IEEE International Symposium on Applied Computational Intelligence and Informatics*, 455-460.

[5] B. Grobauer and T. Schreck (2010); Towards incident handling in the cloud: challenges and approaches, *Proceedings of the 2010 ACM workshop on Cloud computing security workshop*, New York, DOI: 10.1145/1866835.1866850, 77-86.

[6] A. Pătraşcu, C. Leordeanu, C. Dobre and V. Cristea (2012); ReC2S: Reliable Cloud Computing System, *European Concurrent Engineering Conference*, Bucharest, 1-9.

[7] M. Simmons and H. Chi (2012); Designing and implementing cloud-based digital forensics, *Proceedings of the 2012 Information Security Curriculum Development Conference*, 69-74.

[8] T. Takahashi, Y. Kadobayashi and H. Fujiwara (2010); Ontological Approach toward Cybersecurity in Cloud Computing, *SIN'10 Proceedings of the 3rd international conference on Security of information and networks*, DOI: 10.1145/1854099.1854121, 100-109.

[9] NIST SP800-86 Notes, *Guide to Integrating Forensic Techniques into Incident Response*, http://cybersd.com/sec2/800-86Summary.pdf

[10] A. Pătraşcu and V. Patriciu (2014); Logging system for cloud computing forensic environments, *Journal of Control Engineering and Applied Informatics*, 16(1): 80-88.

[11] http://cee.mitre.org/language/1.0-beta1/cls.html

[12] http://uw714doc.sco.com/en/UDI_spec/m_mgmt.html