Beyond Digital Forensics. A Cloud Computing Perspective Over Incident Response and Reporting

Alescsandru Pătrașcu¹ and Victor-Valeriu Patriciu¹
¹Military Technical Academy, Computer Science Department, Bucharest, Romania
Email: alescsandru.patraescu@gmail.com, victorpatriciu@yahoo.com

Abstract—Cloud computing represents a different paradigm in distributed computing that involves more and more researchers. In this context, we can see that there is a need for knowing where and when a certain data is processed or stored. Compared with classic digital forensic, the field of cloud forensic poses a lot of difficulties since data is not stored on a single storage unit and furthermore it involves the use of virtualization technologies.

In this paper we will present in detail a new and novel way of monitoring user activity in cloud environments using a secure cloud forensic framework. We talk about the architecture of such framework and we emphasize the way in which our research can be applied on top of new or existing cloud infrastructures. Also, for testing purposes, we have applied our findings over our previous developed cloud computing framework.

Keywords - cloud computing; secure data forensics; cloud computing incident response; cloud computing forensics; Linux kernel virtualization; KVM; XEN.

I. INTRODUCTION

Cloud computing has become in the last years a paradigm that attracts more and more researchers. One of the main research areas in this field is the way in which common data and processing power can be shared and distributed across single or multiple datacenters that are spread across a specific geographical area or even the entire globe. In this context a new need for IT experts is increasing: the need to know exactly how, where and in what condition the data are 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 case of classic computer forensics, the purpose is to search, preserve and analyze information on computer systems to find potential evidence for a trial. In cloud environments the entire paradigm changes because we don’t have access to a physical computer, and even if we have access it is a great chance that the data stored on it is encrypted or split across multiple other computer systems.

Taking in account all the variables that have appeared in cloud computing technologies, modern hypervisors and virtualization technologies implement more or less simple mechanisms for data monitoring across datacenters. Starting from the basic building blocks composed by simple logs that are gathered from the entire cloud infrastructure, every monitoring module must have a precise target and must not affect the proper function of the systems from the datacenter. All virtualization technologies have a difficulty in this area.

We are going to present a new and novel 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 that stands at it ground 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 [5]. Of course we are not neglecting the security part of our proposal and we will present briefly a mechanism that helps us secure the transmissions across our cloud infrastructure modules.

Our paper is organized 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 and 4 we talk about what is incident response and incident reporting and how it is related to cloud environments together with challenges that arise from it. Section 5 contains our detailed architecture for a cloud computing forensic framework and in section 6 we conclude our document.

II. RELATED WORK

In the field of classic incident response there are a lot of active researches, books, guides and papers. Nevertheless, in the field of cloud computing incident response the papers are mostly theoretical and present only an ideal model for it.

In the direction of classic incident response, one of the most interesting guides is the one from NIST [11]. In it we can find a summary containing a short description and recommendations for the field of computer forensics, along with the basic steps that must be made when conducting a forensic analysis: collection, examination, analysis and reporting. A great deal of attention is payed to the problem on incident response and how should an incident be detected, isolated and analyzed.

Bernd Grobauer and Thomas Schreck talk in their paper [3] about the challenges imposed by cloud computing incident handling and response. This problem is also analyzed by Chen [2] and in their paper they consider that incident handling should be considered a well-defined part of the security process. Also it is presented a description for current processes and methods used in incident handling and what changes can be made when migrating towards a cloud environment from the point of view of a customer or an security manager.
Furthermore, the integration of cloud incident handling and cybersecurity is presented in two papers, one written by Taka-hashi et al [8] and the other written by Simmons et al [7]. They talk about how Internet development lead to a widespread deployment of various IT technologies and security advances. In their paper they also propose an ontological approach for integration of cybersecurity in the context of cloud computing and present how information should be used and analyzed in such environments.

III. INCIDENT RESPONSE AND INCIDENT REPORTING IN CLOUD ENVIRONMENTS

Since twenty years ago, experts working in incident response field were taught that it is better to shutdown down an entire system in case of a breach. Today, with cloud computing surrounding us, this task is impossible and we need new approaches. For example, in a virtual machine memory can be stored important data for an investigator, such as network connections and running processes, that in case of a power failure will be lost.

Given these realities, we need to handle incident response and reporting with great care. Incident responders and forensic investigators must now rely more and more on live collection and analysis of system RAM. Live response analysis has been used for the last six to eight years only in important high-end cases, under the command of highly qualified computer forensics specialists, but modern reality forces this approach to almost all data cases.

Current tools that are used in forensic analysis work in a simple way: you upload a remote agent to the target system, wait for and gather snapshots of data from memory and send it back to the investigators computers, where a human or a software program reviews them.

Furthermore, forensic investigators need to avoid copying the entire disk images. In today’s cloud environments data can easily reach large sizes and it is not always technically possible or cost effective to do it. Even more, judges, agents and investigators were classically taught that only a perfect data copy is a “forensically sound” copy that can be used. This must change because there is no need to copy an entire disk image in order to retrieve only a few files representing evidence and to accomplish this the forensic investigator will need also a way to view and analyze “live remote” data.

A. Cloud computing incident response

We must be careful how we deal with incident response in the cloud because we must make sure how we will respond to it. Traditional incident response is easy to do because everything is located inside the same network or datacenter. It is an environment in which you can reach the physical devices, you can take forensic copies of them. Traditional incident response consists from seven basic steps and procedures that we are going to present briefly.

The first step is detection, in which we identify the threat(s). After we detected a certain threat, we must contain it so it cannot spread to the rest of the systems and eradicate it. To prepare ourselves for future attacks we must remediate the problem(s) and fix the vulnerabilities in order to close all gaps that can breach the system. The recovery is important because we must restore the system from backups to the point before the threat emerged. Finally, after everything is back to normal, we must review the system and check it again and communicate what has been done to the administrators, key stockholders, media, etc.

The cloud model is different because we cannot “grab” the actual data because we cannot access the physical disks. And if we look at the standard definition of cloud computing we can see that the data can be stored even across multiple datacenters. The problem in this case, and one of the main challenges in cloud, is how can we take a forensic image of a cloud instance. Containing the cloud environment is also a problem because we make sure that a potential threat does not spread to other cloud instances.

The only solution for these problems is to change our mindset and try seeing the big picture differently than before. Of course we must still use the same basic principles existing in traditional incident response. The key is not to focus on the datacenter, but to focus on the actual data. We must move along datacenters and networks and pay attention to where is the data. An approach like the one used in airports to keep the airplanes secure can be used - starting from the first time a traveler enters an airport, up to getting in the plane, reaching the destination and safely getting down from the plane.

So, the main directions for our research is to focus on the data, to know where it is and keep it secure.

A step towards this is made in today’s cloud computing environments, as we encrypt the data stored online. This means that if something happens at the Cloud Service Provider (CSP) location, the user data is kept safely, because only the user has the decryption key. This is a great responsibility for the user because he is now the manager of his own keys. The important fact is not to permit the CSP to handle the users key management because this represents a “backdoor” in the user’s data.

B. Cloud computing incident reporting

Cloud computing incident reporting represents a way to monitor the data and its location across datacenters and present the report in a human readable form. Since in the following years almost 80% of organizations will store their data on various CSPs, they will be dependent of this technology. We can say for sure that large cloud providers will be serving tens of millions of end-users and it is also fair to say that a part of the public cloud computing services can be included as a part of the critical information infrastructure.

Our paper presents a solution to the problems mentioned in the previous sections. Since we gather data starting from the hypervisors, we can know for sure how and where the data is stored and processed.

IV. CLOUD COMPUTING AND FORENSICS CHALLENGES

A. Cloud computing
them into a service utility. Computing as a public utility is a
wish that dates from the beginning of the computing history.
But only the advances made in the last ten years, that made
possible the development of a set of new technologies, together
with the need for more efficient and affordable computing, has
enabled an on-demand system creation.

The NIST definition [16] classifies cloud computing into the
tree service models (SaaS, IaaS, and PaaS) and four cloud
types (public, private, community, and hybrid). Also assigns
five essential characteristics that cloud computing systems must
offer, such as on-demand self-service, broad network
access, resource pooling, rapid elasticity and measured service.
While these five core features of cloud computing are on
almost anybody’s list, we should also keep in mind additional
advantages, like ease of utilization, lowering costs, quality of
service, reliability, outsourced IT management and simplified
maintenance and upgrade.

If we have to choose a single area of concern in cloud
computing, that area would be privacy and security. When our
data is stored and processed on computing systems that are not
our direct control, we open the possibility to data interception
data alterations.

B. Classic digital forensics

Digital forensics, also called computer forensics or cy-
berforensics, represents the application of computer analysis
and investigation techniques to gather evidence suitable for
presentation in a court of law. The goal of computer forensics
is to perform an organized investigation, while keeping the
chain of evidence thoroughly documented in order to find out
exactly what happened on a computer.

Forensic investigators must follow a standard set of proce-
dures: after physically isolating the targeted computer, so that
its data cannot be accidentally altered, they make a digital copy
of the hard drive. Once the drive has been copied, it is put
in a secure storage facility to maintain it in proper conditions.
All following investigation is done on the digital copy.

Digital forensics can be found in various forms, depending
on the targeted problem. The most important ones, that are
also applicable and relevant to our paper are: computer
crimes, including downloading and transmitting illegal digital
files, recovery of destroyed or altered files, hacking and
cell phone data such as contacts and personal mail.

Digital forensics is comparable to a crime scene, because
law enforcements must use proper investigation techniques
to gather evidence. According to NIST [13], the electronic
forensics must have a well defined set of procedural models.
We will present briefly some of these models, that apply to
classic digital forensics:

- **Securing and evaluating the scene.** Steps should be
taken to ensure the safety of individuals and to identify
and protect the integrity of potential evidence

- **Documenting the scene.** Create a permanent record of
the scene, accurately recording both digital-related and
conventional evidence.

- **Evidence collection.** Collect traditional and digital evi-
dence in a manner that preserves their evidential value

- **Packaging, transportation and storage.** Take adequate
precautions when packaging, transporting, and storing
evidence, maintaining chain of custody

But as we naturally progress and move more and more data
to the cloud environments, the classic digital forensics model
reaches its limits because it did not evolve at the same speed
as the rest of the environment and all the methods applicable
to a single computer, became useless in the context of cloud
computing.

C. Cloud computing and forensics

As a result, the need for cloud forensics models and
procedures exists. But why is this really necessary? In order
to respond to this question we must first fully understand the
context in which the entire cloud paradigm exists.

Currently, a digital forensics investigator conducting an
analysis in a cloud environment faces a series of obstacles,
such as laws, standard operating procedures, the involvement
of a third party CSP. All these because classic and traditional
rules do not apply in the cloud.

Multiple consulting companies like Gartner Consulting
[14], warn that “Investigating inappropriate or illegal activity
may be impossible in cloud computing. Cloud services are
especially difficult to investigate, because logging and data
for multiple customers may be co-located and may also be
spread across an ever-changing set of hosts and data centers.
If you cannot get a contractual commitment to support specific
forms of investigation along with evidence that the vendor has
already successfully supported such activities - then your only
safe assumption is that investigation and discovery requests
will be impossible”. While this is not good news, there are
options available to the investigators, but not as many as there
are for traditional forensic investigations.

The user must also be prepared for incident response. The
nature of cloud computing creates difficulties in figuring who
to alert in case of intrusion detection, security breach, data
leakage or any other form of attack and data theft. For
Infrastructure-as-a-Service environments, the biggest problem
is that it does not handle custom application security and data
integrity; even more, in case of Software-as-a-Service, the user
does not know any implementation detail about the software
he uses.

Although there are currently some drawbacks, cloud com-
cputing can be interesting in the area of conducting a forensic
investigation from a cloud environment. In papers such as
[1], [4] and [9] we are presented a brief introduction
into it, but without any practical implementations. In the
future, companies can offer this as a Forensic-Investigation-
Environment-as-a-Service with on-demand resource allocation
needed for such problems.

Even though cloud computing can offer some interesting
opportunities for forensic investigation, this technology will
have to be thoroughly tested at every step of the implementa-
tion. Reliability and how to keep evidence intact will need to
be examined and implemented with great care so they can be
eligible for accepting in a court of law.

Besides the properties presented in the previous sections
we must introduce a new concept that stands at the ground of
cloud computing: cloud native applications [10]. A cloud native application is designed and created to make use of specific engineering practices that have proven successful in some of the world’s largest and most successful software applications. Many of these practices are unconventional, yet the need for unprecedented scalability and efficiency drove to the adoption in the environments that truly needed them.

V. CLOUD COMPUTING FORENSIC ENABLED FRAMEWORK ARCHITECTURE

A. General architecture

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

![Figure 1. Basic cloud computing architecture](image)

In the virtualization layer we find the actual workstations that host the virtual machines and have virtualization enabled hardware. In the management layer we find the modules responsible with enabling all the operations specific to the cloud, as presented in the previous sections. These modules are:

- **Security.** This module is responsible with all security concerns related to the cloud system. For simplicity we can consider it as an intrusion detection and alarming module.

- **Validation engine.** This module receives requests to add new jobs to be processed. Every new request is checked for consistency and it is validated and if it is legit, the new lease is transformed in a job for our system and it is properly inserted in the job queue.

- **Virtual jobs.** This module creates an abstraction between the data requested by the user and the payload that must be delivered to the cloud system.

- **Scheduler.** This is one of the most important modules in a cloud framework. Its main purpose is to efficiently schedule the jobs to the virtualization layer. It also must communicate with the other modules in order to find new instances, new services, virtual machine managers, load balancers in the system.

- **Hypervisor interface.** This module acts like a translation layer that is specific to a virtualization software vendor. It must implement each vendor API specifications.

- **Load distribution.** This module is responsible with horizontal and vertical scaling of the requests received from the scheduler. It must run a distinct application framework in order to decouple the code from the existing underneath runtime. The algorithm must be applied automatically and in the process of this analysis, the number of workstations must be taken in account.

- **Internal cloud API.** This module is intended as a link between the virtualization layer and the cloud system. In order to be more scalable and also maintain a high degree of abstraction, a common interface must be provided and every implementation of the specific API must implement this.

- **External cloud API.** This module offers a way to the user to interact with the system. It must provide means to add new jobs in the cloud system. The requests are registered and sent to the validation engine module. This API must be flexible enough to permit adding details to the jobs, like the hardware specifications of the virtual machine, operating system to be used, packages to be installed.

Now that the notion of a cloud computing framework was presented, we 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.

![Figure 2. Forensic enabled cloud computing architecture](image)

More exactly, we see a new module, the **Cloud Forensic Module.** Its main goal is to gather all forensic and log data from the virtual machines that are running inside the virtualization layer. Furthermore, we must attribute to the security module greater responsibilities and permit it communicate with all the other modules in the management layer.

Of course, in order to gather data reliably from the virtual machines we must interact with the hypervisors existing in the workstations kernel. In our paper we present only what modifications must be made to a Linux kernel. We have chosen this alternative because in a Linux kernel we can find at least two distinct, free and open source virtualization techniques: KVM and XEN. An image of an forensic enabled kernel can be seen in Figure 3.
On our research we will focus more on the KVM technology. KVM (Kernel-based Virtual Machine) is a full virtualization solution for Linux on x86 hardware containing virtualization extensions on Intel or AMD processors. It consists of a loadable kernel module called kvm.ko, that provides the core virtualization infrastructure and a processor specific module called kvm-intel.ko or kvm-amd.ko. Using KVM, an user can run multiple virtual machines running unmodified Linux or Windows images. Each virtual machine has private virtualized hardware such as a network card, disk and a graphic adapter.

Our cloud forensic interface will be implemented as a series of stand-alone kernel modules and user-space applications that can be activated or disabled at runtime. Our goal is to provide the users a way to manage it from the kernel building menu. We need this segregation because we want to have access to all the modules from the kernel that help in the entire process of virtualization. Parts like system calls, process management, virtual file system, memory management, networking management are extremely important to our research because they represent the basic building blocks between a virtual machine running on a host and the operating system. We will detail furthermore the main interest directions for our research.

The first step toward a full kernel integration is to have a proper API both to the kernel and to the external system. We are interested mostly about KVM internal API. This API is a set of ioctl instructions that are issued in order to control different aspects of a running virtual machine. In computing, ioctl (short for “input-output control”) represents a system call made to a specific device which cannot be done using regular system calls. Logically, this API is split across three different main parts: main system ioctls, meant to set proper KVM internal variables and used when creating a new virtual machine, virtual machine ioctls, meant to set different attributes of a virtual machine, like the memory size or layout and virtual CPU ioctls, meant to be used to control the operation of a virtual machine virtual CPU.

In order to be fully compatible with the Linux API, the entire KVM API is centered around the concept of file descriptors. This means that once activated, KVM creates a new device called “/dev/kvm”. On initial open of the device we get a handle of the internal KVM system that we can use to issue proper ioctl commands. For example, sending a KVM_CREATE_VM command to the kernel, we get a response containing a virtual machine file descriptor that we can further use to set different values.

It is easy to see that we can get all details concerning about the status of the “virtual hardware” that is used for a certain virtual machine. This is useful because, for example, we can get details such as:

- the memory pages that are dirtied since a last call, using the KVM_GET_DIRTY_LOG ioctl and the following data structure:

  ```c
  struct kvm_dirty_log {
    {...}
    union {
      __u64 __dirty_bitmap;
      __u64 padding;
    };
  }
  ```

- getting processor registry values, using the KVM_GET_REGS ioctl and the following data structure:

  ```c
  struct kvm_regs {
    {...}
    __u64 rsi, rdi, rsp, rbp;
    __u64 r8, r9, r10, r11;
  }
  ```

- setting processor registry values, using the KVM_SET_REGS ioctl;

- translation of a memory virtual address according to virtual CPU own translation mode, using the KVM_TRANSLATE ioctl and the following data structure:

  ```c
  struct kvm_translation {
    __u64 linear_address;
    __u64 physical_address;
  }
  ```

From the point of view of live cloud computing forensics, a great impact is given by the Memory Management Unit (MMU). For a virtualization software it is very important to have a proper MMU module. In our case, KVM uses its own MMU modules with the purpose of translation from guest physical address to host physical address. This gives us a real advantage because interacting with this modules gives us a full map of what is going on inside a virtual machine memory space.

In order to be considered reliable, a virtual machine MMU must respect a set of particular requirements, like correctness (the virtual machine must not be able to determine that it is running using an emulated MMU) and security (the virtual machine must not allocate memory beyond the limits imposed by the MMU). These requirement are going to be monitored by our forensic module and at any time we will have a full memory footprint and the whole previous states.

It is important for a forensic investigator to have access to the network communication devices and to the storage devices. In case of virtual machines running under KVM it is done using the virtio interface. Virtio is a virtualization standard for network and disk device drivers where just the guest’s device driver is aware of the fact that it is running in a virtual environment, and cooperates with the hypervisor and this enables guests to achieve high performance network and disk operations. Its design allows the hypervisor to export a common set of emulated devices and make them available through a common API. Using this interface we gain full
access to everything related to the disk and/or network devices than an investigator can use.

B. Performance

During the development of our forensic module, besides the problem of intercepting all virtual machine activity, a great attention must be payed to its performance. 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.

There is also the problem of network transmission overhead. 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.

In order to properly test and compare the results a test suite must be organized and implemented. As stated in the previous paragraphs, in this paper we present in detail our top view architecture; taking in consideration the entire complexity of the framework, the actual results are not yet ready for publishing.

C. Logging metalanguage

During our research we have focused on choosing an intermediate representation of data that is sent between the local and central forensic modules. We have analyzed different existing metalanguages for logging.

The first one is the “Management metalanguage” [15] 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. This is due to the fact that this metalanguage is designed to be used only locally over a system.

On the other side, the CEE (Common Event Expression) organization [12] 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.

Using the information gathered at this step and taking in consideration that our project is novel in this field, we will provide our own logging metalanguage, based on the CEE specifications for compatibility. Also we will keep in mind suggestion made by previous research in this direction, such as the one by Sang et al [6], for making digital forensic logging easier. Due to advantages it offers, we are going to use the JSON markup language as data envelope because it is a simple, clean, concise and human-readable format. It is also good for decoupling our cloud forensic modules because we can implement each module using its own programming language and having only a common JSON interface for using it. Along with this format we intend to use it along with a storage module that is fit for our needs. We have chosen this approach in order to have the collected data from a host in a single database and only provide management messages to the above central forensic modules.

VI. CONCLUSION

As we have seen, the field of cloud computing forensics and incident response is a new field for research that attracts more and more scientists. It poses a lot of challenges due to the distributed nature of the cloud but steps are starting to be made in this direction. In our paper we have presented a novel and new way in which user actions can be monitored and reproduced inside a cloud environment, even if it spreads over multiple datacenters.

REFERENCES