Network Forensics for Cloud Computing

Tobias Gebhardt, Hans P. Reiser

Network Forensics for Cloud Computing

Tobias Gebhardt[1] and Hans P. Reiser[2]

[1] TÜV SÜD AG Embedded Systems, Barthstr. 16, 80339 Munich, Germany
tobias.gebhardt@tuev-sued.de
[2] University of Passau, Innstr. 43, 94032 Passau, Germany
hans.reiser@uni-passau.de

Abstract. Computer forensics involves the collection, analysis, and reporting of information about security incidents and computer-based criminal activity. Cloud computing causes new challenges for the forensics process. This paper addresses three challenges for network forensics in an Infrastructure-as-a-Service (IaaS) environment: First, network forensics needs a mechanism for analysing network traffic remotely in the cloud. This task is complicated by dynamic migration of virtual machines. Second, forensics needs to be targeted at the virtual resources of a specific cloud user. In a multi-tenancy environment, in which multiple cloud clients share physical resources, forensics must not infringe the privacy and security of other users. Third, forensic data should be processed directly in the cloud to avoid a costly transfer of huge amounts of data to external investigators. This paper presents a generic model for network forensics in the cloud and defines an architecture that addresses above challenges. We validate this architecture with a prototype implementation based on the OpenNebula platform and the Xplico analysis tool.

Keywords: Cloud Computing, Network Forensics, Incident Investigation

1 Motivation

Cloud computing has become highly popular over the past decade. Many organizations nowadays use virtual resources offered by external cloud providers as part of their IT infrastructure. As a second significant trend, IT-based services are getting more and more into the focus of criminal activities [11, 24]. As a result, technology for computer forensics has become increasingly important. Cloud computing imposes new challenges for this technology. This paper addresses some of these challenges that concern network forensics in an Infrastructure-as-a-Service (IaaS) model.

Computer forensics is the science of collecting evidence on computer systems regarding incidents such as malicious criminal activities. Noblett et al. define it as "the science of acquiring, preserving, retrieving, and presenting data that has been processed electronically and stored on computer media" [16]. Traditionally, forensics has been associated with the collection of evidence for supporting or

refuting a hypothesis before the court. In this paper, we use the term in its broader sense, which includes collecting information for purposes such as internal incident investigation and intrusion analysis.

Network forensics is the investigation of network traffic of a live system. This means that for network forensics it is necessary to capture and analyse the network traffic of the system under investigation. If an organization runs a service on its own local IT infrastructure, the responsible administrator (or any investigator with physical access to the infrastructure) can easily apply local forensics measures. Today, a great variety of forensics frameworks are available for this purpose [9].

With cloud computing, a paradigm shift takes place. Virtualization and multi-tenancy create novel challenges. If a cloud user wants to investigate an incident on a virtual resource, it is usually not possible for him to use traditional forensics tools that require direct access to networks and physical machines [13]. Even if such tools can be used at the physical facilities of the cloud provider, this easily creates privacy and security issues: Investigations typically target a specific system, which might be running in a virtual machine on a physical host shared with other completely unrelated systems. These other systems should not be affected by the investigation. As an additional complication, due to dynamic elasticity and migration of virtual resources, the geographical location of systems under investigation is no longer constant [15]. These issues create a new research field called cloud forensics [3, 4].

The aim of this work is to propose a solution for some of these challenges. Its focus is on network forensics for the Infrastructure-as-a-Service (IaaS) model. Specifically, this paper makes the following contributions:

- It defines a generic model for network forensics in IaaS.
- It defines an architecture for "Forensics-as-a-Service" in a cloud management infrastructure. This architecture offers an API that authorized subjects can use to remotely control the forensics process at the cloud provider. Both data acquisition and data analysis can be handled directly at the cloud provider.
- It describes and evaluates a prototype implementation of this architecture for the OpenNebula cloud management infrastructure. The prototype includes a daemon running on all cloud hosts for collecting network traffic, filtered for a specific system under observation, and the integration of an existing network forensics analysis tool as a cloud-based service.

The paper is structured as follows. The next section discusses related work on computer forensics. Section 3 describes the system model and security assumptions in our approach. Section 4 presents our generic forensics model. Section 5 focuses on the forensics architecture and describes details of the prototype for OpenNebula. Section 6 evaluates this prototype, and finally Section 7 presents our conclusions.

2 Related Work

Computer forensics is a field that has undergone thorough investigation in various directions over the past decades. Within the scope of this paper, we discuss related work in the areas of *network forensics* and *cloud computing forensics*.

2.1 Network Forensics

The term *network forensics* was coined by Ranum [18]. In his paper, the author describes a forensics system called "Network Flight Recorder". This system offers functionality similar to an Intrusion Detection System (IDS), but it makes it possible to analyse data from the past.

Some existing approaches cover only parts of the forensics process. For example, practical tools such as WireShark[3] and tcpdump[4] support only the acquisition of data, without providing mechanisms for further analysis and reporting. These tools assume that you can run them on the host under investigation and have no dedicated support for remote forensics in the cloud.

Similarly, intrusion detection systems focus on the detection and reporting of attacks as they happen, without providing specific evidence gathering functionality [21]. In the context of network forensics, intrusion detection systems can be used as a trigger point for forensic investigations, as they create events upon the detection of suspicious behaviour.

Several publications in the area of network forensics target the question of how to manage and store forensic data efficiently. For example, Pilli et al. [17] focus on reducing the file size of captured data. They consider only TCP/IP headers and additionally reduce file size with a filter. Other publications focus on the analysis step of the forensics process. For example, Haggerty et al. [12] describe a service that helps to identify malicious digital pictures with a file fingerprinting service. Such research is orthogonal to the contribution of our paper.

Some existing approaches address the full forensics process. For example, Almulhem et al. [1] describe the architecture of a network forensics system that captures, records and analyses network packets. The system combines network-based modules for identifying and marking suspect packets, host-based capturing modules installed on the hosts under observation, and a network-based logging module for archiving data. A disadvantage of this approach is that the host-based capturing module cannot be trusted after an attack has compromised the host. Shanmugasundaram et al. [23] have proposed a similar approach that suffers from the same disadvantage. Wang et al. [25] propose an approach that adds a capturing tool on a host at the moment it should be inspected. Again, this approach suffers from the same integrity weakness, as data is captured by tools running on a possibly compromised system.

[3] http://www.wireshark.org
[4] http://www.tcpdump.org

All of these approaches imply the assumption that there is a single entity that has permission to perform forensic investigations over all data. While this is not a problem if all data is owned by a single entity, this model is not appropriate for a multi-tenancy cloud architecture.

2.2 Cloud Forensics

Cloud forensics is defined as a junction of the research areas of cloud computing and digital forensics [19]. In the recent years, cloud forensics has been identified as an area that is still faced with important open research problems [2, 5].

Zafarullah et al. [26] proposed an approach to analyse log files in an IaaS environment. They use a centralized approach in which the cloud service provider is responsible for forensic investigations. In contrast, in our approach we want to offer forensics services to cloud users to investigate problems and incidents in their own virtual resources.

The works of Birk et al. [4] and Grobauer et al. [10] share same aspects of our approach, as they propose a cloud API for forensics data. However, both publications list this idea together with other high level approaches, without presenting many details. Our forensics model is a more specific approach that also discusses aspects of a real prototype implementation.

3 System Model and Security Assumptions

This paper considers an IaaS model in which a cloud provider executes virtual machines on behalf of a cloud client. The client has full control over the software running within the virtual machines. The cloud provider manages the physical machines, and the client has no direct access to them. Multiple clients can share a single physical machine.

Client virtual machines can be compromised by malicious attacks. This work proposes an architecture in which cloud clients (or authorized third parties) can autonomously perform forensic investigations targeted at cloud-based virtual machines, based on support provided by the cloud provider. For this purpose, we assume that only the cloud provider and the cloud infrastructure are trusted, whereas client virtual machines are untrusted.

Having untrusted client virtual machines means that we make no assumptions on their behaviour. An attacker that compromises a virtual machine can fully modify the virtual machine's behaviour. We therefore do not want to collect forensic data with the help of processes running within the client virtual machine, as an attacker can manipulate these. We make no assumptions on how the attacker gains access to the client virtual machine. For example, the attacker might use access credentials he obtained from the cloud user by some means, or he might be able to completely take over the virtual machine by exploiting some vulnerability of the software running within the virtual machine.

Even if the adversary has full control over an arbitrary number of client virtual machines, we assume that the virtualization infrastructure guarantees

isolation of virtual machines. This means that the attacker has no access to the virtual machine monitor (VMM) itself or to other VMs running on the same physical host.

We assume that the cloud provider and the cloud infrastructure can justifiably be trusted. This means that they always behave according to their specification. Under this model, the possibility that an attacker compromises the cloud infrastructure (i.e., the VMM and the cloud management system) is not considered. This might be regarded as a strong assumption, but it is the usual assumption that is made by practically all cloud users.

In addition, several research projects have shown that the trust into the cloud infrastructure can be further justified by technical mechanisms. For example, Garfinkel et al. [8] introduce a trusted virtual machine monitor (TVMM). Santos et al. [20] address the problem of creating a trustworthy cloud infrastructure by giving the cloud user an opportunity to verify the confidentiality and integrity of his own data and VMs. Doelitzscher et al. [7] designed a security audit system for IaaS. This means that trust into a cloud infrastructure can be enforced not only by contracts (SLAs), but also by technical mechanisms. The exact details of such mechanisms are beyond the scope of this paper.

4 Network Forensics Architecture for the Cloud

In this chapter we define our model for network forensics in IaaS environments and describe a generic network forensics architecture.

4.1 Forensics Process Model

The model for our forensics process is shown in Fig. 1. Five horizontal layers interact with a management component, which is needed as central point of control. The layers of the model are adopted from the process flow of the NIST forensics model [14]. The layers represent independent tasks regarding the investigation of an incident. The tasks are executed in a distributed multi-tenant environment, as described before in Section 3.

The first layer is the *Data Collection* layer. All network data can be captured at this point. The management component interactions with the data collection layer for starting and stopping the capture of network traffic. The data collection also needs to be coordinated with migration of virtual machines in the IaaS environment. For this purpose, the management component coordinates a continuous capture of network traffic for a migrating virtual machine.

On top of the data collection layer resides the *Separation* layer. The task of this layer is to separate data by cloud users. At the output of the separation layer, each data set must contain data of only a single cloud client. Optionally, the separation layer can additionally provide client-specific compression and filtering of network traffic to reduce the size of the collected forensics data.

The third layer is called *Aggregation* layer. It combines data from multiple sources belonging to the same cloud client. Data is collected at multiple physical

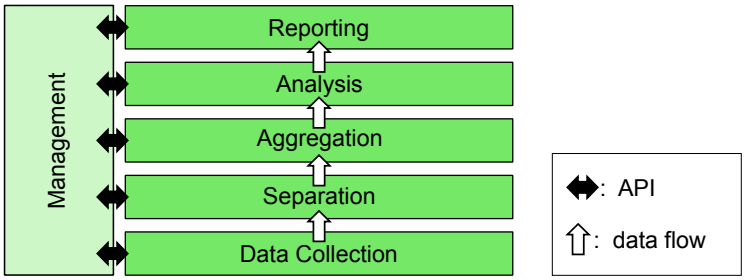

Fig. 1. The forensics process for the cloud is modelled by five basic layers, controlled by a central management block.

locations if a cloud client uses multiple virtual machines (e.g., for replication or load balancing), or if a virtual machine migrates between multiple locations. All network data is combined into a single data set at this layer.

The next layer is the *Analysis* layer. This layer receives pre-processed data sets from the aggregation layer and starts the analysis starts of the investigation. The management layer configures the transmission of collected data from aggregation to analysis. Typically, the analysis is run as a service within the same cloud.

The top layer is the *Reporting* layer in which the analysis results and consequences are presented.

4.2 Network Forensics Architecture

Our network forensics architecture directly translates the conceptional blocks of the model into separate services.

- The *management* layer is realized as part of the cloud management infrastructure. This infrastructure typically offers a central point of access for cloud clients. The infrastructure is augmented with an interface for configuring and controlling the forensics process. This component also handles authorization of forensics requests. A cloud client can control forensic mechanisms targeted at his own virtual machines, and this control privilege can also be delegated to a third party.
- The *data collection* layer is realized in the architecture by a process that executes on the local virtual machine monitor of each physical host. In a typical virtualization infrastructure, all network traffic is accessible at the VMM level. For example, if using the Xen hypervisor, all network traffic is typically handled by the Dom0 system, and thus all traffic can be captured at this place. The management layer needs to determine (and dynamically update upon reconfigurations) the physical hosts on which virtual machines of a specific client are running, and start/stop the data collection on the corresponding hosts.

- The *separation* layer is responsible for filtering data per cloud user. It is possible to investigate multiple clients on the same physical hosts, but all investigations must be kept independent. This layer separates network traffic into data sets each belonging to a single cloud client, and drops all traffic not pertaining to a client under investigation. To monitor the network data of a specific cloud client, a means of identification of the corresponding traffic is required. In our architecture, we use a lookup table from virtual machine ID to MAC address and use this address for filtering.
- The *aggregation* layer is realized by a simple component that can collect and combine data from the separation layer at multiple locations.
- For *analysis* layer, our architecture potentially supports multiple possibilities. A simple approach is to transfer the output of the aggregation layer to the investigator, who then can locally use any existing analysis tools. In practice, the disadvantage of such approach is the high transfer cost (in terms of time and money). A better option is to run the analysis within the cloud. For this purpose, the cloud user can deploy the analysis software as a service within the cloud.
- The task of the *reporting* layer is to produce reports on the analysis results that are suitable for further distribution. This step is identical to other forensics frameworks.

5 Prototype Implementation

We have implemented a prototype according to the architecture described in the previous section. As a basis for this implementation, we have selected and extended existing projects in order to create a prototype system that is usable for network forensics in IaaS clouds. The basic idea of the prototype is the following: A cloud user has an account at a cloud provider to manage VMs. This user should be able to start and to stop the forensics process for his VMs and access analysis results using a web-based system inside the cloud.

The user's central point of interaction is the management software for IaaS. The user is able to work with the VMs he owns, to monitor or to change their status and parameters. We extend the management software by adding API commands for controlling network forensics actions. Two established systems for cloud management are Eucalyptus[5] and OpenNebula[6] [22]. Both are open source systems and have an active community. We have chosen OpenNebula for our prototype, as it supports more different VMMs and we want our prototype to be usable independent of specific VMM products.

Our goal is not to develop new analysis tools, but instead make existing tools and approaches applicable to cloud-based systems. Because of this, we did not want to implement a custom integration of the analysis part into OpenNebula. Instead, the idea is to create an interface for existing analysis software and run it as a service in the cloud. With this approach it is easy to replace it by another

[5] http://www.eucalyptus.com
[6] http://www.opennebula.org

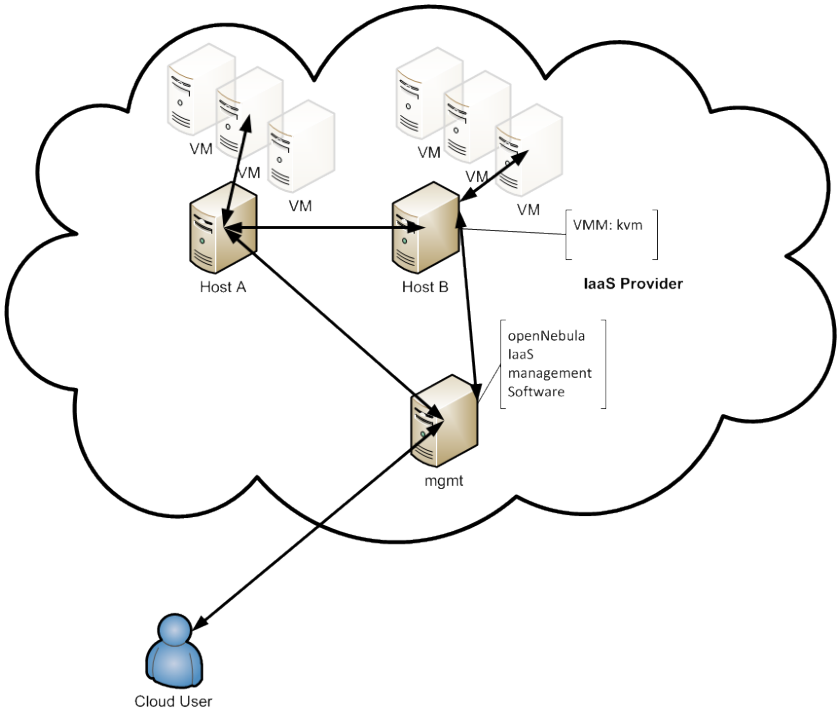

Fig. 2. In an OpenNebula-based IaaS environment, the user interacts with a central OpenNebula management instance, which in turn controls the execution of user VMs on various physical hosts.

software if requirements changed for the analysis steps. PyFlag[7] and Xplico[8] are both network forensics analysis frameworks that could be used for analysis within our work. Cohen et al. have presented PyFlag in 2008 [6], but apparently it is not being maintained any more. Xplico has a more active community and thanks to the helpful author Gianluca Costa, a developer version of Xplico is available.

In the prototype, we have implemented the *Management* layer of our architecture as an extension to the OpenNebula API. The *Aggregation* and *Analysis* layers have been implemented by a cloud-based service running Xplico, and currently the *Reporting* functionality is limited to the visualization of the Xplico output. The *Data Collection* and *Separation* layers have been realized by a custom implementation called *nfdaemon*.

The original cloud environment as provided by OpenNebula is shown in Fig. 2. Our modified environment with additional forensics components is shown in Fig. 3. The cloud user is interacting with OpenNebula, e.g. through the Open-

[7] http://code.google.com/p/pyflag/
[8] http://www.xplico.org

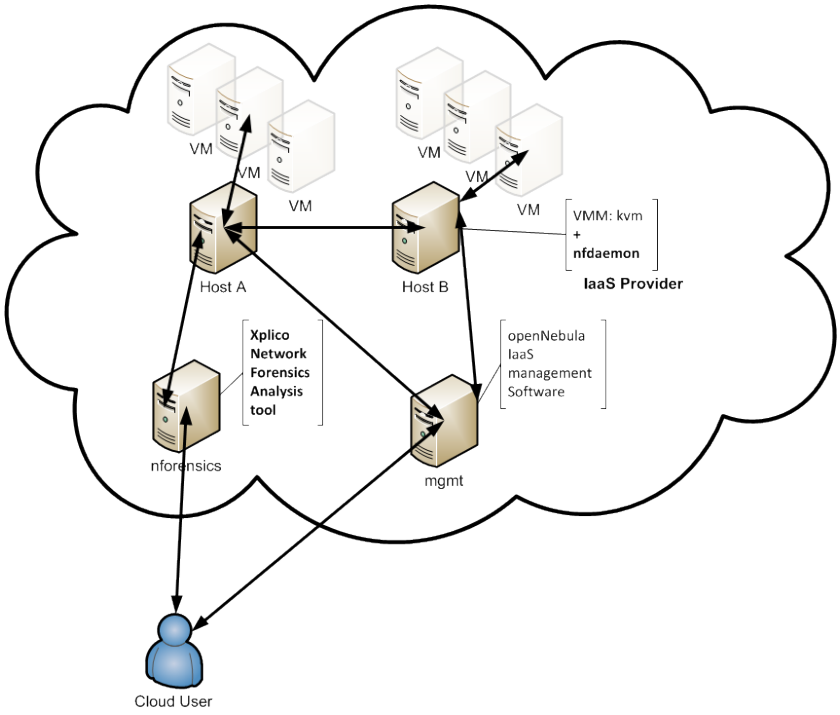

Fig. 3. The user controls forensics processes via our extended OpenNebula management component. The analysis software (Xplico) runs as a service within the cloud and offers the user a direct interface for accessing the analysis results.

Nebula CLI. Actions to start, stop or restart virtual machines are examples for actions that are already implemented by the OpenNebula project.

Beside these actions we added the actions *startnf* and *stopnf*. The first one, *startnf*, triggers the process of starting a network forensics session. The VM-ID from OpenNebula is used as a parameter for the action to identify a particular VM. *stopnf* is doing the opposite and stops the process.

A *nfdaemon* (network forensics daemon) needs to be running an each VMM (see Fig. 3). The main tasks of *nfdaemon* are collecting data, separating data per VM, and transferring it to the aggregation and analysis system. The communication interfaces are shown in Fig. 4.

The control FIFO channel is the interface for interaction between *nfdaemon* and the OpenNebula management system. The *nfdaemon* waits for input on this channel. Each command sent by the management system triggers actions of *nfdaemon*.

For filtering network data pertaining to a specific virtual machine, a mechanism for translating the VM-ID (which is used for identifying VMs by OpenNebula) to MAC network address is needed. The information about the corresponding MAC address is obtained from OpenNebula.

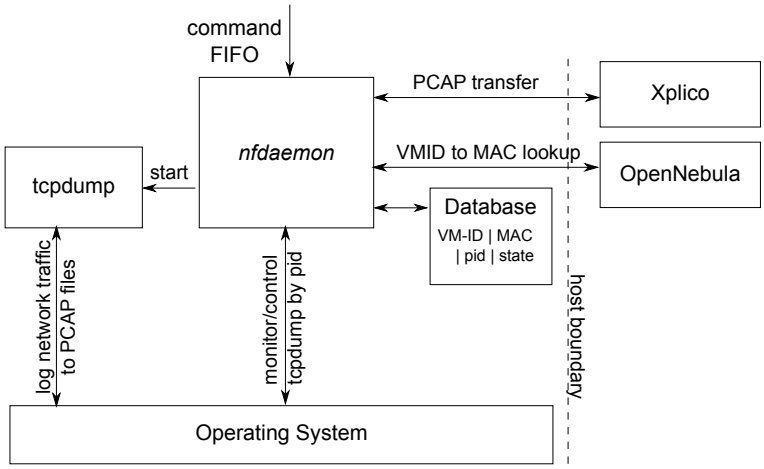

Fig. 4. The *nfdaemon* receives command via the command FIFO. It stores its state in a local database, manages tcpdump processes, and interacts with remote OpenNebula and Xplico components.

The *Data Collection* internally uses tcpdump to capture data. The *Separation* layer is realized on the tcpdump output on the basis of filtering by MAC addresses. The monitored data is periodically written into PCAP files ("Packet CAPture", a widely used format for network traffic dumps).

The *nfdaemon* periodically transfers PCAP files to the aggregation and analysis system. In our prototype, Xplico handles all analysis. This tool already contains an interface for receiving PCAP files over the network (PCAP-over-IP). This standard interface, however, is insufficient for our prototype, as it accepts connections from any source. We wanted to make sure that the only data used for analysis is traffic collected by *nfdaemon*. Therefore, we implemented a date source authentication mechanism based on TLS, using a private key stored within *nfdaemon*. PCAP files from multiple virtual machines of the same cloud users can be aggregated within the same Xplico analysis.

6 Evaluation

As a first step for evaluating our approach, we have performed basic functionality tests using a vulnerable web application running within an OpenNebula-based cloud environment. The web application was vulnerable for remote code execution due to inappropriate input validation of a CGI script. Using our architecture, we could analyse attacks targeted at this web application using Xplico in the same way as we were able to analyse the same attack on a local system.

These basic tests convinced us that we can use our approach for forensic investigations in the cloud. For a real-world deployment of our approach, however, we wanted to analyse two additional questions: What is the performance

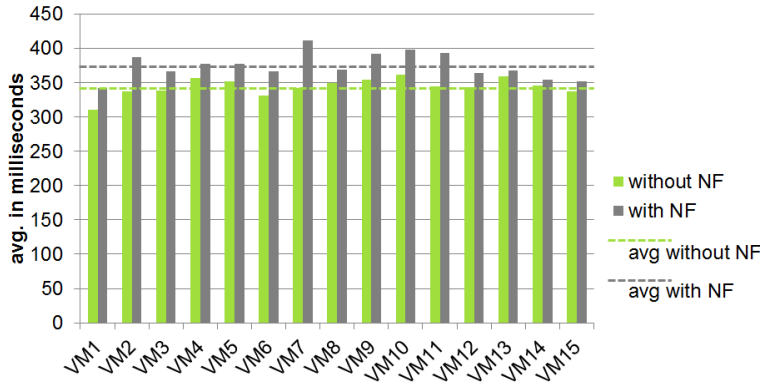

Fig. 5. Measurements show that ongoing data acquisition (NF) has an only minor impact on the running system.

impact of such analysis, and what are over-all benefits and implications of our approach?

6.1 Performance Evaluation

The purpose of the following performance evaluation is to verify whether our modified cloud system suffers from significant performance degradations if the network forensics functionality is activated. The setup for measuring the performance impact uses a running *nfdaemon* on each VMM in a cloud environment. This process consumes computation and communication resources when capturing, processing, and transferring network traffic. The following experiment intends to quantify this performance impact by comparing two configurations with and without *nfdaemon*.

The measurements have been done on hosts with 2.8 GHz Opteron 4280 CPUs and 32 GB RAM, connected via switched Gigabit Ethernet. 15 VMs are used in parallel on one VMM for each scenario. Each VM hosts a web service that executes a computationally intensive task. The web service chooses two random values, calculates the greatest common divisor of the two values is calculated, and shows the result on its output. The calculation is repeated 1000 times for each client request. Clients iteratively call functions of this web service. Client and Xplico analysis is running on a separate host.

Clients and the service calls are realized with the tool *ab*[9]. For each VM, the web service is called 2,500 times with 25 concurrent requests. The time for each request and a summary of every 2500 calls are stored in text files. This procedure is repeated 60 times every two minutes. 150,000 measurement values are collected for each VM.

[9] http://httpd.apache.org/docs/2.0/programs/ab.html

Fig. 5 shows the results of these measurements, with network forensics turned on and off. The results show a measurable but insignificant impact on the performance of the VMs. The average performance reduction between the two runs is between 2% to 17%. On average, the difference between active and inactive forensic data collection is 9%. The measurement shows that it is possible to transfer the concept of the network forensics service to a real life scenario.

6.2 Discussion of Results

The measurements show the overhead for monitoring a single physical host. For a real cloud deployment, the question of scalability to a large number of hosts arises. Furthermore, the business model for the cloud provider and benefits for the cloud client need to be discussed.

Regarding the *scalability* of our approach we note that the observed overhead does not depend on the number of physical machines of the cloud provider. If a cloud client uses a single virtual machine in a large cloud, the previously shown overhead exists only on the physical machine hosting the client VM. Our approach can, however, be used to observe multiple client VMs simultaneously. In this case, a single analysis VM can become a bottleneck. A distributed approach in which multiple VMs are used to analyse forensic data could be used to leverage this problem, but this is beyond the scope of this paper.

For cloud providers, offering *Forensics-as-a-Service* is a *business model*. On the one hand, the client uses additional virtual resources for the analysis of forensics data. This utilization can easily be measured by existing accounting mechanisms. On the other hand, the overhead caused by the *nfdaemon* cannot be accounted by existing means. Therefore, we propose to estimate this overhead by measuring the amount of forensic data transferred to the aggregation and analysis component.

For the cloud user, the benefits of our approach are two-fold: Most importantly, the user re-gains the possibility to perform network-based forensic investigations on his services, even if they are running on a remote cloud. This way, the lack of control caused by cloud computing is reduced. Second, the client needs resources for performing the analysis of forensic data. With our approach, he can dynamically allocate cloud resources for the duration of the analysis.

7 Conclusion

The growing use of cloud-based infrastructure creates new challenges for computer forensics. Efficient means for remote forensics in the cloud are needed. The location of a virtual resource is often hard to determine and may even change over time. Furthermore, forensics needs to be limited to the specific system under observation in multi-tenant environments.

In this paper, we have analysed these problems with a special focus on network forensics for the Infrastructure-as-a-Service (IaaS) model in the cloud. We have defined a generic model for network forensics in the cloud. Based on this

model, we have developed a system architecture for network forensics and developed a prototype implementation of this architecture. The implementation integrates forensics support into the OpenNebula cloud management infrastructure. Our approach solves three basic issues:

- It provides a remote network forensics mechanism to cloud clients. Network data acquisition and processing can be controlled remotely, independent of the physical location of virtual machines. If virtual machines migrate, the data acquisition transparently follows the location of the migrating VM.
- It ensures separation of users in a multi-tenant environment. The mechanism of acquiring data is limited to network traffic of the user's virtual machines, without infringing the privacy and security of others.
- It avoids the cost of transferring captured network data to external investigation tools by implementing the analysis step by a cloud-internal service under to control of the investigator.

An evaluation shows that the additionally needed computing power for running our data collection and processing service in parallel to an existing service is at an acceptable level. All in all, with the results of this work an organization will be able to use network forensics for a service hosted on an IaaS cloud infrastructure. This contribution eliminates a disadvantage that cloud-based services have compared to traditional services running locally on the organization's internal IT-infrastructure.

References

1. Almulhem, A., Traore, I.: Experience with engineering a network forensics system. In: Proc. of the 2005 Int. Conf. on Information Networking, Jeju (2005)
2. Beebe, N.: Digital forensic research: The good, the bad and the unaddressed. In Peterson, G., Shenoi, S., eds.: Advances in Digital Forensics V. Volume 306 of IFIP Advances in Information and Communication Technology. Springer Boston (2009) 17–36
3. Biggs, S.: Cloud computing: The impact on digital forensic investigations. In: Proc. of the 4th Int. Conf. for Internet Technology and Secured Transactions (ICITST). (2009)
4. Birk, D.: Technical Challenges of Forensic Investigations in Cloud Computing Environments. In: Workshop on Cryptography and Security in Clouds. (2011) 1–6
5. Catteddu, D., Hogben, G.: Cloud Computing – Benefits, risks and recommendations for information security. ENISA Technical Report (2009)
6. Cohen, M.I.: PyFlag – an advanced network forensic framework. Digit. Investig. **5** (September 2008) 112–120
7. Doelitzscher, F., Reich, C., Knahl, M., Clarke, N.: Incident Detection for Cloud Environments. In: EMERGING 2011, The Third International Conference on Emerging Network Intelligence. (2011) 100–105
8. Garfinkel, T., Pfaff, B., Chow, J., Rosenblum, M., Boneh, D.: Terra: a virtual machine-based platform for trusted computing. SIGOPS Oper. Syst. Rev. **37**(5) (October 2003) 193–206

9. Glavach, S., Zimmerman, D.: Cyber Forensics in the Cloud. IAnewsletter **14**(1) (2011) 1–36
10. Grobauer, B., Schreck, T.: Towards incident handling in the cloud. In: Proceedings of the 2010 ACM workshop on Cloud computing security workshop - CCSW '10, New York, New York, USA, ACM Press (2010) 77–85
11. Grobauer, B., Walloschek, T., Stocker, E.: Understanding Cloud Computing Vulnerabilities. IEEE Security & Privacy Magazine **9**(2) (March 2011) 50–57
12. Haggerty, J., Llewellyn-Jones, D., Taylor, M.: FORWEB: file fingerprinting for automated network forensics investigations. In: Proceedings of the First International Conference on Forensic Applications and Techniques in Telecommunications Information and Multimedia eForensics 2008. (2008)
13. Hoopes, J., Bawcom, A., Kenealy, P., Noonan, W., Schiller, C., Shore, F., Willems, C., Williams, D.: Virtualization for Security. Syngress Publishing, Burlington (2009)
14. Kent, K., Chevalier, S., Grance, T., Dang, H.: SP800-86: Guide to Integrating Forensic Techniques into Incident Response. National Institute of Standards and Technology, Gaithersburg (2006)
15. Mather, T., Kumaraswamy, S., Latif, S.: Cloud Security and Privacy – An Enterprise Perspecive on Risks and Compliance. O'Reilly Media, Sebastopol (2009)
16. Noblett, M.G., Pollitt, M.M., Presley, L.A.: Recovering and examining computer forensic evidence. Forensic Science Communications **2**(4) (2000)
17. Pilli, E.S., Joshi, R.C., Niyogi, R.: Data reduction by identification and correlation of TCP/IP attack attributes for network forensics. In: Proceedings of the International Conference & Workshop on Emerging Trends in Technology - ICWET '11, New York, New York, USA, ACM Press (2011) 276–283
18. Ranum, M.J.: Network forensics and traffic monitoring. Computer security journal (1997) 35–39
19. Ruan, K., Carthy, J., Kechadi, T., Crosbie, M.: Cloud Forensics. Advances in Digital Forensics VII **361/2011** (2011) 35–46
20. Santos, N., Gummadi, K.P., Rodrigues, R.: Towards trusted cloud computing. In: Proceedings of the 2009 conference on Hot topics in cloud computing. HotCloud'09, Berkeley, CA, USA, USENIX Association (2009)
21. Scarfone, K., Mell, P.: Guide to intrusion detection and prevention systems. NIST Special Publication 800-94 (2007)
22. Sempolinski, P., Thain, D.: A Comparison and Critique of Eucalyptus, OpenNebula and Nimbus. In: 2010 IEEE Second International Conference on Cloud Computing Technology and Science, IEEE (November 2010) 417–426
23. Shanmugasundaram, K., Memon, N., Savant, A.: ForNet: A distributed forensics network. In: Second International Workshop on Mathematical Methods, Models and Architectures for Computer Networks Security. (2003)
24. Somorovsky, J., Heiderich, M., Jensen, M.: All your clouds are belong to us: security analysis of cloud management interfaces. In: Proceedings of the ACM Cloud Computing Security Workshop (CCSW). (2011)
25. Wang, H.M., Yang, C.H.: Design and implementation of a network forensics system for Linux. In: 2010 International Computer Symposium (ICS2010), IEEE (December 2010) 390–395
26. Zafarullah, Anwar, F., Anwar, Z.: Digital forensics for Eucalyptus. In: Proceedings of the 2011 Frontiers of Information Technology. FIT '11, Washington, DC, USA, IEEE Computer Society (2011) 110–116