FraaS: A Framework for Digital Forensic Services in a Cloud-based Environment

Mohit Soni¹, and Manish Kumar Bharti²

(1) Amity School of Applied Science, Amity University Haryana
(2) Amity School of Engineering and Technology, Amity University Haryana

Abstract: Digital forensics is the application of computer science to cater to legal needs. Quality digital forensic services are often encountered by various availability issues. The invention of cloud-based services has now enabled researchers to build software or platform level based service channels. In this paper, a multi-tenant capacity framework where digital investigation can be provided as a service in the cloud has been proposed. One can use this service to perform forensic analysis on digital evidence. Services can be procured vide the forensic tools provided on a pay-per-use basis. This paper also suggests a proto-architecture of the proposed Forensics-as-a-Service framework along with its implementation module using SDNs. The architecture of FraaS is built around the NIST guidelines for the same.

Key words: FraaS (Forensic-as-a-Service), NFAT (Network Forensic Analysis Tool), Digital Forensics, Cloud Services, SaaS, PaaS, SDN (Software Defined Networks), OpenContrail, OpenStack, API (Application Program Interface)

1. Introduction

With the increased dependence on technology in the twenty first century, digital crimes have also paced up. Today almost all of the electronic devices are vulnerable in one way or another.

Digital Forensics can be classified in different segments such as computer forensics, mobile forensics, network/cyber forensics, plastic & smart card forensics, etc. The scope of this paper concentrates specifically on two segments, i.e. computer forensics and network forensics. The other segments of the digital forensics classifications can be implemented similarly. To understand the model of forensic as a service in the cloud, the recommendations of National Institute of Standards and Technology (NIST) have been considered. The digital forensics procedures involve four major processes namely Collection, Examination, Analysis and Reporting [1]. The proposed Forensics-as-a-Service (FraaS) also provides services based on these basic recommendations where specific applications are designed for collection of data from the actual evidence, the proper examination of integrity and authenticity of the data which further eases the output to analyze as per requirement. The storage database than allows the researchers and analysts to use the reporting module to
document each and every phase of the evidence analysis to keep the chain of custody maintained.

1.1 Chief Features

As per NIST, cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to shared pool of configurable computing resources like networks, servers, storage, applications and services that can be rapidly provisioned and released with minimal management effort or service providers’ interaction [2]. The FraaS framework provides the applications and tools that are forensically approved by NIST and can be used with minimum service providers’ interaction [3]. The features which FraaS proposes are given below:

- **Forensic Imaging**: It involves copying the sectors from the disk into an uncompressed, un-split, raw, header less image file. The analysis and investigation is carried out on the image copy while retaining the original evidence in its initial form. The image file and other artifacts need to be authenticated and uploaded to a cloud server for further analysis [4].

- **Hashing**: Hash functions are used for mathematical authentication of data. In digital forensics, it is computationally infeasible to find two different inputs that can yield the same hash output. The hash function ensures that if input is changed in any way an entirely different hash value is produced. The mathematical authentication of digital evidence must be done using either SHA-1 or SHA-2 [ref].

- **Keyword Search**: String searches aid in processing large amounts of data to find keywords [6]. A keyword search involves extensive disk reads and takes reasonable amount of time e.g. single keyword search on a 120 GB image takes about 85 minutes. Searches may result in thousands or even lakhs of results which take time to review such as single keyword search on a 120 GB image resulted in 216,000 hits after 2% completion.

- **Data Recovery**: The image file may need to be examined to discover what data was stored on the suspect system. This examination is time consuming and needs to be performed thoroughly. The recovered data of evidence may include:
  - Normal files,
  - Deleted files,
  - Password protected files,
  - Hidden files,
  - Data in free and unallocated space,
  - Slack Ram,
  - Slack Drive.

- **Indexing and Searching**: Data recovery operations may result in very large quantity of data for analysis. Powerful indexing and searching tools are used for effective analysis of large volumes of data. The applications further provide Fuzzy searching, Phonic searching, Synonym searching, Numeric searching and Stemming [7]. Indexes are created and updated at runtime simultaneously as data is being recovered by forensic machine. This enables searching through terabytes of data instantly.

- **Password and Encryption Cracking**: In most cases investigators are confronted with password protected or encrypted files, so the FraaS provides with a specific applications which decrypts such protected files. Tools enabling dictionary or brute force attacks are generally employed [8]. The proposed FraaS cloud ecosystem may also provide standalone password cracking tools, distributed password recovery tools or cloud based password cracking services.

- **Network Traffic Analysis Tool**: The concept of network forensics deals with the data found across a network connection. This is mostly ingress and egress traffic from one host to another. Network forensics analyzes the traffic data logged through firewalls and/or intrusion detection systems or at network devices like routers [9]. The goal is to trace back to the source of the attack so that the cybercriminals are
prosecuted. FraaS may provide applications for deep traffic analysis. This could allow administrators to monitor the networks, gather all information about anomalous traffic, assist in network crime investigation and help in generating a suitable incident response. Such NFATs may also aide in analyzing insider thefts, analyze misuse of resources, predict potential attack targets, perform risk assessment, evaluate network performance, and protect intellectual propriety.

This proposed framework of FraaS uses a generic framework for live network forensics as its skeleton for live network/cyber forensics. Zhong [10] in his work utilizes the apriori algorithm which enables an examiner the flexibility to connect to an attacked network through a virtual private network (VPN) and develop a crime signature. Similar forensic framework was also found to be proposed earlier by Gang Zeng [11]. However, this is found to contain several limitations. The work is limited to a conceptual outline of digital forensics as a service. The proposed framework in this paper, on the other hand, proposes a true frame work using OpenContrail, which is a software defined networking module. This is integrated with the OpenStack framework to achieve robust connectivity. These also enable the implementation of the ‘pay-per-use’ concept to materialize an economic roadmap for FraaS for tenants and service providers alike. As mentioned earlier, this proposal is NIST compliant as well.

Further, live network forensics is being introduced as a tool in this paper, along with additional applications to maximize the availability of network forensics as a service too. This paper also talks about ‘tenant categorization’ and their related/respective networking architectures.

2. Overview: FraaS Architecture

The proposed model for FraaS is a virtualized multi-tenancy architecture where the users share the same infrastructure (storage, networking, servers). The software defined controllers play the role of orchestrating the network and networking services like load balancing and security based on the applications to compute and store resources. The orchestrator uses the north bound interface of the SDN controller at a very high level of data abstraction to create a virtual network for a tenant within a data center. They also attach VM’s to tenants’ virtual network. To perform the live network forensics it can connect tenants’ virtual network to some victim network over the internet or VPN.

Fig. 1: Overview of FraaS

2.1 Use of OpenContrail

The architecture for the frame work of FraaS is both the combination of IaaS and SaaS in the cloud environment. This section talks about an architectural layout to accomplish the implementation of FraaS, stating the individual roles of all the modules. The complete architecture is further sub-divided based on the virtualized modules which are important to implement such a frame work along with the installation details.
To construct a system which can hold the forensic utility and functions, one must distribute the actions symmetrically. A symmetric flow of events can be achieved by a set of nodes which are inter-connected. These nodes can be implemented as virtual machines or physical servers. Figure (1) describes all the different nodes integrated together.

![Diagram of the proposed architecture](image)

**Fig. 1: The Complete Layout of the proposed Architecture**

### 2.2 Controller Node

It implements a logical connection throughout the system to maintain the continuity of the state. Controller node interacts with all the network elements present in the system to check the connectivity. The configuration node takes care of the conversion part from high level data to lower level data. It also maintains a copy of current configuration for all the virtual machines. As the name suggests the analytics node store, correlates and analyzes information of the network and system logs, events and errors.

The minimum hardware requirements for an OpenContrail system environment are five servers [ref.]. The ones to be used here are defined below:

- **Server 1**: OpenStack with Horizon and webui, config-node, api-server, collector, and analytics
- **Server 2**: Controller node
- **Server 3**: Controller node
- **Server 4**: Compute node, including vrouter, vnagent, and OpenStack Nova
- **Server 5**: Compute node, including vrouter, vnagent, and OpenStack Nova

These servers need to have a certain hardware configuration. The following are recommended specifications for the servers in the Contrail system:

**Servers 1, 2, and 3:**
- 12 GB memory
- 24 GB hard drive
- Dual-core x86_64 processor
- At least one Ethernet port

**Servers 4 and 5:**
- 64 GB memory
- 120 GB hard drive
- Quad-core x86_64 processor
- At least one Ethernet port

The controller needs to be installed and logged into next. Steps for installation of the Controller are as below:

1. Download OS image and install on PC
2. Run setup.sh
   - cd /opt/contrail/contrail_packages; ./setup.sh
3. Create testbed file
4. Install system
   - cd/opt/contrail/utils; fab install_contrail
     .....(rebooted automatically)
   - cd/opt/contrail/utils; fab setup_all
     (rebooted automatically)

**testbed file**

- cd /opt/contrail/utils/fabfile/testbeds
- cp testbed_singlebox_example.py testbed.py
- Edit vi testbed.py

```python
ext_routers = ['srx1',
    '192.168.192.79']
(if external router does not exist, comment out)
host1 = 'root@192.168.192.64'
host_build = 'root@192.168.192.64'
env.passwords = {
    host1: '<host password>',
    host_build: '<host password>',
}
```
If the installation is successful, the log-in prompt Contrail Web screen would read:

```
Host://(host ipaddress:8080/; username: admin, password:contrail123
```

The following management page can be viewed with the successful login, as displayed in Fig. 2.

![Contrail Web screen](image)

**Fig. 2: Screenshot of Login Page**

### 2.3. Compute Node

The compute node basically contains the virtual machines. The virtual machines can be those of the tenants or virtual servers. Compute node can also be used to create the service chains with virtualized services. In this framework, the base OS remains as Ubuntu 12.04 Server and KVM is the hypervisor. The vRouter forwarding plane sits in the Linux Kernel; and the vRouter Agent is the local control plane. Below is the method to configure the vRouter:

```
interfaces
ge-0/0/0
{
    unit 0 {
        family inet { address 192.168.192.79/24; } }
    ge-0/0/1 {
        unit 0 {
            family inet { address 10.0.0.1/24}]
```

### 2.4. Analytics Node

In this proposed FraaS framework, the analytics node serves the concern of storage. It is also the rules engine. It logs errors and the network events. It also communicates with other analytic nodes spread over the distributed environment to analyze metadata and perform state updates from the peer nodes. This framework recommends the use of this module as the storage node. The remaining resources can be utilized the additional services provided by the OpenContrail Analytics node. This feature will create a flawless connection for the tenants connecting to the network with real-time updates and patch. The NoSQL database is being used to store the data in Analytics node. The below given is the working structure of the Analytics’ Node.

![Analytics Node](image)

**Fig. 3: The Analytics Node**

### 2.5. Tenant Network

The tenant access level and computational usage of the provided services are subjective issues. There are three ways to configure the networking of the OpenContrail for the required framework proposed in this paper. This network configuration can be done from the OpenContrail web screen. The OpenStack also gives us an option to configure the networking using the quantum module but some features are not
available as of now until the next release occurs. The architecture for the network is given in Fig. 4.

![Fig. 4: Network Architecture](image)

In this method, the tenants are divided into three categories, namely gold, silver, and platinum. The gold members have access to the forensic tools with full functionality, whereas the silver has limited access. Platinum has access only for a limited amount of time frame which entitles them to a service-level agreement. Therefore on a virtual network design, all three servers are made independent along with their respective application retrieving servers.

Another method may be the use of OpenContrail REST API (representational state transfer).

### 2.6. Network Configuration using Open Contrail

The network configuration for the above mentioned network can be achieved by the Open Contrail’s SDN framework which provides a GUI for the configuration and management of the private and public subnets and their required persistent routes.

The GUI presented in Fig. 5 helps us to configure and to create the network names and network policies for individual networks. Subnet management and IP address pool management can also be done using the same GUI.

![Fig. 5: Screenshot: OpenContrail’s Network Configuration GUI](image)

### 2.7. Connecting the Forensic API to the Open-Contrail Architecture

The main goal of this paper is to provide an interface to the users who wish to use, limited or full functionality, of forensic tools and applications. In a multi-tenant architecture where the resources are being shared it is believed that the dynamic or polymorphic multitenant application can serve the needs of independent users. The API for the FraaS therefore, needs a multi-tenant application which can use the run-time engine that generates application components from the metadata [12].

In a metadata driven architecture, the run time kernel is separated and so is the metadata. One piece of metadata describes the core functionality of the application and another which responds to tenants' data and applications. Such boundaries help independently manage the core application functionality and the tenants' metadata. It also updates the system kernel with virtually no risk [13].

In this architecture as well, we expect a metadata driven polymorphic application to handle the distributed forensic applications and tools.

The development of web based application for FraaS must contain this independent functionality.
so that the fetching of the resources from the user interface remains specific for each call of a function.

3. Limitations of Digital Forensics as a Service

The error reporting of the digital forensics tool is one of the key limitations which remain on this model. Once the output is generated from the evidence, the verification of that output completely depends on the applications and tools employed. The Forensic design framework has very little role to play in this effort.

- In case of the Live Network forensics proposed in this model, if the tenant network is already affected by some malicious program or malware which multiplies through the network, then the virtual private network connection from the cloud to such a physical network means getting infected. Hence proper measures must be taken to isolate such applications and tools while configuring SDNs.

- As this model, is completely based on dynamic Forensic approach the official evidences or their copies must need to be uploaded to the cloud for their analysis.

- On the wire protocol; encryption, authentication and security of the data on transit (both egress and ingress) must be dealt with strict policies and strong a defense system so as to discourage intrusion.

4. Conclusion

Making forensic services available to any and all underfunded organizations/start-ups, students, researchers, law enforcement bodies, etc. was the motivation behind proposing such a framework. Needless to say, development of a FraaS is a start-up idea in itself.

By using SDN tools like OpenContrail and GUIs like OpenStack Nova, it has become evident that the proposed framework is extremely cost effective as well.

Use of APIs to enforce tenant categorization is novel and also a motivation for further research and easy services.

References


