A Survey on Providing Reliability to Forensic Data by Cloud Forensic Architecture and Linear Network Coding

Lashkare Hiral N.[1], Gayatri S Pandi(Jain)[2]

[1] *Master Of Engineering, Computer Engineering, L.J.I.E.T, Gujarat, India*
[2] *Head Of Department, Computer Engineering, L.J.I.E.T, Gujarat, India*

ABSTRACT

Cloud computing is rapidly being adopted by business and IT organizations since it offers a high degree of scalability, convenient pay-as-you-go services, Location independency, Device Independency and low cost computing. As cloud use increases the crime conducted using cloud also increased there is always need of forensic architecture. In this paper survey on forensic architecture and survey on reliability for forensic data. In first architecture present modifications needed to be made in order to create a digital forensic framework, In Second architecture provide reliable forensics in current cloud infrastructures by the open cloud forensics(OCF) model and also survey technique to provide reliability to forensic data using Linear Network coding(LNC).

Keyword: - *Cloud Forensic, Reliability, Linear Network Coding*

1. Introduction

Cloud computing is rapidly being adopted by business and IT organizations since it offers a high degree of scalability, convenient pay-as-you-go services, device independency, location independency and low cost computing. Cloud Computing can be defined as "On demand delivery of infrastructure, applications, and business processes in a security-rich, shared, scalable, and based computer environment over the Internet for a fee"

As cloud use increase crime conducted using cloud also increase so the required forensic architecture. Forensics computing is the process of identifying, preserving, collecting, analysing, and presenting evidence in a manner that is legally acceptable [6]. The uncovering and examination of evidence located on all things electronic with digital storage including computers, cell phones, and networks [6]. Any crime conducted using the cloud is considered a cloud computing crime [6]. Building a case based on evidence located in the cloud computing crime is considered as a cloud investigation [6]. Applying forensics computing process on cloud evidence is called as cloud forensics.

2. Related Work

In first paper, author Alecsandru Patrascu and Victor Valeriu Patriciu proposed detail the architecture and the modifications needed to be made in order to create a digital forensic compliant framework. The framework consists in a some of parts that are designed to work together. The application is composed from seven modules. These modules are: Frontend, User Manager, Lease Manager, Scheduler, Hypervisor Manager, Monitor and Database Layer. The actual implementation is made using a multi-tier architecture, more exactly a three-tier architecture containing the Presentation tier in Fig. 2, Application tier in Fig. 3 and Data tier in Fig. 4.

The "GUI" module is responsible for user interaction. It is presented as a web interface. From it, a user can choose a number of virtual machines - a minimum and a maximum instance count. The "Frontend" module maps over the "Intrusion detection filter" and "API" layers. The "User manager" module is responsible for storing the details for the system users, like username and password. The "Lease manager" module implements the "Job and Lease Manager" layer. It is responsible for getting the leases from the user; transform them into proper jobs and saving them both on local physical memory and a remote database. The "Scheduler" module is responsible with choosing on what physical node will a lease be run. The "Hypervisor manager" module manages virtual machine, creating, starting, stopping, pausing, resuming and cloning a new or existing virtual machine. The "Monitor" module is responsible for monitoring the entire system activity. The "Database" module is responsible with the underneath database(s) (DB). Since it is implemented separate from the entire system it can be easily adapted to work with different DB software. It is also implemented using the same plug-and-play architecture and the only thing that is needed to do from the part of the system administrator is writing the proper driver to interface with the particular Database vendor.

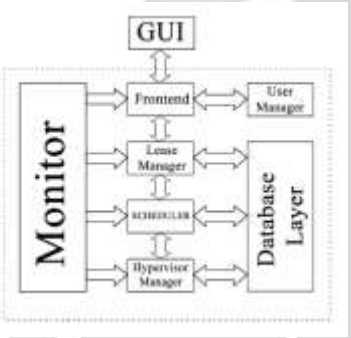

Fig -1:System Architecture[1]

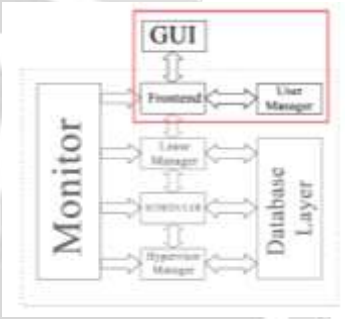

Fig -2: Presentation tier[1]

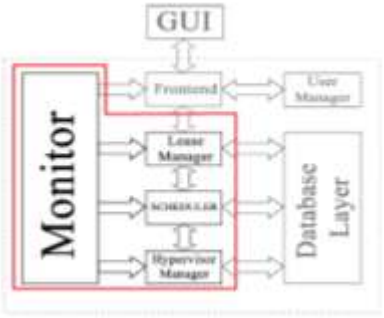

Fig -3:Application tier[1]

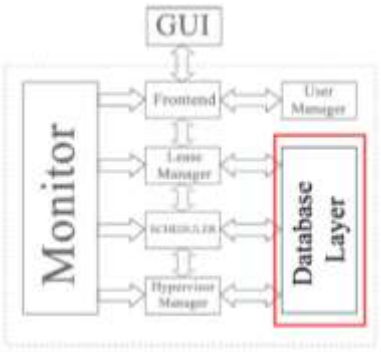

Fig -4:Data tier[1]

Datacenter Enabled Architecture is Fat Tree topology as presented in Fig. 5 composed from multiple building blocks. The basic building block is called a "cluster". A cluster is composed from multiple racks, each rack having multiple servers and one switch called top of the rack switch (ToR). This allows communications between adjacent servers to be made really fast. Using the same principle of data locality, each ToR is linked by a level 2 switch (L2S) and each L2S is linked to an aggregation switch. Each cluster is linked to an cluster router (CR) and finally, each CR to a border router.

In the first place, the modifications will start at the physical servers that are stored in each rack. For this, they need a dedicated forensics network port, just like a management port, as can be seen in Fig. 6. Also, this port must have a

correspondent in the ToR switch. This port will be used by our Cloud infrastructure for collecting and processing data and also it will be used by the authenticated forensic investigators.

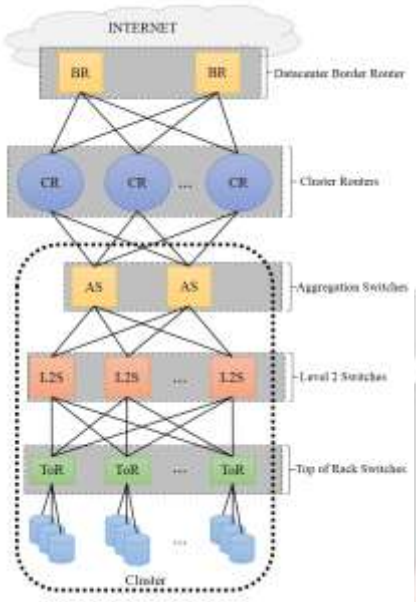

Fig -5:Datacenter topology[1]

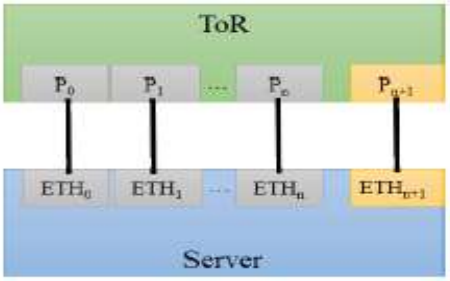

Fig -6:Dedicated forensics port on a servers[1]

In second paper, author Shams Zawoad, Ragib Hasan, and Anthony Skjellum proposed OCF model; they design a cloud computing system, which is illustrated in Fig 7. They introduce the features to support the OCF model. First, to prevent the loss of volatile electronically stored information (ESI), they propose a continuous synchronization feature, which will store sufficient volatile ESI efficiently in a persistent storage without hampering the CSP's business model. Second, to translate the ESI to verifiable ESI, we propose a proof publisher module that will create cryptographic proof of all the ESI and publish to the Internet, so that neither a dishonest cloud provider nor an investigator can alter evidence after-the-fact. Third, all the ESI will be made available to the investigators through APIs so that investigators do not need physical access to the cloud infrastructure to acquire possible evidence. Finally, the court authority can use the published proofs of the ESI to verify the integrity of the evidence.

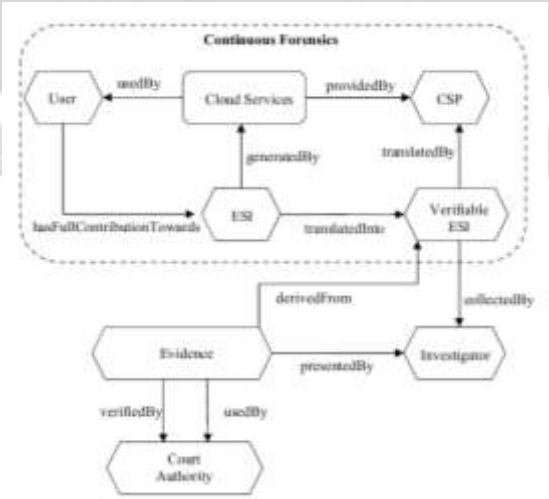

Fig -7: Open Cloud Forensics Model [2]

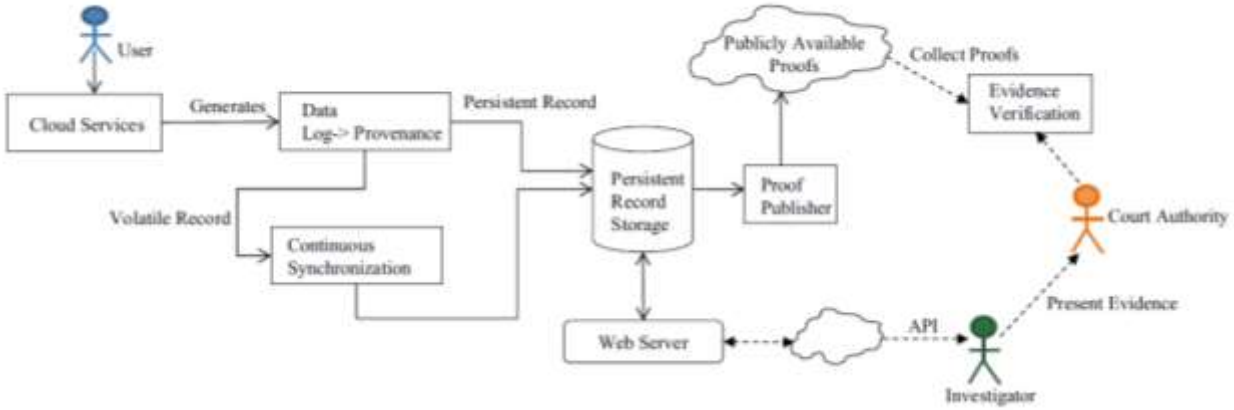

Fig -8: The OCF-Supported Cloud[2]

As above two papers give description about the cloud forensic architecture in that paper cloud forensic data is stored in the cloud data center and that data required to be reliable so the technique applied on to that data is linear network coding and survey on linear network coding is given in next papers [2].

In third paper, author Frank H.P. Fitzek, Tamas Toth, Aron Szabados, Morten V. Pedersen, Daniel E. Lucani, Marton Sipos, Hassan Charaf, Muriel Medard proposed, random linear network coding (RLNC) to generate coded data in our cloud storage systems. RLNC linearly combines uncoded packets into any number of coded packets using random coding coefficients from a finite field. The coded packet has the same size as the uncoded packets plus some additional information referred to as the encoding vector, which comprises the values of the random coefficients used to generate that particular coded packet. In contrast to any other end–to–end coding approach, RLNC allows for recoding, i.e. any intermediate node is able to code over any number of already code packets without needing to decode the data. This feature is vital for our distributed cloud approach in order to reduce the amount of data to be conveyed to a newly added cloud storage device without compromising reliability[3].

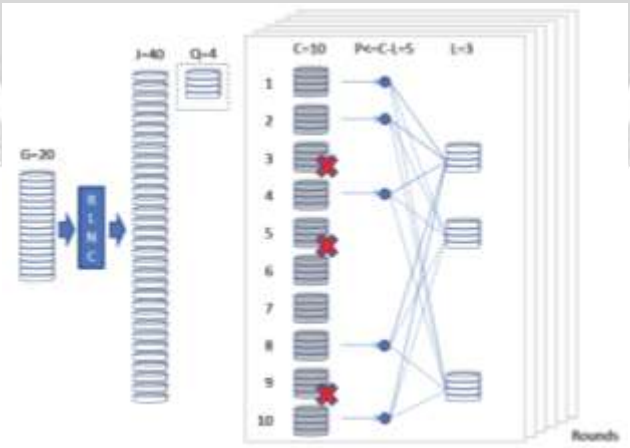

Fig -9: Model of the dynamic distributed storage system loosing L clouds and recovering by exploiting data from a subset of the remaining clouds[3]

In forth paper, author Marton Sipos, Frank H.P. Fitzek, Daniel E. Lucani, Morten V. Pedersen proposed work focuses on distributed storage solutions using RLNC. In this paper, they present a system that employs

commercially available clouds to store files reliably. Proposed system is comprised of a client application that uploads and downloads data to the storage nodes and handles all computations related to encoding, decoding and recoding.

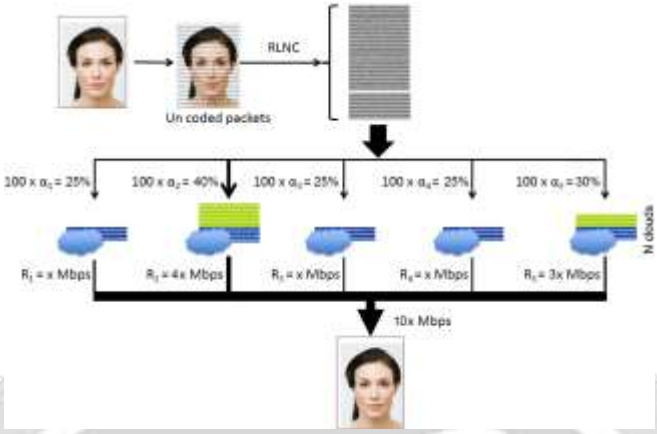

Fig -10: Main idea of distributed clouds with network coding [4]

Fig 10 shows an overview of the system, both the initial distribution of data as well as retrieval (top and bottom part, respectively). The original file is divided into a number of uncoded packets by the client, which are then linearly combined using random coefficients to generate a somewhat larger number of coded packets. The packets are then distributed by the client evenly to N (5 in the example) clouds. The extra redundant coded packets ensure that content stays available if one of the clouds becomes unreachable. We call this part of the data critical data storage and show it in dark blue on each cloud. This storage is kept unchanged unless the value of N changes in the system. During content retrieval (bottom part of the figure), all clouds send over as many packets as they can supply. Once sufficient coded packets are collected, decoding occurs on the client side where the original file is reconstructed.

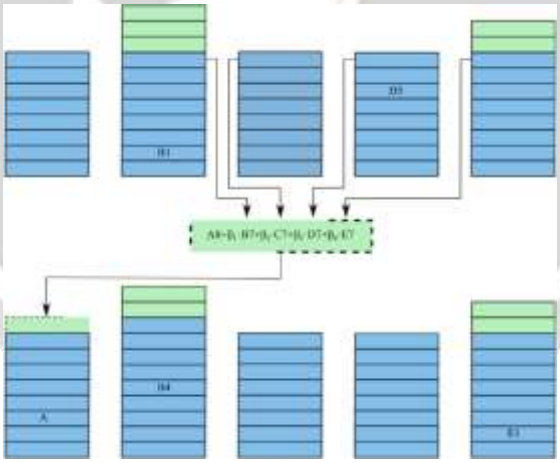

Fig -11: An example of a new packet being generated with sparse recoding for Cloud A [4]

Because some clouds will inevitably be faster, they should store a larger part of the data to improve download time. The faster Cloud B and Cloud E already store some performance enhancing data. In the example, the download speed of Cloud A increases, which determines that a new performance enhancing packet should be added to it. Our proposed sparse recoding solution generates the new packet A8 by creating a linear combination of packets B7, C7, D7 and E7 using random coefficients. This is then uploaded to Cloud A. Similarly, if Cloud A becomes even faster in the future, packet A9 would be generated by combining packets B6, C6, D6 and E6[4].

In fifth Paper, author Zhengwei Qi, Chengcheng Xiang, Ruhui Ma, Jian Li, Haibing Guan and David S. L. Wei proposed work for live forensics is an important technique in cloud security but is facing the challenge of reliability.

Most of the live forensic tools in cloud computing run either in the target Operating System (OS), or as an extra hypervisor. The tools in the target OS are not reliable, since they might be deceived by the compromised OS. Furthermore, traditional general purpose hypervisors are vulnerable due to their huge code size. However, some modules of a general purpose hypervisor, such as device drivers, are indeed unnecessary for forensics. In this paper, they propose a special purpose hypervisor, called ForenVisor, which is dedicated to reliable live forensics.

The reliability is improved in three ways: reducing Trusted Computing Base (TCB) size by leveraging a lightweight architecture, collecting evidence directly from the hardware, and protecting the evidence and other sensitive files with Filesafe module. We have implemented a proof-of-concept prototype on the Windows platform, which can acquire the process data, raw memory, and I/O data, such as keystrokes and network traffic. Furthermore, they evaluate ForenVisor in terms of code size, functionality, and performance. The experiment results show that ForenVisor has a relatively small TCB size of about 13 KLOC, and only causes less than 10% performance reduction to the target system. In particular, our experiments verify that ForenVisor can guarantee that the protected files remain untampered, even when the guest OS is compromised by viruses, such as *'ILOVEYOU'* and *Worm.WhBoy*. Also, our system can be loaded as a hypervisor with no need of pausing the target OS, and thus our system can also gather live evidence of a target OS running on hardware directly[5].

3. CONCLUSIONS

To provide reliability to forensic data need and forensic cloud architecture that have plug-in-play data center that shown in first two papers. To provide reliability to forensic data in cloud data center by linear network coding is described in third and fourth paper and in the fifth paper hypervisor convert to forenvisor with minimal code to make it forensic reliable. All survey based on to provide reliability to digital forensic data.

4. REFERENCES

[1]. Alecsandru Patrascu and Victor Valeriu Patriciu "Implementation of a Cloud Computing Framework for Cloud Forensics" In International Conference on System Theory, Control and Computing, 2014 IEEE, DOI:10.1109/IAdCC.2014.6779427, pp 440-445

[2]. Shams Zawoad, Ragib Hasan, and Anthony Skjellum "OCF: An Open Cloud Forensics Model for Reliable Digital Forensics" In 8th International Conference on Cloud Computing, 2015 IEEE, DOI: 10.1109/COMSNETS.2014.6734930, pp 437-444

[3]. Frank H.P. Fitzek, Tamas Toth, Aron Szabados, Morten V. Pedersen, Daniel E. Lucani, Marton Sipos, Hassan Charaf, Muriel Medard "Implementation and Performance Evaluation of Distributed Cloud Storage Solutions using Random Linear Network Coding" ICC'14 - W13: Workshop on Cooperative and Cognitive Mobile Networks , 2014, DOI:10.1109/ICDSE.2014.6974610, pp 249-254

[4]. Marton Sipos, Frank H.P. Fitzek, Daniel E. Lucani, Morten V. Pedersen "Dynamic Allocation and Efficient Distribution of Data Among Multiple Clouds Using Network Coding" IEEE 3rd International Conference on Cloud Networking (CloudNet), 2014, DOI: 10.1109/CloudNet.2014.6968974, pp 90-95

[5]. Zhengwei Qi, Chengcheng Xiang, Ruhui Ma, Jian Li, Haibing Guan and David S. L. Wei "ForenVisor: A Tool for Acquiring and Preserving Reliable Data in Cloud Live Forensics" IEEE Transactions on Cloud Computing , Volume: PP, Issue: 99, DOI: 10.1109/TCC.2016.2535295, pp1-14.

[6]. Sameera Almulla, Youssef Iraqi and Andrew Jones "Cloud forensics: A research perspective" Computer published by the IEEE computer society, 2013 IIT'13 1569711939, pp1-6