An MTD-based Self-Adaptive Resilience Approach for Cloud Systems

Miguel Villarreal-Vasquez*, Bharat Bhargava*, Pelin Angin†, Noor Ahmed‡, Daniel Goodwin§, Kory Brin§, Jason Kobes§
*Computer Science, Purdue University
Email: {mvillar, bbshail}@purdue.edu
†Computer Engineering, Middle East Technical University
Email: pangin@ceng.metu.edu.tr
‡Air Force Research Laboratory
Email: norman.ahmed@us.af.mil
§Northrop Grumman Corporation
Email: {daniel.goodwin, kory.brin, jason.kobes}@ngc.com

***Abstract*—Advances in cloud computing have made it a feasible and cost-effective solution to improve the resiliency of enterprise systems. However, the replication approach taken by cloud computing to provide resiliency leads to an increase in the number of ways an attacker can exploit or penetrate the systems. This calls for designing cloud systems that can accurately detect anomalies and dynamically adapt themselves to keep performing mission-critical functions even under attacks and failures. In this paper, we propose a self-adaptive resiliency approach for cloud enterprise systems that employs a live monitoring and moving target defense based approach to automatically detect deviations from normal behavior and reconfigure critical cloud processes through software-defined networking to mitigate attacks and reduce system downtime. The proposed solution is promising to present a unified framework for resilient cloud systems.**

***Keywords*-moving target defense; resiliency; adaptability; cloud security**

I. Introduction

Recent advances in cloud computing infrastructures have given increased traction to the adoption of cloud-based systems for reliable and elastic computing needs of enterprises. However, in a cloud-based environment, the enlarged attack surface hampers attack mitigation, especially when attacks originate at the kernel level. In a virtualized environment, an adversary that has fully compromised a virtual machine (VM) and has system privileges, exposes the cloud processes to attacks that might compromise their integrity, jeopardizing mission-critical functions.

A major issue with existing cloud defense solutions is that they target specific threats, which makes them ineffective for fighting against attacks lying outside their protection perimeter. In order to provide effective threat mitigation across various cloud systems, it is critical to design a resiliency solution in which the protection against attacks is integrated across all layers of the system at all times.This requires designing cloud enterprise frameworks that can accurately detect system anomalies and dynamically adapt through *starting secure*, *staying secure*, and returning to *secure+* [1] state in cases of cyber-attacks.

We propose an approach for cloud system resiliency that is capable of dynamically adapting to attack and failure conditions through performance/cost-aware process replication, automated software-based monitoring and reconfiguration of virtual machines. The proposed approach offers many advantages over existing solutions for resiliency in trusted and untrusted clouds, among which are the following:

- The solution is generic and targets multiple layers of the cloud software stack, as opposed to traditional techniques for mitigation targeting specific attacks.
- The proposed resiliency framework facilitates proactive mitigation of threats and failures through active monitoring of the performance and behavior of services and can incorporate new tools to resiliency and antifragility under various failures and attacks.
- Continuous monitoring, restoration and healing of cloud system operations allows for starting secure, staying secure and returning secure+ by learning from the attacks and failures and reconfiguring processes accordingly to increase resiliency.

The rest of this paper is organized as follows: Section II provides an overview of monitoring and security approaches in distributed computing. Section III introduces the proposed resiliency framework. Section IV discusses the results of preliminary experiments for the feasibility of the approach. Section V concludes the paper.

II. Related Work

Current industry-standard cloud systems such as Amazon EC2 [1] provide coarse-grain monitoring capabilities (e.g. CloudWatch) for various performance parameters for services deployed in the cloud. Although such monitors are useful for handling issues such as load distribution and

Approved for public release by Northrop Grumman, Case # 17-1053

[1] https://aws.amazon.com/ec2

elasticity, they do not provide information regarding potentially malicious activity in the domain. Log management and analysis tools such as Splunk[2], Graylog[3] and Kibana[4] provide capabilities to store, search and analyze big data gathered from various types of logs on enterprise systems, enabling organizations to detect security threats through examination by system administrators. Such tools mostly require human intelligence for detection of threats and need to be complemented with automated analysis and accurate threat detection capability to quickly respond to possibly malicious activity in the enterprise and provide increased resiliency by providing automation of response actions.

Various moving target defense (MTD) solutions have been proposed to provide protection against specific threats in systems. However, these are only effective against attacks within their scope. For instance, while application-level replication schemes mitigate attacks targeting the application code base, they fail in the case of code injection attacks targeting runtime execution. Randomizing runtime [2], and system calls [3], instruction set randomization [4] and address space randomization [5], have been successfully used to mitigate system-level attacks. Althought most of these security mechanisms are effective for attacks they target, modern complex attacks against cloud systems call for defense approaches that are deeply integrated into the architecture, at all system layers and at all times.

III. Proposed Approach

We propose a novel approach that uses cloud-based domain activity monitors to audit service behavior and performance changes to detect anomalies that trigger the reconfiguration of the system. The reconfiguration is based on our virtualization-based MTD strategy for distributed applications, which benefits from the flexibility offered by software-defined networking (SDN) and its capability of dynamically configuring network devices via OpenFlow[5]. By integrating components for service performance monitoring and dynamic reconfiguration, the proposed model aims to provide a unified framework for agile and resilient computing in trusted and untrusted clouds. Figure 1 illustrates a high level view of the framework, based on the idea of *starting*, *staying*, and *returning* secure in the cloud process lifecycle as proposed by Goodwin et al. [1].

General characteristics of the solution are as follows:

- The operations of each cloud-based service and domain are monitored using monitoring tools (e.g. Heat[6] and Monasca[7] for OpenStack[8]) built on top of the cloud platform. These tools report performance and security parameters such as response time, response status, CPU usage, memory usage, etc. to anomaly detection tools built on top of the same infrastructure.
- The analysis results by the anomaly detection tools are reported to a central monitor in the form of summary statistics for the services/VMs. The central monitor utilizes data submitted by the monitors to update trust values of services and reconfigure services to provide resiliency against attacks and failures.
- A moving target defense approach that migrates services to different platforms periodically to narrow the exposure window of a node to attacks is utilized, which increases the cost of attacks on a system and lowers the likelihood of success. Detection of service failures and/or suboptimal service performance, as well as integrity violations detected with virtual machine introspection also trigger restoration of optimal behavior through replication of services and adaptable migration of virtual machines to different platforms.

[2]https://www.splunk.com
[3]https://www.graylog.org
[4]https://www.elastic.co/products/kibana
[5]http://archive.openflow.org
[6]https://wiki.openstack.org/wiki/Heat
[7]https://wiki.openstack.org/wiki/Monasca
[8]http://www.openstack.org

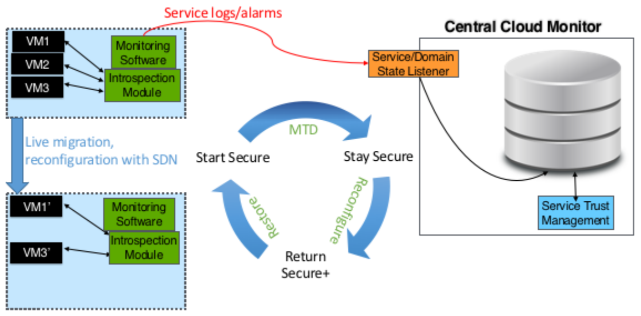


Figure 1: High-level view of resiliency framework

The following subsections provide details of the main components of the proposed resiliency approach.

A. Live Monitoring

Cyber-resiliency is the ability of a system to continue degraded operations, self-heal, or deal with the present situation when attacked [1]. For this we need to measure the assurance level (integrity/accuracy/trust) of the system from the Quality of Service (QoS) parameters such as response time, throughput, packet loss, delays, consistency, etc.

The solution developed for dynamic reconfiguration of service compositions as described in [6] involved a distributed set of monitors in every service domain for tracking performance and security parameters and a central monitor to keep track of the health of various cloud services. Even though the solution enables dynamic reconfiguration of entire service compositions in the cloud, it requires replication, registration and tracking of services at multiple sites, which could have performance and cost implications for the enterprise. To overcome these challenges, the framework proposed in this work utilizes *live monitoring* of cloud resources to dynamically detect deviations from normal behavior and integrity violations, and *self-heal* by reconfiguring service compositions through software-defined networking [7] of automatically migrated service/VM instances.

As the goal of the proposed resiliency solution is to provide a generic model, for detection of possible threats and failures in a cloud-based runtime environment, limiting the utilized anomaly detection models to supervised learning algorithms will not provide the desired applicability. Hence, unsupervised learning models such as *k-means clustering* [8] and *one-class SVM classification* [9] to detect outliers (i.e. anomalies) in service and VM behavior will be more appropriate. Algorithm 1 shows an adaptation of the k-means algorithm to cluster service performance data under normal system operation conditions and algorithm 2 shows how to detect outliers by measuring the distance of the performance vector of a service at a particular point in time to all clusters formed during training. Additionally, virtual machine introspection (VMI) [10] techniques need to be utilized to check the integrity of VMs at runtime to ensure that the application's memory structure has not been modified in an unauthorized manner. The results of the monitoring and anomaly detection processes help decide when to reincarnate VMs as described in the next section.

```
Input: Set $S$ of performance vectors
Output: Set $S^m$ of performance vector clusters
assign $K$ centroids $C^1$,...,$C^k$ for each cluster in $S^m$;
while stopping condition not met do
    for $x^i \in S$ do
        find $C^j$ s.t. $d(x^i, C^j)$ is min across all $C^j$;
        assign $x^i$ to cluster $S^j$
    end
    for $S^i \in S^m$ do
        $C^i = \sum_{x^j \in S^i} x^j / |S^i|$
    end
end
```

Algorithm 1: Anomaly training algorithm

```
Input: performance vector $x^t$ at timepoint $t$
Output: normal or anomalous
$status$ = anomalous;
for $S^i \in S^m$ do
    if $d(C^i, x^t) \leq max\_distance\_in\_S^i$ then status =
     normal ;
end
return status;
```

Algorithm 2: Anomaly detection algorithm

B. Moving Target Defense

Moving target defense (MTD) as defined by the US Department of Homeland Security is *controlling change across multiple system dimensions to increase uncertainty and complexity for attackers to increase the cost of their attack efforts* [11]. The proposed MTD-based attack-resilient virtualization-based framework is based on [12], a solution that reduces the vulnerability window of nodes (virtual machines) mainly through three steps:

1) Partitioning the runtime execution of nodes in time intervals
2) Allowing nodes to run only with a predefined lifespan on heterogeneous platforms (i.e. different OSs)
3) Live monitoring

The main idea of this MTD-technique is allowing a node running a distributed application on a given computing platform for a controlled period of time before vanishing it. The allowed running time is chosen in such a manner that successful ongoing attacks become ineffective and a new node with different computing platform characteristics is created and inserted in place of the vanishing node. The new node is updated by the remaining nodes after completing the replacement. The required synchronization time is determined by the application and the amount of data that needs to be transferred to the new node. as the reincarnation process do not keep the state of the old node.

The randomization and diversification technique of vanishing a node to appear in another platform is called *node reincarnation* [12]. One key question is determining when to reincarnate a node. One approach is setting a fixed period of time for each node and reincarnating them after that lifespan. In this first approach nodes to be reincarnated are selected either in Round Robin or randomly. However, attacks can occur within the lifespan of each machine, which makes live monitoring mechanisms a crucial element. Whether an attack is going on at the beginning of the reincarnation process determines how soon the old node must be stopped to keep the system resilient. When no threats are present both the old node and new node can participate in the reincarnation process. The old node can continue running until the new node is ready to take its place. On the contrary, in case an attack is detected the old node should be stopped immediately and the reincarnation should occur without its participation, which from the perspective of the distributed application represents a greater downtime of the node.

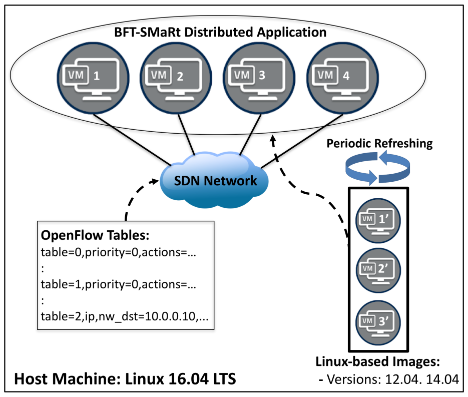
Our main contribution here is the design and implementation of a prototype that speeds up the node reincarnation process using SDN, which allows configuring network devices on-the-fly via OpenFlow. We avoid swapping virtual network interfaces of the nodes involved in the process as proposed in [12] to save time in the preparation of the new virtual machine. The new virtual machine is created and automatically connected to the network. The machine then starts participating in the distributed application when routing flows are inserted to the network devices to redirect the traffic directed to the old VM to the new one.

Measurements	Times
VM restart time	$\sim 7s$
VM creation time	$\sim 11s$
Open vSwitch flow injection time	$\sim 250ms$

Table I: Reincarnation process times

IV. Experiments

Experiments to evaluate the operation times of the proposed MTD solution were conducted. Figure 2 shows the experiment setup. A Byzantine fault tolerant (BFT-SMaRt) distributed application was run on a set of Ubuntu (either 12.04 or 14.04 randomly selected) VMs in a private cloud, which are connected with an SDN network using Open vSwitch[9]. The reincarnation is stateless, i.e. the new node (e.g. VM1') does not inherit the state of the replaced node (e.g. VM1). The set of new VMs are periodically refreshed to start clean and the network is reconfigured using OpenFlow when a VM is reincarnated to provide continued access to the application. Table I presents the results: virtual machine restarting and creation time, and Open vSwitch flow injection time. Note that the important factor for system downtime here is the Open vSwitch flow injection time, as VM creation and restart take place periodically to create fresh backup copies, and do not affect the downtime.



```
OpenFlow Tables:
table=0,priority=0,actions=...
:
table=1,priority=0,actions=...
:
table=2,ip,nw_dst=10.0.0.10,...
```

Figure 2: Experiment setup

V. Conclusion

We proposed a novel approach to introduce resiliency into cloud systems such that they can mitigate attacks and failures to provide uninterrupted operation of critical functions. The solution is based on distributed monitoring of cloud service/VM behavior and periodic refreshing of the related cloud resources to allow self-adaptive reconfiguration through SDN with a moving target defense approach. We demonstrated with preliminary experiments that the MTD-based solution is able to achieve acceptable reconfiguration times. In future work we will focus on the development and evaluation of a full resiliency framework for cloud systems based on the ideas presented in this work, not only for stateless but also for stateful distributed applications.

[9]http://openvswitch.org/

Acknowledgment

This work was funded by the Northrop Grumman Cybersecurity Research Consortium. The prototype was implemented in collaboration with Northrop Grumman and internally presented to them in April 2017. The authors would also like to thank Dr. Leszek Lilien and Dr. Weichao Wang for their valuable comments.

References

[1] S. Norman, J. Chase, D. Goodwin, B. Freeman, V. Boyle, and R. Eckman, "A condensed approach to the cyber resilient design space," *INSIGHT*, vol. 19, no. 2, pp. 43–46, 2016.

[2] J. Xu, Z. Kalbarczyk, and R. K. Iyer, "Transparent runtime randomization for security," Tech. Rep. UILU-ENG-03-2207, 2003.

[3] C. Warrender, S. Forrest, and B. Pearlmutter, "Detecting intrusions using system calls: Alternative data models," in *Proceedings of the 1999 IEEE Symposium on Security and Privacy*, 1999, pp. 133–145.

[4] G. S. Kc, A. D. Keromytis, and V. Prevelakis, "Countering code-injection attacks with instruction-set randomization," in *Proceedings of the 10th ACM Conference on Computer and Communications Security*, 2003, pp. 272–280.

[5] H. Shacham, M. Page, B. Pfaff, E.-J. Goh, N. Modadugu, and D. Boneh, "On the effectiveness of address-space randomization," in *Proceedings of the 11th ACM Conference on Computer and Communications Security*, 2004, pp. 298–307.

[6] B. Bhargava, P. Angin, R. Ranchal, and S. Lingayat, "A distributed monitoring and reconfiguration approach for adaptive network computing," in *Proceedings of the 2015 IEEE 34th Symposium on Reliable Distributed Systems Workshop (SRDSW)*, 2015, pp. 31–35.

[7] K. Kirkpatrick, "Software-defined networking," *Communications of the ACM*, vol. 56, no. 9, pp. 16–19, Sep. 2013.

[8] M. H. Marghny and A. I. Taloba, "Outlier detection using improved genetic k-means," *International Journal of Computer Applications*, vol. 28, no. 11, pp. 33–36, August 2011.

[9] L. M. Manevitz and M. Yousef, "One-class svms for document classification," *Journal of Machine Learning Research*, vol. 2, pp. 139–154, Mar. 2002.

[10] T. Garfinkel and M. Rosenblum, "A virtual machine introspection based architecture for intrusion detection," in *Proceedings of the Network and Distributed Systems Security Symposium*, 2003, pp. 191–206.

[11] DHS, "Moving target defense," https://www.dhs.gov/science-and-technology/csd-mtd, Accessed Feb. 2017.

[12] N. Ahmed and B. Bhargava, "Mayflies: A moving target defense framework for distributed systems," in *Proceedings of the 2016 ACM Workshop on Moving Target Defense*, 2016, pp. 59–64.