IOSR Journal of Electrical and Electronics Engineering (IOSR-JEEE)
e-ISSN: 2278-1676,p-ISSN: 2320-3331, Volume 12, Issue 2 Ver. I (Mar. – Apr. 2017), PP 23-29
www.iosrjournals.org

Design of Intrusion Tolerance System based on Service Redundancy Level

Hyun Kwon[1], Yongchul Kim[2], Hyunsoo Yoon[1]
[1](School of Computing / Korea Advanced Institute of Science and Technology, Korea)
[2](Department of Electrical Engineering / Korea Military Academy, Korea)

Abstract: *The Internet is an open space where a great number of computer systems are connected. Since many services are provided through the Internet, malicious users can easily intrude on any of those systems by using the vulnerabilities of the Internet. Although Intrusion Detection and Prevention System (IDPS) can be used to defend against such malicious activities, it is not always possible to completely protect a targeted system against the attacks. For this reason, Intrusion Tolerance Systems (ITS) has been proposed to maintain services even in threatening environments, where some malicious attacks have intruded into a system successfully. In this paper, we propose a new ITS based upon maintaining a service redundancy level to ensure that all services are properly provided to users even if a malicious intrusions such as VM (virtual machine) escape attack exists. The simulation results show that the proposed scheme can guarantee the operation of every ongoing service by maintaining the service redundancy level of all services.*

Keywords: *Intrusion Tolerance System (ITS), Service redundancy level, Self Cleansing Intrusion Tolerance (SCIT), Virtual Machine (VM)*

I. Introduction

The Internet is the global system of interconnected computer networks, and allows users to perform many tasks such as information exchange, communication, purchasing, etc. For the simplicity of providing services, servers are implemented in the Internet so that the users can access the server whenever and wherever they are. Since the number of users is growing dramatically, malicious users can also easily access the Internet to attack target systems. In other words, the Internet security vulnerability is considered as a serious problem and the total number of vulnerabilities continues to grow as depicted in Figure 1. To solve these problems, many security alternatives have been introduced such as Firewall, Intrusion Detection System (IDS), and Intrusion Prevention System (IPS) [1]. However the attack incidence has not decreased. In recent years, government control systems are often attacked by unknown zero-day attacks [2]. The IDS and IPS cannot completely avoid evolving malicious attacks. The intrusion tolerance system (ITS) is recently receiving a great deal of attention as a potential solution for any kind of cyber attack. The focus of ITS is to maintain services under attack unlike the conventional security solutions that focus on detecting and blocking all the attacks. Therefore the ITS can tolerate some level of attacks but ensures that the whole system does not collapsed as a result of the attacks. Cloud computing technology has also developed at a phenomenal rate and made it possible for users to access data, applications and services over the Internet. Accordingly much of the work to date has focused on ITS techniques in the cloud computing environment. One of the well-known techniques is the self cleansing intrusion tolerance (SCIT) system [3]. The SCIT scheme uses virtual machines (VM) that periodically repeat the online and offline processes, i.e., a VM provides services during online periods and then goes to the recovery process to tolerate potential attacks. However in the virtual environments, a special attack pattern like VM escape is always possible. When a VM escape attack successfully intrudes on a VM in a host, then the rest of the VMs in that host will be easily affected by the VM escape attack. For example, when there are several VMs providing the same service to clients and operating in the same host, VM escape attack is able to paralyze that host, resulting in a specific service down. This phenomenon is considered a common vulnerability in the conventional SCIT systems.

In this paper we propose a new SCIT system that guarantees none of the ongoing services are down from the VM escape attack by maintaining service redundancy level. The meaning of service redundancy level is that a number of hosts provide the same service. For example, if a Web service is provided by host1, host2, and host3 at the same time, then the service redundancy level will be three, while when a Web service is only provided by host1, the service redundancy level is one. The detailed explanation will be addressed in Section III.

The remainder of the paper is structured as follows: section II reviews related works as to ITS, SCIT, and virtual environments. After introducing the proposed SCIT system in section III, performance results and findings are subsequently presented in Section VI. Finally conclusions are drawn in Section V.

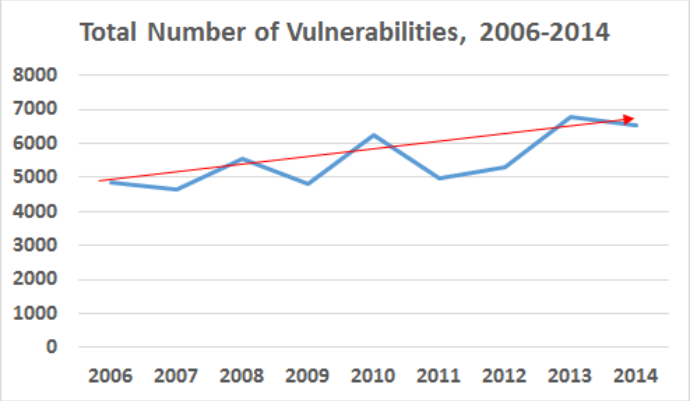

Fig.1: Total Number of Vulnerabilities (2006~2014) [2]

II. Related Works

The main goal of ITS is to preserve a certain level of services under any kind of malicious attacks [4]. Figure 2 shows the concept of ITS [5]. Unauthorized user can be able to intrude a specific system by using the vulnerability of Internet or fault operation of system administrator. IDS and Firewall are used to block well-known attacks but they do not provide perfect security for the system since there are always unknown attackers. After successful intrusion, these attackers are able to use the services for leaking information or modify the system intentionally to cause the operating system to malfunction. Therefore the ITS is needed to keep the system from going down under the intruded attackers.

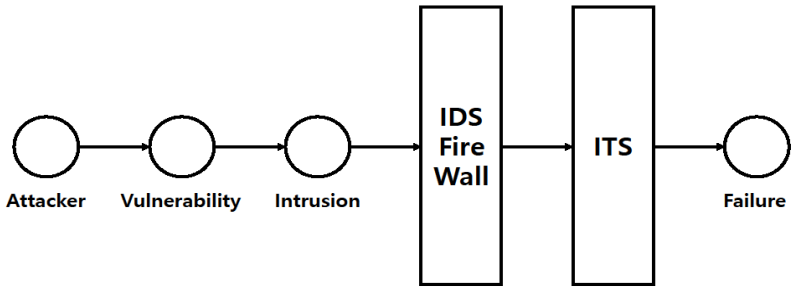

Fig. 2: The concept of Intrusion Tolerance System (ITS)

To achieve the goal of ITS, in general, redundancy and diversity are the two fundamental principles in designing ITS [6][7]. Redundancy refers to the duplication of elements or services in a system. When some of the elements or services are failed, the whole system can be maintained by using duplicated substitutes. However, due to the fact that the simply duplicated elements or services are having the same vulnerability, it will be easier and faster to be attacked. For this reason, diversity principle has also to be considered, i.e., the duplicated substitutes have same function for redundancy but different attributes for diversity. One of the well-known ITSs taking into account fundamental principles is the SCIT in the virtual environment. The aim of the SCIT technique is to prevent system failure by applying preemptive recovery process as depicted in Figure 3.

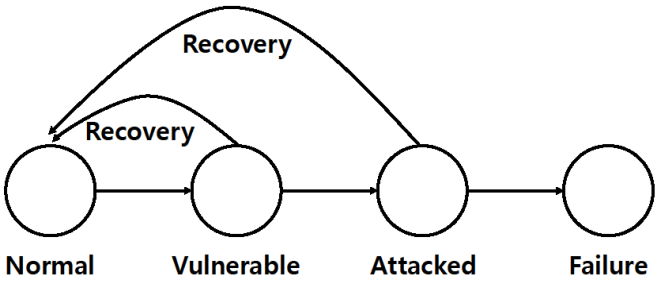

Fig. 3: SCIT state transition diagram [8]

In general, an SCIT system consists of several hosts and each host containing VMs that are providing services. Every VMs in a system are controlled by a central controller and follows four step circular process; Active – Grace period – Cleansing – Live spare [3]. During active state period, a VM is providing service by receiving and processing incoming requests, thus this period is denoted as exposed window time. Once an

exposed window time of a VM is expired, it goes to grace period where a VM stop receiving incoming requests but processing the requests in its queue. After that a VM undergoes cleansing period to be recovered to a good state. When a VM is recovered, it is ready to be used again, hence standing by period is called live spare state. The both active and grace period are considered online states whereas cleansing and live spare are considered offline states. In an SCIT state transition, the average time consumed from normal state to failure state is denoted as MTTSF (Mean Time To Security Failure) and it was shown that the MTTSF value can be increased by reducing the exposed window times of VMs [8]. An overview of an SCIT cluster is shown in Figure 4. The central controller replaces VMs in active state (online) with VMs in live spare state (offline). For security purposes, VMs in online state are not allowed to communicate with the central controller by using two way path communication, i.e., only one way path communications from the controller to online VMs are possible in order to prevent the external clients from breaking into the central controller [3].

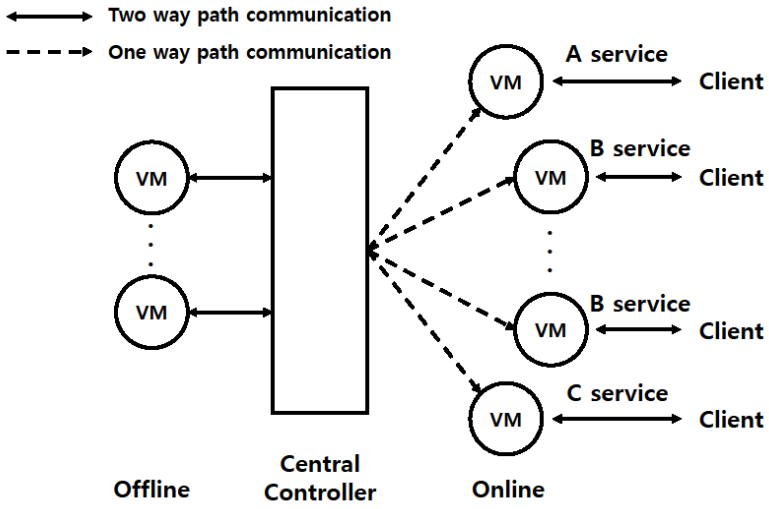

Fig. 4: The overview of an SCIT cluster

The authors in [9] introduce security issues in virtualization environments. The hypervisor also known as VM monitor is a vulnerability since it has root privilege to access every VMs in a host. One of the well-known attacks that uses this vulnerability is a VM escape. After bypassing the first attacked VM, it gains access to the host machine where all VMs are controlled. To maximize system throughput and response time, a dynamic load balancing using VM migration technique is presented in [10]. The idea of VM migration is to move VMs from a heavy host to a slight host to increase the system performance. Another work [11] proposes a load balancing scheme based on VM migration to guarantee fault-tolerant requirement. On the other hand, a migration scheme causes degradation of service process speed due to additional usage of CPU and the waste of service time for moving VMs. To correct a weak point of migration scheme while minimizing damages from the VM escape attack, a new SCIT using VM balancing is proposed in our previous paper [12]. The balancing controller determines priorities of VMs to maintain the balance between hosts. However a variety of services are not taken into account. When a certain service is only provided by VMs that are in the same host, that service can be down from a VM escape attack. Therefore, in this paper we propose another SCIT based on service redundancy level to ensure that every ongoing service is maintained even under a VM escape attack.

III. Proposed SCIT System

In this section we present the proposed SCIT system. Figure 5 shows the architecture of the proposed scheme. A central controller controls VMs in every host and all VMs are divided into two groups: online state VMs and offline state VMs. The VMs in the online state provide services to clients during the exposed time and the VMs in the offline state are undergoing recovery process. The central controller determines the switching time of VMs in different groups. We use another controller denoted as tolerance controller in our proposed system to preserve the service redundancy level. The tolerance controller must monitor every VM in the online state, i.e., service and host information of VMs are used to determine service redundancy level. To perform this task, the tolerance controller uses a VM service distribution table. Table 1 shows an example of VM service distribution in a system. From the table 1, we can see that there are three VMs providing Web service and one live spare state VM in host 1. The Web service redundancy level is three while the DNS service redundancy level is two.

Figure 6 shows flow chart of the proposed SCIT algorism. When the system starts, a variety of services are provided by online VMs. Whenever the exposed time of online VM expires, the tolerance controller checks the VM distribution table to determine whether there is a live spare VM that can maintain service redundancy level. If so, the live spare VM is substituted for the expired online VM immediately. If not, the online VM waits until the live spare VM maintaining redundancy level is ready. We describe this algorism through an example.

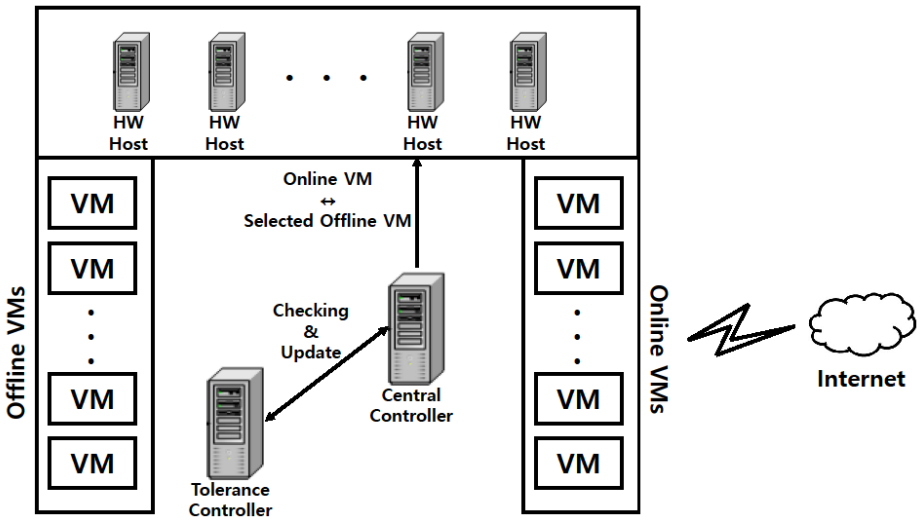

Fig. 5: Architecture of the proposed scheme

Table. 1: An example of a service distribution table

Type		Host1	Host2	Host3	Total	Service redundancy level
Online	Web	3	1	2	6	3
	DNS	0	1	1	2	2
Offline	Cleansing	0	2	1		

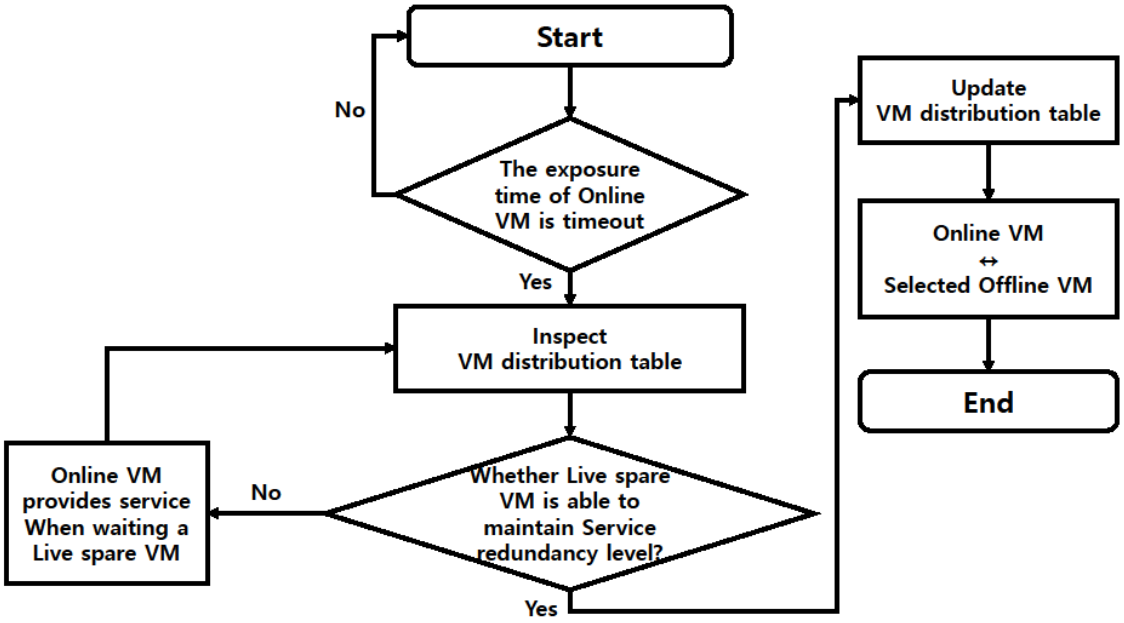

Fig. 6: Flow chart of the proposed algorism

Table 2 shows an example of required number of online VMs for Web and DNS services in a system. This system has three hosts and each host consists of four VMs to provide services as depicted in Figure 7. One of the online VMs in a host 1 has expired its exposure time, which is marked with red circle in Figure 7. Then one of the live spare VMs will be chosen to switch with the online VM. When a conventional SCIT is applied in this scenario, a randomly chosen live spare VM will be switched with the online VM without considering service redundancy level. Thus, the live spare VM in host 3 can be chosen in this example scenario.

Consequently, the DNS service redundancy level is reduced to one as shown in Figure 8. However if we apply our proposed scheme in this example scenario, the tolerance controller will choose the live spare state VM in host 2 to maintain the DNS service redundancy level and sends this information to the central controller. Then the central controller switches online VM with offline VM as depicted in Figure 9 and the service redundancy level is always maintained.

Table. 2: An example of required number of VMs for services

Description	Web service	DNS service
The required service guarantee	6 VMs	2 VMs
The minimum service guarantee	1 VM	1 VM

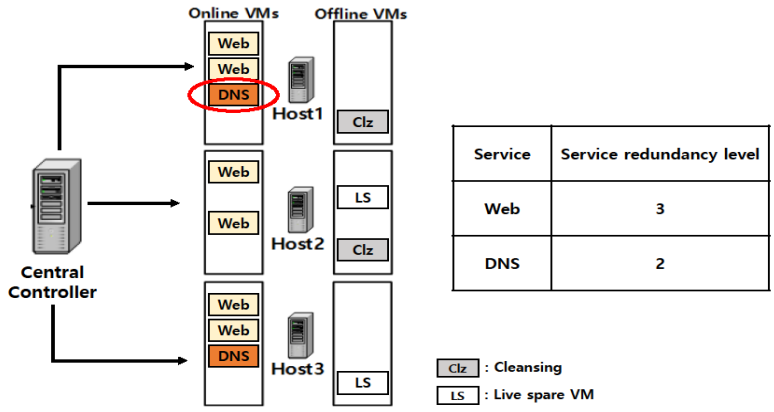

Fig. 7: An example of a system with three hosts and each host has four VMs

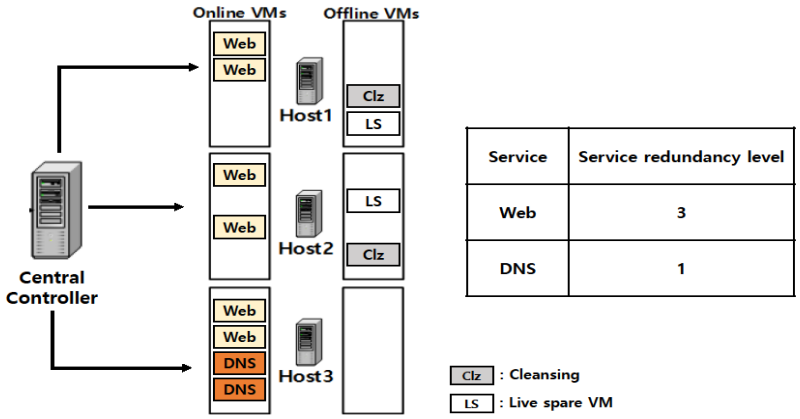

Fig. 8: An example of a conventional SCIT applied in Fig. 7

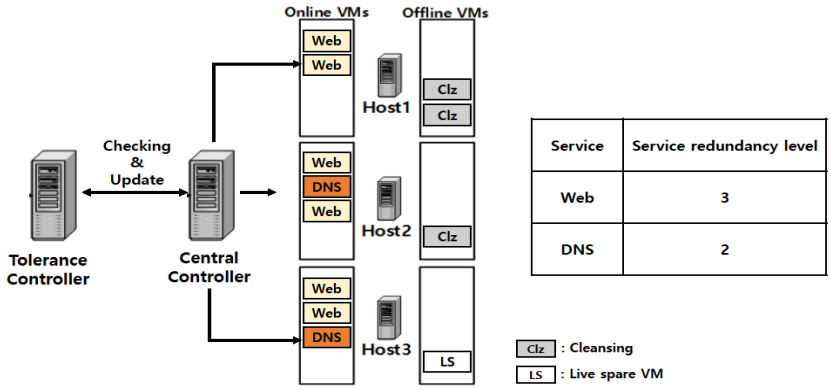

Fig. 9: An example of the proposed scheme applied in Fig. 7

IV. Performance Analysis

In order to analyze the performance of the proposed SCIT system, we used CloudSim simulator in this work [13]. CloudSim simulator is written in Java and has become one of the most popular open source cloud simulators for modeling virtual environments. The experimental environments and values used in this work are listed in Table 3. The considered system consists of three hosts and four VMs are assigned in each host. Six VMs are assigned for Web service and two VMs are assigned for DNS service in a normal situation. For minimum service guarantee, at least one VM has to be assigned for each service.

Table. 3: An experimental environment and values

Parameters	Description and values
N	The total number of VMs : 12
H	The total number of hosts : 3
N_H	The total number of VMs in each host : 4
$\Omega_{requirement}$	The required service guarantee : 6 Web service, 2 DNS service
$\Omega_{minimum}$	The minimum service guarantee : 1 Web service, 1 DNS service
$W_{exposure}$	The exposure time of Online VM : 3 minutes (180 seconds)
$T_{cleansing}$	The cleansing time of VM : 1 minute (60 seconds)

We run the simulation for both conventional SCIT and proposed SCIT systems with equal generated packets and clock cycle is used as the time unit. Whenever one of the online VMs is switched with a live spare VM, clock cycle is increased by one. Figure 10 shows the results of service redundancy levels of Web and DNS services for the conventional SCIT system. When the clock cycle is 8$\sim$11 and 24$\sim$27, DNS service redundancy level is reduced from two to one. In other words, DNS service is only provided from one host during that clock cycle. If a VM escape attack intrudes on that host during those periods, DNS service will be down. We can also see that Web service redundancy level is also reduced from three to two during the clock cycle 26$\sim$27. In contrast to the conventional SCIT system, the simulation results from the proposed system show that the service redundancy levels for both Web and DNS services are constantly stable throughout the simulation as shown in Figure 11. Therefore, it is guaranteed that none of the ongoing services is down even if a VM escape attack exists in our proposed system.

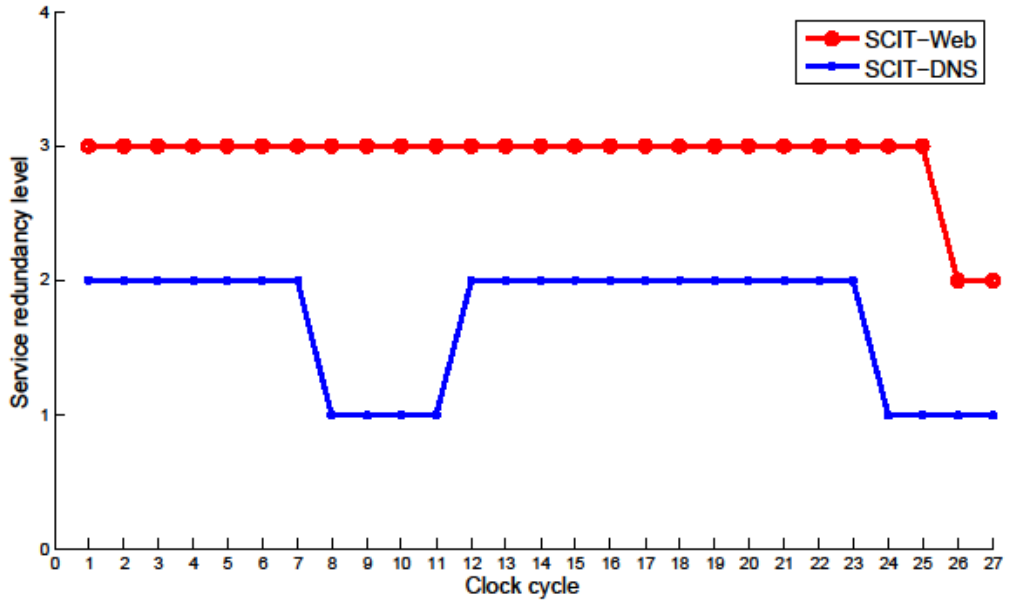

Fig. 10: Service redundancy levels of Web and DNS service in a conventional SCIT

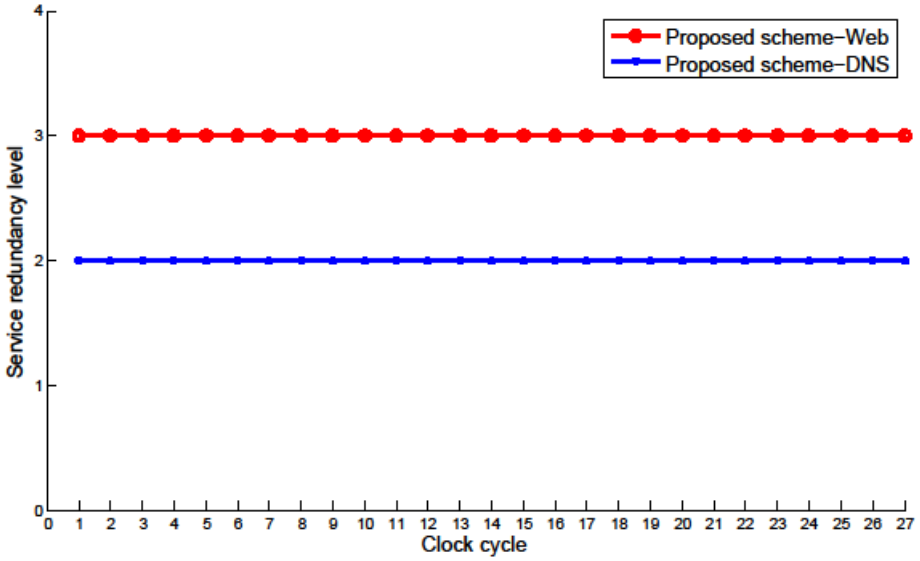

Fig. 11: Service redundancy levels of Web and DNS service in the proposed scheme

V. Conclusion

In this paper we proposed a new SCIT system in order to ensure that every service is maintained under a variety of attacks. We have illustrated that a VM escape attack can gain access to a host machine after bypassing the first attacked VM. Thus, it is likely that every VMs in the same host can be attacked at the same time from the VM escape attack. The main focus of our proposed scheme is to spread out the online VMs that are providing same service over most of the hosts. To accomplish this, we have introduced the tolerance controller to support the central controller in order to determine which live spare VM should be chosen to replace the online VM while maintaining the service redundancy level. Our simulation results show that the service redundancy levels for ongoing services fluctuated from the conventional SCIT system, however the service redundancy levels from our proposed system were constantly stable throughout the simulation.

References

[1] Kwon, Ohmin, et al, "A Survey on Intrusion-Tolerant System," *Korea Computer Congress 2012*, 2012.
[2] Internet Security Threat Report 2015 - *Symantec*, https://www.symantec.com
[3] Y. Huang, D. Arsenault, and A. Sood, "Closing cluster attack windows through server redundancy and rotations*." Proc of the sixth intl sysmp on Cluster Computing and the Grid Workshops*, 2006.
[4] Q. Nquyen, and A. Sood, "Realizing S-Reliability for Services via Recovery-driven Intrusion Tolerance Mechanism" *International conference on dependable systems and networks workshops*. 2010
[5] Lim, Jungmin, et al. "A novel Adaptive Cluster Transformation (ACT)-based intrusion tolerant architecture for hybrid information technology*." The Journal of Supercomputing 66.2 (2013):* 918-935.
[6] P. Verrissimo, N. Neves, and M. Correia, "Intrusion Tolerance : Concepts and Design Principles", *Architecting Dependable Systems*, volume 2677 of LNCS. 2003.
[7] Heo, Seondong, et al. "A Survey on Intrusion-Tolerant System." *Journal of Computing Science and Engineering (JCSE) 7.4 (2013)*: 242-250.
[8] Q. Nquyen, and A. Sood, "Quantitative Approach to Tuning of a Time-Based Intrusion-Tolerance System*," 3rd Workshop on Recent Advances in Intrusion Tolerance System*, Portugal, 2009.
[9] Joshi, Nilambari, and J. N. Varshapriya. "Analytical Survey of Security in Virtualized Environment." *International Journal of Computer Science and Information Technology & Security 3.1(2013)*: 66-71.
[10] Wilcox Jr, "Dynamic load balancing of virtual machines hosted on Xen," Master's Thesis. USA : Brigham Young University, 2009.
[11] Yao, Lin, et al. "Guaranteeing Fault-Tolerant Requirement Load Balancing Scheme Based on VM Migration." *The Computer Journal(2013)* : bxt012.
[12] Hyun Kwon, Yongchul Kim "Intrusion Tolerance System Using VM Balancing In Virtual Environments" *Published at International Journal of Engineering Research and Development*, Volume 12, Issue 8 (August 2016)
[13] Rodrigo N. Calheiros, CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms, *Softw. Pract. Exper. 2011*; pp 23–50.