applied sciences

MDPI

Article

Optimal Cluster Expansion-Based Intrusion Tolerant System to Prevent Denial of Service Attacks

Hyun Kwon [1,†], Yongchul Kim [2], Hyunsoo Yoon [1] and Daeseon Choi [3,*]

1 School of Computing, Korea Advanced Institute of Science and Technology, Daejeon 34141, Korea; khkh@kaist.ac.kr (H.K.); hyoon@kaist.ac.kr (H.Y.)
2 Department of Electrical Engineering, Korea Military Academy, Seoul 01805, Korea; kyc6465@kma.ac.kr
3 Department of Medical Information, Kongju National University, Gongju-si 32588, Korea
* Correspondence: sunchoi@kongju.ac.kr; Tel.: +82-041-850-0340
† Current address: KAIST, 291 Daehak-ro, Yuseong-gu, Daejeon 305-701, Korea.

Received: 3 November 2017; Accepted: 15 November 2017 ; Published: 17 November 2017

Abstract: In this study, we propose an optimal cluster expansion-based intrusion-tolerant system (ITS) that can maintain quality of service (QoS) under a massive denial of service (DoS) attack. Our proposed scheme conserves resources while maintaining good QoS by optimally increasing and decreasing cluster size. To evaluate the performance of the proposed scheme, we use a CloudSim simulator and compare our proposed scheme with an existing conventional adaptive cluster transformation (ACT) scheme. Our simulation results show that the proposed scheme outperforms the conventional ACT scheme in terms of better QoS and lower resource consumption.

Keywords: intrusion-tolerant system; optimal cluster expansion; denial of service (DoS); virtual machine (VM); queuing theory

1. Introduction

Internet access and mobile communication technologies have developed very rapidly, and have enabled users to access useful services. This accessibility provides convenience to the users, but simultaneously makes users vulnerable to malicious attacks. Many security countermeasures have been introduced, such as firewalls, intrusion detection systems (IDSs), and intrusion prevention systems (IPSs) [1,2]. However, these systems are still vulnerable to evolving malicious attacks, such as zero-day attacks that can target publicly known but unpatched vulnerabilities. To alleviate this problem, an intrusion-tolerant system (ITS) has been recently introduced in the literature. The goal of ITS is to provide reliability and survivability by maintaining quality of service even while under attack [3]. When designing an ITS, the fundamental principles are redundancy, diversity, and recovery [4,5]. Redundancy refers to the duplication of elements or services in a system. Redundant systems provide stable services by using duplicated elements or services as substitutes when the original elements or services are attacked. Diversity refers to the variety of attributes of an element or service; thus, the duplicated substitutes can have the same function for redundancy, while having different attributes for diversity. Recovery refers to the restoration process of an element or service, and conserves resources.

As cloud computing techniques evolve, virtual machine (VM)-based ITSs have attracted the attention of researchers [6,7]. Multiple VMs are used in a host to provide redundancy principles instead of using a backup system. One well-known ITS is a self-cleansing intrusion tolerance (SCIT) system [8,9], which uses VM snapshots to periodically recover the system. Hence, the exposure time to attackers is reduced. A SCIT system consists of a central controller and several hosts. Each host contains duplicated VMs. Each VM provides service during the designated exposed time, and then undergoes the recovery process. Every VM follows a four-step circular process: active–grace period–cleansing–live

spare. During the active state period, a VM provides service by receiving and processing incoming requests. Therefore, this period is referred to as the exposed window time. This four-step process can be divided into two groups: online VMs and offline VMs, as shown in Figure 1. However, an SCIT system can be significantly affected by a massive-scale attack like a denial of service (DoS) or distributed DoS (DDoS), because the number of online VMs providing services are fixed in a SCIT system. That is, the total number of processing packets is limited [6,8,9].

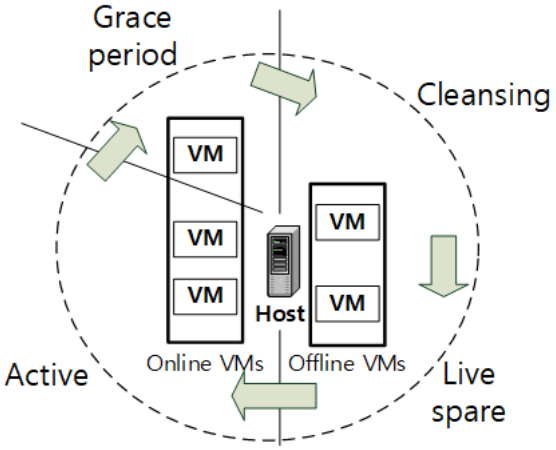


Figure 1. Overview of a self-cleansing intrusion tolerance (SCIT) system. VM: virtual machine.

To alleviate those problems, an adaptive cluster expansion scheme is necessary to preserve system QoS by rapidly and adaptively increasing the number of online VMs according to a massive packet attack. Moreover, the performance of a cluster expansion scheme must consider a waste-of-resources analysis. That is, the total number of VMs from the cluster expansion must be compared with the appropriate number of VMs necessary for preserving QoS. Therefore, in this paper, we propose an optimal cluster expansion-based ITS scheme that can optimally increase or decrease the number of online VMs to preserve system QoS. The optimal cluster size is computed using queuing theory [10], and resource waste efficiency is also considered.

The remainder of this paper is structured as follows: in the next section, we discuss related work. In Section 3, we propose an optimal cluster expansion-based ITS. The performance results for the proposed system and findings are presented in Section 4. A discussion of the proposed system is presented in Section 5. Finally, we conclude the paper and give some future works in Section 6.

2. Related Work

The generic architecture for an intrusion-tolerant system consists of various Web servers and proxies [11]. Duplicated Web servers provide the same services but manage different applications and OS on different hardware platforms. Proxies play an important role in monitoring Web servers and other proxies. Thus, they are used to control user service requests, and can be applied to IDS. In a generic ITS, a malicious attacker can reside inside the system until it shuts down the whole system. Therefore, a preemptive recovery process is needed to prevent systems from shutting down due to the malicious attacks. However, this general ITS system is not cost-effective, because it requires multiple servers and various hardware platforms.

As virtualization computing technologies evolve, VM-based ITS is considered as a cost-effective system. Redundancy is achieved by increasing the VMs, and diversity is achieved with various operating systems and applications. A recovery method is also achieved by using snapshots in the VM-based ITS systems [6,7]. Heo et al. [12] introduce a method for finding an optimal exposure time by considering resources in the rotation process of the VM. Using reactive recovery, they proposed

an additional scheme that restores abnormal VMs first by changing the order of VM rotations [13]. Most existing SCIT-related works have focused on adjusting VM rotation schedules to improve system performance. However, these SCIT systems can be significantly affected by a massive-scale attack like DoS or DDoS, because the number of online VMs are fixed in a SCIT system.

Lim et al. [14] proposed an adaptive cluster transformation (ACT) scheme to enable systems to survive various types of intrusions. This ACT scheme was able to maintain quality of service (QoS) by adopting a variable cluster size depending on the system status. However, the ACT increases the number of VMs by two at a time, regardless of the number of current online VMs. Therefore, the efficiency of cluster expansions under a massive packet attack is limited. Moreover, the ACT scheme only focuses on analyzing service response time, without considering available resources. In other words, it does account for resource waste phenomena, because it assumes unlimited resources. Shin et al. [15] introduced a VM migration scheme to preserve QoS by replacing exhausted VMs. Additionally, a cluster expansion scheme like ACT was applied in order to mitigate DoS attacks. Jang et al. [16] proposed another ACT-based cluster expansion scheme to preserve QoS under malicious intrusions by using proactive and reactive recovery. However, none of the above-mentioned schemes considered the number of current online VMs at the point where the cluster is expanded. As a result, QoS is significantly degraded under massive DoS attacks. Therefore, in this paper, we propose an optimal cluster expansion scheme to maintain good QoS even under massive DoS attacks, by optimally adjusting the number of online VMs. We also analyze resource waste efficiency by comparing the total number of increased or decreased VMs from the proposed scheme with the necessary number of VMs for preserving QoS.

3. Proposed ITS System

In this section, we present an optimal cluster expansion scheme for ITS. Figure 2 shows the architecture of the proposed scheme. The considered system consists of a robust cluster controller (RCC) and multiple hosts. The RCC controls the whole system in order to provide a guaranteed QoS under any circumstances, including a massive DoS attack. The RCC also determines the cluster size and redistribution of packets in queues by monitoring the VMs in every host and checking overall system performance with the incoming packets. Detailed RCC operations are addressed in Section 3.4. The proposed scheme is based on the following assumptions:

- The maximum number of VM processing packets is constant
- The main target scenarios are massive DoS attacks
- The existing attacks are prevented by IDS and IPS
- The considered cloud system has sufficient VMs and hosts

The entire system process can be divided into three cases: cluster size expansion, unprocessed packet distribution, and cluster size reduction. To preserve QoS service without degradation, it is important to expand the cluster size at the right time. The expansion and reduction timing should be determined by considering service response time, VM queue size, and central processing unit (CPU) usage.

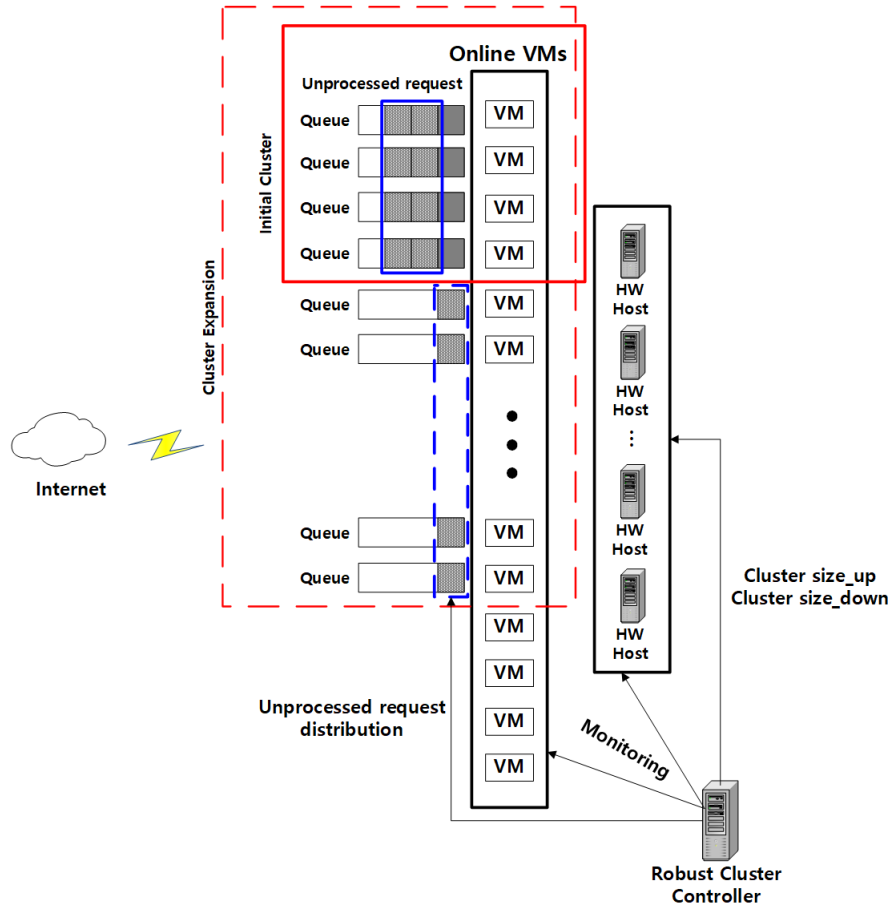


Figure 2. Architecture of the proposed scheme. HW: hardware.

3.1. Cluster Size Expansion

To continuously maintain reliable service under a massive DoS attack, an appropriate cluster size expansion method must be used. That is, when online state VMs are not properly processing requested packets, cluster size expansion is needed to increase the number of online VMs. In a conventional ACT scheme, the cluster size increases when the response time for a packet is greater than a predetermined threshold value:

$$X_{delta} = f(res_{time}),$$

$$f(res_{time}) = \begin{cases} 2 & \text{if } res_{time} > \Theta_{res\_time\ size_{up}} \\ -2 & \text{if } res_{time} < \Theta_{res\_time\ size_{down}} \\ 0 & \text{otherwise} \end{cases}$$

$$X_{next} = X_{current} + X_{delta}, \quad (1)$$

where res_{time} is the requested packet response time and X is the number of online VMs. Θ is a specific threshold value. The cluster size will be increased by two regardless of the $X_{current}$ value. When the $X_{current}$ value is small, adding two VMs can be sufficient to process the packets in the queues. However, when the $X_{current}$ value is large, a two-VM increase cannot be sufficient to process every packet in the queues. That is, the service response time of a conventional ACT scheme can be significantly delayed as the $X_{current}$ value increases. To overcome this problem, it is possible to exponentially expand the number of online VMs under a massive DoS attack, thereby maintaining QoS. However, it is then inevitable that some resources will be wasted. Therefore, it is important to calculate an optimal cluster

size that can minimize the waste of resources while preserving good QoS. In our proposed scheme, we consider both the incoming packet amount and the online VM packet processing time to compute the exact number of VMs that are needed to preserve QoS. We also consider additional parameters such as CPU usage and queue length, as well as packet response time, in order to determine the exact timing of the changing cluster size:

$$Z = \{res_{time},\ use_{cpu},\ len\},$$

$$X_{next} = f(Z, \lambda),$$

$$f(Z,\lambda) = \begin{cases} \frac{\lambda}{a\gamma} & \text{if } res_{time} > \Theta_{res\_time\ size_{up}} \\ & \text{and } use_{cpu} > \Theta_{cpu\_use\ size_{up}} \\ & \text{and } len > \Theta_{len\ size_{up}} \\ \frac{\lambda}{a\gamma} & \text{if } res_{time} < \Theta_{res\_time\ size_{down}} \\ & \text{and } use_{cpu} < \Theta_{cpu\_use\ size_{up}} \\ & \text{and } len < \Theta_{len\ size_{up}} \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

where Z is a set of three input parameters: packet response time (res_{time}), CPU usage (cpu_use), and VM queue size (len). λ, γ, and a represent the incoming packet amount per second, the user-requested service level, and the VM processing packet amount per second, respectively. Cluster size expansion is performed when all three parameters satisfy the predetermined conditions. The number of increased online VMs is determined by simultaneously considering the incoming packet numbers and the VM packet processing capabilities. Therefore, the increased number of online VMs can process the requested packets, maintaining QoS under a massive DoS attack.

3.2. Unprocessed Packet Distribution

Whenever a cluster needs expansion because of a massive DoS attack, it takes time for the cluster to be extended. During the cluster expansion time, the packets from DoS attacks accumulate in the online state VM queues. Thus, QoS will be necessarily degraded. To mitigate this problem, an additional packet distribution process is performed while the cluster expands. The total number of unprocessed packets, denoted as S, can be computed by summing all unprocessed VM packets. S can be expressed as

$$S = \sum_{i=1}^{n} Z_i, \quad (3)$$

where n is the number of online VMs before cluster size expansion, and Z_i is the number of unprocessed packets of the ith VM. After cluster size expansion, the unprocessed packets are distributed to all online VMs, including newly created VMs. Let D be the average number of packets allocated to each online VM, which can be calculated as follows:

$$D = \frac{S}{N}, \quad (4)$$

where N is the total number of online VMs after cluster size expansion. Therefore, in our proposed scheme, an unprocessed packet distribution process is performed whenever the cluster size is extended, in order to preserve the required QoS under a massive DoS attack.

3.3. Cluster Size Reduction

A cluster size reduction process is performed when the requested packets decrease. Similar to the cluster expansion process, the exact time at which the cluster size changes is determined using three parameters: packet response time, CPU usage, and queue length. Equation (2) shows how to reduce cluster size by considering the incoming packet amounts and VM packet processing capabilities. Decreasing the number of VMs helps to reduce resource waste. Unlike in a cluster expansion process,

the active state online VM is not immediately changed to an inactive state when the cluster size is reduced, because the packets in a queue must be processed before changing into an inactive state. The overall cluster size reduction process is the inverse of the cluster size expansion process.

3.4. Cluster Management

Generally, the service response time to users in a cloud environment should be shorter than 0.77 s, as indicated in [17]. Additionally, it is known that the cluster size must be increased when CPU usage reaches 80%, and decreased once again when usage falls below 50% [15]. The RCC is the monitoring service response time, CPU usage, incoming packet numbers, and queue length in real time. When the service response time is longer than 770 ms, CPU usage is higher than 80%, and queue length is higher than 50%, the RCC proceeds with an optimal cluster expansion process by considering incoming packet numbers and VM packet processing capabilities. During the cluster expansion process, an additional packet distribution process is performed to preserve QoS by preventing packets from converging into a specific VM. On the other hand, the RCC performs a cluster reduction process when the service response time is shorter than 770 ms, CPU usage is under 50%, and queue length is under 30%. The active VMs are changed into an inactive state to conserve resources. The detailed process of cluster expansion and reduction is described in Algorithm 1 using pseudo code.

Algorithm 1 Cluster expansion and reduction process

Input:

N_{max} ▷ Maximum number of online VMs
$cluster_{size}$ ▷ Current number of online VMs
use_{cpu} ▷ Mean value of CPU usage
res_{time} ▷ Mean value of response time
len ▷ Mean value of queue length
λ ▷ Number of incoming packets
a ▷ Number of packets being processed by a VM
γ ▷ Required service level
$flag$ ▷ Boolean value of system ability

Cluster expansion & reduction process:

while $flag$ *is true* **do**
  if $use_{cpu} > 0.8$ **and** $res_{time} > 0.77$ **and** $len > 0.5$ **and** $cluster_{size} < N_{max}$ **then**
    $X_{next} \leftarrow \frac{\lambda}{a\gamma}$
    $cluster_{size} \leftarrow Online\ VM(X_{next})$
    $submit\_Vm\_List(cluster_{size})$
    $Update\_Host\_in\_Cluster$
    $create\_Vm\_in\_Host$
    $Unprocessed\ packet\ distribution()$
  end if
  if $use_{cpu} < 0.5$ **and** $res_{time} < 0.60$ **and** $len < 0.3$ **then**
    $X_{next} \leftarrow \frac{\lambda}{a\gamma}$
    $cluster_{size} \leftarrow Online\ VM(X_{next})$
    $submit\_Vm\_List(cluster_{size})$
    $Update\_Host\_in\_Cluster$
    $delete\_Vm\_in\_Host$
  end if
end while

4. Performance Analysis

Through extensive simulations, we show that our proposed scheme can provide reliable and secure service under massive DoS attacks. We used a CloudSim simulator in this work [18]. CloudSim is written in Java and has become one of the most popular open source simulators for modeling virtual environments. We ran simulation on a Windows 7 (64 bit) system with a 3.3 GHz Intel i3 processor and 8 GB memory. The details of the simulation environments and setup of our simulations are as Table 1.

Table 1. Simulation environments. VM: virtual machine.

Parameters & Descriptions	Values
VM processing element capacity (PE)	512 MIPS
Maximum number of VMs in system	10,000
Highest acceptable response time [17]	0.77 s
Initial number of VMs	14
VM cleansing time [7]	146 s
VM exposure time	900 s
VM attack follows a Poisson distribution with μ	0.000485

To evaluate our proposed system in terms of availability, reliability, and resource waste, we conducted simulations in three different scenarios: normal packets, continuous DoS packets, and one-pulse DoS packets. We compare our results with a conventional SCIT and ACT scheme to show the effectiveness of the proposed scheme. Moreover, we add an additional scheme (namely, an exponential-ACT) to show an extreme case: we expand cluster size exponentially to rapidly adjust cluster size. Exponential-ACT schemes are advantageous because they can maintain good QoS under a massive attack. However, it is likely that the number of VMs is increased above the necessary amount, leading to resource waste. Therefore, we simultaneously examined four different systems.

Figure 3a shows the incoming requested packets per second during a simulation without a DoS attack. Approximately 100∼220 packets(count) are incoming to the system every second. Figure 3b,c shows the response time and VM resource results for four different systems: SCIT, ACT, exponential-ACT, and the proposed scheme. Because the incoming packets are not from a DoS attack, a cluster expansion is not performed, and all four systems produce the same results.

Figure 4a shows the incoming requested packets per second during a DoS attack. Approximately 1000∼1120 packets(count) are incoming to the system every second. Figure 4b shows the service response time results under a DoS attack. The response times for SCIT and ACT increase linearly with time, but the response times for exponential-ACT and our proposed scheme are shorter than 770 ms, and are almost constant for the whole duration (i.e., QoS is maintained). Figure 4c shows the number of online VMs as a function of time. The number of VMs from the exponential-ACT system increased to 1038, while the remaining systems had fewer than 117 VMs. Therefore, the resource waste from the exponential-ACT scheme is significant, whereas our proposed scheme optimally expanded cluster size to maintain good QoS without wasting resources.

Figure 5a shows a one-pulse DoS attack scenario. Approximately 2000∼2150 packets are incoming between 1600∼3900 s. Figure 5b shows the response time results for each system. The response times for SCIT and ACT increase continually, even after the DoS attack ends. This occurs because the requested packets from the DoS attack remain in the queue, and a limited number of online VMs is available to process them. In contrast, the response times for exponential-ACT and our proposed scheme are stable for the whole time, although there is a slight increase at the beginning of the one-pulse DoS attack. The performance of our proposed scheme is similar to that of an exponential-ACT scheme with respect to packet response time. However, the number of VMs in the exponential-ACT system is substantially higher than the number of VMs in our proposed scheme, as shown in Figure 5c.

To more efficiently analyze resource waste, we define the VM processing rate parameter as the number of actual VMs that are used to process packets divided by the total number of VMs at a given time. Figure 6 shows the VM processing rate of each scheme under three different scenarios. When there is no DoS attack, the initial number of VMs is maintained and is fully used to process incoming packets for the duration of the simulation time, as shown in Figure 6a. In contrast, when there is a continuous or one-pulse DoS attack, the number of VMs increases in every system to process incoming DoS packets. The VM processing rates for the SCIT, ACT, and our proposed scheme preserve 100% without wasting resources, because all the additional VMs are fully used to process packets. In contrast, the VM processing rate of the exponential-ACT scheme decreases significantly because of the redundant VMs that are created by the expansion process but are not used for processing packets, as shown in Figure 6b,c.

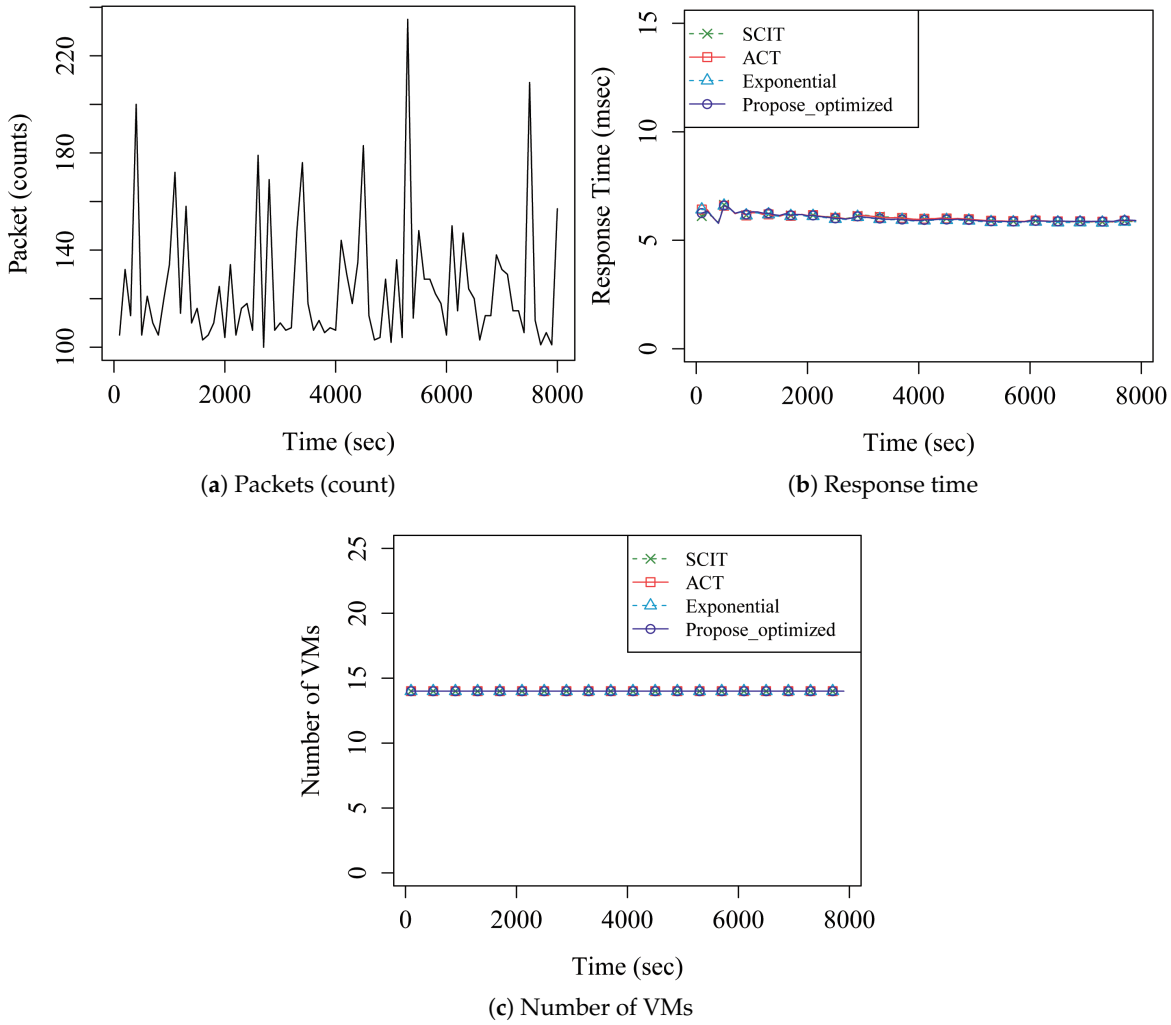
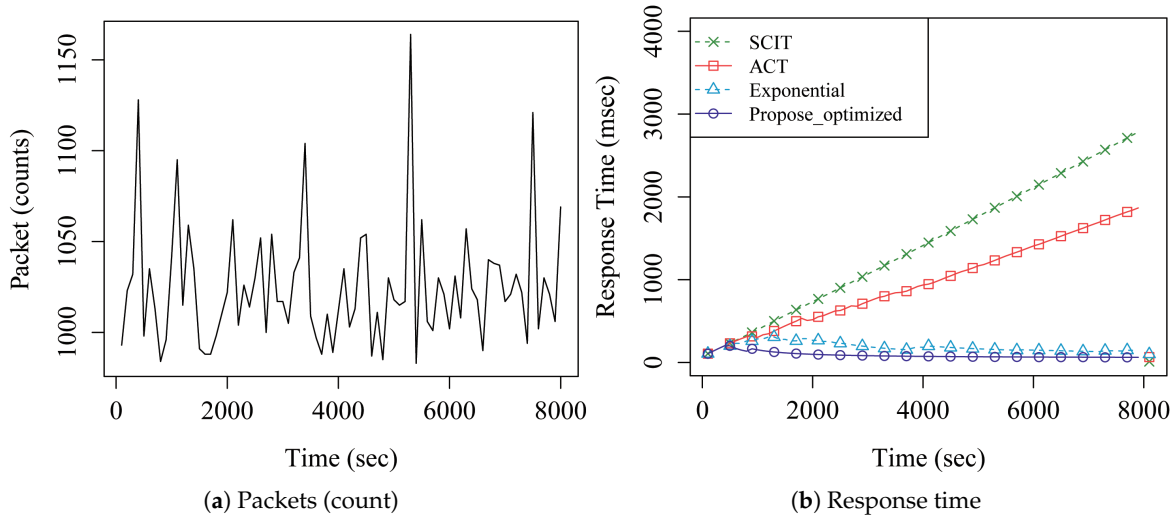
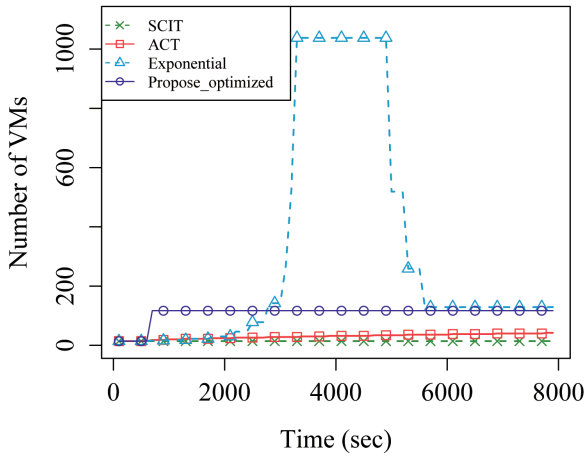


Figure 3. Scenario 1: normal packets. ACT: adaptive cluster transformation. (**a**) packets (count); (**b**) response time; (**c**) number of VMs.

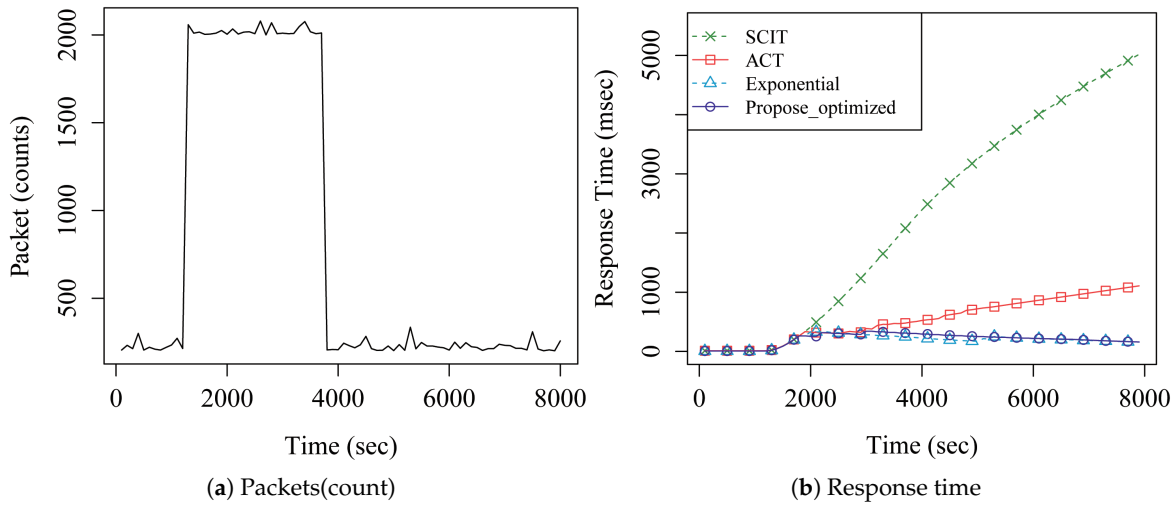


(**a**) Packets (count) (**b**) Response time



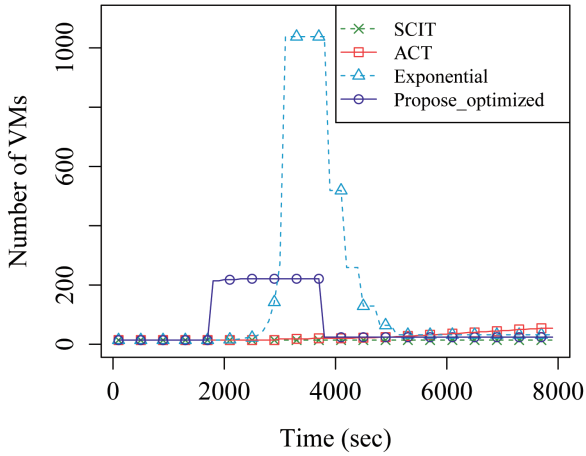
(**c**) Number of VMs

Figure 4. Scenario 2: continuous denial of service (DoS) packets. (**a**) packets (count); (**b**) response time; (**c**) number of VMs.



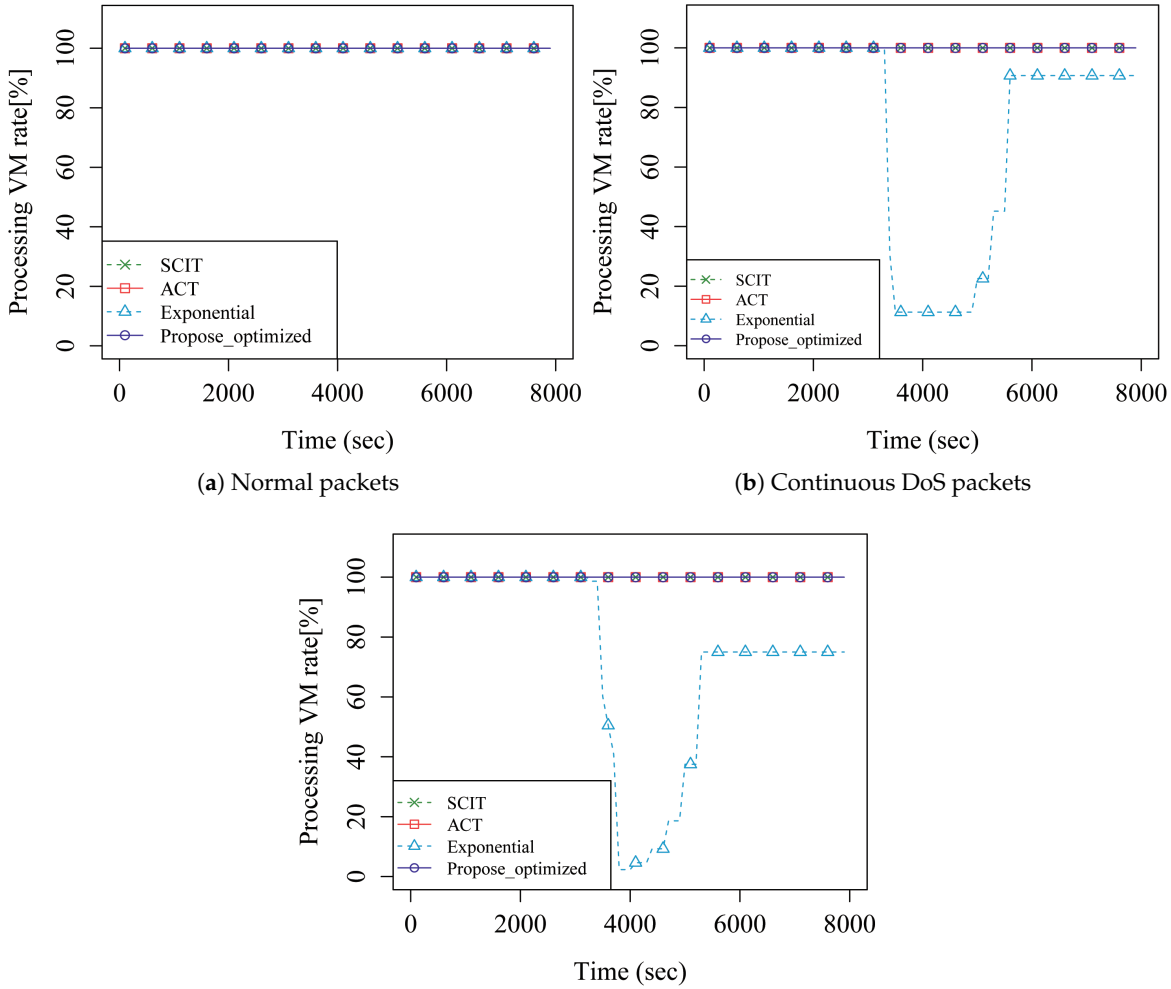
(**a**) Packets(count) (**b**) Response time

Figure 5. *Cont.*



(**c**) Number of VMs

Figure 5. Scenario 3: one-pulse DoS packets. (**a**) packets (count); (**b**) response time; (**c**) number of VMs.



(**a**) Normal packets

(**b**) Continuous DoS packets

(**c**) One-pulse DoS packets

Figure 6. VM processing rate in three scenarios. (**a**) normal packets; (**b**) continuous DoS packets; (**c**) one-pulse packets.

Table 2 lists average response time, average VM processing rate, and total VM running time under three different scenarios. When there is no DoS attack, the average response times for four different systems are the same. However, under a DoS attack, the average response times for the SCIT and ACT schemes increase significantly, while the average response times for the exponential-ACT and our proposed scheme is reasonably small compared to the response times for the SCIT and ACT schemes. Although the response time performance of the exponential-ACT is similar to that of our scheme, the average VM processing rate of the exponential-ACT is much smaller than it is in our proposed scheme, because of the waste of resources. Therefore, we show that our proposed scheme can provide good QoS under a massive DoS attack without wasting resources by properly expanding cluster size.

Table 2. Total average response time and VM processing rate and total VM running time in three scenarios. DoS: denial of service, SCIT: self-cleansing intrusion tolerance, ACT: adaptive cluster transformation.

Description		Normal Packet	Continuous DoS	One-Pulse DoS
Total average response time (ms)	SCIT	7.41	1406.36	1810.25
	ACT	7.41	955.95	529.78
	Exponential	7.41	186.61	197.19
	Proposed	7.41	87.41	200.98
Total average VM processing rate (%)	SCIT	100	100	100
	ACT	100	100	100
	Exponential	100	74.42	73.29
	Proposed	100	100	100
Total VMs running time (s)	SCIT	1106	1106	1106
	ACT	1106	2374	2002
	Exponential	1106	25,082	12,699
	Proposed	1106	8625	5499

5. Discussion

In this section, we discuss several considerations in designing our proposed ITS. First, it is necessary to determine the threshold value at which a cluster size expansion or reduction process will be performed, because overall system performance can vary under different threshold values. Additionally, we need to account for the user-requested service level in order to adjust the threshold value. For example, when the user requested service level is high, the threshold value can be adjusted to a lower value in order to provide stable service. However, when the user-requested service level is low, the higher threshold value can be used to provide less stable service. Second, the total number of available VMs and hosts must be sufficient to protect the system from a massive DoS attack. Even if we use an optimal cluster expansion scheme, system QoS cannot be guaranteed when a limited number of VMs are available to support a massive DoS attack. Thus, our proposed scheme is appropriate for cloud environment scenarios where the number of VMs is not limited. Third, whenever a VM is created by the expansion process, it takes time for the VM to be able to provide actual services. This VM waiting time is not negligible. Therefore, it would be interesting to study how to minimize this VM waiting time. Estimating the cluster expansion timing is important, because it then becomes possible to prepare available VMs in advance and begin providing services whenever needed. Analyzing packet history and patterns in an ITS using machine learning techniques can also be applied to the estimation of the cluster expansion. For example, when we analyze the daily packet patterns by using machine learning, we can easily observe that the packet amount during daytime is significantly different from the nighttime packet amount. Thus, it is easy to detect a DoS attack when the packet amount is increasing during the nighttime. Fourth, an appropriate application service for the proposed ITS is a Web service or Domain Name System (DNS) service, because those services have short-term sessions. In contrast, a database service that does not have a short-term session will require more time whenever

a cluster expansion or reduction process is performed. Fifth, the proposed ITS must cooperate with the pre-existing IDS and IPS. It is possible that the ITS can protect a system from unknown attacks that the IDS and IPS cannot detect. Then, the ITS will inform the IDS and IPS of the unknown attack information in order to prevent future attacks. Sixth, for security purposes, the RCC that controls the overall system must be physically unreachable from outside of the system. If an attacker can reach the RCC, the complete ITS system can be broken down. Therefore, we believe that the proposed ITS system will provide good QoS to Web-based cloud services, by taking into account the multiple considerations mentioned above.

6. Conclusions and Future Works

In this study, we proposed an optimal cluster expansion-based ITS in order to provide stable and reliable services under massive DoS attacks. We took queuing theory into account for incoming packet numbers in order to determine the optimal number of increased or decreased VMs. Moreover, our proposed scheme increases system performance by minimizing resource waste while preserving good QoS. Our simulation results show that the proposed ITS scheme outperforms conventional ITS schemes such as SCITs, ACTs, and exponential-ACTs. Our future work will include the development of a new cluster expansion scheme that can control the rotation scheduling of VMs under a massive attack. For example, when the rotation of VM is rapid, system security will be enhanced because of the short exposure time of the online VMs. However, the VM rotation overhead will increase, and more resources will be required for the fast VM rotation. Therefore, we aim to develop an optimal scheme for maintaining appropriate VM rotation and system performance, while minimizing resource waste. We will examine how to improve system performance and security by optimally controlling the VM rotation schedule, and we will also investigate different types of attack models.

Acknowledgments: This work was supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIT) (No.2017-0-00380, Development of next generation user authentication and No. 2016-0-00173, Security Technologies for Financial Fraud Prevention on Fintech).

Author Contributions: Hyun Kwon designed the entire architecture and performed the experiments. Yongchul Kim, Hyunsoo Yoon and Daeseon Choi gave some advice and ideas about the article. All authors analyzed the result of this article. Hyun Kwon and Yongchul Kim wrote the article. All authors discussed the contents of the manuscript. Daeseon Choi supervised the whole process as a corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Kenkre, P.S.; Pai, A.; Colaco, L. Real time intrusion detection and prevention system. In *Proceedings of the 3rd International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA) 2014*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 405–411.
2. Patel, A.; Taghavi, M.; Bakhtiyari, K.; JúNior, J.C. An intrusion detection and prevention system in cloud computing: A systematic review. *J. Netw. Comput. Appl.* **2013**, *36*, 25–41.
3. Veríssimo, P.E.; Neves, N.F.; Correia, M.P. Intrusion-tolerant architectures: Concepts and design. In *Architecting Dependable Systems*; Springer: Berlin/Heidelberg, Germany, 2003; pp. 3–36.
4. Heo, S.; Kim, P.; Shin, Y.; Lim, J.; Koo, D.; Kim, Y.; Kwon, O.; Yoon, H. A Survey on Intrusion-Tolerant System. *J. Comput. Sci. Eng.* **2013**, *7*, 242–250.
5. Guo, M.; Bhattacharya, P. Diverse virtual replicas for improving intrusion tolerance in cloud. In Proceedings of the 9th Annual Cyber and Information Security Research Conference, Oak Ridge, TN, USA, 8–10 April 2014; ACM: New York, NY, USA, 2014; pp. 41–44.
6. Smith, M.; Schridde, C.; Freisleben, B. Securing stateful grid servers through virtual server rotation. In Proceedings of the 17th International Symposium on High Performance Distributed Computing, Boston, MA, USA, 23–27 June 2008; ACM: New York, NY, USA, 2008; pp. 11–22.
7. Sousa, P.; Bessani, A.N.; Correia, M.; Neves, N.F.; Verissimo, P. Highly available intrusion-tolerant services with proactive-reactive recovery. *IEEE Trans. Parallel Distrib. Syst.* **2010**, *21*, 452–465.

8. Huang, Y.; Sood, A. Self-cleansing systems for intrusion containment. In Proceedings of the Workshop on Self-Healing, Adaptive, and Self-Managed Systems (SHAMAN), New York, NY, USA, 23 June 2002.
9. Huang, Y.; Arsenault, D.; Sood, A. Closing cluster attack windows through server redundancy and rotations. In Proceedings of the IEEE Sixth International Symposium on Cluster Computing and the Grid, Singapore, 16–19 May 2006; Volume 2, p. 12.
10. Hanczewski, S.; Stasiak, M.; Weissenberg, J.; Zwierzykowski, P. Queuing model of the access system in the packet network. In *International Conference on Computer Networks*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 283–293.
11. Saidane, A.; Nicomette, V.; Deswarte, Y. The design of a generic intrusion-tolerant architecture for web servers. *IEEE Trans. Dependable Secur. Comput.* **2009**, *6*, 45–58.
12. Heo, S.; Lee, S.; Doo, S.; Yoon, H. Design of a secure system considering quality of service. *Symmetry* **2014**, *6*, 938–953.
13. Heo, S.; Lim, J.; Lee, M.; Lee, S.; Yoon, H. A novel intrusion tolerant system based on adaptive recovery scheme (ARS). In *IT Convergence and Security 2012*; Springer: Berlin/Heidelberg, Germany, 2013; pp. 71–78.
14. Lim, J.; Kim, Y.; Koo, D.; Lee, S.; Doo, S.; Yoon, H. A novel adaptive cluster transformation (ACT)-based intrusion tolerant architecture for hybrid information technology. *J. Supercomput.* **2013**, *66*, 918–935.
15. Yongjoo, S.; Sihu, S.; Yunho, L.; Hyunsoo, Y. A novel intrusion tolerant system using live migration. *IEICE Trans. Inf. Syst.* **2014**, *97*, 984–988.
16. Bumsoon, J.; Seokjoo, D.; Soojin, L.; Hyunsoo, Y. Hybrid Recovery-Based Intrusion Tolerant System for Practical Cyber-Defense. *IEICE Trans. Inf. Syst.* **2016**, *99*, 1081–1091.
17. Alhamad, M.; Dillon, T.; Wu, C.; Chang, E. Response time for cloud computing providers. In Proceedings of the 12th International Conference on Information Integration and Web-Based Applications & Services, Paris, France, 8–10 November 2010; ACM: New York, NY, USA, 2010; pp. 603–606.
18. Calheiros, R.N.; Ranjan, R.; Beloglazov, A.; De Rose, C.A.; Buyya, R. CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Softw. Pract. Exp.* **2011**, *41*, 23–50.