EDoS Attack Defense Shell (EDoS-ADS): An Enhanced Mitigation Technique Against Economic Denial of Sustainability (EDoS) Attacks for Controlling the Access to Cloud Resources

BY

AHMAD IBRAHIM SHAWAHNA

A Thesis Presented to the
DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

COMPUTER ENGINEERING

APRIL, 2016

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN- 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

This thesis, written by **AHMAD IBRAHIM SHAWAHNA** under the direction of his thesis advisor and approved by his thesis committee, has been presented and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE IN COMPUTER ENGINEERING.**

Marwan

Dr. Marwan H. Abu-Amara
(Advisor)

Dr. Ahmad Almulhem
Department Chairman

Ashraf

Dr. Ashraf S. Mahmoud
(Member)

Dr. Salam A. Zummo
Dean of Graduate Studies

عمادة الدراسات العليا
DEANSHIP OF GRADUATE STUDIES

Dr. Yahya E. Osais
(Member)

12/5/16

Date

To my loving

I dedicate this thesis to my parents, my wife, my brothers, and sisters.

Thank you for supporting me along the way.

Without your help, I could not have completed this work.

ACKNOWLEDGMENTS

I thank Allah for granting me the guidance, patience, health, and determination to successfully accomplish this work.

I would like to express my deep appreciation to my research supervisor, **Dr. Marwan H. Abu-Amara**, for guidance me throughout the research. I also would like to thank the members of my thesis committee, **Dr. Ashraf S. Mahmoud**, and **Dr. Yahya E. Osais**, for their help in accomplishing this work. Thanks also to **Mr. Mohammed Y. Alkaff** for giving me nice comments.

I would also like to thank the King Fahd University of Petroleum and Minerals for creating the research environment for the students and for supporting them until they accomplish their objectives.

I also would like to thank my parents, and my wife who always support me with their love, patience, encouragement, and constant prayers. I would like to thank my brothers and sisters for their love and support throughout my study.

Finally, I would like to say thanks to all my friends, mostly those in the King Fahd University of Petroleum and Minerals and all others who helped in this work.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

EDoS	:	Economic Denial of Sustainability
EDoS-ADS	:	EDoS Attack Defense Shell
DoS	:	Denial of Service
DDoS	:	Distributed Denial of Service
EDDoS	:	Economic Distributed Denial of Sustainability
GTT	:	Graphics Turing Test
URL	:	Uniform Resource Locator
TF	:	Trust Factor
UTF	:	User Trust Factor
TRPM	:	Total Requests Per Minute
CRPS	:	Concurrent Requests Per Second
MRPS	:	Maximum Requests Per Second
TRC	:	Total Request Count
MRC	:	Malicious Request Count
LB	:	Load Balancer
DB	:	DataBase

NAT	:	Network Address Translation
IaaS	:	Infrastructure as a Service
PaaS	:	Platform as a Service
SaaS	:	Software as a Service
VM	:	Virtual Machine
EC2	:	Elastic Compute Cloud
IDC	:	International Data Corporation
sPoW	:	Self-verifying Proof of Work
vFirewall	:	virtual Firewall
V-Node	:	Verifier cloud Node
TTL	:	Time to Live
BYOD	:	Bring Your Own Device
IAM	:	Identity and Access Management
DDoS-MS	:	DDoS-Mitigation System
IPS	:	Intrusion Prevention System
RP	:	Reverse Proxy
IPA-Defender	:	Index Page Attack Defender

TSP	:	Time Spent on a web Page
MAD	:	Mean Absolute Deviation
FCMDPF	:	Flexible, Collaborative, Multilayer, DDoS Prevention Framework
OB	:	Outer Blocking
STBOA	:	Service Trace Back Oriented Architecture
FAEB	:	Flexible Advanced Entropy Based
HR-DDoS	:	High Rate DDoS
FC	:	Flash Crowd
ML	:	Machine Learning
JRE	:	Java Runtime Environment
TCP	:	Transmission Control Protocol
CAPTCHA	:	Completely Automated Public Turing test to tell Computers and Humans Apart
Req/sec	:	Request per second
APT	:	Attack Period Time
PMMLCG	:	Prime Modulus Multiplicative Linear Congruential Generator

HTTP	:	Hypertext Transfer Protocol
SLAs	:	Service Level Agreements
AWS	:	Amazon Web Services
ELB	:	Elastic Load Balancer
REP	:	Relative Error Percentage
LR-EDoS	:	Low Rate EDoS

ABSTRACT

Full Name	:	Ahmad Ibrahim Shawahna
Thesis Title	:	EDoS Attack Defense Shell (EDoS-ADS): An Enhanced Mitigation Technique Against Economic Denial of Sustainability (EDoS) Attacks for Controlling the Access to Cloud Resources
Major Field	:	Computer Engineering
Date of Degree	:	April, 2016

Cloud computing has become one of the most promising technologies for the future of IT industry due to its benefits for business. Many organizations are willing to employ this technology for hosting their services in order to achieve economies of scale, reduce spending on technology infrastructure, streamline processes, reduce capital costs, improve the performance, availability, accessibility, and flexibility. However, the level of security provided by a cloud computing service model has been specified as the biggest challenge facing the cloud services providers and a major concern for the cloud services customers. Cloud computing has attractive features such as elasticity, auto scaling, and utility computing. These features could help the adopters to minimize their operating costs and maximize resource utilization. However, if the attackers take the advantage of these features and launch a Distributed Denial of Service (DDoS) attack on the cloud computing resources, DDoS attack will be diverted to a new strain of attack called Economic Denial of Sustainability (EDoS) attack. An EDoS attack occurs when attack machines send a huge amount of service requests to the cloud computing servers, exploiting the elasticity and auto scaling features of the cloud, to charge a cloud adopter's bill an exorbitant extra amount of costs due to the pay per use model of the cloud, leading to large scale service withdraw or bankruptcy.

In this work, we study several existing mitigation techniques for the EDoS attack and state their major drawbacks. Then, a new reactive approach, implemented at the cloud provider’s end, is proposed to mitigate such attacks taking into account most of the drawbacks of the existing mitigation techniques. Through the proposed technique, limited access permission for cloud services is granted to each user based on different factors such as Graphics Turing Test (GTT), Uniform Resource Locator (URL) redirection technique, Trust Factor (TF), and Maximum Requests Per Second (MRPS). Initially, the proposed technique will monitor the auto scaling feature and the auto scaling thresholds to detect if there is an EDoS attack. Once an attack behavior is detected, the cloud service will trigger a checking component for differentiating between legitimate users and automated attackers (Zombies). Subsequently, the traffic or requests generated by an automated attacker will be dropped while the legitimate users’ requests will be directed to the cloud servers. The proposed approach has the ability to identify the legitimacy of clients behind a Network Address Translation (NAT) router and avoid blocking an entire NAT-based network that may hosts legitimate clients from accessing the cloud servers. The effectiveness of the proposed mitigation technique is evaluated using CloudSim simulator. In addition, we conduct a comparison between our new approach and the EDoS-Shield technique. The simulation results show that the proposed technique successfully differentiates the legitimate and attacker clients. Moreover, the proposed technique outperforms the existing techniques especially when those clients belong to the same NAT-based network. In addition, the results show that the EDoS attacks will not force auto scaling of the cloud service when implementing the proposed technique at the cloud service provider end.

ملخص الرسالة

الاسم الكامل : أحمد ابراهيم شواهنة

عنوان الرسالة : قوقعة الدفاع ضد هجمات الحرمان الاقتصادي للاستدامة: تقنية محسنة لتخفيف هجمات الحرمان الاقتصادي للاستدامة في الحوسبة السحابية والتحكم في الوصول إلى موارد السحابة

التخصص : هندسة الحاسوب

تاريخ الدرجة العلمية : أبريل ، 2016

الحوسبة السحابية هي واحدة من التقنيات الواعدة لمستقبل صناعة تكنولوجيا المعلومات نظراً لفوائدها الجمة للأعمال التجارية. تتجه العديد من المنظمات لتوظيف هذه التكنولوجيا لاستضافة خدماتها من أجل تخفيض تكاليف التوسع ، والحد من الإنفاق على البنية التحتية للتكنولوجيا ، وتبسيط العمليات ، وخفض تكاليف رأس المال ، وتحسين الأداء ، وضمان توافر الخدمات وسهولة الحصول عليها ، والمرونة أيضاً. ومع ذلك يعتبر مستوى الأمن الذي يوفره نموذج خدمة الحوسبة السحابية واحد من أكبر التحديات التي تواجه مقدمي الخدمات السحابية ومصدر قلق كبير لعملاء الخدمات السحابية.

للحوسبة السحابية العديد من الملامح الجذابة مثل : وفرة الموارد ، وقابلية التوسع التلقائي ، والدفع لمزود الخدمة بحسب استخدام الموارد. هذه الميزات يمكن أن تساعد عملاء الحوسبة السحابية على تقليل تكاليف التشغيل وتعظيم الاستفادة من الموارد. لكن إذا استغل المهاجم هذه الميزات واطلق هجوم الحرمان من الخدمة الموزعة (DDoS) على بيئة الحوسبة السحابية ؛ فإنه سيتم تحويل هذا الهجوم الى نوع جديدة من الهجمات ، يطلق عليه هجوم الحرمان الاقتصادي للاستدامة (EDoS).

يحدث هجوم الحرمان الاقتصادي للاستدامة عندما ترسل آلياً كميات كبيرة من طلبات الخدمة نحو الحوسبة السحابية من قبل أجهزة مخترقة وموجهة من المهاجم. الأمر الذي ينجم عنه نمو كبير في الموارد لإستعاب وخدمة طلبات المهاجم ، وبوجود آلية الدفع حسب الاستخدام في نموذج الحوسبة السحابية ؛ فإن العميل سيضطر الى دفع كل التكاليف الناتجة عن الهجوم ، والتي قد تصل الى درجة لن يقدر بعدها على الاستمرار اقتصاديا ؛ بسبب الخسارة الكبيرة التي تؤدي إلى الإفلاس.

يهدف هذا البحث إلى دراسة تقنيات التخفيف القائمة حالياً التي تمنع أو تخفف من هجوم الحرمان الاقتصادي للاستدامة ، وبيان العيوب الرئيسية الموجودة في كل منها. وبعد ذلك سوف نقدم تقنية جديدة ؛ لتخفيف ومنع هجوم الحرمان الاقتصادي للاستدامة ، مع الأخذ بعين الاعتبار معظم العيوب الموجودة في تقنيات التخفيف القائمة ، من خلال التقنية المقترحة. سوف يتم منح إذن الوصول الى موارد الخدمات السحابية لكل مستخدم بناءً على عوامل مختلفة مثل : اجتياز اختبارات الرسم تورينج (GTTs) ، وتقنية إعادة توجيه محدد موقع المعلومات (URL) ، وعامل الثقة للمستخدم (TF) ، والطلبات المتزامنة المسموح في الثانية (MRPS).

في البداية ، سوف تقوم التقنية المقترحة بمراقبة ميزة التوسع التلقائي ، وعتبات التوسع التلقائي لموارد الخدمة السحابية ؛ للكشف عما إذا كان هناك وجود لهجوم الحرمان الاقتصادي للاستدامة. بمجرد اكتشاف الهجوم ، فإن الخدمة السحابية تحيل فوراً جميع الطلبات القادمة الى عنصر الفحص من أجل التحقق من شرعية المستخدمين والتمييز بين المستخدمين الشرعيين والمهاجمين الآليين (Zombies). في وقت لاحق ، سيقوم عنصر الفحص بإسقاط حركة المرور أو الطلبات الناتجة عن المهاجمين الآليين ، في حين سيتم توجيه طلبات المستخدمين الشرعيين إلى خوادم الحوسبة السحابية. المنهج المقترح لديه القدرة على تحديد شرعية المستخدمين حتى وإن كانوا خلف جهاز توجيه ترجمة عنوان الشبكة (NAT). ولديه القدرة كذلك على تجنب حجب الشبكة المعتمدة على ترجمة عنوان الشبكة بشكل كامل والتي قد تستضيف المستخدمين الشرعيين. وبالتالي تمكينهم من الوصول إلى الخوادم السحابية.

سوف يتم تقييم فعالية تقنية التخفيف المقترحة باستخدام محاكاة (CloudSim) ، وبالإضافة إلى ذلك نحن نخطط لإجراء موازنة بين نهجنا الجديد وتقنية درع هجوم الحرمان الاقتصادي للاستدامة (EDoS-Shield) ؛ لتقييم أسلوبنا المقترح.

أظهرت نتائج المحاكاة أن التقنية المقترحة تميز بنجاح بين المستخدم الشرعي والمهاجم الآلي. علاوة على ذلك ؛ فإن التقنية المقترحة تتفوق على التقنيات الموجودة حالياً وخصوصاً عندما ينتمي هؤلاء المستخدمين الى نفس الشبكة المعتمدة على ترجمة عنوان الشبكة. إضافة إلى ذلك ، فقد بينت النتائج أن هجمات الحرمان الاقتصادي للاستدامة لن تجبر الحوسبة السحابية على التوسع التلقائي عند تنفيذ التقنية المقترحة لدى طرف مزود الخدمات السحابية.

CHAPTER 1

INTRODUCTION

Cloud computing is a new pattern of computing in which on-demand network access to the computing resources, utility-based pricing model, and dynamic resource assignment are granted as a service over the Internet [1, 2]. The cloud resources can be rapidly provisioned and freed almost without any interaction and low management effort by the service provider. Cloud computing technology has been identified as the topmost technology with the potential for significant impact on organizations in 2011 [3]. Moreover, Gartner technology research company in the US has highlighted cloud computing as one of the top 10 strategic technologies group that companies should look to make deliberate decisions about them during 2015 and the next two years [4].

Cloud computing has provided a novel paradigm to address the critical need for affordable and convenient resource availability to meet the large scale computing demands of contemporary applications. The cloud paradigm provides various features that are different from traditional network such as on-demand network access, dynamic resource assignment, shared resource pooling, service oriented, multi-tenancy, self-organizing, and pay for use pricing model.

Pay per Use is the base model in cloud computing wherein vendors and service providers do not have to procure and maintain hardware and software resources to sustain their respective computing activity. Rather, cloud services are purchased on a need basis. The cloud service provider will automatically scale the cloud resources when the user’s traffic increases and accordingly the cost on the consumer also increases.

Auto scaling model may put the providers of cloud application at risk by skillful attackers who intend to increase the cost for a consumer. For example, the attackers synthetically generate a huge traffic of HTTP requests to the cloud web application exploiting the elasticity and auto scaling features on the cloud to charge a cloud adopters bill an exorbitant extra amount of costs due to the pay for use model. These requests are not profitable for the attackers but they are accounted for billing the customers. Thus, such an attack, technically labeled as Economic Denial of Sustainability (EDoS) attack, will cause a sustainable descent for the consumers’ economy.

In this work, we study several existing mitigation techniques that prevent or mitigate the EDoS attack and state all drawbacks of these techniques. Then, a new approach is proposed to mitigate the EDoS attack. This new mitigation technique takes into account most of the drawbacks of the existing mitigation techniques.

1.1 Background and Terminology

Cloud computing is based on some existing technologies such as virtualization, utility computing, grid computing, and automatic computing. In particular, the virtualization technology mainly forms the foundation of cloud computing technology [5]. Virtualization is the layer responsible to pool the computing resources from the available hardware by allocating and reallocating virtual resources based on demand.

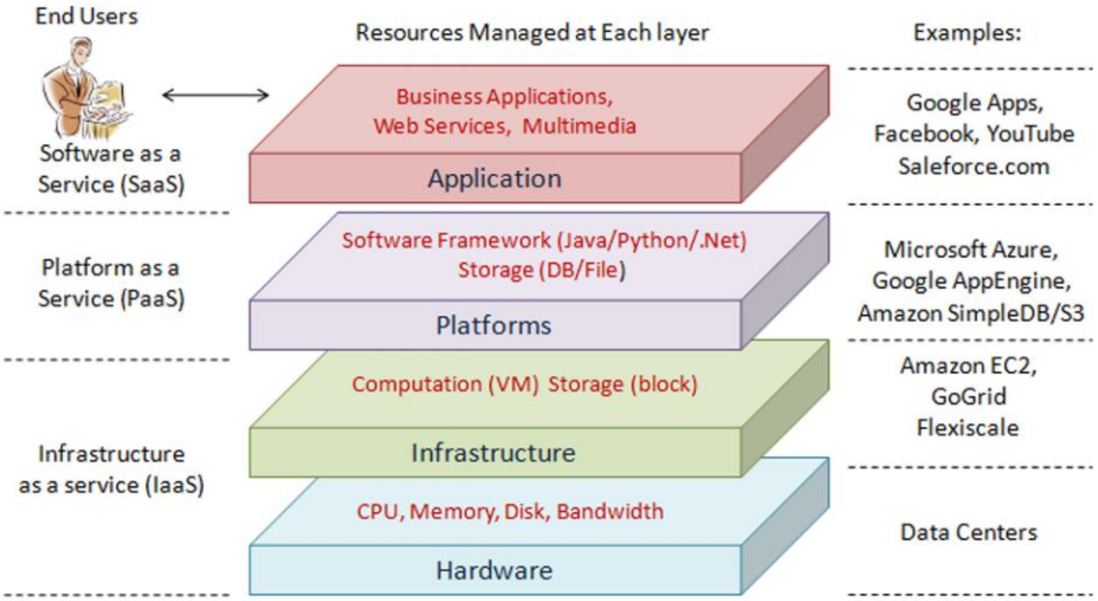


Figure 1 Cloud Computing Architecture [5]

The cloud computing services can be classified based on the services delivered to the end users into the Infrastructure offered as a Service (IaaS), Platform offered as a Service (PaaS), and Software offered as a Service (SaaS), as shown in Figure 1. IaaS refers to providing storage and compute capabilities as services, usually through spawning of Virtual Machines (VMs) based on the service providers demands. Amazon Elastic Compute Cloud (EC2) [6] is an example suite that is built on the IaaS service model. PaaS provides a layer of software, such as operating system, as a service for leasing out to the service providers in order to sustain design and deployment of application layer services. Google App Engine [7] is an example of a PaaS service model. SaaS refers to provisioning of software based on the end user demand as a service over the Internet. Saleseforce.com [8] is an example of a SaaS service model.

There are four types of cloud computing models based on its location. These types are private, public, hybrid, and community clouds [5, 9, 10]. A public cloud is a cloud where the infrastructure layer is offered by a third party and shared between customers. In addition, the client has no control over the data, network, and security settings. A private cloud can be of two types, the first one is on premise private cloud in which the infrastructure layer is devoted to a specific organization and not common with others. In addition, the organization has full control over the data, network, and security settings. The second type of private cloud is an externally hosted private cloud, virtual private cloud, in which the infrastructure layer is used by one organization but offered by a third party. The virtual private cloud is less costly than the on premise private cloud. A hybrid cloud is defined as a blend of private and public clouds where the customers could host critical applications on their private cloud and non-critical applications on the public cloud. Also, this type of cloud computing is used for the infrastructure layer. For example, private computing infrastructure could be used for daily and normal activity. However, if there is a high activity in the network such as flash crowd effect, then the computing infrastructure could be expanded by renting more resources from the public cloud infrastructure. Then, these resources may be released when not needed or the activity returns to normal. Finally, a community cloud is used to share the infrastructure layer between organizations or companies of the same community.

According to a recent survey under the supervision of the International Data Corporation (IDC) [11], security has been positioned as the major challenge of cloud computing model. Nearly 87% of IT executives reported the cloud security as the principal challenge prohibiting the adoption of the cloud computing services model, as shown in Figure 2.

Security anxieties have driven organizations to hesitate in moving their basic assets and resources to the cloud. Companies and individuals are generally worried about how security, consistency, and trustworthiness can be kept up in this new environment. Also, adoption of public or community cloud environments for hosting critical applications and sensitive data will exacerbate anxiety for corporations since their datacenters network boundary protection is not on hand. With security being one of the top concerns that hinders cloud computing environment [12-16], it has become a major field of study.

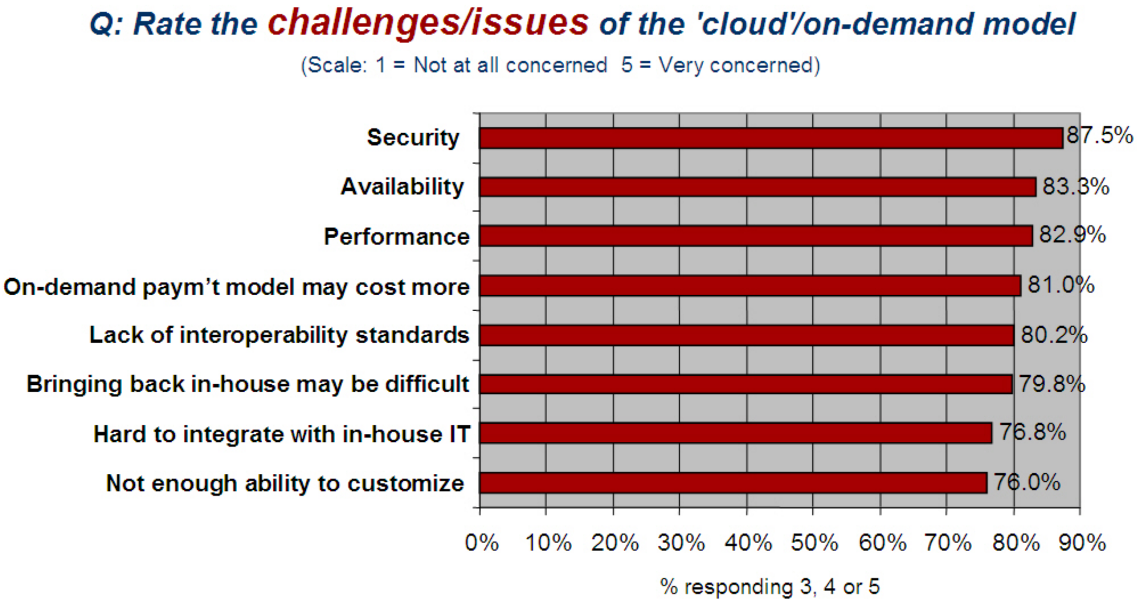


Figure 2 Cloud Challenges Survey [11]

The aspect of cloud computing security is wide and general. Therefore, it is imperative to introduce two types of network security threats, Denial of Service (DoS) and Distributed Denial of Service (DDoS). The DoS and the DDoS attacks overwhelm a network infrastructure or service by employing a distributed number of malicious or infected machines to perform unwanted operations intended to cause damage to the IT infrastructure of an organization. For example, a botnet that is made up of a collection of malicious machines participating in an attack can be activated to overwhelm a web server using an asynchronous DDoS attack. Such an attack makes the site unavailable to end users due to an exhaustion of its computing or network resources [17]. The next subsection explains the effect of DDoS on the cloud computing service.

1.2 Economic Denial of Sustainability (EDoS)

The cloud computing model permits the customers to scale their resources in size and availability. Note that the customers of the cloud computing model are charged depending on the pay as you go premise of the cloud's instances and network resources, otherwise known as the utility computing.

Such a service model may appear to overcome the effects of a DDoS attack where the resource bottlenecks are eliminated. However, this model merely transforms the traditional DDoS attack to a new strain of attacks that target the cloud customer’s economic resource to charge their bill an exorbitant extra amount of costs, originally labeled as Economic Denial of Sustainability (EDoS) attack [18].

The EDoS attack takes advantage of some attractive features in the cloud computing environment such as elasticity, auto scaling, and pay per use model. Auto scaling is one of the attractive features of a cloud computing environment, this feature allocates automatically more instances or resources to handle the high load (scale up) and release automatically these resources when the load or traffic returns back to normal (scale down). The auto scaling could be activated by monitoring some parameters such as resources utilization, memory usage, response time, and network bandwidth.

The EDoS attack is a major threat in the cloud computing environment. EDoS occurs when zombie machines send a large amount of undesired traffic towards the cloud computing system [19]. This attack is not only causing the service to be unavailable or down but also it is causing a tremendous economic loss to the cloud customer.

The main source of the EDoS attack is the DDoS attack targeting the cloud resources. Because of the elasticity and auto scaling features, the resources will grow according to the demand of the DDoS attack, and due to the pay as you go model of the cloud the customers will be charged for the scaling of resources until it reaches a point that it cannot sustain economically [20]. Hence, unlike a DDoS attack that could prevent the legitimate users from accessing the service for a certain amount of time, an EDoS attack could prevent the service provider from delivering the service forever if the attack leads to bankruptcy.

1.3 Problem Statement

Cloud computing provides a model to reduce the customers costs related to under-provisioning, over-provisioning, and under-utilization by moving the organization resources to the cloud system [21]. Moreover, it reduces the provisioning time of resources for scaling up or down the applications servers when the traffic changes.

However, the scalability feature in the cloud may cause economy loss or bankruptcy when the attackers target the cloud service with a huge number of requests.

Economic Denial of Sustainability (EDoS) attack is considered as one of the security concerns that have hindered the migration of many organizations to the cloud technology. This is because an EDoS attack targets the financial constraints of the service provider. It exploits the elasticity feature of the cloud by increasing the resources usage which in turn they will scale up to accommodate the demand. As a consequence of pay for use model of the cloud, service provider will be charged for the attackers activities. Ultimately, the economic viability of the service provider becomes unsustainable.

Figure 3 illustrates the idea of the EDoS attack where the attackers launch a high number of requests to the cloud computing services, and subsequently the servers scale up by the auto scaling feature to handle this high traffic. Then, the customer has to pay for all duplicated servers due to the pay as you go model. Accordingly, the cost for the customer increases and economic loss occurs.

For these reasons, a mitigation technique to identify suspicious service requests that target the service provider’s financial resources must be present. Therefore, we propose a technique to mitigate the effects of the EDoS attack through controlling the usage of resources.

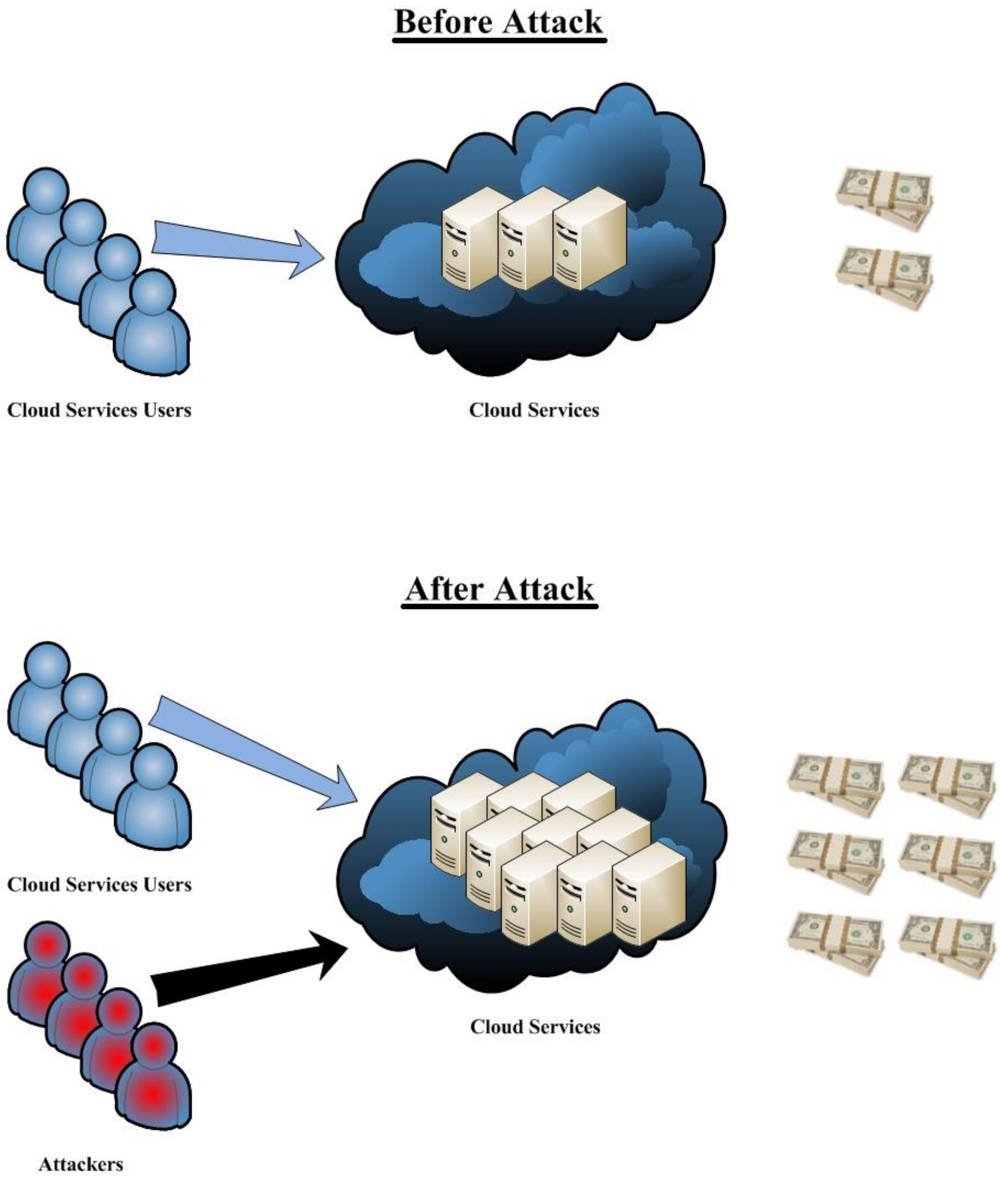


Figure 3 Effect of an EDoS Attack against the Cloud

1.4 Motivation

According to a survey conducted by International Data Corporation (IDC) [11], the issue of security has been rated as the greatest challenge of cloud computing system. In particular, malicious attacks against the cloud, such as EDoS attacks, have remained largely unaddressed in the literature.

The disruption of any service provisioned through the cloud can cause large scale damage to the end users comprising both novice private end users applications and sophisticated business applications. Users of the cloud pay as they use based on application needs. Elasticity of resource availability at the service provider is the key driving aspect for the growing popularity of the cloud paradigm. Hence, the ability of the service provider to differentiate between legitimate and malicious users is critical for smooth and affordable resource usage at the cloud computing system. However, a comprehensive mechanism for identifying routine cloud usage activity and distinguishing it from malicious activity is largely unaddressed in the literature.

The Economic Denial of Sustainability (EDoS) attack is a serious threat to the cloud computing environment due to its impact on the economic side. We have explored the existing mitigation techniques for the EDoS attack. These techniques have a lot of weaknesses such as blocking an entire Network Address Translation (NAT) based-network due to an EDoS attacker that belongs to the NAT-based network. Thus, legitimate clients belonging to the blocked NAT-based network will not have the ability to access the cloud resources. Accordingly, this work attempts to rectify such a weakness in the existing EDoS mitigation techniques with minimal impact on both the cloud service customers and the legitimate clients.

1.5 Objective

The objective of this research is to propose an efficient mitigation technique against EDoS attack in the cloud infrastructure. The proposed technique should be able to detect and prevent such an attack with a high degree of accuracy while allowing the legitimate clients access to the cloud services and use its resources.

The effectiveness of the proposed mitigation technique will be evaluated through simulations. In addition, we will conduct a comparison between the proposed technique and other techniques under different scenarios.

1.6 Contribution

The main contributions of this research are as follows:

- We summarize almost all of research work found in the literature for addressing EDoS attack.
- We propose a novel and reactive approach to detect and mitigate the EDoS attack in cloud systems with low cost and overhead.

- Our scheme is able to detect malicious users using the Graphics Turing Test (GTT), Uniform Resource Locator (URL) redirection technique, Trust Factor (TF), and Maximum Requests Per Second (MRPS).
- Our proposed scheme tracks the users' behavior and avoids the false positives based on the user TF.

1.7 Thesis Organization

The rest of the thesis is organized as follows. Chapter 2 provides an extensive literature review on EDoS attack mitigation techniques and the methods used for identifying the clients that belong to a NAT-based network. We provide an elaborate explanation of our proposed scheme in Chapter 3. In Chapter 4, we discuss the simulator implementation used to evaluate the proposed mitigation technique. Experimental results and their analysis are presented in Chapter 5. Finally, Chapter 6 includes the conclusions and future directions for our work.

CHAPTER 2

LITERATURE REVIEW

Cloud computing security is a critical issue that concerns many enterprises that are considering moving their services to the cloud. Subsequently, many researchers have focused on the Economic Denial of Sustainability (EDoS) attack due to its severe impact on the cloud service customer's bill. In this section, we present a detailed literature review on EDoS attack mitigation techniques and the methods used to identify the clients that belong to a NAT-based network.

2.1 EDoS Attack Mitigation Techniques

An auto scaling technique, namely CloudWatch [22], has been enabled by Amazon to reduce the effects of the EDoS attack by monitoring the cloud resources. CloudWatch gives the customers a control to define the limits of the cloud platforms elasticity and thus reduce the EDoS attacks effects. However, this control technology will freeze the cloud service when elasticity reaches the pre-defined upper pound threshold and thus the legitimate users will not be able to access the cloud services until refreshing the quota again which leads to similar demeanor of DoS and DDoS attacks. In addition, the cloud service customers will be charged to some degree determined by the pre-defined thresholds of the cloud platforms elasticity during the attack. For these reasons, CloudWatch cannot be deemed as one of the effective techniques for mitigating the EDoS attack.

Self-verifying Proof of Work (sPoW) is another approach that focuses on proving client commitment through solving crypto-puzzles [23]. The proposed approach introduced an asymmetric step before committing the server's resources. Clients request for server access at the cloud provider’s end by first defining the crypto-puzzle difficulty level, k, and subsequently requesting access. A server has to generate a crypto-puzzle to protect the connection server channel. A crypto-puzzle consists of both the encryption of channel information and the concealed encryption key with k bits representing the puzzle difficulty. A puzzle requester running on the client-side expands the client resources by brute forcing these k bits to discover the server channel information. The proposed approach prioritizes the traffic based on the difficulty of the puzzle. If an initial connection request is not successfully made during a given frame of time, the client may request for a more difficult puzzle to solve. Upon succeeding in solving a puzzle of a given complexity, the server establishes a secure communication channel to exchange messages with the client. sPoW has several shortcomings such as asymmetric consumption power problem due to generating and solving the puzzles by servers and clients respectively, puzzle accumulation attack when the attacker responds with high difficult unsolved puzzles to the server, and puzzle’s difficulty impact on false positives.

Sqalli et al. [24] proposed a mitigation technique called EDoS-Shield to protect the cloud against EDoS attacks. The key factor proposed for differentiating between legitimate and EDoS requests is through verification of human presence to control an end-user machine. The EDoS-Shield architecture consists of two main components. The first component is a virtual Firewall (vFirewall) and it works as a filter mechanism to the incoming requests based on two lists, white list and black list, which hold the IP addresses of clients targeting the cloud application. The second component is a verifier cloud node (V-Node) and it uses the Graphic Turing test to verify legitimate requests. Then, it updates the whitelist and the blacklist based on the verification process outcome. If a user passes the Graphic Turing test, the user's IP address will be held in the white list and subsequent requests from the same IP address will be forwarded directly to the cloud scheduler for providing necessary services. In contrast, if a user fails the Turing test, the user's IP address will be held in the black list and subsequent requests from this IP address will be dropped by the firewall. However, the proposed approach has shortcomings. One of them is its vulnerability to IP spoofing. This problem might cause an EDoS attack if an attacker spoofs an IP address belonging to the whitelist of the verifier node. Another disadvantage is the false positives, in which the EDoS-Shield may unknowingly block a NAT IP address that corresponds to thousands of legitimate users due to the misbehaving of one attacker that belongs to the same NAT-based network. Likewise, the problem of false negatives can take place when IP addresses that belong to the whitelist change their behavior to harm the system. Finally, the EDoS-Shield adds an overhead on the firewall when checking the IP address of each incoming request, whereas our proposed mitigation technique checks the user's legitimacy only when there is abnormal behavior on the requested cloud service.

An enhanced version of the EDoS-Shield is proposed in [25]. The authors improved the EDoS-Shield and proposed an Enhanced EDoS-Shield that takes into account the spoofed IP addresses. This technique appends a Time To Live (TTL) value with the packet's IP address of cloud service request and adds a counter of unmatched TTL values to the white and black lists to decide whether the packet is having a spoofed IP address or not. The mean absolute variance of the TTL values was used to identify anomalous network traffic in [26, 27]. The results show that the Enhanced EDoS-Shield is an efficient technique to mitigate EDoS attacks especially with those using spoofed IP addresses. However, the Enhanced EDoS-Shield mitigation technique is based on the IP addresses lists approach that has many drawbacks such as blocking an entire NAT-based network if an attacker that belongs to the NAT-based network is added to the black list. In addition, cloud services are accessible from everywhere, so it is difficult to recognize clients fingerprint and their TTL values. Moreover, there are some attack tools that could change the value of TTL, so this value is not always correct and cannot be trusted [28]. Finally, the Enhanced EDoS-Shield adds an extra overhead on the firewall when checking the IP address and its TTL value for each request. This overhead will affect the response time of the legitimate requests even with the normal behavior on the requested cloud service.

A discussion and a description of Economic Distributed Denial of Sustainability (EDDoS) in cloud computing has been presented in [29]. The authors proposed a mitigation technique for EDDoS attack. This mitigation technique is an in-cloud EDDoS mitigation web service comprising three modules, namely, packet filtering, proof-of-work technique, and egress filtering to avoid EDDoS attacks to the cloud. The clients of the cloud service must prove their commitment for gaining service access by solving the crypto puzzle. Only clients succeeding in solving the crypto puzzle are granted access to the cloud services. The authors focused on building an effective crypto puzzle [30], but this puzzle adds a computation overhead on the client side. In addition, in the cloud computing environment, there are varieties of clients such as mobile clients, tablet clients, etc. These clients cannot handle the computation overhead to solve the crypto puzzle in order to gain access to cloud services. Moreover, Gligor et al. [31] stated that the client puzzles used as a proof of work are superfluous and ineffectual as they force a high overhead on the legitimate users requests and just offer extremely feeble assurances. Another limitation of the proposed scheme is the puzzle accumulation attack at the puzzle's generation server.

In-Cloud Scrubber Service [32] is another mitigation technique for the EDoS attack provided by the cloud service provider as a separate service to check the users legitimacy. In-Cloud Scrubber Service will be responsible for the generation and verification of puzzles, so there is no overhead on the cloud service. The proposed mitigation technique uses two modes of operation, normal mode and suspected mode. When the web server is working as expected, then the system will work in the normal mode. But when the service provider notices that the traffic that targets the web server exceeds an acceptable threshold, then the operation will be switched to the suspected mode. In-Cloud Scrubber Service is used while the cloud service is operating in the suspected mode. It will send crypto puzzles to the clients to distinguish legitimate requests from bot requests. The proposed mitigation has some limitations such as the disadvantage of the crypto puzzles, as mentioned earlier, and the threshold limit. The system could be unstable due to a fluctuation in the traffic activity around the threshold limit frequently.

The authors in [33] propose an approach for ensuring that HTTP and XML based DDoS attacks do not trigger the auto scaling feature of the cloud, thus ensuring that an EDoS does not transpire through such an attack. The contribution mainly focuses on studying the ability of a DDoS attack to cause an EDoS attack against the cloud through protocol vulnerability exploitation. They conducted some experiments on Amazon public cloud. Their results show that the proposed protection technique will not eliminate completely the EDoS attack and more research is needed to prevent and mitigate the EDoS attack.

Alosaimi et al. [34] proposed a new mitigation technique to encounter the EDoS attack in a new cloud environment. This environment is where “Bring Your Own Device” (BYOD) policies in enterprises are defined. The attack is targeting the Identity and Access Management (IAM) weaknesses in the BYOD implementation of the organizations to gain access to the internal resources of the organizations and launch an EDoS attack. This attack is taking advantage of the missing of resource control and management of platforms of the BYOD devices. This attack could cause Direct DDoS to the organization itself or cause indirect DDoS to other organizations that use the same cloud service provider. Their mitigation technique is called DDoS-Mitigation System (DDoS-MS). It investigates only two packets from the source of the requests by performing two tests, the Crypto Puzzles test and Graphic Turing test. The two types of testing are used to authenticate the packet and the user, respectively. In addition, this technique uses the black and white lists based on IP addresses to control the access to the cloud services, a verifier node and puzzle server for verification process, a DNS server, and a filtering router. The proposed framework reduces the end-to-end delay of the users’ requests. However, problems such as false positives and false negatives are still unaddressed by this technique.

An enhanced DDoS-Mitigation technique [35] has been proposed to improve the DDoS-MS mitigation technique. The proposed system investigates only the first packet by using

Graphic Turing test. The enhanced DDoS-MS consists of a virtual firewall, verifier nodes, a client puzzle server, an Intrusion Prevention System (IPS) device, and a Reverse Proxy (RP) server. The virtual firewall has four different lists, the black list, the white list, the suspicious list, and the malicious list. The IPS is used to investigate the packet for malicious content such as malware. The RP server is responsible for hiding the locations of cloud servers, and for balancing the load between these servers. In addition, it monitors the rate of traffic to detect DDoS attacks. The client puzzle server is used as a reactive step to delay the requests of a user who tries to overwhelm the system. DDoS-MS and Enhanced DDoS-MS have many drawbacks such as blocking an entire NAT-based network, as mentioned earlier, adding a huge overhead on the system because of the use of three layers of filtering; firewall, IPS, and RP, and using the TTL value which is not always correct and cannot be trusted.

The author of [36] proposed a mitigation technique called EDoS Armor against EDoS attack for e-commerce applications in cloud environment. The EDoS Armor has dual defense system; the admission control and the congestion control. The admission control is used to limit the number of clients who are accessing the cloud services at the same time. The congestion control is used to assign priority to the permitted clients based on a browsing behavior learning mechanism. The learning mechanism is used to classify the clients into good and bad, based on the client activities in the system. Moreover, a challenge server is used to authenticate users into the system by sending a challenge for each client at the beginning of the session. The EDoS Armor has some limitations such as restraining the elasticity feature of the cloud in which the simultaneous user requests for cloud service are limited by the admission control. In addition, the average response time of the good clients is rather high because of the continuous learning mechanism and the priority updating process. Another disadvantage is the false positives, in which the EDoS Armor may unknowingly block a NAT IP address that corresponds to thousands of legitimate users due to the misbehaving of one attacker that belongs to the same NAT-based network.

A novel approach for detecting and mitigating the effects of an EDoS attack against the auto scaling feature of the cloud is proposed in [37]. This scheme is implemented at the service provider’s end with normal or suspicious classification to the incoming requests. Subsequently, further investigation is directed to guarantee that priority to access the cloud service is given to legitimate clients, while suspect clients are given less primacy to access the cloud system until they are ultimately expelled from the suspicion list. The architecture of this technique consists of virtual Firewall (vFirewall), Job Scheduler, VM Observer and VM Investigator. The vFirewall is used with white and black lists with the white list user requests being directed to the VMs. On the other hand, those requests found to be originating from blacklisted users are directly sent forth to the VM Investigator. The Job Scheduler is responsible for distribution of jobs to individual VMs. The VM Observer process resides within each VM and it is responsible for probing the resource usage within the VM. Upon observing exceeding resource usage at a VM, the VM Observer redirects subsequent requests to the VM Investigator for further analysis. The VM Investigator uses a User Trust Factor (UTF) parameter to calculate the number of accesses that can be provided to a given user in a given time frame. The UTF is updated depending on the result of the user’s Turing test. When the UTF value reaches zero, all subsequent requests from this IP address are dropped. Through this scheme, none of the requests that surpass the defined thresholds of resource usage are dropped, but rather the users are provided with a delayed access to the cloud services based on the UTF value. However, the proposed approach sets a limit on the number of requests that a user can submit to the cloud infrastructure. This can be significantly less than the number of requests a legitimate user may have. Hence, this will result in an increase in the response time for such users. Such a limitation can be magnified for legitimate users that belong to a NAT-based network.

The authors in [38] proposed a detection technique against EDoS attack based on the Time Spent on a web Page (TSP) that represents the duration spent on viewing a website. A massive quantity of very few TSP values indicates a botnet targeting the web page [39]. The average TSP value resulting from the attack payload will be different from the mean TSP of a website. The TSP deviation from the mean value can be calculated in terms of Mean Absolute Deviation (MAD). A MAD plot method and foot step graph method plot the deviation and the TSP's respectively to identify the various types of attack traffic. However, the proposed detection technique requires human intervention to monitor and interpret the plots.

Modi et al. [40] surveyed the intrusion detection techniques that can be used in the cloud system. They listed many types of attacks that target the cloud system. They describe the flooding attack and how this attack could increase the costs for the customers significantly as the cloud system would not be able to differentiate the legitimate traffic from the malicious traffic. They described the EDoS attacks and they claimed that some of the proposed mitigation techniques are not efficient since they only use traditional firewalls as firewalls cannot differentiate the legitimate requests from DoS attack requests. In addition, they state that more research is needed to prevent and mitigate the EDoS attack.

A mitigation technique called Index Page Attack Defender (IPA-Defender) is presented in [41]. The IPA-Defender focuses on detecting and mitigating an EDoS attack that targets the index page of any website because it is provided freely and without authentication. The Iptables is used to implement the scheme. The idea of the IPA-Defender is to check each request for the index page. If the page count threshold of the requester is crossed, the IPA-Defender drops the current request and the subsequent requests of that requester by maintaining its IP address in the blacklist for a period of time. Then, the IP address will be removed from the blacklist upon completion of the blocking period allowing the user to re-request the index page. The IPA-Defender might be inefficient to detect fraud requests that rely on the page count threshold. Moreover, the proposed scheme is susceptible to the false positive problem.

Saleh et al. [42] proposed a framework to address all sorts of HTTP DoS/DDoS attacks. Their research aims to design a solution called a Flexible, Collaborative, Multilayer, DDoS Prevention Framework (FCMDPF). The innovative design of the FCMDPF framework handles all aspects of HTTP-based DoS/DDoS attacks through three subsequent framework's schemes. Firstly, an Outer Blocking (OB) scheme that blocks attacking IP source addresses when they are listed in a black list table. Secondly, the Service Trace Back Oriented Architecture (STBOA) scheme validates whether the incoming request is launched by a human or by an automated tool. Then, it traces back the true attacking IP source. Thirdly, the Flexible Advanced Entropy Based (FAEB) scheme is used to eliminate High Rate DDoS (HR-DDoS) and Flash Crowd (FC) attacks.

The FCMDPF framework is flexible because it eliminates the impact of flash crowd attacks gradually by decreasing the maximum connection's timeout value plus decreasing the maximum allowed requests per this timeout until these two values reach zero. Once the values of the timeout and the maximum allowed requests reach zero, the FAEB scheme disables the Keep-Alive feature of the HTTP connection. Therefore, the detection and prevention mode switches from a flash crowd attack to a high rate DDoS attack. In the meanwhile, the FCMDPF framework blocks high rate HTTP DoS/DDoS attacks immediately. This framework's design provides a novel alternative protective framework to protect web applications from all sorts of HTTP DoS/DDoS attacks, such as HR-DDoS and FC. In addition, it is quite able to validate and trace back the real attacking IP sources, and block them at the edge router using the OB scheme. In the contrast, it suffers from the false negatives, since it was not able to detect and prevent all of the flash crowd attacks. In addition, it failed to validate and trace back some of the incoming requests such as the requests targeting the system that originated from a NAT-based network.

Baig et al. [43] proposed a novel technique for detecting and mitigating the effects of an EDoS attack. The architecture of this technique consists of virtual Firewall (vFirewall), Load Balancer (LB), DataBase (DB) and VMInvestigator. The vFirewall is used with a black list table to filter the incoming requests to the cloud computing system. The black list user requests are directly sent forth to the VMInvestigator for extra analysis. The load balancer is responsible for distributing requests evenly to individual VMs, monitoring and auto-scaling the cloud resources. In addition, the LB is responsible for redirecting the incoming requests to the VMInvestigator when the CPU utilization of the cloud resources exceeds 80%. The DB contains information about the users past behavior. This information includes the user’s IP address, the time of the last activity on the cloud system, the number of requests per minute originated by each individual user, the User Trust Factor (UTF) value, and the number of requests per a single second made by each individual user. The VMInvestigator uses a Turing test and a UTF parameter to determine the user’s legitimacy. The VMInvestigator randomly selects numbers, referred to as Random Check (RC) values, from the interval [1, TRPM]. The TRPM is the allowable total requests per minute which is equal to 60 multiplied by the pre-defined allowable Concurrent Requests Per Second (CRPS). The count of the selected RC values is equivalent to the CRPS. Initially, the VMInvestigator checks if the user violates the system by sending requests per second greater than the CRPS. The request from a user who breaches the CRPS threshold with UTF less than 0.25 will be dropped, whereas the user will be requested to provide an answer for the Turing test when breaching the CRPS but with UTF greater than or equal to 0.25. On the other hand, the VMInvestigator will check if the number of requests per minute from the requester matches any of those randomly selected RC values when the user does not breach the CRPS. Subsequently, the VMInvestigator will send a Turing test to that user upon matching the RC values. Otherwise, the VMInvestigator will check the UTF value when the request number does not match any of the RC values. Then, the user will be requested to perform the Turing test when the UTF less than 0.25. Otherwise, the user’s request will be directly sent to the cloud VMs. Note that if the user failed the Turing test by giving a wrong answer or by not providing an answer at all, the UTF value will be decremented by 0.02 and the user’s request will be dropped. However, if the user passed the Turing test, the UTF value will be incremented by 0.01 and the user’s request will be directly sent to the cloud VMs.

The proposed technique by Baig et al. has some limitations such as forcing the legitimate users to answer the Turing test even when they are targeting the cloud service with requests less than the CRPS. This is due to the case of matching their requests number to the RC values. Another disadvantage is the false positives, in which the proposed technique may unknowingly block a NAT IP address that corresponds to thousands of legitimate users due to the misbehaving of one attacker that belongs to the same NAT-based network.

2.2 Methods for Identifying Clients that belong to a Network Address Translation (NAT) - based Network

Network Address Translation (NAT) devices [44-46] have become a convenient way to hide the source of malicious behaviors. In [47], the authors explore how far we can push a Machine Learning (ML) approach to identify the behavior of NAT devices using only network flows. They showed that detecting the NAT behavior in the analyzed traffic is challenging. Moreover, they mentioned that using the TTL field of the IP header to differentiate the types of users that belong to a NAT-based network will not be accurate. This is mainly because some NAT implementations might not decrement the TTL values for some reason, or the TTL values can be hidden or modified by some tools.

A recognition method of computers that belong to a NAT-based network was proposed in [48]. The method uses a proxy authentication that is implemented in a proxy server. Their target application was HTTP. They used a realm field in the authentication header and associated it to the MAC address of the client computer after the authentication is succeeded. That realm field is shown to the user as a prompt message and is included in the authentication header of all HTTP messages sent from the client to the proxy server. Their proposed system requires a Java Runtime Environment (JRE) on each client machine to run a Java applet. This Java applet is developed by the authors and to be downloaded from the proxy server. The purpose of the Java applet is to collect the MAC address of the client PC and subsequently used in the realm field. They assumed that a web browser has the capability to associate the authentication header with the request as soon as the verification has succeeded. Even though this method is successful, it could only be used for their target application and the proxy conditions require the JRE code that the authors developed.

Maier et al. [49] analyzed the HTTP user-agent strings to obtain the OS and the browser information such as the browser family and version. Then, based on this information, they made a decision regarding the presence of a NAT device in the traffic. This approach can be used to distinguish between users of different types of browsers such as Firefox, Internet Explorer, Safari and Opera. However, the approach cannot distinguish between users that utilize the same type of browser, and, therefore, is inefficient in distinguishing between legitimate and malicious users when they all belong to the same NAT-based network.

A method for controlling a system against DDoS attacks which prevents a normal user from being blocked in Network Address Translation (NAT) was presented in [50]. The proposed system used a black list rule table to store the client IP address, the web server IP address, and the web server port when excessive web page request traffic or an abnormal connection request is detected through the detection engine. The system that is located between a client and a web server receives data requesting information of the web server from the client. Subsequently, it halts the transmission of packets other than a packet for Transmission Control Protocol (TCP) session connection according to a matching result obtained from a black list rule table. As a result, the system transmits a virtual IP information data of the corresponding web server to serve the current session. Accordingly, the system determines that a user of the received information is a normal user when the corresponding client requests information from the web server through the use of the received virtual IP information of the corresponding web server. However, the proposed method may unknowingly block a NAT IP address that corresponds to thousands of legitimate users due to the misbehaving of one attacker that belongs to the same NAT-based network. This will occur when the NAT IP address, the web server IP address, and the web server port are added to the black list rule table. As a result, the proposed system will transmit a URL redirection packet as a response to the first web page request after the TCP session connection request to prevent the legitimate users from being blocked. However, since both an attacker and the legitimate users may belong to the same NAT-based network it cannot be guaranteed that the URL redirection packet will reach the legitimate user who has sent the TCP session connection. In addition, there will be no benefit from the use of the web server port in the black rule table, knowing that the web server port is always 80.

CHAPTER 3

THE PROPOSED APPROACH

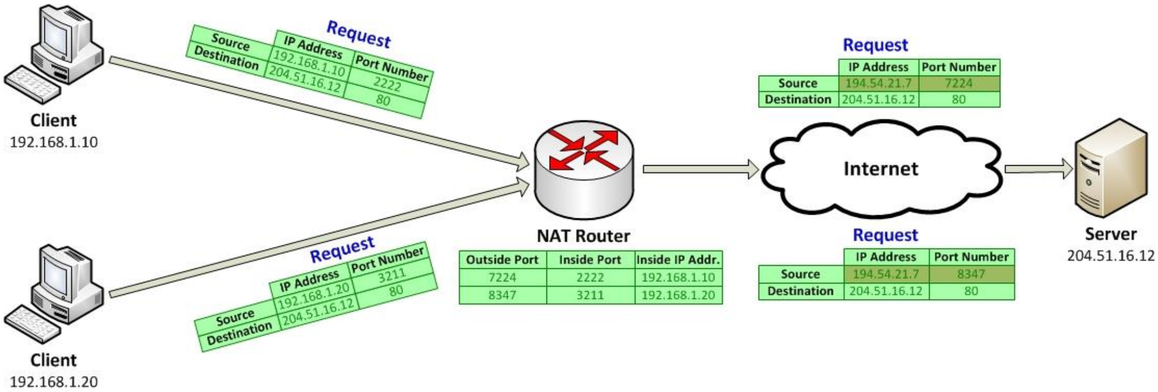
In this chapter, we present the proposed EDoS attack mitigation technique for the cloud infrastructure that will be referred to as the EDoS Attack Defense Shell (EDoS-ADS). It is a reactive technique that detects and mitigates the effects of an EDoS attack against the auto scaling feature of the cloud computing model.

Auto scaling allows the allocation of new resources with minimal management effort for the purpose of handling the increased demands on the services and releasing some existing resources when these demands decrease. Auto scaling is triggered based on a pre-defined threshold duration of some parameters. The cloud computing platform administrator could select the average CPU utilization, memory, or response time as the auto scaling parameter with predefined threshold and duration based on the specific application. The threshold parameter is the point at which the auto scaling is triggered. The duration parameter is the period of time during which the auto scaling condition is true to trigger the auto scaling.

The proposed technique makes use of the threshold and the duration parameters that accompany the auto scaling conditions. We will use the average CPU utilization threshold as a parameter to trigger the auto scaling condition since it is a good indicator of the abnormal behavior as well as its importance to maintain the cloud system performance and save the cost. The duration value is used in our proposed technique to ensure that auto scaling is triggered only for legitimate requests in order to mitigate EDoS attack effects. So, the proposed technique is considered a reactive because it starts running only when the threshold parameter is crossed at which time it verifies whether the incoming requests are coming from legitimate users or are generated by compromised machines (Zombies). Subsequently, it only allows the legitimate users requests to access the cloud service and drops all attack traffic. Since the auto scaling feature is managed by the cloud provider, the cloud provider is encouraged to implement and offer the EDoS mitigation technique as a security feature for the cloud customers.

3.1 Address the Problem of Blocking an Entire NAT-Based Network

It is necessary for EDoS attack mitigation techniques to be able to distinguish between legitimate users and attackers when both belong to Network Address Translation (NAT)-based network. Such a distinction will help in preventing the blocking of an entire NAT-based network. A NAT is defined as a method of sharing one public IP address for several different devices by replacing the source IP address and the source port number with the NAT public IP address and a randomly chosen unused port number, respectively, as shown in Figure 4 [44, 45, 51]. Note that the port number assigned by the NAT router can be used to distinguish between different clients that belong to a NAT-based network since it is impossible to assign the same port for several active users. Subsequently, the source IP address along with the source port number will be used in the proposed mitigation technique to address the problems of IP spoofing, the blocking of NAT-based networks, and distinguishing the different types of users targeting the cloud services.



	IP Address	Port Number
Source	192.168.1.10	2222
Destination	204.51.16.12	80

	IP Address	Port Number
Source	192.168.1.20	3211
Destination	204.51.16.12	80

Outside Port	Inside Port	Inside IP Addr.
7224	2222	192.168.1.10
8347	3211	192.168.1.20

	IP Address	Port Number
Source	194.54.21.7	7224
Destination	204.51.16.12	80

	IP Address	Port Number
Source	194.54.21.7	8347
Destination	204.51.16.12	80

Figure 4 Network Address Translation (NAT) Working Principle

3.2 Differentiate Between Legitimate Users and Automated Attackers

It is essential for an EDoS attack mitigation technique to differentiate between legitimate users and automated attackers. The proposed mitigation technique will differentiate the traffic using two techniques. Graphic Turing Tests (GTTs) [52] and Uniform Resource Locator (URL) redirection [53]. Subsequently, it will drop the request in case of a failure to respond to either the GTT or the URL redirection, and will mark the request generator as an attacker.

3.2.1 Graphic Turing Test (GTT)

All mitigation techniques that have been studied in the literature used Completely Automated Public Turing test to tell Computers and Humans Apart (CAPTCHA) [54-58] to differentiate between legitimate users and automated attackers. CAPTCHAs are designed to be easy for humans but unbreakable by machines. However, Bursztein et al.

[59] proved that CAPTCHAs are often difficult for humans and are breakable by machines. In [60], the authors provided a study about reCAPTCHA and they mentioned that reCAPTCHAs are currently more secure than the conventional CAPTCHAs because they use only words from scanned books to which OCR fails, while CAPTCHAs generate their own randomly distorted characters. Additionally, they reported that the overall success rate for reCAPTCHA was 96.1%, based on more than 1 billion responses from English speaking, and the average time spent in solving reCAPTCHA was lower than the time required to solve a standard CAPTCHA. Subsequently, the proposed mitigation technique will make use of reCAPTCHA Turing test for differentiating between legitimate users and automated attackers. Figure 5 shows an example of a reCAPTCHA Turing test.

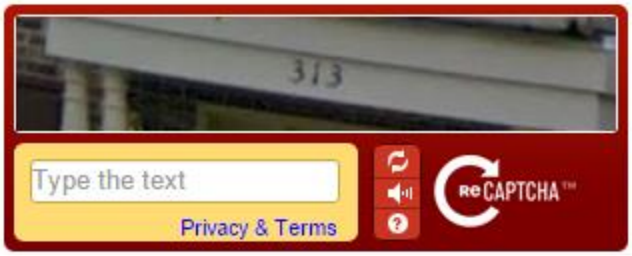


Figure 5 reCAPTCHA Turing Test [60]

3.2.2 Uniform Resource Locator (URL) Redirection

The URL redirection occurs when a client sends HTTP requests to access a web resources located at specific URL. Then, the requested web server issues an HTTP response to the client browser with a location field in the header. This implies that the client has to redirect to a different URL [61, 62]. We make use of the URL redirection for detecting certain clients who send requests without the use of a web browser. Subsequently, these clients will be marked as attackers.

The concept of the URL redirection has been used in preventing the DDoS attacks [50]. The proposed mitigation technique determines the client as a normal client if the received request includes the virtual IP address of the requested web server that is used in the URL redirection technique as discussed in subsection 2.2. This indicates that the client is using a web browser to request the cloud computing service. On the other hand, the attackers will keep requesting the cloud service using the actual IP address of the web server. The attackers most likely run a script in an automated machine to generate a huge number of requests to target the cloud web servers. The attackers typically do not wait for the cloud service response packet and accordingly they will not react to the URL redirection.

The cloud computing system workflow under the use of the EDoS Defense Shell with the URL redirection technique is shown in Figure 6. The proposed mitigation technique will generate a TCP SYN-ACK response signal on behalf of the web server if the matched client request is a TCP session connection request. Furthermore, the proposed mitigation technique transmits the generated TCP SYN-ACK response signal to the client. Moreover, it sends a URL redirect response packet to the client system upon receiving an

HTTP web page request that includes the actual IP address of the cloud server. The rest of the process follows the URL redirection technique explained in subsection 2.2.

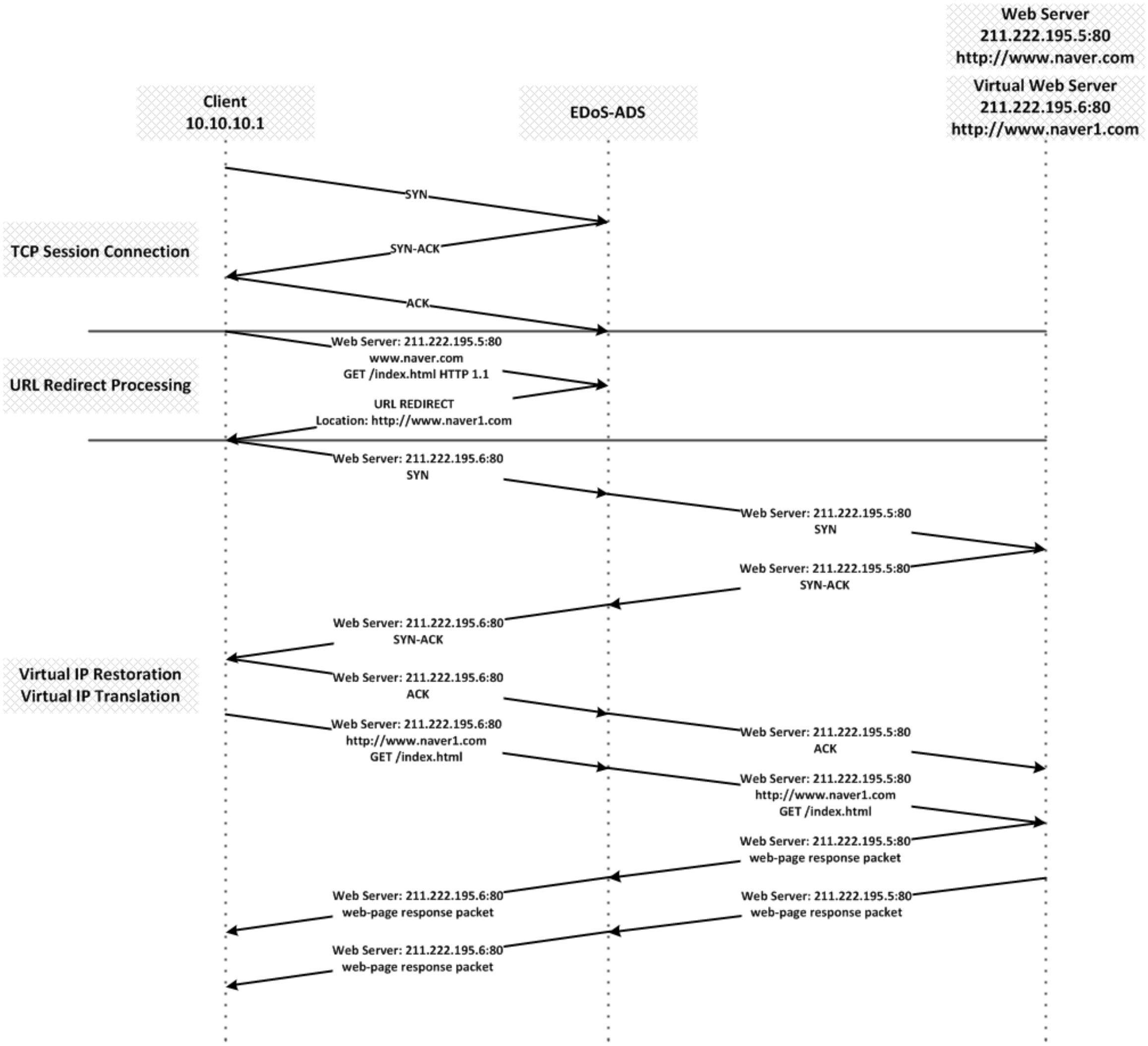


Figure 6 Actions of Individual Packet Processing Operations

3.3 EDoS Attack Defense Shell (EDoS-ADS)

The proposed EDoS Attack Defense Shell mitigation technique depends operationally on the use of CPU utilization thresholds. In order to avoid the fluctuation of the cloud average CPU utilization, we will use four thresholds with two timers as shown in Figure 7. These thresholds are scaling-up upper threshold, scaling-up lower threshold, scaling-down upper threshold, and scaling-down lower threshold with assumed values of 80%, 75%, 30%, and 35%, respectively [43, 63, 64]. The auto scaling timers are the upper threshold timer and the lower threshold timer with assumed durations of 5 minutes and 1 minute, respectively [43, 64].

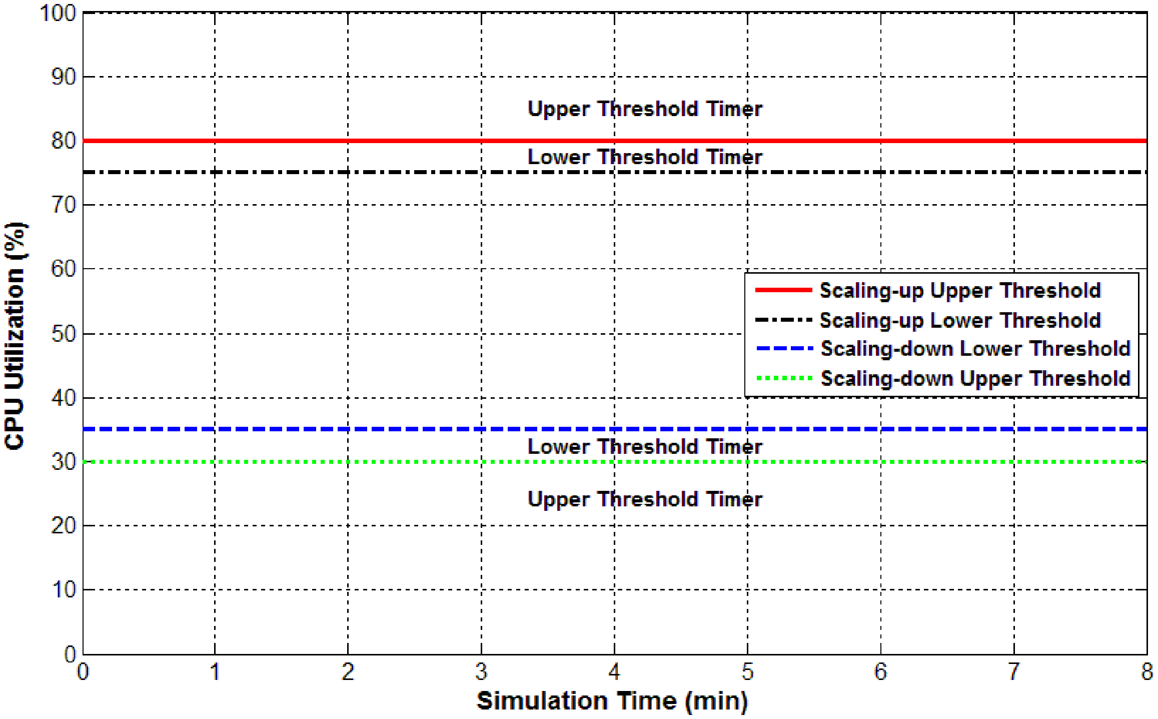


Figure 7 CPU Utilization Thresholds

To illustrate how these thresholds and timers will be used, consider the cloud service without a mitigation technique. The allocation of new resources will occur when the average CPU utilization is above 80% for 5 minutes. Now consider that the utilization surpasses the 80% threshold for less than 5 minutes, then goes down but still above 75% and lasts for less than 1 minute then returns back up to exceed the 80% threshold, then it will be counted as if it was above the 80% threshold for the whole time. Thus, auto scaling will be triggered and will not be affected by the momentary drop in the utilization. The same technique will be used in the case of releasing some existing resources.

The proposed mitigation technique will make use of the scaling-down upper and lower thresholds for releasing some of the existing resources. On the other hand, the scaling-up upper and lower thresholds will be used to trigger the EDoS-ADS. Further clarification will be provided in the rest of this section.

The EDoS-ADS mitigation technique has four modes for the cloud computing environment. These modes are ***normal mode***, ***suspicion mode***, ***flash overcrowd mode***, and ***attack mode*** as illustrated in Figure 8. The ***normal mode*** indicates that the number of requests that arrive at the cloud service component are below the scaling-up utilization thresholds. However, the system transitions to the ***suspicion mode*** when the current utilization surpasses the scaling-up utilization thresholds. If this increase in the number of requests comes from malicious clients, the system moves to the ***attack mode***. Otherwise, the system transitions to the ***flash overcrowd mode*** indicating that these requests are from legitimate clients. The EDoS-ADS will work differently in the ***suspicion mode*** than in the ***attack mode***. For simplicity, we will use the term **Suspicion Shell** when referring to the proposed technique when working under the ***suspicion mode***, and the term **Attack Shell** when referring to the proposed technique when working under the ***attack mode***.

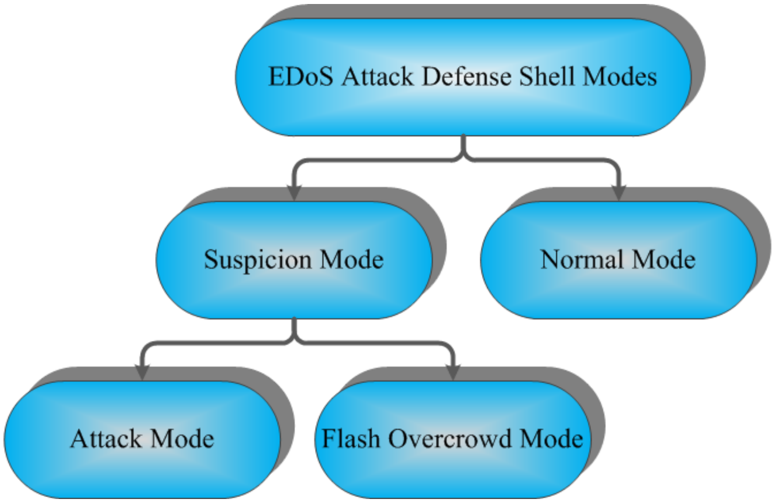


Figure 8 EDoS Attack Defense Shell Modes

The cloud service will start operating in the ***normal mode***. The load balancer will forward all incoming requests to the cloud servers. The cloud service responds to the clients' requests without any overhead or checking as shown in Figure 9. We consider the cloud computing environment in the ***normal mode*** if the average CPU utilization did not violate the scaling-up upper threshold. The ***normal mode*** indicates that the number of requests that arrive at the cloud service is as usual.

The cloud service load balancer will monitor the current utilization of a cloud service system and the auto scaling utilization thresholds. If the system current utilization is greater than the scaling-up upper threshold, referred to as the suspicion region, then the ***normal mode*** will be changed to ***suspicion mode*** and all incoming requests will be directed to the **Suspicion Shell** for further investigations. For example, the cloud mode will immediately change to ***suspicion mode*** when the average CPU utilization exceeds 80%.

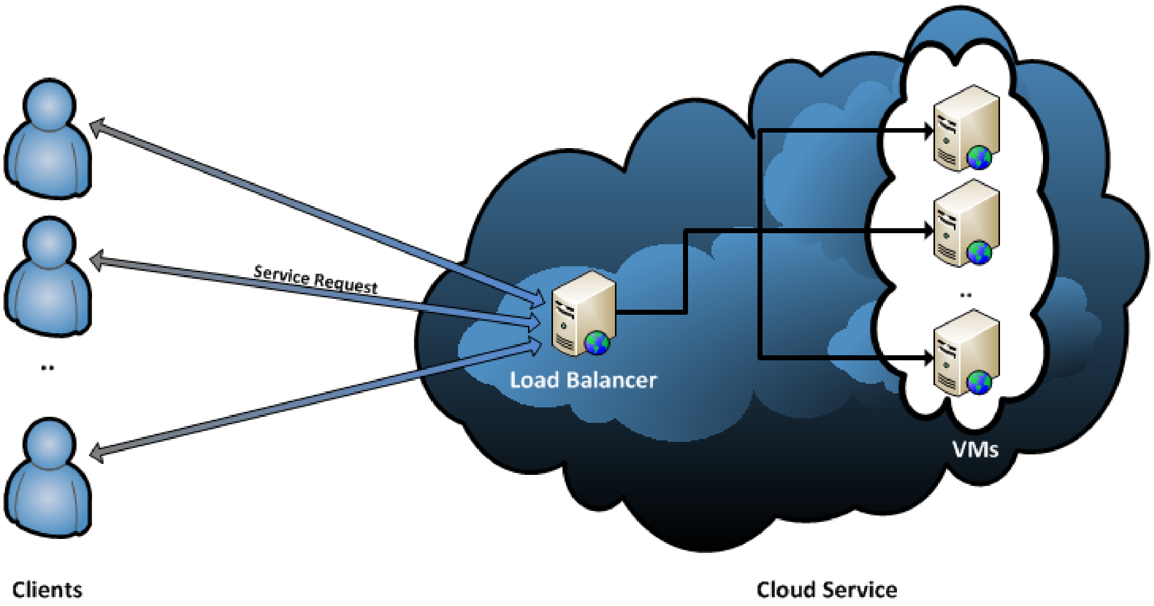


Figure 9 Cloud Service Architecture

Hence, the load balancer is responsible for auto scaling and for monitoring the average CPU utilization on the cloud. The load balancer will redirect the incoming requests to the cloud servers or to the EDoS-ADS (**Suspicion Shell** or **Attack Shell**) depending on the current mode and the current CPU utilization level. Figure 10 shows the workflow for this component. In addition, the load balancer will be responsible for switching the cloud mode from the ***flash overcrowd mode*** to the ***normal mode*** after provisioning of new VM instances.

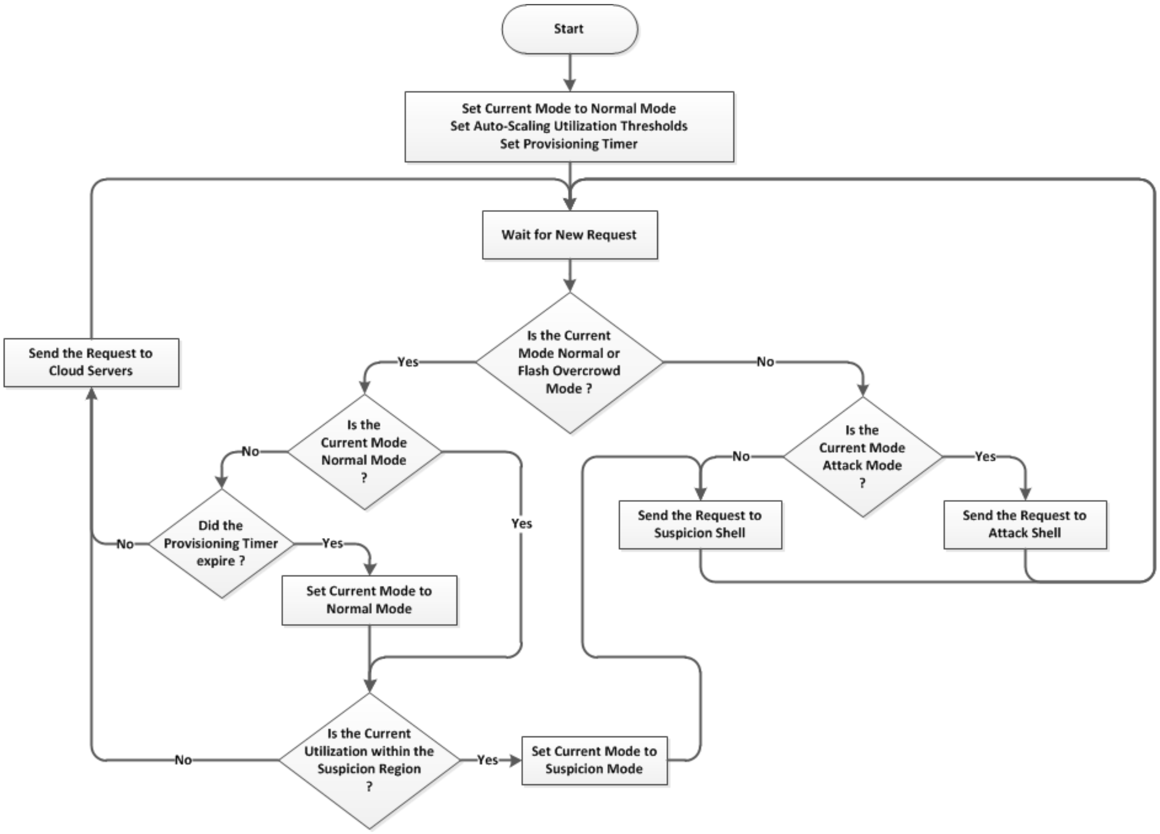

Start
Set Current Mode to Normal Mode
Set Auto-Scaling Utilization Thresholds
Set Provisioning Timer
Wait for New Request
Is the Current Mode Normal or Flash Overcrowd Mode ?
Yes
No
Send the Request to Cloud Servers
Is the Current Mode Normal Mode ?
No
Yes
Is the Current Mode Attack Mode ?
No
Yes
Send the Request to Suspicion Shell
Send the Request to Attack Shell
Did the Provisioning Timer expire ?
No
Yes
Set Current Mode to Normal Mode
Is the Current Utilization within the Suspicion Region ?
No
Yes
Set Current Mode to Suspicion Mode

Figure 10 Cloud Service Load Balancer Workflow

The EDoS-ADS is responsible for differentiating between legitimate users and automated attackers by sending GTTs to the request generator. The EDoS-ADS will drop the request in case of a failure to respond correctly to the GTT and considers the request generator as an attacker. On the other hand, the EDoS-ADS will forward all requests that pass the GTT to the cloud service. However, by continuously sending GTTs to the normal clients will likely make them less motivated to use the cloud computing services. Subsequently, the EDoS-ADS will make use of the URL redirection technique as explained in subsection 3.2.2 to eliminate such a problem. Figure 11 shows the architecture of the EDoS-ADS.

Once the current utilization within the suspicion region, the system needs to determine whether the cloud service is under an ***attack mode*** or a ***flash overcrowd mode***. To assist the system in making that decision, the system tracks the past behavior of clients. As such, the system stores information about each client in a Database. The information stored includes the IP address, source port number, status, a Trust Factor (TF), Last Seen, and the Concurrent Requests Per Second (CRPS). Table 1 represents an example of the stored information for each client in the Behavior table.

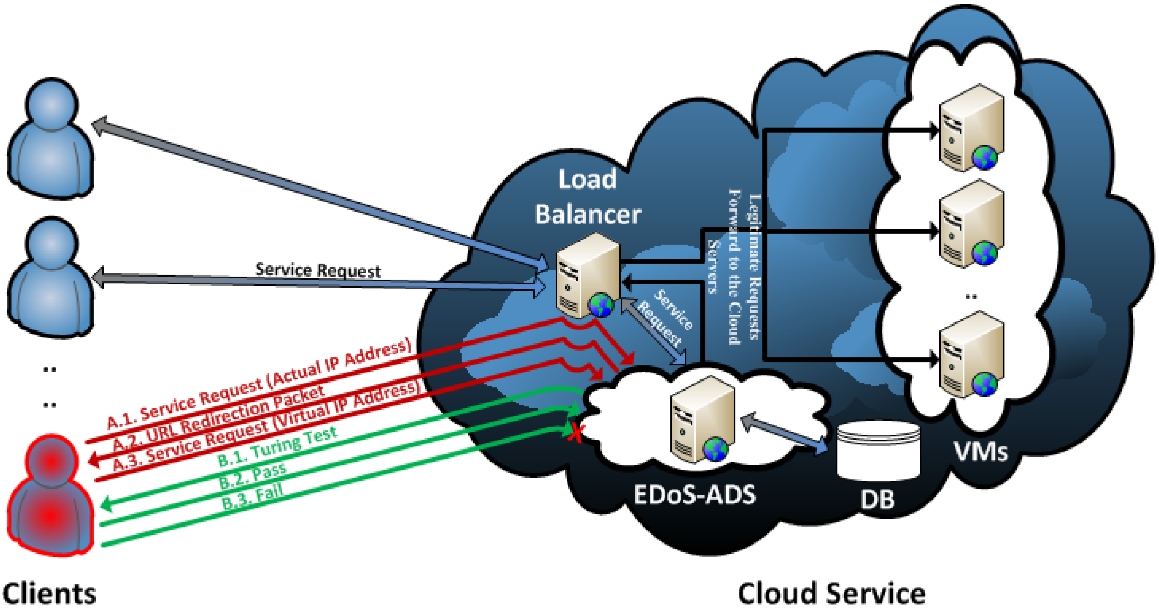

Load Balancer
Legitimate Requests Forward to the Cloud Servers
Service Request
Service Request
..
A.1. Service Request (Actual IP Address)
A.2. URL Redirection Packet
A.3. Service Request (Virtual IP Address)
B.1. Turing Test
B.2. Pass
B.3. Fail
EDoS-ADS
DB
VMs
Clients
Cloud Service

Figure 11 EDoS Attack Defense Shell Architecture

Table 1 Behavior Table

IP	Port	Status	TF	Last Seen	CRPS
11.10.10.1	3254	Legitimate	0.79	2014-10-07 01:00:10	2
194.66.82.11	9842	Suspicious	0.52	2014-10-07 07:10:26	1
…	…	…	…	…	…
22.231.113.64	2314	Malicious	0.13	2014-10-07 09:25:03	6
22.231.113.64	6405	Legitimate	0.86	2014-10-07 04:25:03	5

The EDoS-ADS uses the IP address and the port number as the client key. The status field represents the current status of a client in the cloud system which could be one of three status types; Legitimate, Suspicious, or Malicious. The TF is a value in the range of [0, 1] assigned to each client depending on the client's response to the GTT. A TF value close to 1 indicates a high trust factor and that the associated client is likely trust worthy. The Last Seen keeps the time for the last request to each client and it is used to update the value of the CRPS. The CRPS is used to keep the total count of the client's requests per second. The value of the CRPS will be incremented by 1 when the request arrival time matches the Last Seen time. Otherwise, the EDoS-ADS will reset the value of the CRPS to 1 and the value of the Last Seen will be updated to the request arrival time.

The TF value is initially set by the **Suspicion Shell** when a new request from a new client passes the EDoS-ADS. A value of 0.5 is used as a default value for the TF. The TF value is incremented or decremented according to the finite state machine shown in Figure 12. The client transitions between the different states considering the source IP address, the source port number, and the current status that are stored in the database according to the client's response to the GTT. Note that the amount of increment of the client's TF should be less than the amount of TF decrement because the failure in the GTT is a possible indicator of an attack [65]. Both the status and the TF are affected by the results of the reCAPTCHA Turing test. If a new client failed the test by giving a wrong answer or by not providing an answer at all, the TF value will be decremented by 0.02 and the client's status will be set to Suspicious. However, if the new client passed the test, the TF value will be incremented by 0.01 and the client's status will be set to Legitimate.

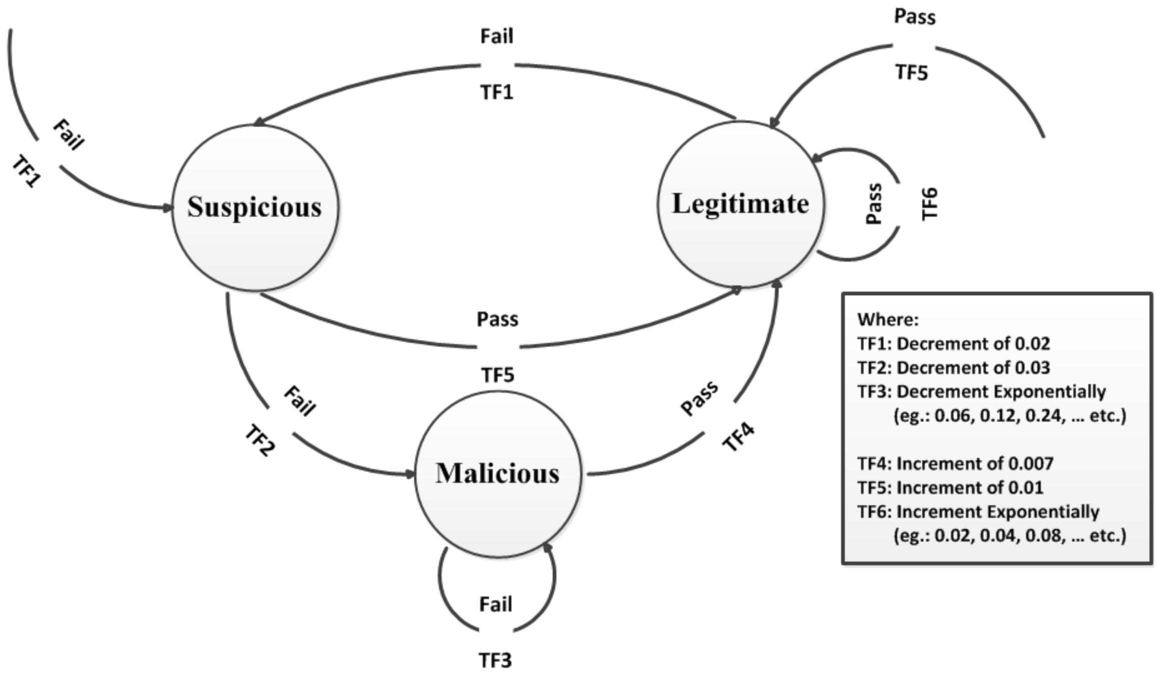


Figure 12 Finite State Machine for Updating the Current Status and the TF Value

In [66], the authors classified trust factors into three different levels each with a specific interval as follows: bad TF is [0, 0.25), average TF is [0.25, 0.75], and good TF is (0.75, 1] as shown in Table 2. The proposed EDoS-ADS uses these levels to check the client's legitimacy. The clients within the good TF level are likely trust worthy.

Table 2 Trust Factor Levels

TF Level	Interval
Bad TF	[0, 0.25)
Average TF	[0.25, 0.75]
Good TF	(0.75, 1]

Figure 13 shows the flowchart of the **Suspicion Shell**. The **Suspicion Shell** will check the type of the incoming request as shown in the dashed border (I). Subsequently, if the request is a TCP session connection, the **Suspicion Shell** will generate a TCP response packet on behalf of the cloud web server and will update the Behavior table. The **Suspicion Shell** will update the Behavior table by inserting or resetting the record related to the client. It resets the record in the Behavior table because the TCP connection packet is an indication of a new client establishing a connection to request the cloud servers using the same IP address and port number that was used by another client.

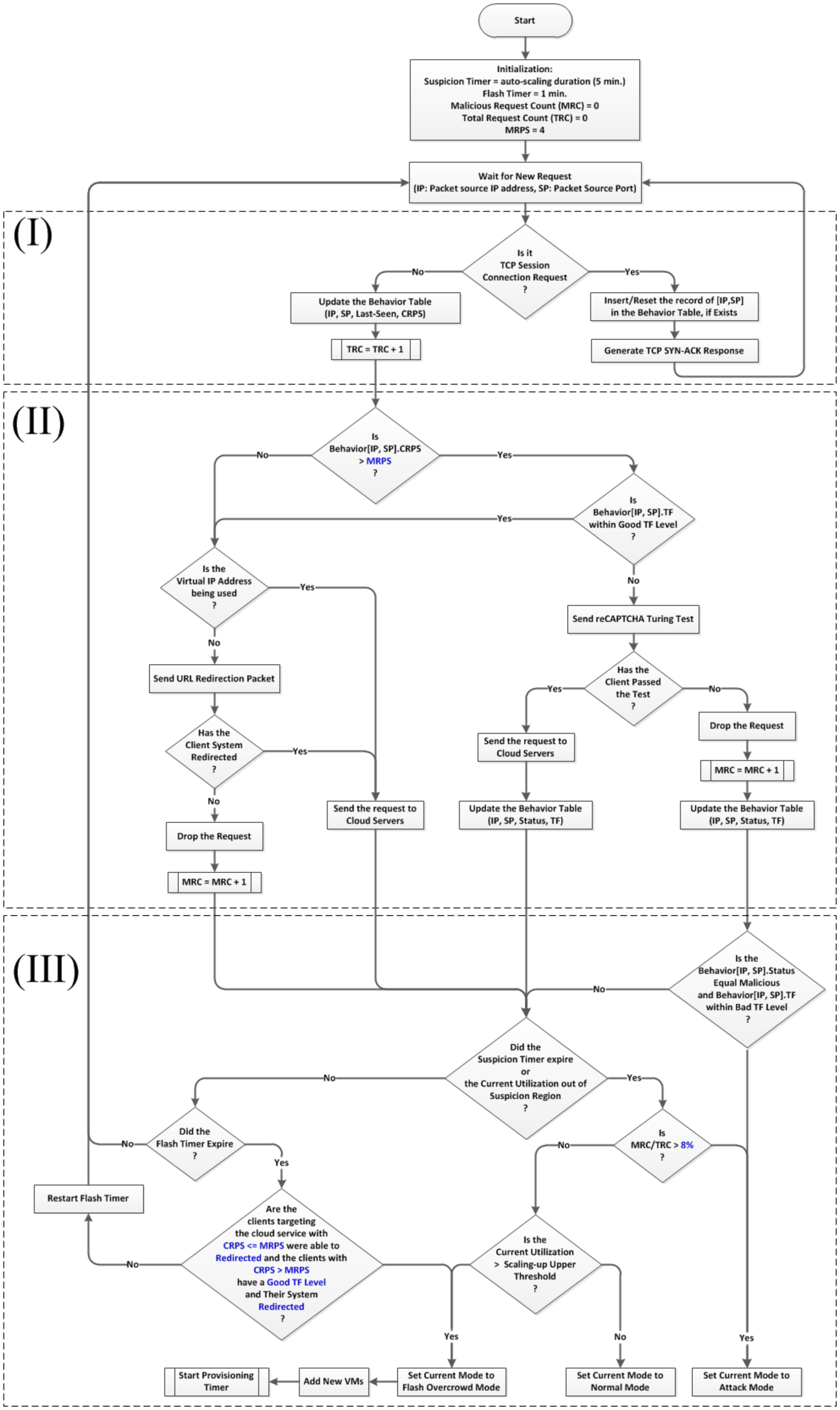

Start
Initialization:
Suspicion Timer = auto-scaling duration (5 min.)
Flash Timer = 1 min.
Malicious Request Count (MRC) = 0
Total Request Count (TRC) = 0
MRPS = 4
Wait for New Request
(IP: Packet source IP address, SP: Packet Source Port)
(I)
Is it TCP Session Connection Request ?
No
Yes
Update the Behavior Table (IP, SP, Last-Seen, CRPS)
Insert/Reset the record of [IP,SP] in the Behavior Table, if Exists
TRC = TRC + 1
Generate TCP SYN-ACK Response
(II)
Is Behavior[IP, SP].CRPS > MRPS ?
No
Yes
Is Behavior[IP, SP].TF within Good TF Level ?
Yes
No
Is the Virtual IP Address being used ?
Yes
No
Send reCAPTCHA Turing Test
Send URL Redirection Packet
Has the Client Passed the Test ?
Yes
No
Has the Client System Redirected ?
Yes
No
Send the request to Cloud Servers
Drop the Request
MRC = MRC + 1
Send the request to Cloud Servers
Update the Behavior Table (IP, SP, Status, TF)
Update the Behavior Table (IP, SP, Status, TF)
Drop the Request
MRC = MRC + 1
(III)
Is the Behavior[IP, SP].Status Equal Malicious and Behavior[IP, SP].TF within Bad TF Level ?
No
Did the Suspicion Timer expire or the Current Utilization out of Suspicion Region ?
No
Yes
Did the Flash Timer Expire ?
No
Yes
Is MRC/TRC > 8% ?
No
Restart Flash Timer
Are the clients targeting the cloud service with CRPS <= MRPS were able to Redirected and the clients with CRPS > MRPS have a Good TF Level and Their System Redirected ?
No
Is the Current Utilization > Scaling-up Upper Threshold ?
Yes
No
Yes
Start Provisioning Timer
Add New VMs
Set Current Mode to Flash Overcrowd Mode
Set Current Mode to Normal Mode
Set Current Mode to Attack Mode

Figure 13 Suspicion Shell Flowchart

On the other hand, the **Suspicion Shell** will update the values of Last Seen and CRPS related to the client in the Behavior table upon receiving an HTTP web page request. The difference between the current time and the Last Seen time of a client is checked when a new request from a certain user comes to the Suspicious Shell. If the time difference is less than or equal to 1 second, the value of the CRPS will be incremented by 1. However, if the time difference is more than 1 second, the value of the CRPS will be reset to 1. Then, the Last Seen will be updated to the current time. In addition, the proposed EDoS-ADS counts the total number of HTTP web page requests. It uses the Total Request Count (TRC) parameter for this purpose. The value of the TRC will be incremented upon receiving an HTTP web page request, and then it will be used later to determine the next cloud service mode.

The **Suspicion Shell** differentiates the legitimate and attacker requests as explained in the dashed border (II). A new request from a client who sends requests with a CRPS less than or equal to the Maximum Requests Per Second (MRPS) threshold will be requested to perform the URL redirection. The proposed mitigation technique will send a URL redirect packet to the web browser of those clients with CRPS less than or equal to MRPS. Later, the client's browser will use the redirected URL for the current request and for any future request to the cloud web server. In such a way, the requests of clients requesting the cloud service with CRPS less than or equal to MRPS will be forwarded to the cloud servers immediately without the need to send reCAPTCHA Turing test since the requests are using the virtual IP address of the cloud web server. The purpose of using the URL redirection in such a case is to defend against a DDoS attack and identify smart attackers who can guess the MRPS value and send requests without exceeding this value.

Accordingly, the Suspicion Sell will protect the cloud servers from such requests by dropping them.

On the other hand, the **Suspicion Shell** will send a reCAPTCHA Turing test to a client sending a new request to the cloud service, if the client's CRPS is larger than the MRPS, and the TF is either average or bad. The purpose of the Turing test is to allow the legitimate users to improve their TF and reach a point where they can be treated as good users. At the same time, the Turing test will help in detecting the attack requests, and accordingly the **Suspicion Shell** will protect the cloud server from such requests by dropping them. Moreover, the **Suspicion Shell** will reduce the TF of the users generating the attack requests.

In order to reduce the amount of Turing test sent to the legitimate users, the **Suspicion Shell** will make use of the URL redirection technique to eliminate such a problem. The proposed mitigation technique will send a URL redirect packet to the web browser of those clients with CRPS larger than MRPS and having a good TF. Later, the client's browser will use the redirected URL for the current request and for any future request to the cloud web server. In such a way, the requests of clients having a good TF will be forwarded to the cloud servers immediately without subjecting them to reCAPTCHA Turing tests even when the CRPS is larger than the MRPS threshold since the requests are using the virtual IP address of the cloud web server.

The **Suspicion Shell** component will keep counting the number of malicious responses and record the total in the Malicious Request Count (MRC) parameter. The MRC will be incremented by one for every failed reCAPTCHA Turing test and URL redirection. Later, the values of the MRC and the TRC will be used to determine the next cloud service mode as represented by the dashed border (III). For early detection of the ***attack mode***, we will check the past behavior for clients who failed the reCAPTCHA Turing test. A client with Malicious status and bad TF and a CRPS more than the MRPS threshold will be marked as an attacker and the current mode will change to ***attack mode***. In addition, we will check the TF level and the URL redirection ability for clients targeting the cloud service system every 1 minute. This period is referred to as the Flash Timer. If the clients targeted the cloud service with CRPS less than or equal to MRPS were able to redirect and the clients with CRPS larger than MRPS have a good TF level and their system redirected successfully, the cloud mode will change to ***flash overcrowd mode*** directly without waiting until the expiration of the Suspicion Timer. On the other hand, the **Suspicion Shell** will restart the Flash Timer if the previous condition has not been met.

The **Suspicion Shell** will be triggered for a period equal to the Upper Threshold Timer. This period is referred to as the Suspicion Timer. If the Suspicion Timer does not expire, and the current system utilization decreases to less than the scaling-up lower threshold (75%) or it is in between the scaling-up upper and lower thresholds (75% - 80%) for more than the Lower Threshold Timer, the cloud mode will be transitioned from ***suspicion mode*** to ***normal mode***. Moreover, when the Suspicion Timer expires, the **Suspicion Shell** will change the mode of the cloud computing environment to either a ***flash overcrowd mode*** or an ***attack mode*** depending on the percentage of malicious responses. The percentage of malicious responses is calculated by dividing the number of malicious requests (MRC) over the total number of requests (TRC). If this value is below or equal 8%, the current mode will be set to ***flash overcrowd mode***. On the other hand, if it is greater than 8%, then the current mode will be set to ***attack mode***. The use of 8% to determine the mode attempts to mimic the real life when some legitimate users cannot solve the reCAPTCHA Turing test for some reason, while taking into account the overall success rate of solving a reCAPTCHA Turing test is 96.1% based on more than 1 billion responses [60]. Moreover, this success rate is in the range of 92.6-96.9% for clients whose native language is not English [60]. Therefore, we select 8% as a threshold for mode determination.

During the ***flash overcrowd mode***, new cloud virtual machines will be added, and a Provisioning Timer will start. Accordingly, the cloud mode will remain in the ***flash overcrowd mode*** for the duration of the timer. Due to the allocation of the new VM instances the cloud will return to the ***normal mode*** [67]. When the cloud is in the ***flash overcrowd mode***, all requests will be served directly by the cloud service as in the ***normal mode***.

The cloud mode will be changed to ***attack mode*** in two cases as previously mentioned in the **Suspicion Shell**. The first case is when a client fails the reCAPTCHA Turing test while having a Malicious status, the client has bad TF, and the client CRPS is more than the MRPS threshold. On the other hand, the second case is when the Suspicion Timer expires and the percentage of malicious responses is greater than 8%. A detailed inner working of the **Attack Shell** is provided in Figure 14.

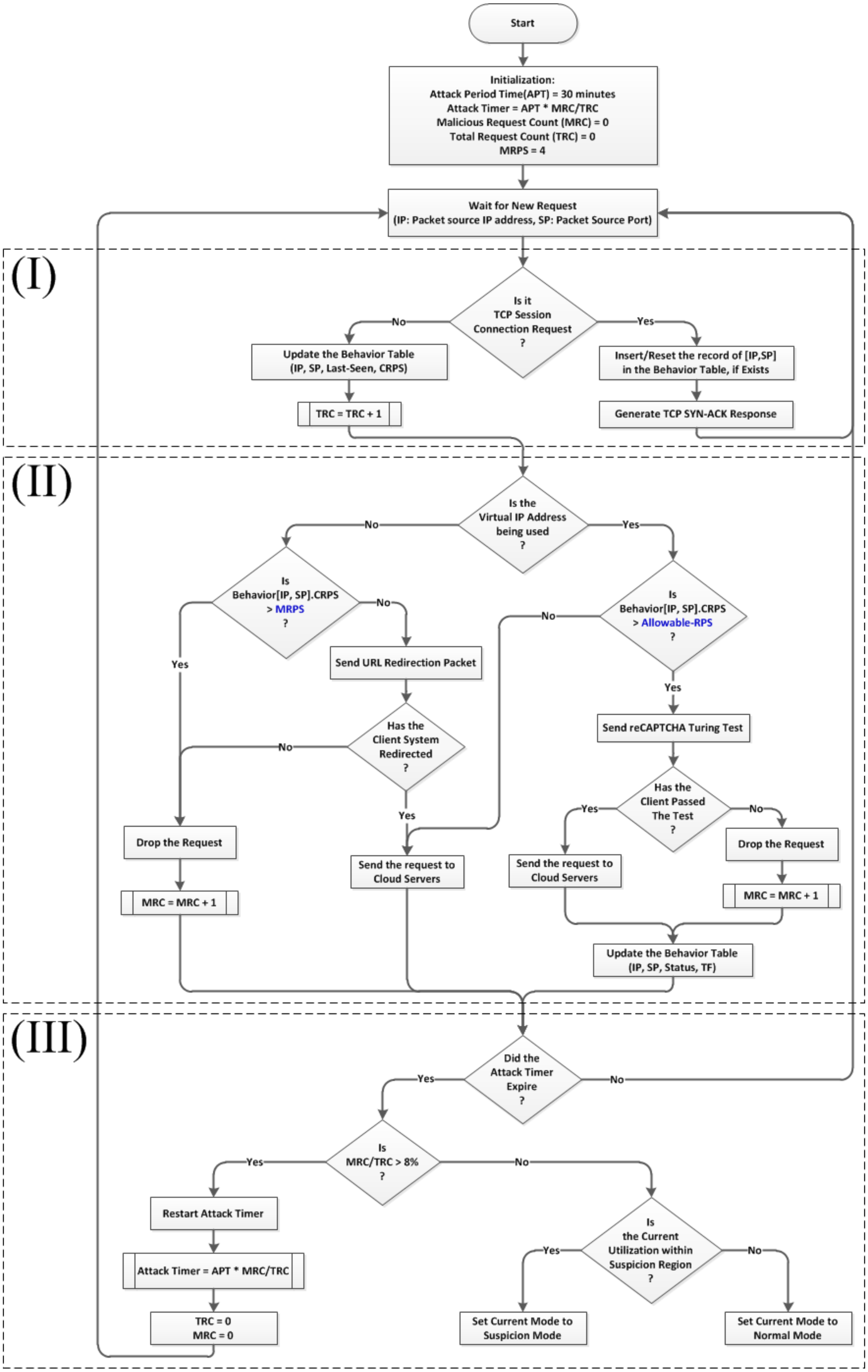

Start
Initialization:
Attack Period Time(APT) = 30 minutes
Attack Timer = APT * MRC/TRC
Malicious Request Count (MRC) = 0
Total Request Count (TRC) = 0
MRPS = 4
Wait for New Request
(IP: Packet source IP address, SP: Packet Source Port)
(I)
Is it TCP Session Connection Request ?
No
Yes
Update the Behavior Table (IP, SP, Last-Seen, CRPS)
Insert/Reset the record of [IP,SP] in the Behavior Table, if Exists
TRC = TRC + 1
Generate TCP SYN-ACK Response
(II)
Is the Virtual IP Address being used ?
No
Yes
Is Behavior[IP, SP].CRPS > MRPS ?
Yes
No
Send URL Redirection Packet
Has the Client System Redirected ?
No
Yes
Is Behavior[IP, SP].CRPS > Allowable-RPS ?
No
Yes
Send reCAPTCHA Turing Test
Has the Client Passed The Test ?
Yes
No
Drop the Request
MRC = MRC + 1
Send the request to Cloud Servers
Send the request to Cloud Servers
Drop the Request
MRC = MRC + 1
Update the Behavior Table (IP, SP, Status, TF)
(III)
Did the Attack Timer Expire ?
Yes
No
Is MRC/TRC > 8% ?
Yes
No
Restart Attack Timer
Attack Timer = APT * MRC/TRC
TRC = 0
MRC = 0
Is the Current Utilization within Suspicion Region ?
Yes
No
Set Current Mode to Suspicion Mode
Set Current Mode to Normal Mode

Figure 14 Attack Shell Flowchart

During the ***attack mode***, the proposed EDoS-ADS will eliminate the attacking request by using the URL redirection technique while reducing Turing tests for legitimate users. Initially, the **Attack Shell** will check the type of the incoming request whether it is a TCP session connection request or an HTTP web page request. This examination is represented by the dashed border (I). The **Attack Shell** will generate a TCP response packet on behalf of the cloud web server and update the Behavior table in the same way as discussed in the ***suspicion mode*** upon receiving a TCP session connection request. However, the **Attack Shell** will check the request legitimacy based on the URL redirection technique, or the reCAPTCHA Turing test in the case of an HTTP web page request. The Behavior table will be updated in a similar fashion to what was discussed in the **Suspicion Shell**.

The **Attack Shell** will differentiate between legitimate and attacker clients as shown in the dashed border (II). All clients will be asked to perform redirection in the ***attack mode***. The proposed mitigation technique will send a URL redirect packet to the web browser of a client targeting the cloud service with requests rate that are less than or equal to the MRPS threshold, and that are using the actual IP address of the requested server. On the other hand, the requests from a client targeting the cloud service with requests rate larger than the MRPS threshold using the actual IP address of the cloud server will be immediately dropped. This is because it is an indication of an attack behavior where an automated machine is sending huge number of requests to the cloud computing system using the actual IP address instead of the virtual IP address passed on to the client through a prior URL redirection packet as a result of an earlier request by the same client. Hence, there is no need to keep sending URL redirection packets to such a client. In this way the **Attack Shell** will immediately eliminate the attacker requests and prevent them from reaching the cloud servers. The **Attack Shell** will allow the clients to request the cloud service with a dynamic number of requests depending on the client's TF. This dynamic number is referred to as the Allowable-RPS.

The Client's browser will use the redirected URL for re-submitting the current request and for any future requests to the cloud web server. In such a way, the requests of clients targeting the cloud service using the virtual IP address and sending the requests with a rate not more than their Allowable-RPS threshold will be forwarded to the cloud servers without reCAPTCHA Turing tests. However, the **Attack Shell** will send reCAPTCHA Turing test for those clients targeting the cloud service using the virtual IP address but with a CRPS exceeding their Allowable-RPS threshold. The Turing test will help in eliminating requests originating from smart attackers while allowing the legitimate users to improve their TF.

The improvement of users TF results in an increase in their Allowable-RPS threshold and allows them to target the cloud service with higher requests rate without a need for reCAPTCHA tests in subsequent submission of requests. The Allowable-RPS threshold is calculated based on the TF value; raising the TF value results in increasing the Allowable-RPS threshold exponentially. The proposed mitigation technique will allow the legitimate users with TF equal to 1 to target the cloud service with requests up to 36 Req/sec [68] rather than 4 Req/sec as shown in Figure 15. Accordingly, the Allowable-RPS threshold can be calculated using the following equation [69]:

$$\text{Allowable-RPS} = \lfloor 0.444516 \times pow(81, TF) \rfloor \tag{3.1}$$

Note that the Allowable-RPS is changed dynamically only for the clients with good or average TF level. On the other hand, the Allowable-RPS threshold for the clients with bad TF will be fixed to 1.

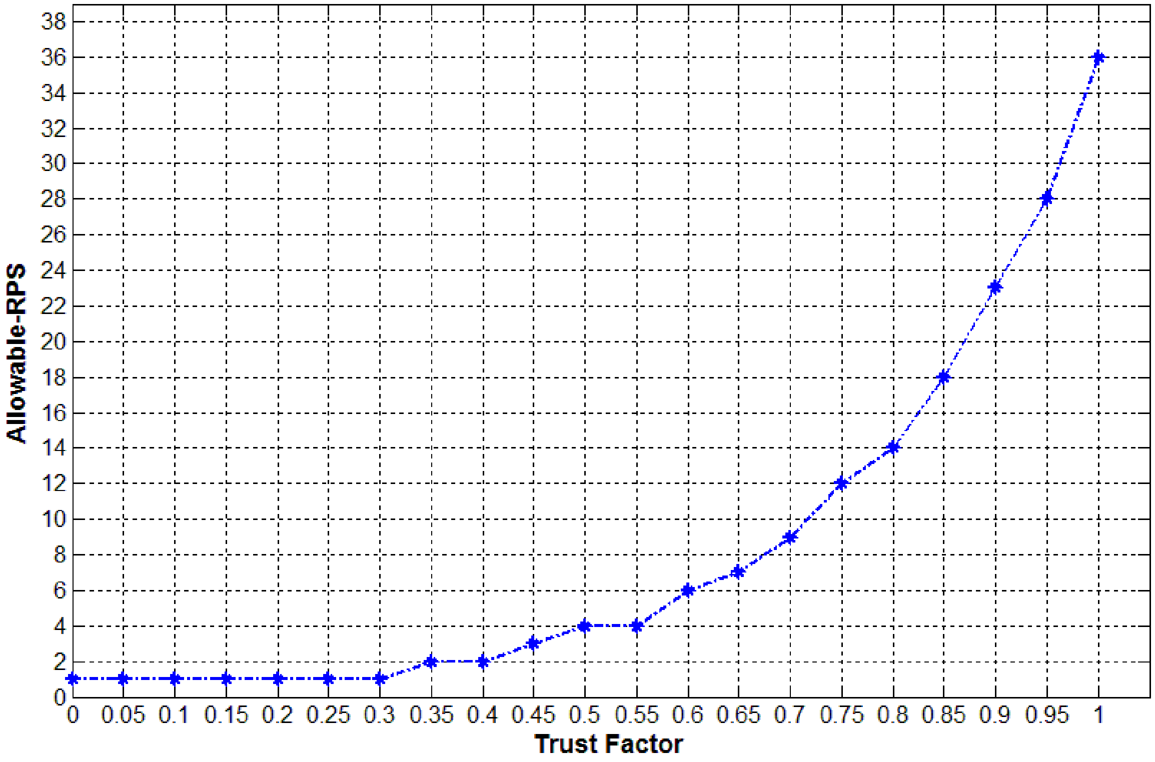


Figure 15 The Allowable Requests Per Second Depending on the Client's Trust Factor

The cloud computing system remains in the ***attack mode*** for a certain amount of duration that is referred to as the Attack Timer. During that duration, the **Attack Shell** continues to send Turing tests to requests originators that are targeting the cloud service using the virtual IP address of the cloud servers where their CRPS is higher than their Allowable-RPS threshold. In order to select the period of the Attack Timer, we consider the average duration of attacks studied in the literature. The authors in [70] mentioned that the majority of the DoS attacks are relatively short, and 80% of the attacks stayed for less than 30 minutes. In addition, NSFOCUS stated in their security report [71] that 93.2% of the DDoS attacks in 2013 lasted for less than 30 minutes, similar to what was observed in 2012. So, we select the Attack Period Time (APT) to be 30 minutes. Subsequently, the duration of the ***attack mode*** will be calculated by multiplying the APT value by the percentage of malicious responses, MRC divided by TRC, which was measured during the ***suspicion mode***.

In the dashed border (III), the **Attack Shell** needs to determine the next mode for the cloud system when its Attack Timer expires. If the percentage of malicious responses is greater than 8%, the current mode will not change and the cloud system will stay in the ***attack mode***. Accordingly, the **Attack Shell** will restart its Attack Timer with a new time duration which is specified by multiplying the APT value by the percentage of malicious responses and reset its TRC and MRC. On the other hand, if the percentage of malicious responses is below or equal 8%, then the current mode will be set to either the ***suspicion mode*** or the ***normal mode*** depending on the current CPU utilization. If the average utilization is within the suspicion utilization region, the cloud mode will be transitioned from the ***attack mode*** to the ***suspicion mode***, otherwise the cloud computing system returns back to the ***normal mode***.

CHAPTER 4

SIMULATOR

In this chapter, we discuss the simulator implementation used to evaluate the proposed mitigation technique. In addition, we present the measured parameters in the simulation. Moreover, we illustrate the experimental setup of the proposed mitigation technique against EDoS attack. We also verify and validate our simulation by comparing the obtained results against the EDoS-Shield mitigation technique results [64].

4.1 Simulator Design and Modeling

The simulation is implemented using Java programming language. We simulated the cloud computing environment using CloudSim [72, 73]. CloudSim is a toolkit used to simulate different scenarios of cloud computing. It provides many classes to describe datacenters, load balancers, hosts, virtual machine instances, cloudlets (requests), users, computational resources, and management policies such as provisioning and scheduling.

CloudSim framework was designed as a multi-layered software. Its layers and architectural components are shown in Figure 16. The CloudSim simulation layer delivers support for simulating and modeling the virtualized cloud based data center environment which includes the virtual machines, memory, storage, and bandwidth management interfaces. This layer handles the fundamental issues such as managing the execution of applications, and the hosts provisioning to the virtual machines. The User

Code layer is concerned with the basic units such as number of users, number of requests and their demand, number of instances and their specifications, virtual machines, and load balancer scheduling policies.

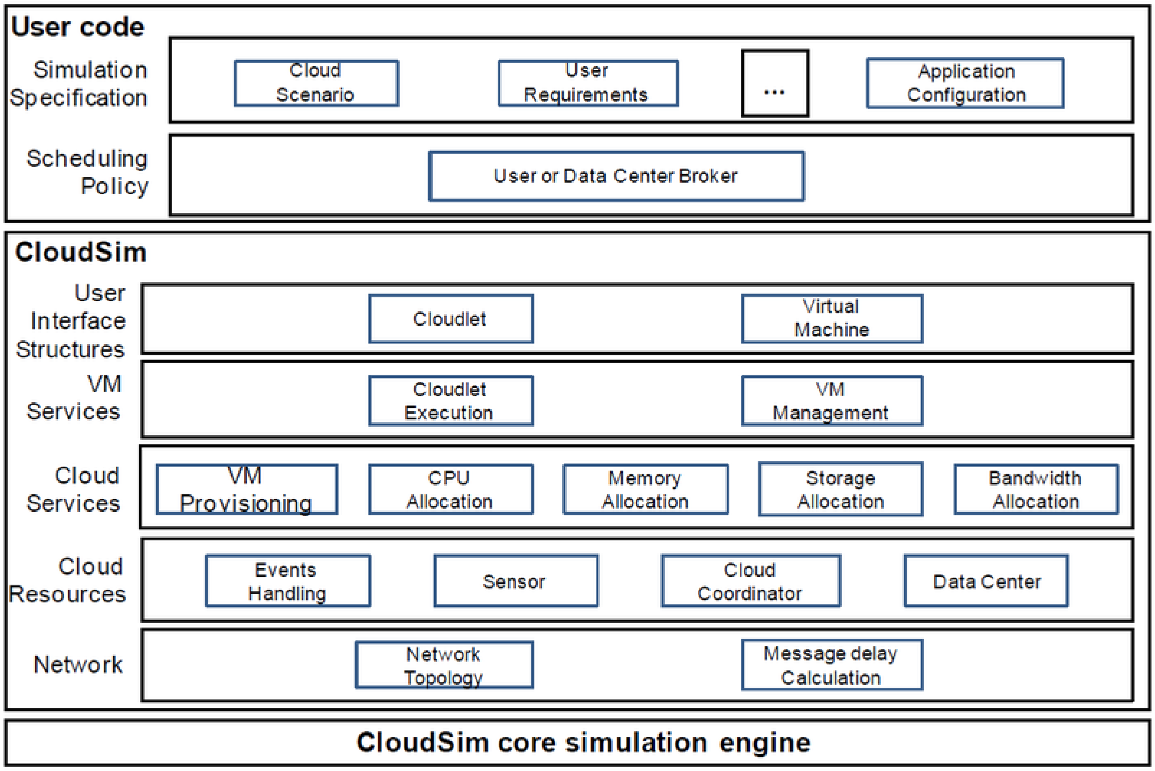


Figure 16 Layered CloudSim Architecture [73]

In regards to clients requests targeting a cloud-based web service, several studies have assumed a random variable having Poisson distribution for both the number of customers' requests and the cloud instances service rate [74-77]. Similarly, the inter arrival time and the service time of client requests are considered an exponential distribution in a cloud service for most of web applications [78-81]. In addition, the assumption of Poisson arrival for the DDoS attack has been discussed in [82-84]. It should be noted that EDoS attack behaves similar to DDoS attack in terms of generating malicious flooding traffic. Subsequently, we have assumed Poisson traffic for the EDoS attack. The Prime Modulus Multiplicative Linear Congruential Generator (PMMLCG) is used for generating random numbers, with the use of initial seeds that were ten million apart, and avoiding any overlapping in the random number streams during the simulation. The PMMLCG is an efficient generator and one of the most popular methods for generating random numbers [85].

In the simulation environment, we are considering a single-class service where all cloud clients' requests have the same processing procedure as it has been discussed by Al-Haidari et al. [64]. For example, we consider delivering the content of a web server such as a web page, using the Hypertext Transfer Protocol (HTTP) over the Internet. Thus, considering the service rate as a Poisson distribution in our simulation is a valid assumption that will simplify the performance model.

We have considered the same setup as that of the queuing model presented in [24]. Several studies have modeled a web service as a network of queues, in which each virtual machine in the distributed system is modeled as a single queue [86, 87]. A cloud-based service usually has multi cloud servers offering the service to the cloud customers. Thus, a parallel *M/M/1* queuing model is used for the cloud service [88-90]. However, cloud servers, in reality, have a restricted queuing buffer like *M/M/1/K*. For convenience, we use *M/M/1* queuing model with the virtual machine instances due to the fact that the cloud computing systems have large finite buffers where the probability of overflow in the buffer is negligible [91].

According to the queuing model shown in Figure 17, the EDoS attack targets the cloud service with rate of λ_m, while the rate of legitimate traffic is λ_l. According to Poisson composition property [92], the aggregated traffic from attack sources and legitimate sources each having a Poisson distribution of arrivals results in a Poisson process having arrival rate of $\lambda = \lambda_l + \lambda_m$.

The Load Balancer (LB), which is also designed as *M/M/1* queuing model, forwards the customers' requests to one of cloud servers to perform the web application service. It ensures an even distribution of the incoming requests among all working cloud servers in such a way that each cloud server has an equal probability, P_i, that is equal to $1/S$ of receiving a request [63, 93], where P_i is the balancer routing probability to the i_{th} server in the auto scaling group, and *S* is the number of running servers in that group.

Auto scaling is one of the attractive features of a cloud computing environment, this feature enables customers to follow the demand curve for their applications closely and run their fleet of servers at optimal utilization. For example, new servers will be added to the auto scaling group when the average CPU utilization is high. Similarly, customers can remove servers when the average CPU utilization of their fleet is low. Auto scaling group is the core of the auto scaling service since it represents the servers' collection.

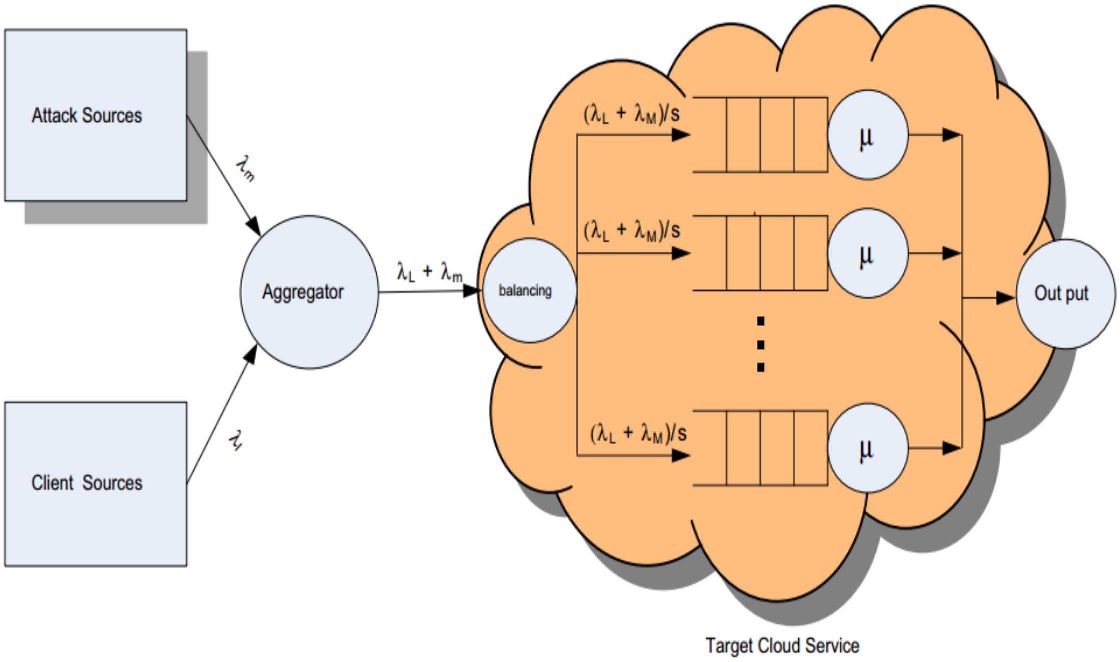


Figure 17 Queuing Model for EDoS Attack Against a Cloud Service [64]

4.2 Simulation Measures

Simulation experiments were conducted to simulate a cloud service system under an EDoS attack for the purpose of evaluating the cloud service system performance with and without the use of a mitigation technique. The key performance indicators include the cloud service response time, the utilization of computing resources, and the cloud service throughput. In addition, we have measured the cost related to the cloud computing resources and the cloud bandwidth allocations due to the EDoS attack.

4.2.1 Response Time

The response time is an important requirement in most of the Service Level Agreements (SLAs). The total end-to-end response time of the path without triggering the EDoS Attack Defense Shell (EDoS-ADS) includes the Load Balancer (LB) delay and the cloud servers' delay. However, the total end-to-end response time of the path after triggering the EDoS-ADS includes the LB delay, the EDoS-ADS delay, and the cloud servers' delay.

The average end-to-end response time is measured experimentally by calculating the average residence time, the departure time (D_τ) minus the arrival time (A_τ), for all requests, N, served by the cloud system as shown in the following equation:

$$RT_{Avg} = \frac{1}{N}\sum_{i=1}^{N}(D\tau_i - A\tau_i) \tag{4.1}$$

4.2.2 Resources Utilization

The scalability of cloud service can be controlled by varying automatically the number of running virtual machine instances in the auto scaling group based on different parameters such as the link utilization, and the computing resources utilization [94]. The average CPU utilization of the running virtual machine instances is being used in our technique to prove the mitigation concept. All incoming requests will be forwarded to the EDoS-ADS upon crossing the CPU utilization threshold for further investigation.

We assume that all cloud servers have the same capacity of computing power, $\mu_i = \mu$, and the request's arrival rate at each server is $\lambda_i = \lambda/S$, as previously mentioned in Figure 17. The mean computing utilization is calculated analytically as follows:

$$\rho = \lambda/(S \times \mu) \tag{4.2}$$

However, the CPU utilization of every VM instance is calculated experimentally by dividing the total server processing time by the total time of simulation. Hence, the average CPU utilization is calculated by taking the average of all running servers' utilization as follows:

$$\rho_{Avg} = \frac{1}{S} \sum_{i=1}^{S} \frac{Processing\ Time_i}{Simulation\ Time_i} \tag{4.3}$$

4.2.3 Throughput

The throughput of the cloud server, with *M/M/1* queuing system, can be calculated directly from Little's formula [95] using the following equation:

$$Throughput = \mu \times \rho = \mu \times \lambda/(S \times \mu) = (\lambda_l + \lambda_m)/S \tag{4.4}$$

Where μ is the service rate of the cloud server, ρ is the utilization of the cloud server $(\rho \leq 1)$, $\lambda = (\lambda_l + \lambda_m)$ is the arrival rate which will be distributed among the S running servers due to load balancing. Thus, each instance will have $(\lambda_l + \lambda_m)\ /S$ as its arrival rate. The average throughput of the cloud service with S running instances will be λ.

However, the throughput of the cloud server is measured experimentally by dividing the number of served requests, N, by the simulation time. The cloud service throughput is the aggregated throughput of all cloud servers in the auto scaling group as follows:

$$Throughput = \sum_{i=1}^{S} \frac{N_i}{Simulation\ Time_i} \tag{4.5}$$

4.2.4 Cost

Several studies [15, 96, 97] have conducted a comparison between the cloud service providers taking into account the most important aspects of cloud computing such as the costs reduction, flexibility, security, level of service, support, and compliance. The results have shown that Amazon Web Services (AWS) is one of the best cloud service providers. Moreover, AWS has been ranked as the most popular enterprise cloud service according to the latest quarterly report from Skyhigh Networks [98]. Subsequently, we have used the pricing model of AWS.

The cloud computing system can use one of several pricing models that provide the flexibility to optimize the customers' costs that are offered by service providers and cloud adopters. Currently, there are three different pricing models provided by Amazon EC2 to the customers. These models are reserved pricing, spot pricing, and on-demand pricing [99]. The reserved instances model allows the cloud user to pay one-time fee for reserving a cloud server and in turn receives a significant discount depending on the server hourly usage. For the spot instances model, the customers will be enabled to bid any price they want for purchasing compute capacity by specifying the maximum price per hour they are ready to pay for running the server. The on-demand instances model allows the customers to pay a constant rate for usage of resources by the hours with no commitment.

In this work, we follow the on-demand pricing model focusing on the costs that are related to the computing resources and network traffic volume [99]. In such a way, a cloud customer will pay for both bandwidth and computation usage. The cost has been calculated as follows:

$$Cost_{BW} = Price_{BW} \times \lambda \times T \tag{4.6}$$

$$Cost_{COMP} = Price_{VM} \times \rho \times S \times T \tag{4.7}$$

$$Cost = (Price_{BW} \times \lambda + Price_{VM} \times \rho \times S) \times T \tag{4.8}$$

Where $Price_{BW}$ is the price per input/output GB, λ is the arrival rate in GB/hr., T is the total running time in hours, $Price_{VM}$ is the price of using the computing resources per hour, ρ is the average CPU utilization of all computing resources during the period time, and S is the number of computing resources in the auto scaling group during time T.

4.3 Simulation Setup

We have conducted numerous experiments to evaluate the proposed technique and demonstrate the effect of the EDoS attack against the cloud services. The simulation was designed to follow the same assumptions of EDoS-Shield work [24, 64]. Table 3 shows the parameters that have been used in the experiments.

The number of requests that target the cloud computing service is selected depending on the arrival rate in such a way to insure reaching the steady state. In addition, we repeated each experiment 10 times and the results are averaged. The Maximum Requests Per Second (MRPS) threshold represents the number of requests permitted for a single user during a second. The MRPS threshold is set dependent on the cloud application. We assume a value of 4 for the MRPS in the simulation [43].

We assumed the use of a small VM instance that has a capability to handle 100 HTTP Req/sec as it was discussed by Catteddu and Daniele [100]. The number of initial running servers in the auto scaling group is 5 servers. The number of computing resources that will be added per provisioning process is 2 instances [64]. Virtual machines provisioning requires some time until they can provide their services to manage the cloud resources

[101], we used 55.4 second as the provisioning overhead caused by committing instances to the cloud service as was measured by Islam et al. [67]. Furthermore, the server response packet size is 580 bytes [102]. Liu and Wee [103] reported that an Amazon EC2 instance can handle 800 Mbps when used as a load balancer. Therefore, we use Amazon's Elastic Load Balancer (ELB) with 5.8 μs, $(580 \times 8)/(800 \times 10^6)$, service time.

The average CPU utilization metric has been used for auto scaling and activating our mitigation technique purposes in the cloud services. We assumed the scaling-up upper and lower thresholds to be 80% and 75% respectively, whereas the scaling-down upper and lower thresholds are set to 35% and 30% respectively. The duration of the upper threshold timer is set to 5 minutes, while 1 minute has been assumed as the duration of the lower threshold timer. The aforementioned parameters are set based on [63, 64, 79]. The Attack Period Time (APT) has been set to 30 minute, where 80% of the attacks last less than this time in the considered set [70].

The cost of a VM instance $(Price_{VM})$ is set to \$0.115 per hour in EDoS-Shield work. This value will be used in our experiments for the comparison purpose with the EDoS-Shield work, but \$0.036 per hour is the current cost as it is recently reported in Amazon for small on-demand VM instances running on the Windows operating system. The price of the large on-demand VM instance, used by the LB and the EDoS-ADS, is \$0.134 per hour. However, \$0.48 per hour will be used as the price of the large on-demand VM instance to compare the cost with EDoS-Shield work [24]. According to the reported prices of Internet input and output data transfer of Amazon EC2, the bandwidth allocation cost is set to \$0.01 per in/out GB. The cost parameters are set based on [104].

Table 3 Experiment Parameters

Parameter	Value	Reference
Auto scaling metric	CPU usage	[64]
Scaling-up upper threshold	80%	[64]
Scaling-up lower threshold	75%	[63]
Scaling-down upper threshold	35%	[63]
Scaling-down lower threshold	30%	[43]
Upper threshold duration	5 min	[64]
Lower threshold duration	1 min	[43]
Cloud instance size	Small	[100]
Cloud instance service rate	100 HTTP Req/sec	[100]
Cloud instance cost	0.115 $/hr	[64]
Initial running servers	5	[64]
Auto scaling size	2	[64]
Provisioning overhead	55.4 sec	[67]
Web server response packet size	580 byte	[102]
Load balancer service time	5.8 µs	[103]
Flash Timer	1 min	[43]
APT	30 min	[70]
MRPS	4	[43]
Large instance cost	0.48 $/hr	[64]
Bandwidth allocation cost	0.01 $/GB	[104]
reCAPTCHA response time	13.06 sec	[60]
URL redirection overhead	0.63 sec	[105]

We assumed that attackers cannot respond to the reCAPTCHA Turing test. Conversely, we have chosen 92% as the success rate of legitimate users to solve reCAPTCHA requests [59], this assumption mimics the real life where some legitimate users cannot solve the reCAPTCHA for some reasons. The average time required by legitimate users to solve reCAPTCHA Turing test is set to 13.06 seconds according to 1000 arbitrarily tested users [59]. The URL redirection overhead is set to 0.63 seconds [103].

4.4 Simulator Validation

The simulation is validated by comparing obtained EDoS-Shield CloudSim simulation results with the EDoS-Shield results [64] for the cloud computing resources. We consider the legitimate users targeting the cloud service with 400 requests per second while the attackers targeting the cloud service with different attack rates ranging from 400 requests per second to 8000 requests per second to show the impact of the attack on the targeted cloud service. Each of the following subsections will compare the results obtained with the results presented in the EDoS-Shield work [64].

The number of cloud VM instances that will be used in the simulation experiments depends on the effective arrival rate. Since the mitigation technique will not be used during the validation process, we need to ensure that the incoming traffic to the cloud resources will not use more than the scale-up upper threshold of the processing resources, $\rho \leq 0.80$ where ρ can be calculated using Eq. (4.2). Hence, the number of required VM instances will be determined according to the following equation:

$$S_{\text{required}} = \lceil 1.25 \times (\lambda_l + \lambda_m)/\mu + 1 \rceil \qquad (4.9)$$

Where λ_l is the rate of legitimate traffic which was assumed as 400 Req/sec, λ_m is the EDoS attack rate targeting the cloud service, μ is the service rate of the cloud server. Figure 18 shows the number of required VM instances, for different arrival rates, used in the simulation based on Eq. (4.9).

The arrival rate will be distributed evenly among the cloud VM instances as shown in Figure 19. For example, if the legitimate users and malicious users target the cloud service with 400 Req/sec and 800 Req/sec respectively, the cloud service will be using 16 VM instances based on Eq. (4.9) and each VM instance will have 75 Req/sec as an arrival rate. In addition, the arrival rate for each VM instance will follow a Poisson distribution as mentioned in EDoS-Shield work.

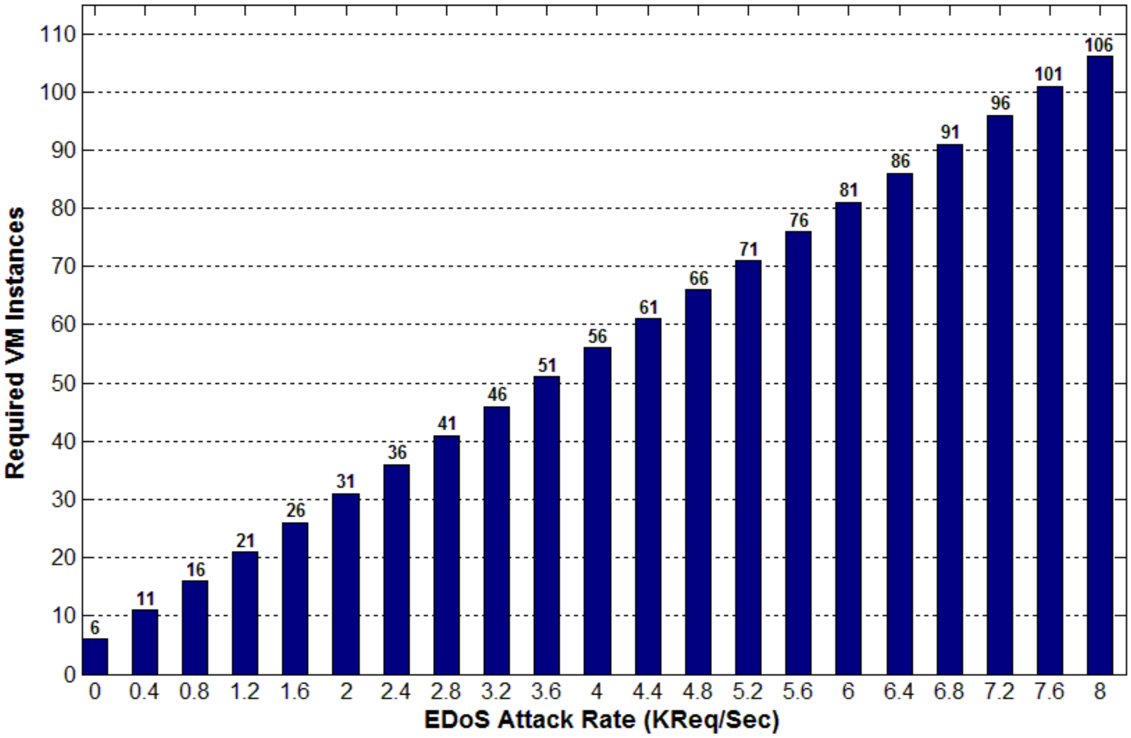


Figure 18 Number of Required VM Instances before Using the EDoS Mitigation Technique

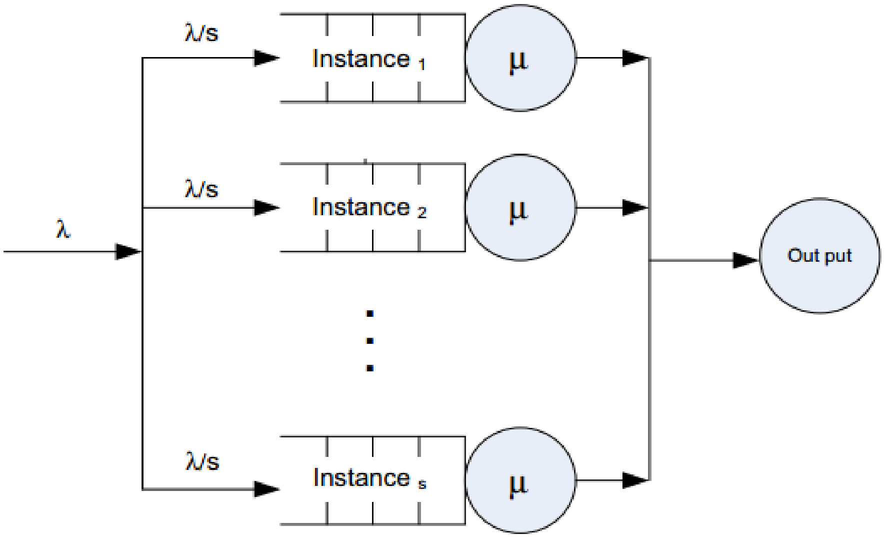


Figure 19 Queuing model for the Cloud Computing Resources [64]

4.4.1 Response Time

The EDoS-Shield CloudSim simulation and the EDoS-Shield [64] have been compared in terms of the end-to-end response time as shown in Figure 20. It is clear that both simulations have similar results for the response time with small differences due to the randomness of the simulation.

The results show that when the load increases, the corresponding end-to-end response time also increases. It is obvious that the response time does not go up considerably when the attack traffic significantly increases. This is due to the auto scaling mechanism that allocates more VM instances to process the high load caused by the attack traffic. However, the results in general show that the attack makes the legitimate clients suffer more response time compared to the optimal case where there is no attack targeting the cloud service and only legitimate users targeting the cloud service with 400 requests per second.

Figure 21 shows the relative error percentage for the cloud computing resources response time. The Relative Error Percentage (REP) can be calculated as follows:

$$REP = \left|\frac{Results_{EDoS-Shield} - Results_{Simulation}}{Results_{EDoS-Shield}}\right| \times 100 \tag{4.10}$$

The REP shows a good accuracy with an error of no more than 0.48% which corresponds to the maximum difference of 192.9 microseconds between the response time results of the EDoS-Shield CloudSim simulation and the EDoS-Shield [64].

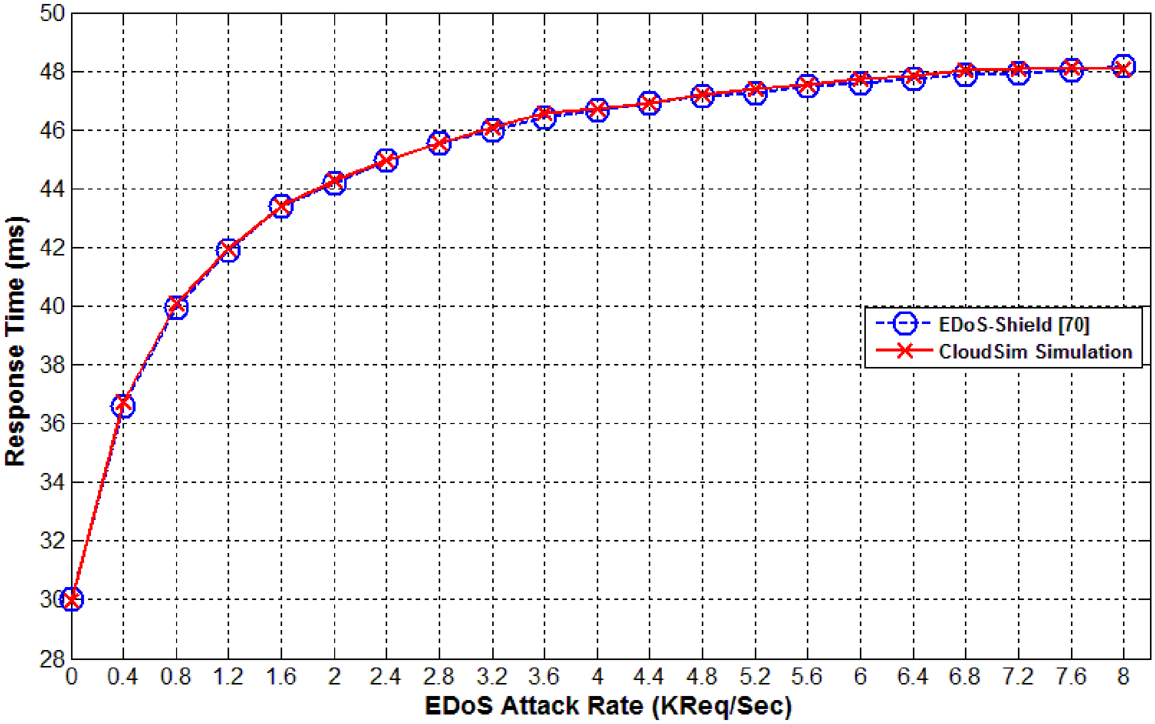


Figure 20 Response Time Results for EDoS-Shield and CloudSim Simulation

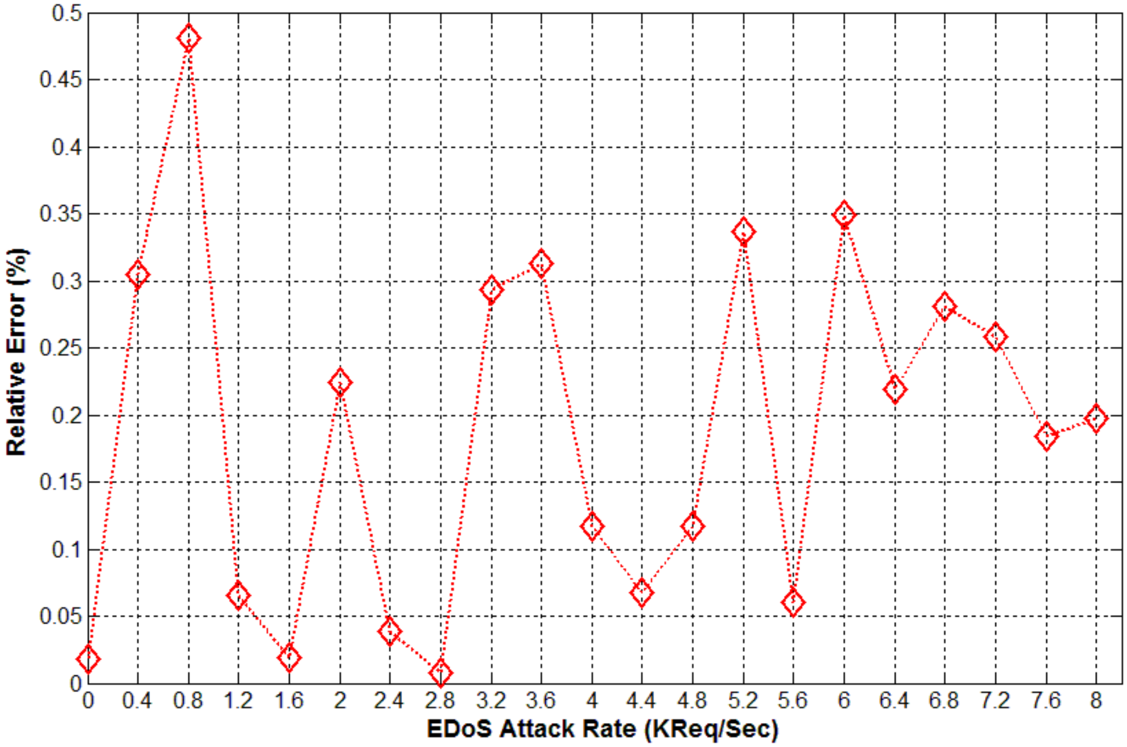


Figure 21 Response Time REP of EDoS-Shield and CloudSim Simulation

4.4.2 Resources Utilization

For the computing resources utilization, the outputs of both the EDoS-Shield CloudSim simulation and the EDoS-Shield [64] are identical as shown in Figure 22 with an error of no more than 0.075% between the CPU utilization results of the EDoS-Shield CloudSim simulation and the EDoS-Shield work as shown in Figure 23.

The results show that the computing resources utilization has a similar trend to the end-to-end response time in Figure 20, the CPU utilization increases whenever the attack rate increases. Thus, the EDoS attack consumes more computing resources when compared with the optimal case.

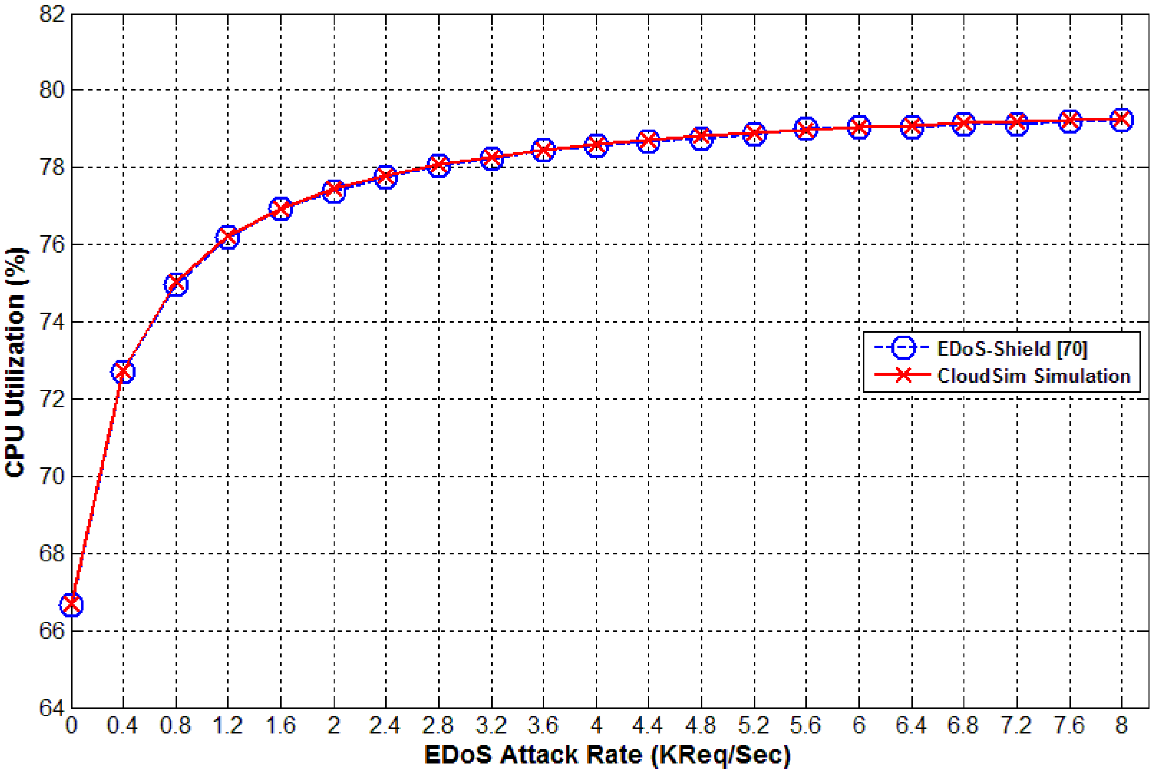


Figure 22 Resources Utilization Results for EDoS-Shield and CloudSim Simulation

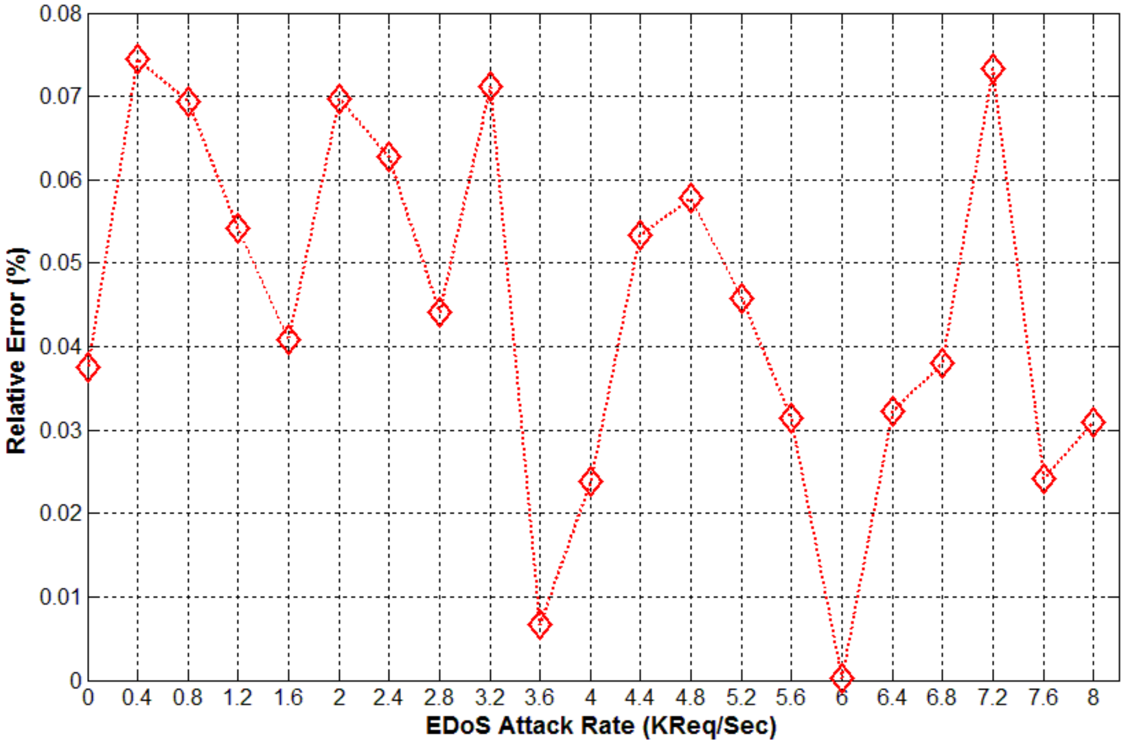


Figure 23 Resources Utilization REP of EDoS-Shield and CloudSim Simulation

CHAPTER 5

SIMULATION RESULTS AND ANALYSIS

In this chapter, we present the simulation results for the proposed mitigation technique. We have conducted four experimental scenarios using the simulation model discussed in chapter 3 and chapter 4. In the first scenario, we study the ***normal mode*** in which we consider different arrival rates of legitimate clients with enough VM instances to allow the cloud computing system to handle the clients requests without the need for auto scaling. Such a scenario is highly needed to study if there is an overhead on the cloud computing service due to the use of the proposed mitigation technique. The second scenario is for the ***flash overcrowd mode*** where a high amount of traffic is coming toward the cloud service from the legitimate clients. This scenario will be used to study the performance of the EDoS Attack Defense Shell (EDoS-ADS) while triggering the auto scaling and the effect of these flash arrivals of requests on the measured parameters. In the third scenario, we have considered an ***attack mode*** where a fixed legitimate arrival rate and different attack rates targeting the cloud service are generated by clients that do not belong to NAT-based networks. Such a scenario will be conducted to study the effectiveness of the proposed mitigation technique under the ***attack mode*** and will demonstrate the efficiency of the **Attack Shell** in detecting attackers. The last scenario is similar to the previous attack scenario, but the legitimate and the malicious clients that request the cloud service belong to the same NAT-based network. This scenario is importantly considered because it is the typical real live scenario, and that it is more serious than the previous scenario since both the legitimate and the malicious clients share the same NAT IP address. In addition, the results of the proposed mitigation technique are compared with the EDoS-Shield mitigation technique results [64].

5.1 Normal Mode Results

In this scenario, we used different legitimate arrival rates ranging from 400 Req/sec to 6400 Req/sec. The objective of this scenario is to find out if there is an overhead associated with the proposed mitigation technique even though the scenario does not consider any attack to the cloud services. In addition, the results of the proposed mitigation technique will be compared with the results of the EDoS-Shield mitigation technique (simulation case 1), the cloud computing system that uses of the auto scaling technique but without the use of an EDoS mitigation technique (simulation case 2), and the cloud computing system that uses neither the auto scaling technique nor the EDoS mitigation technique (simulation case 3).

5.1.1 Required VM Instances

We used different number of VM instances based on the arrival rates. The number of VM instances is selected in the same way that is used in the EDoS-Shield work [64] in order to compare the results and keep the CPU utilization below the auto scaling threshold. The EDoS-Shield work proposed the calculation of the required VM instances, as shown in

Eq. (4.9), to achieve less than 80% of CPU usage to examine the cloud system under the normal activity. Figure 24 shows the number of VM instances needed for different arrival rates. The results show that the required VM instances to handle the initial arrival rate, 400 Req/sec, is 6. It should be noted that the number of VM instances are constant for simulation case 3 since it does not have the ability to auto scale. On the other hand, the proposed mitigation technique, simulation case 1, and simulation case 2 are using varied number of VM instances for different arrival rates that follow Eq. (4.9) because they have the auto scaling feature.

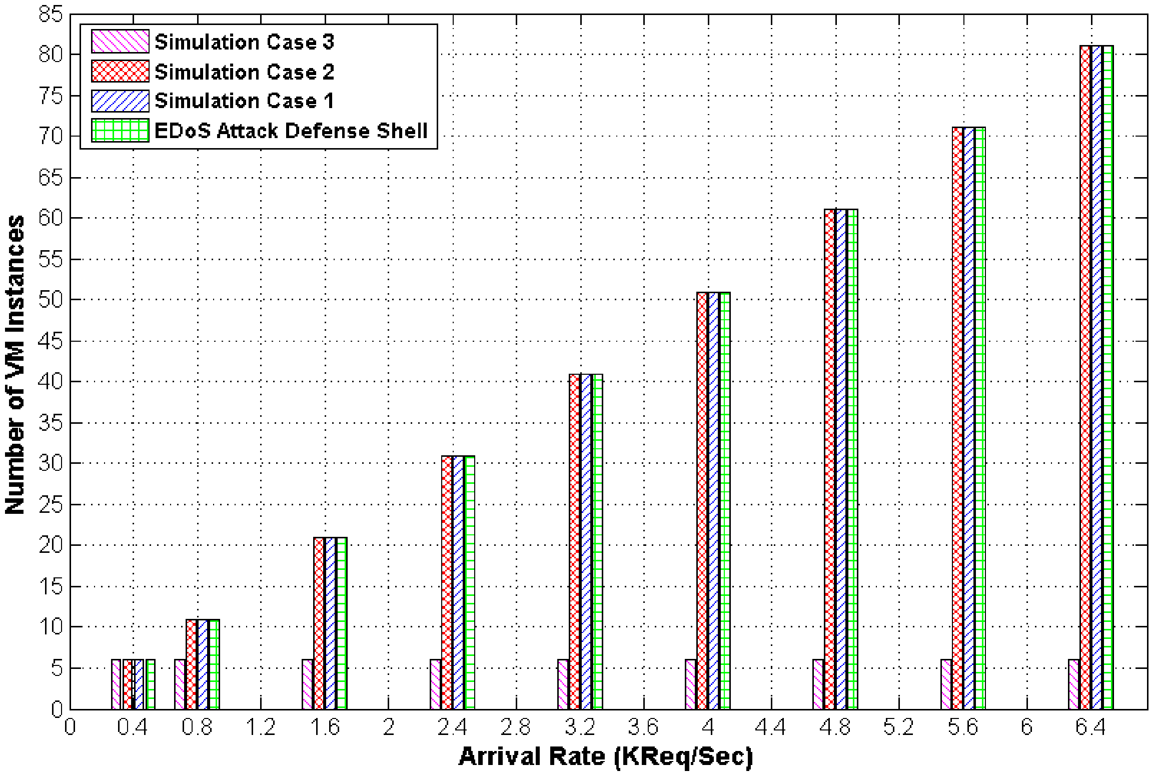


Figure 24 Number of Required VM Instances in the Normal Mode

5.1.2 Response Time

The average end-to-end response time along with the simulation time for our proposed technique at an arrival rate of 400 Req/sec with the use of 6 VM instances is shown in Figure 25. As shown, the average response time changes during the simulation until it reaches the steady state value of around 19 milliseconds.

On the other hand, the response time of the cloud computing system under simulation case 3 is depicted in Figure 26 and Figure 27. It is clear that the average end-to-end response time became extremely high in less than 3.5 minutes from the start of submitting requests to the cloud. Such a case will cause a Denial of Service (DoS) for all clients requesting the cloud service which makes the cloud system unavailable. In addition, the average response time increases exponentially with increasing arrival rate. This is due to the fact that the number of cloud VM instances is fixed, and that the requests will wait for a long time in the cloud VMs until they are serviced.

The end-to-end response time results for simulation case 2 have the same results as those obtained when using the EDoS Attack Defense Shell (EDoS-ADS) as shown in Figure 28. The results are identical because no overhead and no extra processing are required by the EDoS-ADS since the load balancer will forward all incoming requests directly to the cloud servers during this mode.

On the other hand, the simulation case 1end-to-end response time pertaining to the EDoS-Shield is higher than that for our proposed mitigation technique as depicted in Figure 28. This difference is related to the extra overhead on the firewall of the EDoS-Shield that is associated with the checking of the IP addresses of the incoming requests. Moreover, the authors of the EDoS-Shield work mentioned that the load balancer will distribute the clients requests on the cloud servers using a Round Robin mechanism and the inter arrival time of the incoming requests to the cloud load balancer is considered an exponential distribution. However, they actually assumed an exponential distribution for the inter arrival time of the clients requests on the cloud VM instances rather than on the cloud load balancer. Such a hypothesis is not valid as stated in [74-84] since it will make each VM instance in their simulation as a distinct cloud service.

On the other hand, the end-to-end response time results in our simulation have a similar trend to the end-to-end response time in the EDoS-Shield simulation. That is, when the load increases, the corresponding end-to-end response time also increases. The response time does not go up considerably when the arrival traffic significantly increases due to the allocation of more VM instances to process the high load.

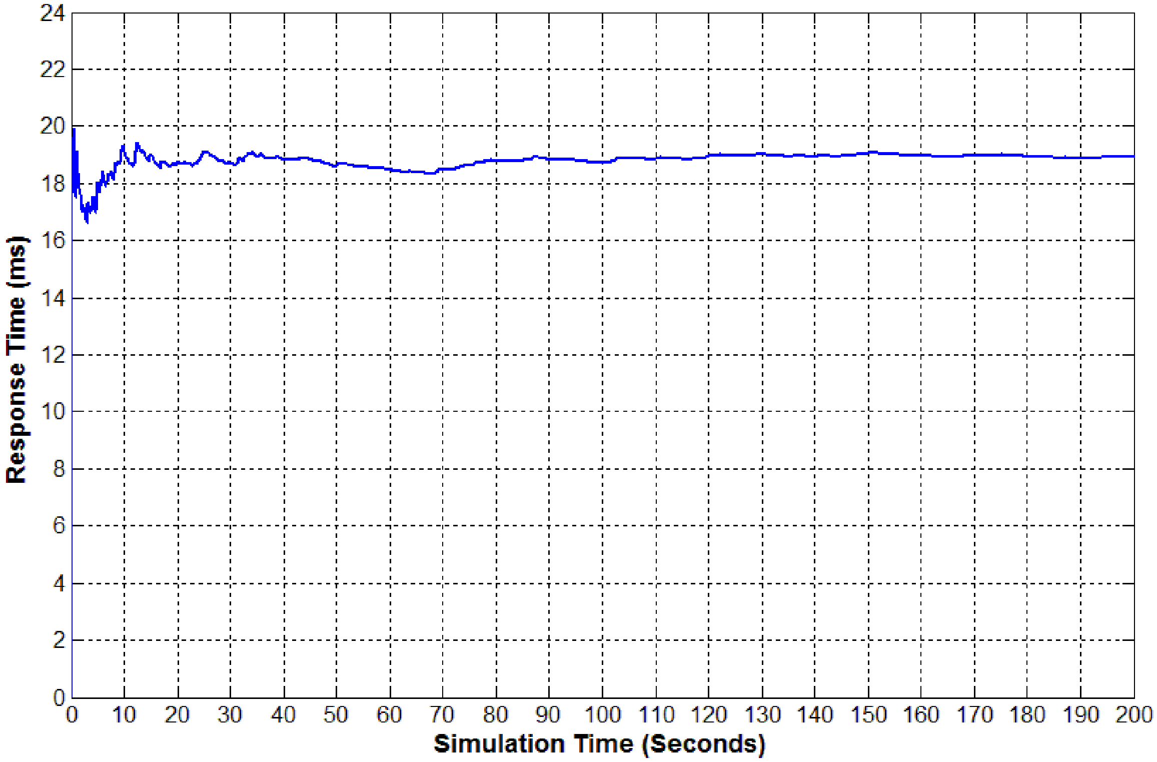


Figure 25 Response Time Evaluation for the EDoS Attack Defense Shell Simulation at Rate of 0.4 KReq/sec in the Normal Mode

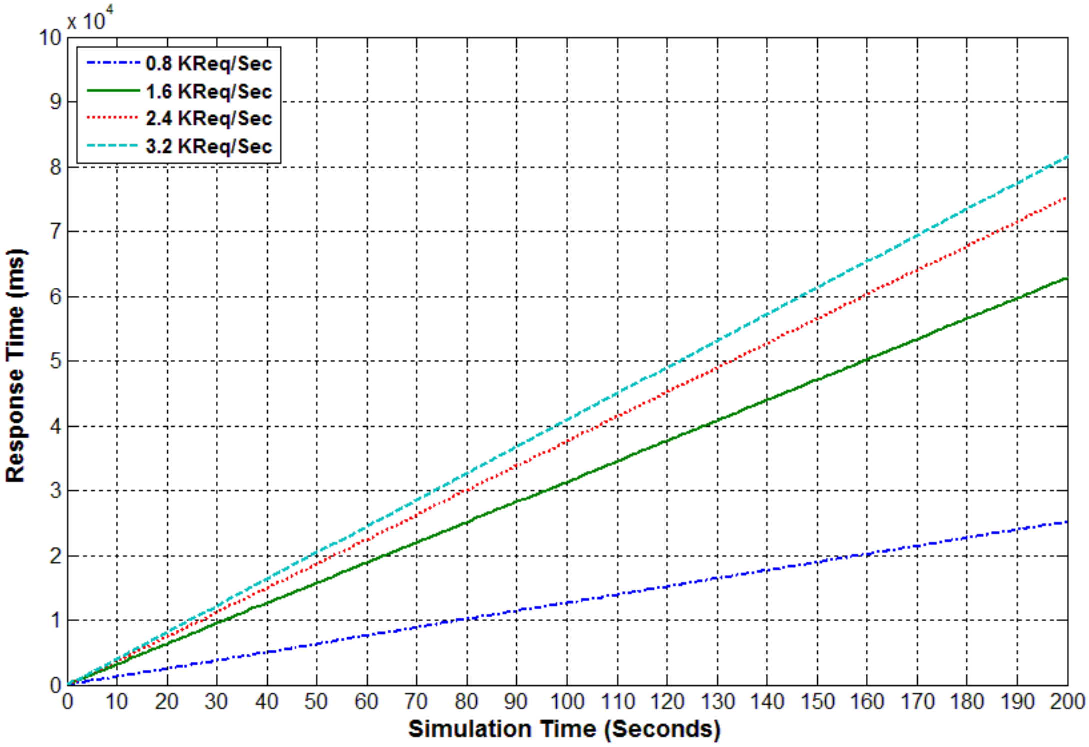


Figure 26 Response Time Evaluation for the Simulation without Auto Scaling and Mitigation Technique (Simulation Case 3) at Rates of 0.8, 1.6, 2.4, and 3.2 KReq/sec in the Normal Mode

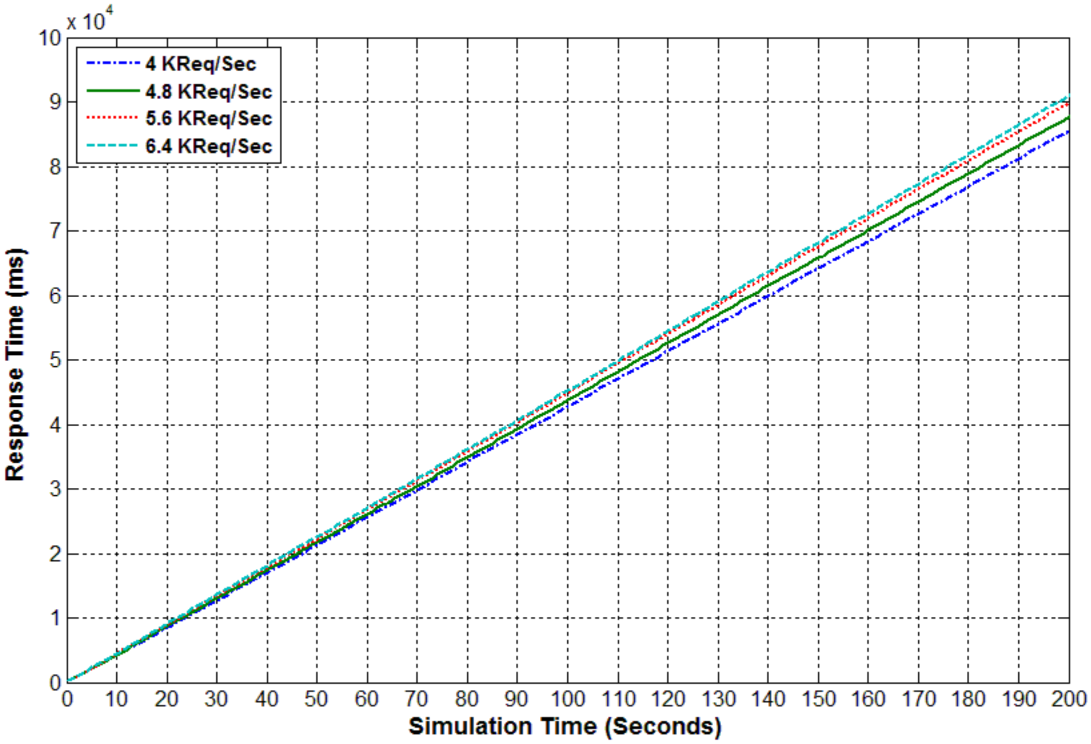


Figure 27 Response Time Evaluation for the Simulation Without Auto Scaling and Mitigation Technique (Simulation Case 3) at Rates of 4, 4.8, 5.6, and 6.4 KReq/sec in the Normal Mode

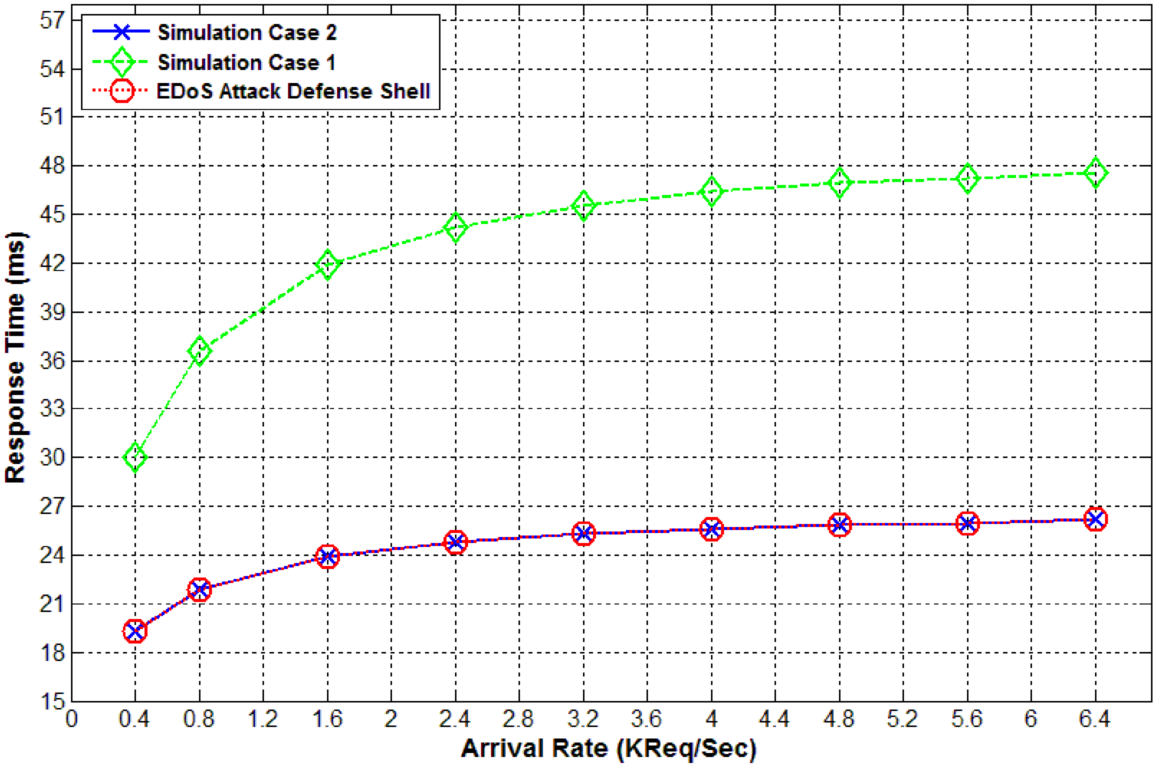


Figure 28 Response Time Evaluation in the Normal Mode

5.1.3 Resources Utilization

Figure 29 shows the average resources utilization for the EDoS Attack Defense Shell (EDoS-ADS) during the simulation time at rates of 400, 1600, and 6400 Req/sec under the ***normal mode***. Obviously, the average CPU utilization will change a lot during the first period of the simulation and after that it will reach the steady state when targeting the cloud service with a high number of requests.

Figure 30 compares the resources utilization of the EDoS-ADS against the three simulation cases for different arrival rates. The resources utilization results of the EDoS-ADS and the simulation case 2 are identical because no overhead and no extra processing will be incurred as a result of the EDoS-ADS since it will not be used in such a case.

Moreover, the results of both the EDoS-Shield simulation and the EDoS-ADS simulation are also identical as shown in Figure 30. The reason for having the same results for the resources utilization while having widely different results for the response time is due to the fact that each cloud VM instance receives equal amount of requests in both mitigation technique while the requests inter arrival time for each technique is different. The number of requests served by each used VM instance after 5 minutes of simulation at rates of 400 Req/sec and 800 Req/sec is shown in Figure 31 and Figure 32, respectively. Note that the increase in the number of served requests for simulation case 3 in Figure 32 is due to the limitation of a maximum of 6 VM instances for that case which makes these VM instances running at a 100% utilization rate.

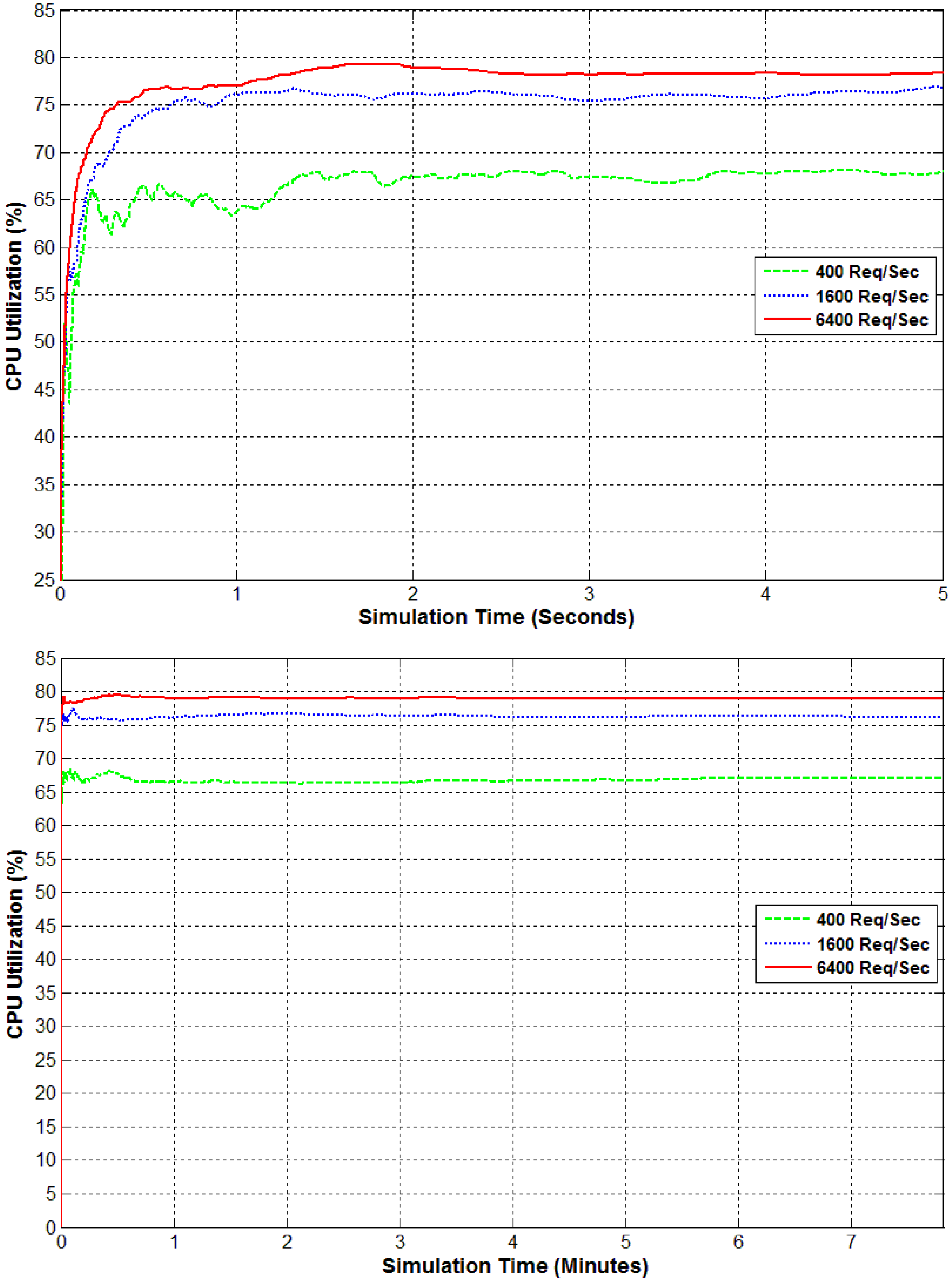


Figure 29 Resources Utilization Evaluation for the EDoS Attack Defense Shell Simulation at Rates of 0.4, 1.6, and 6.4 KReq/sec in the Normal Mode

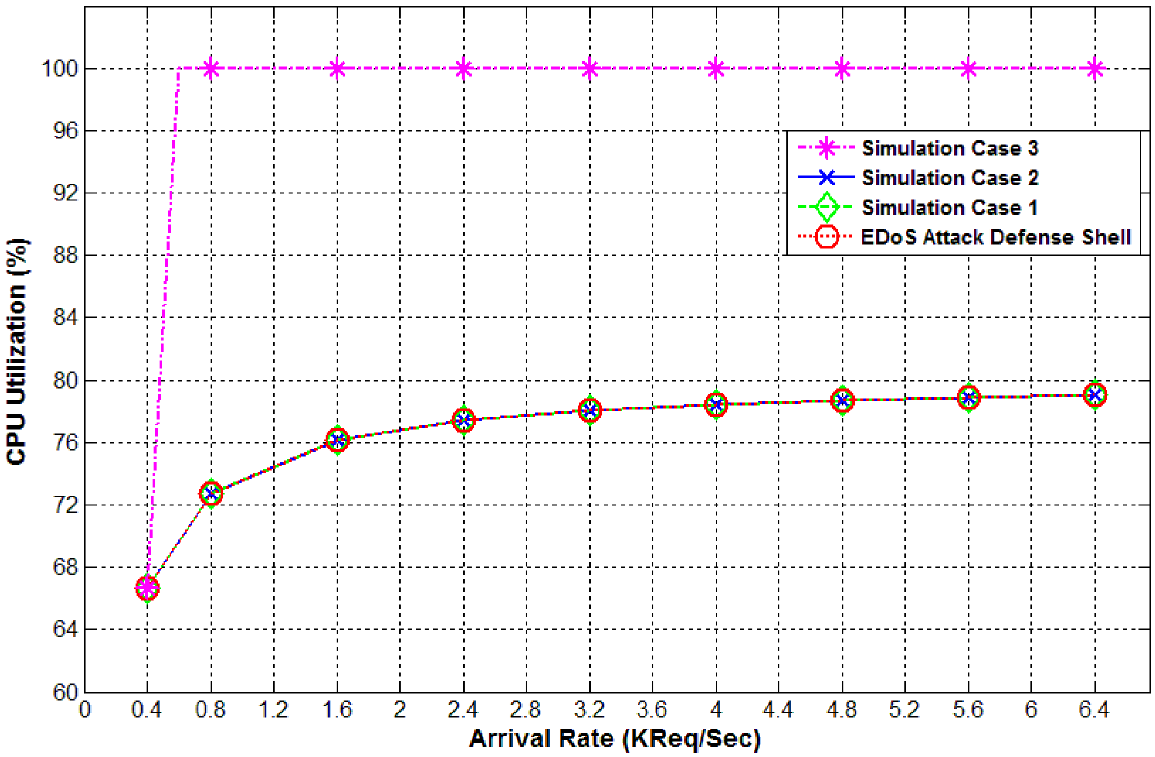


Figure 30 Resources Utilization Evaluation in the Normal Mode

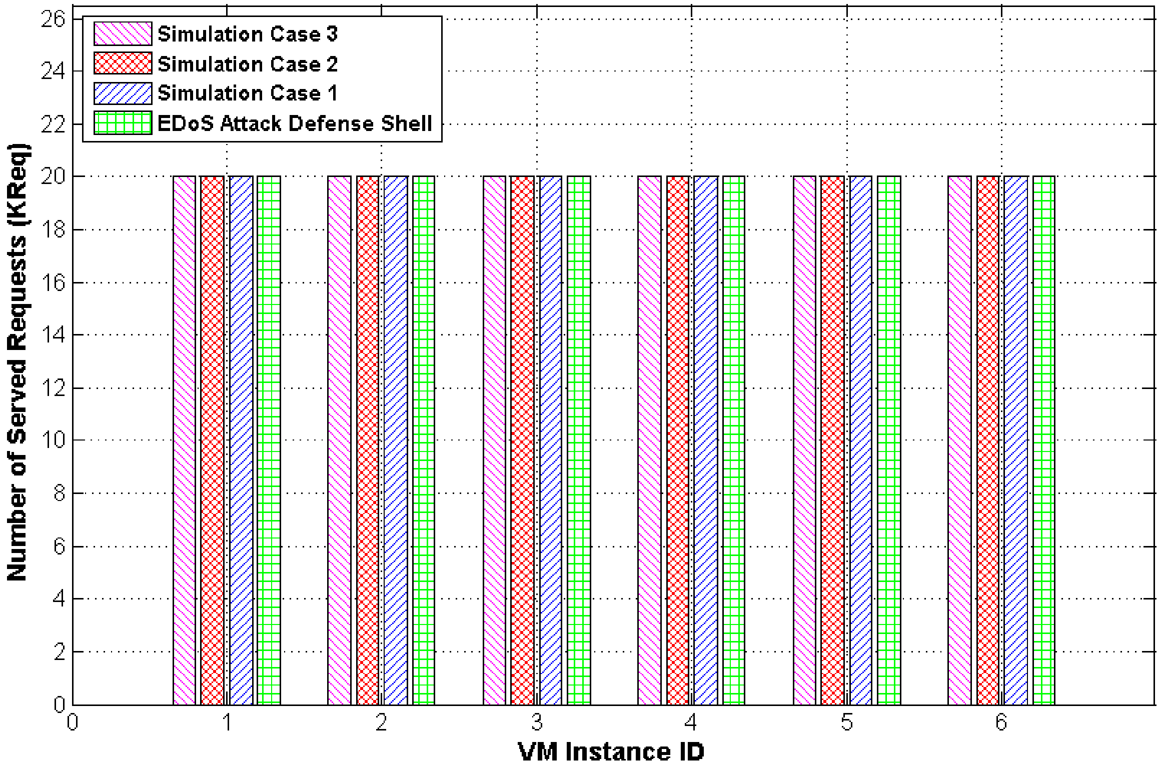


Figure 31 Number of Requests Served in each VM Instance after 5 Minutes of Simulation at Rate of 400 Req/sec

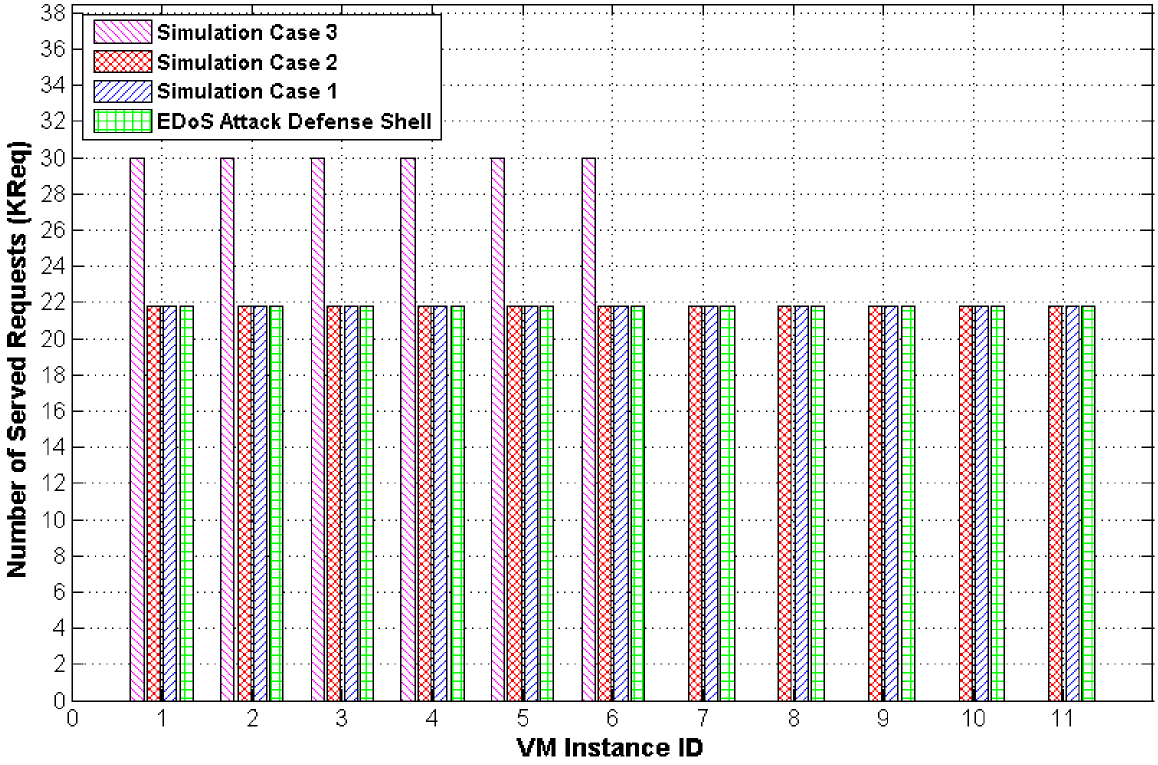


Figure 32 Number of Requests Served in each VM Instance after 5 Minutes of Simulation at Rate of 800 Req/sec

5.1.4 Throughput

The throughput evaluation in the ***normal mode*** for different arrival rates is shown in Figure 33. The throughput results are identical for the EDoS Attack Defense Shell (EDoS-ADS) and simulation case 1 and 2 for all considered arrival rates. This is due to having enough on-demand VM instances in the cloud service. In addition, the results show that the EDoS-ADS and simulation case 1 and 2 do not have an impact on the throughput rate.

On the other hand, the throughput of the cloud computing system under simulation case 3 has different trend when compared with the rest. This is because it has a fixed number of allocated VM instances. Subsequently, it will not be able to serve the incoming requests rate directly, and will result in a limited throughput rate.

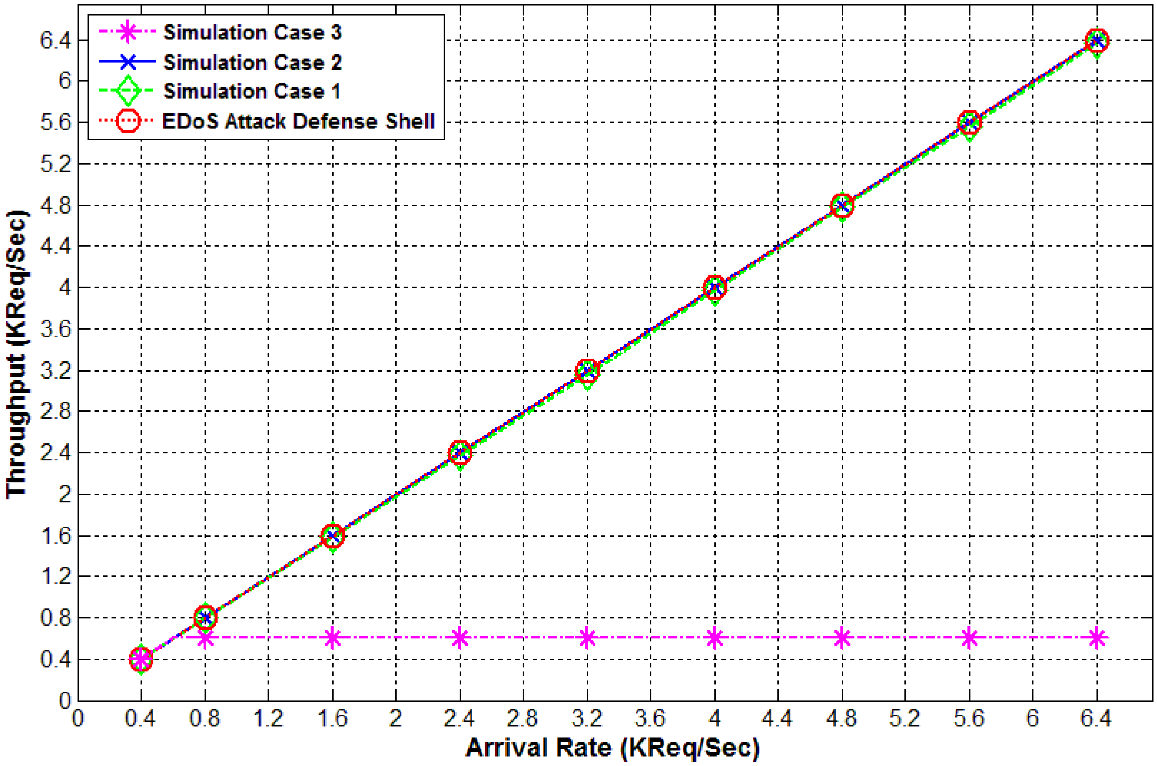


Figure 33 Throughput Evaluation in the Normal Mode

5.1.5 Cost

In this work, we follow the on-demand pricing model focusing on the costs that are related to the computing resources and network traffic volume [99]. In such a way, a cloud customer will pay for both bandwidth and computation usage. The cost has been calculated as shown in Eq. (4.8).

Figure 34 shows the cost evaluation for the EDoS Attack Defense Shell (EDoS-ADS) compared with the three simulation cases for different arrival rates. We considered the cost per month for both of bandwidth allocation and the allocated VM instances.

The costs are identical for the EDoS-ADS and simulation case 1 and 2 since they are all using the same number of VM instances and the average CPU utilization of the cloud computing system is also the same.

On the other hand, the cost of the cloud computing system for simulation case 3 has a different trend when compared with the rest since it has a fixed number of allocated VM instances. Subsequently, the maximum cost in such a system will be the cost of running the available VM instances at full utilization.

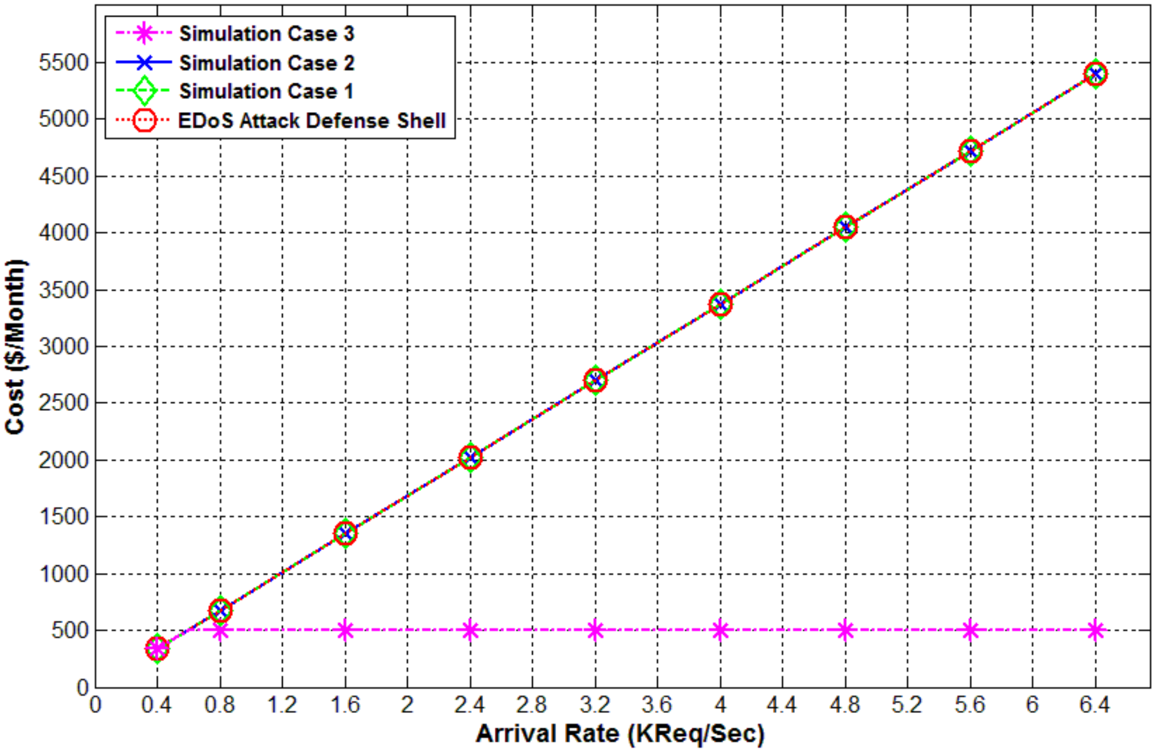


Figure 34 Cost Evaluation in the Normal Mode

5.2 Flash Overcrowd Mode Results

The ***flash overcrowd mode*** will occur when there is a large amount of legitimate traffic referred to as flash traffic that is targeting the cloud computing service. Recall from chapter 3 that the cloud system will be in the ***suspicion mode*** if the average CPU utilization exceeds the scaling-up upper threshold. Subsequently, the load balancer will redirect all incoming requests to the **Suspicion Shell** for differentiating between legitimate users and automated attackers as discussed in chapter 3.

In this scenario, we used different legitimate arrival rates ranging from 400 Req/sec to 6400 Req/sec. The legitimate clients are assumed to target the cloud computing service with CRPS less than the MRPS. In addition, there are no attackers targeting the cloud computing service in this scenario. For every arrival rate, the number of initial running VM instances is set to 5 VM instances. If the arrival rate is considered as flash traffic, the **Suspicion Shell** will request the allocation of 2 more VM instances to be added to the cloud computing service.

The objective of this scenario is to find out if there is an overhead associated with the proposed mitigation technique even though the scenario does not consider any attack to the cloud services. Moreover, this scenario will show the capability of the proposed mitigation technique to auto scale when having flash traffic. In addition, the results of the proposed mitigation technique will be compared with the results of the EDoS-Shield mitigation technique (simulation case 1), and the cloud computing system that uses of the auto scaling technique but without the use of an EDoS mitigation technique (simulation case 2).

5.2.1 Resources Utilization and Number of Allocated VM Instances

The **Suspicion Shell** will allocate new VM instances to the cloud computing service when there is a high amount of traffic coming from legitimate clients. The simulation case 2 follows the same setup mentioned in the EDoS-Shield work in which the CPU utilization will be checked periodically every 5 minutes. Subsequently, an additional 2 VM instances will be allocated after 55.4 seconds, referred to as the provisioning overhead, if the CPU utilization exceeds the 80%. The resources utilization and the number of allocated VM instances for the simulation case 2, and the EDoS Attack Defense Shell (EDoS-ADS) at rates of 0.4, 0.8, 1.6, 2.4, and 3.2 KReq/sec are shown in APPENDIX A. It is clear that the simulations of both cases have been allocated the same number of VM instances. However, the EDoS-ADS mitigation technique is faster in the allocation of these VM instances since it makes use of the clients collected information in the database. Subsequently, it changes the cloud mode to the ***flash overcrowd mode*** and allocates new VM instances to the cloud computing service accordingly.

We have conducted a simulation experiment similar to that presented in the EDoS-Shield work to evaluate the auto scaling mechanism in each technique as shown in Figure 35. The EDoS-Shield work assumed that the arrival rate is 200 Req/sec during the first 25 minutes of the simulation with the use of 5 VM instances as an initial VM instances in the cloud service. Subsequently, the arrival rate increases until it reaches 2400 Req/sec. Note that the results shown in the EDoS-Shield work reflect an ideal case where the newly allocated VM instances service the queued requests instantaneously. Therefore, these results match the analytical model, and can be considered, at best, optimal. The side effect of this on the EDoS-Shield is extremely high response time. The results show that the EDoS-ADS allocates 12 more VM instances compared with the optimal number of VM instances that needed to be allocated. However, the EDoS-ADS allocates the required VM instances with half of the time used to allocate the VM instances in the optimal case.

Figure 36 shows the comparison of the number of allocated VM instances for the simulation case 2, the EDoS-ADS, and the optimal case while considering different legitimate arrival rates. Note that the optimal case allocates the required number of VM instances analytically according to Eq. (4.9). Figure 37 shows the evaluation of the time to allocate the required number of VM instances until the CPU utilization becomes less than 80%. It is clear that our mitigation technique has reached the desired number of VM instances with less time when compared with the other techniques. This is due to the fast detection of such a ***flash overcrowd mode***.

Finally, the evaluation of the resources CPU utilization in the ***flash overcrowd mode*** for different arrival rates is shown in Figure 38. The CPU utilization results are identical for the simulation case 2 and the EDoS-ADS. This is because both of the techniques are using the same number of VM instances for serving the legitimate clients requests during the ***flash overcrowd mode***. However, the CPU utilization when using the EDoS-Shield is greater than the CPU utilization when using the EDoS-ADS since the EDoS-Shield uses less number of VM instances than the EDoS-ADS.

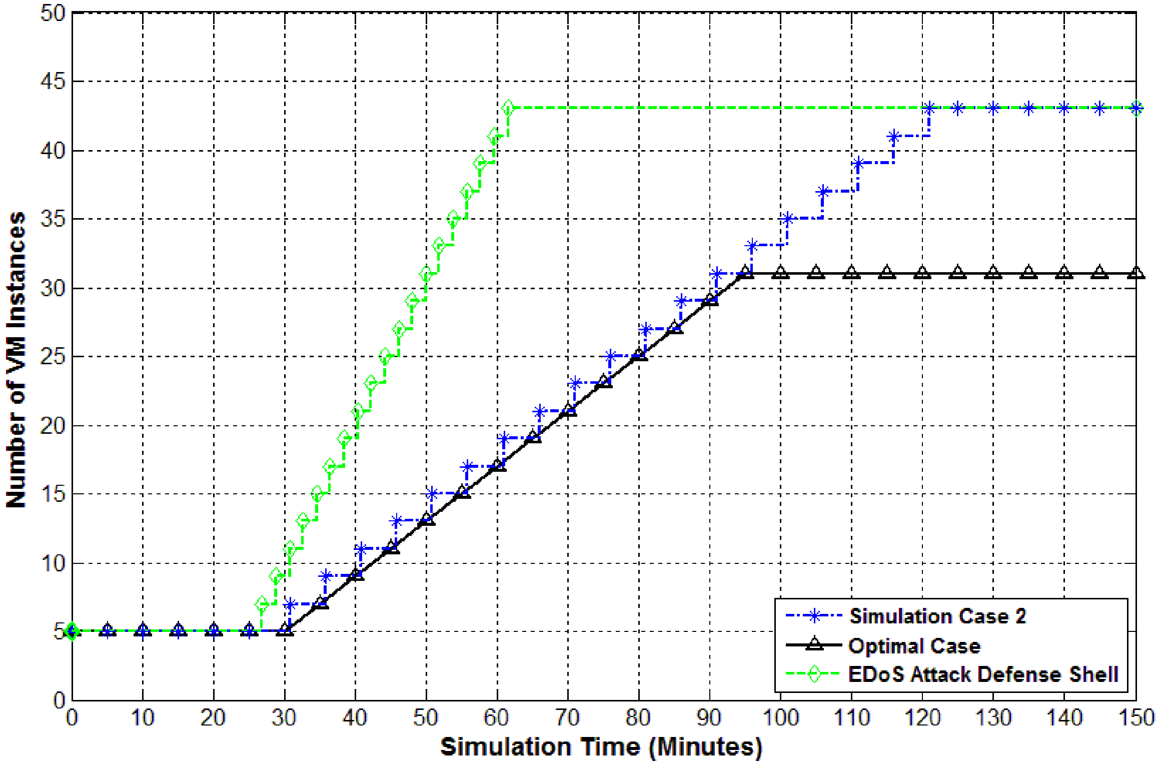


Figure 35 Evaluation of the Number of Allocated VM Instances at Rate of 2.4 KReq/sec in the Flash Overcrowd Mode

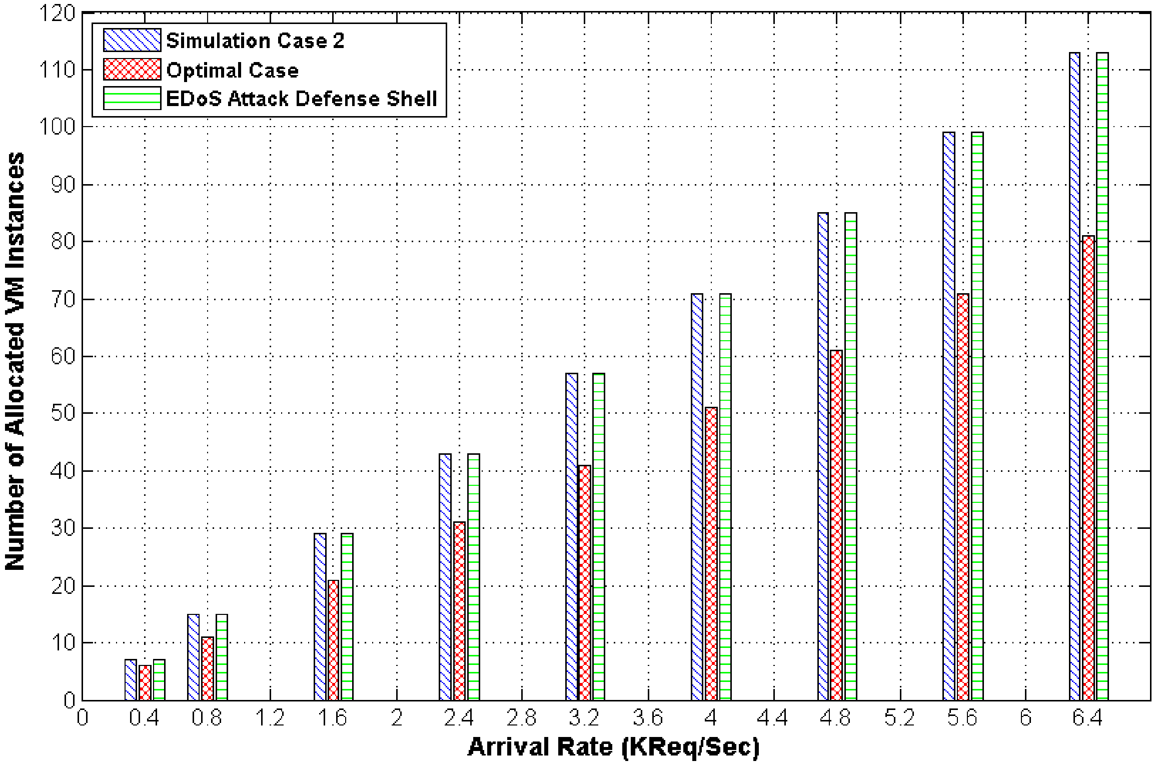


Figure 36 Evaluation of the Number of Allocated VM Instances

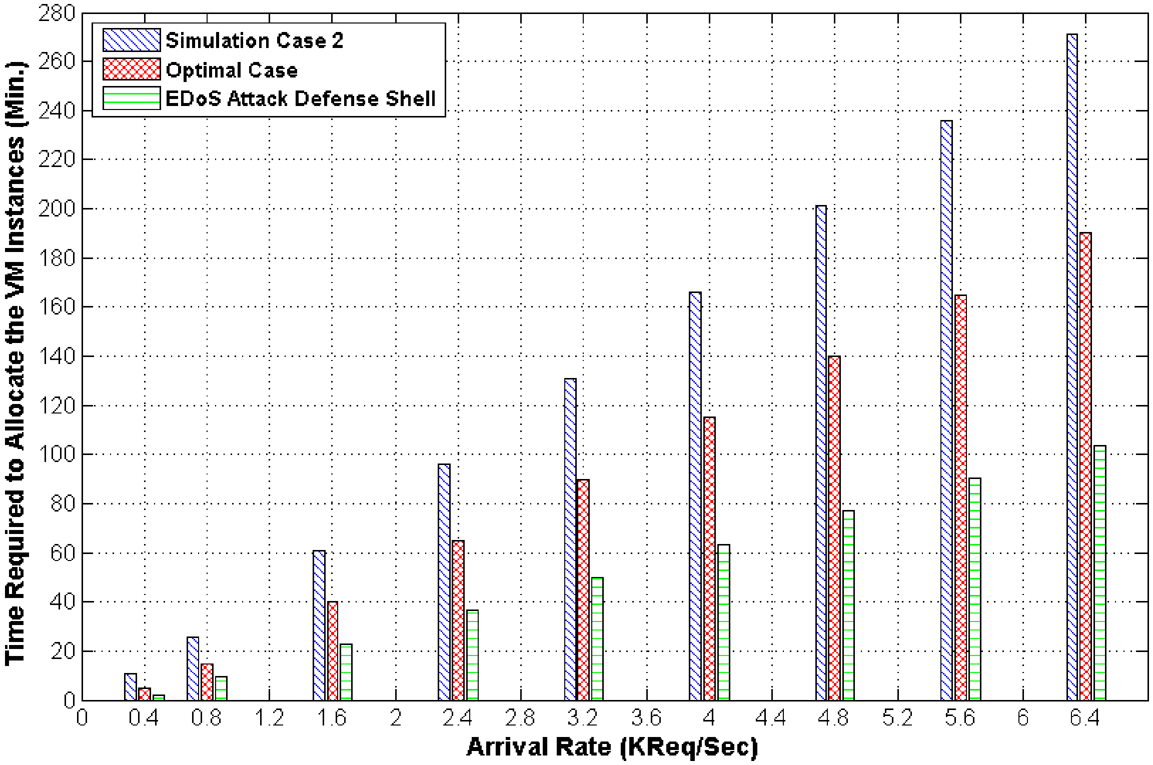


Figure 37 Evaluation of the Time to Allocate the Required Number of VM Instances

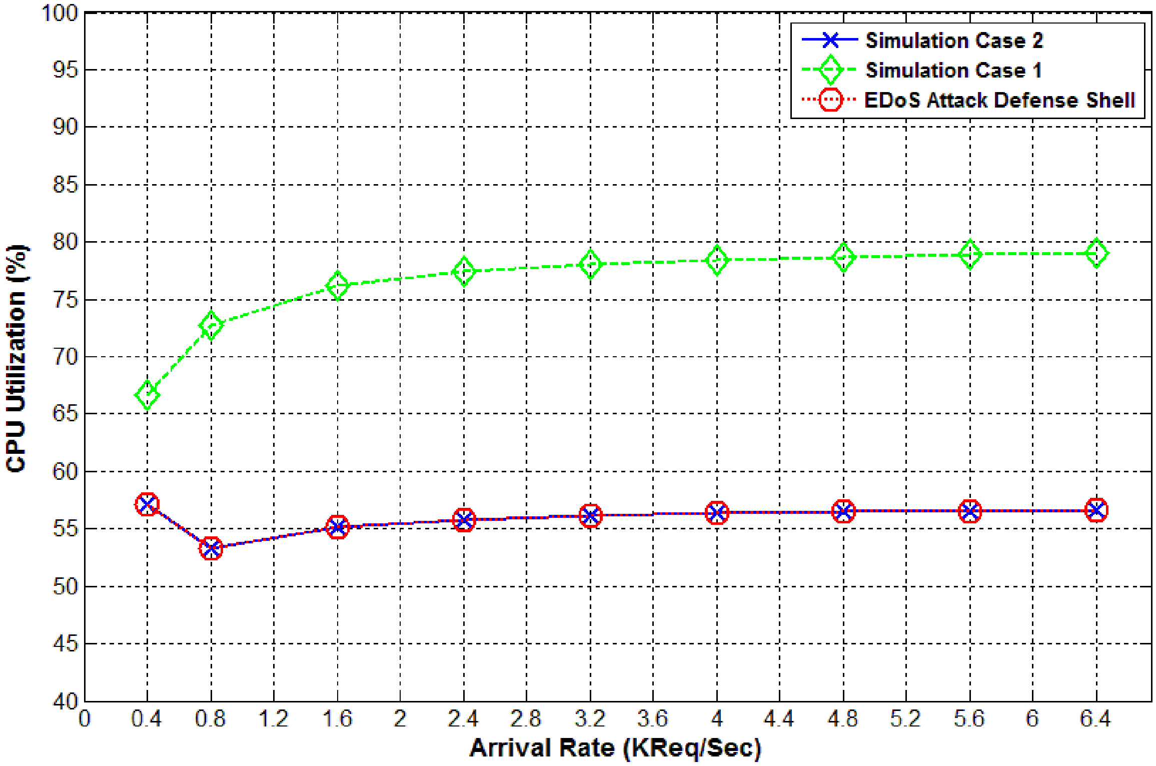


Figure 38 Resources Utilization Evaluation in the Flash Overcrowd Mode

5.2.2 Response Time

The time average end-to-end response time evaluation for the simulation case 2, and the EDoS Attack Defense Shell (EDoS-ADS) at rates of 0.4, 0.8, 1.6, 2.4, and 3.2 KReq/sec are shown in APPENDIX B. It is clear that the EDoS-ADS simulation has less average response time when compared with the average response time of simulation case 2. This is because of the faster allocation of new VM instances in the case of the EDoS-ADS.

The evaluation of the average response time in the ***flash overcrowd mode*** for different arrival rates and for different simulation cases is shown in Figure 39 and Figure 40. For all simulation cases considered, the response time does not go up considerably when the arrival traffic significantly increases. This is mainly due to the auto scaling mechanism that allocates sufficient VM instances to process the additional load. The response time results are almost identical for simulation case 2 and the EDoS-ADS simulation. This is due to that both of the techniques are using the same number of VM instances for serving the legitimate clients requests during the ***flash overcrowd mode***. Note that the average processing time of the large VM instances used for the EDoS-ADS does not add significantly to the response time since it is extremely low, around 6 microseconds for the rate of 6.4 KReq/sec. However, the response time when using the EDoS-Shield is greater than the response time when using the EDoS-ADS due to the different number of VM instances allocated to the cloud service. Note that the less number of VM instances in the EDoS-Shield results in large queuing delay which in turns results in higher response time. In addition, the EDoS-Shield has a much higher response time than the EDoS-ADS due to the extra overhead on the firewall of the EDoS-Shield that is associated with the checking of the IP addresses of the incoming requests, and the EDoS-Shield assumption of an exponential distribution for the inter arrival time of the clients requests on the cloud VM instances rather than on the cloud load balancer.

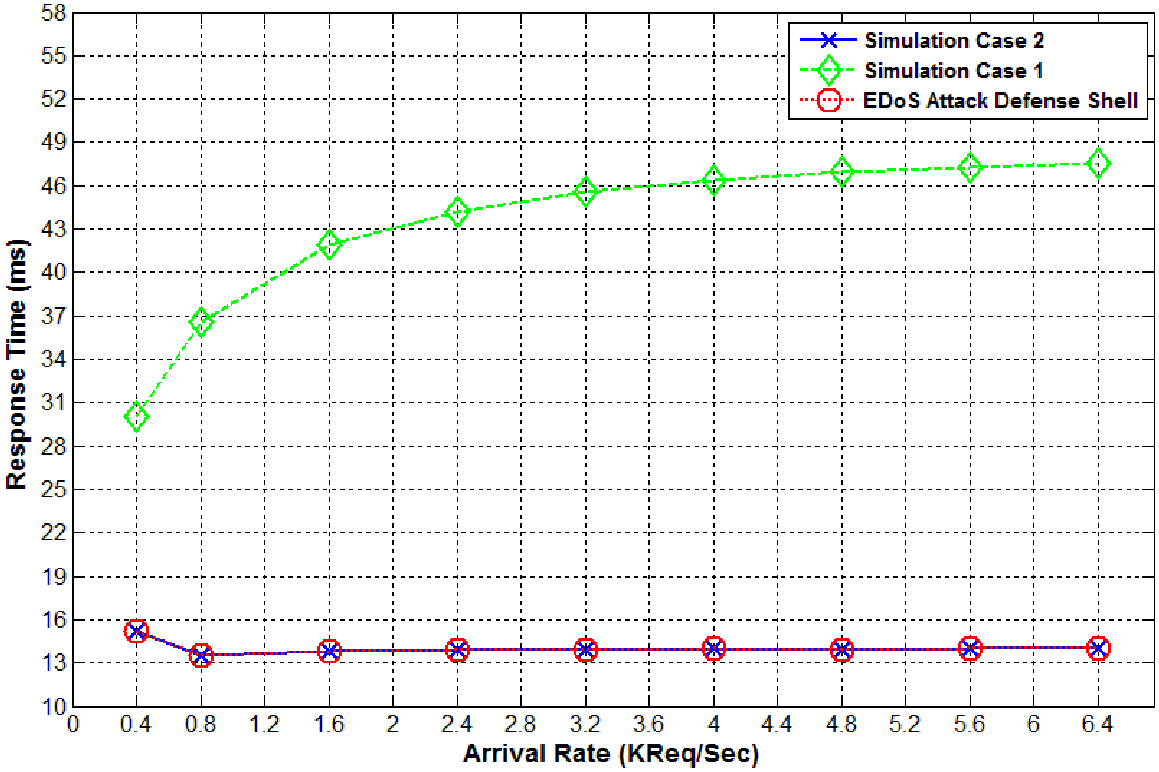


Figure 39 Response Time Evaluation in the Flash Overcrowd Mode

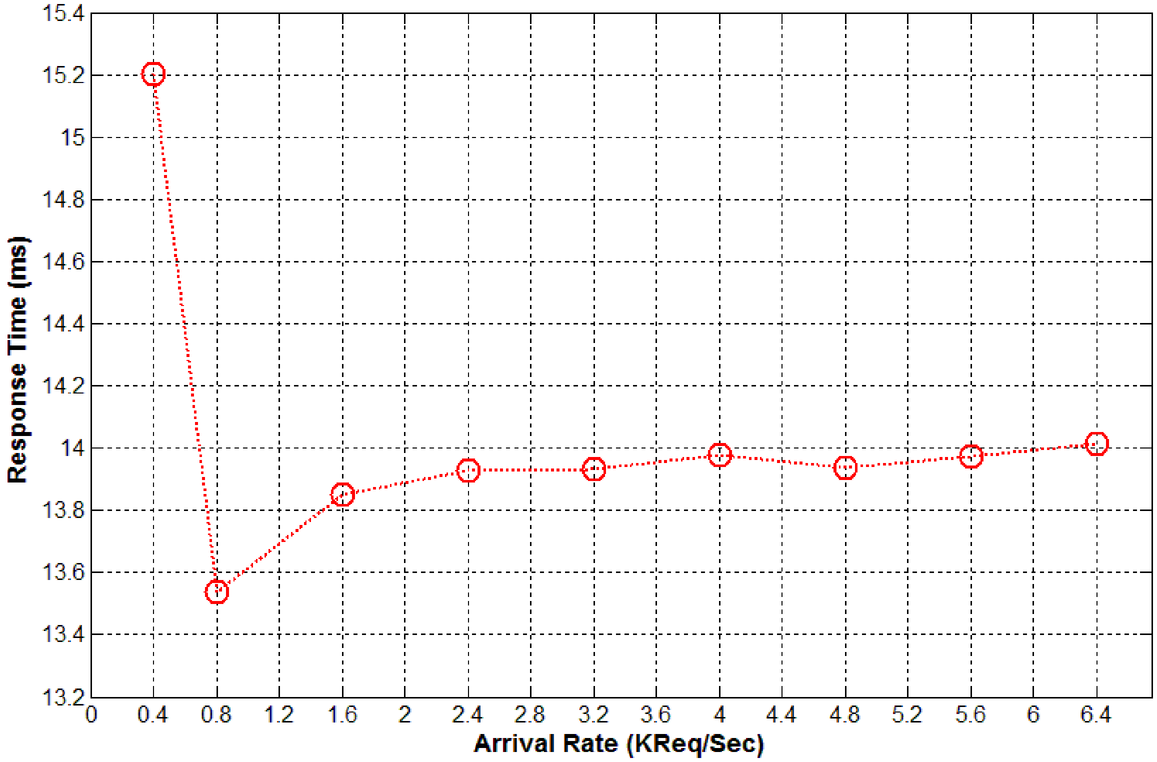


Figure 40 Response Time Evaluation for EDoS Attack Defense Shell in the Flash Overcrowd Mode

5.2.3 Throughput

The throughput evaluation for the simulation case 2, and the EDoS Attack Defense Shell (EDoS-ADS) at rates of 0.4, 0.8, 1.6, 2.4, and 3.2 KReq/sec are shown in APPENDIX C. It is clear that the EDoS-ADS simulation has more throughputs when compared to simulation case 2. This is due to the faster allocation of new VM instances in the case of the EDoS-ADS.

Figure 41 shows the throughput evaluation in the ***flash overcrowd mode*** for different arrival rates and for different simulation cases. The throughput of the cloud computing system is the same for all simulation cases considered due to the elasticity nature of the cloud computing system in which VM instances being allocated as needed.

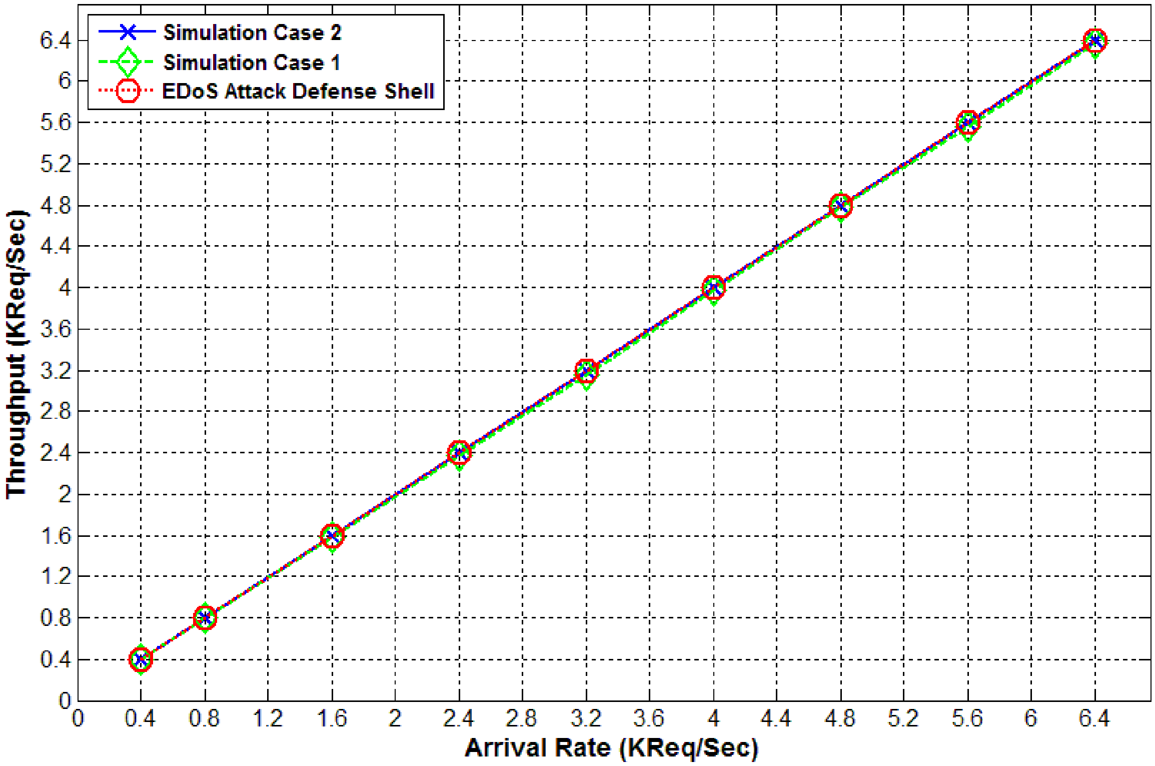


Figure 41 Throughput Evaluation in the Flash Overcrowd Mode

5.2.4 Cost

The cost is computed according to Eq. (4.8). In such a way, a cloud customer will pay for both bandwidth and computation usage. Figure 42 shows the cost evaluation for the EDoS Attack Defense Shell (EDoS-ADS) compared with the other techniques in the ***flash overcrowd mode*** for different arrival rates. We considered the cost of the bandwidth allocation and the allocated VM instances for 10 hours [24].

The costs are identical for the cloud computing system for simulation case 2 and the EDoS-ADS. This is because they allocate the same number of VM instances and the average CPU utilization of the cloud computing system is also the same for these techniques. Note that the cost of the EDoS-ADS large VM instance is extremely low since its CPU utilization is less than 1%. Therefore, the cost of the EDoS-ADS is not affected by the large cost of the large VM instance. Moreover, the cost for the EDoS-Shield is almost identical for the cost of the EDoS-ADS even when it allocates less VM instances when compared with the VM instances allocated in the EDoS-ADS. This is because the average CPU utilization in the EDoS-Shield VM instances is greater than that of the EDoS-ADS.

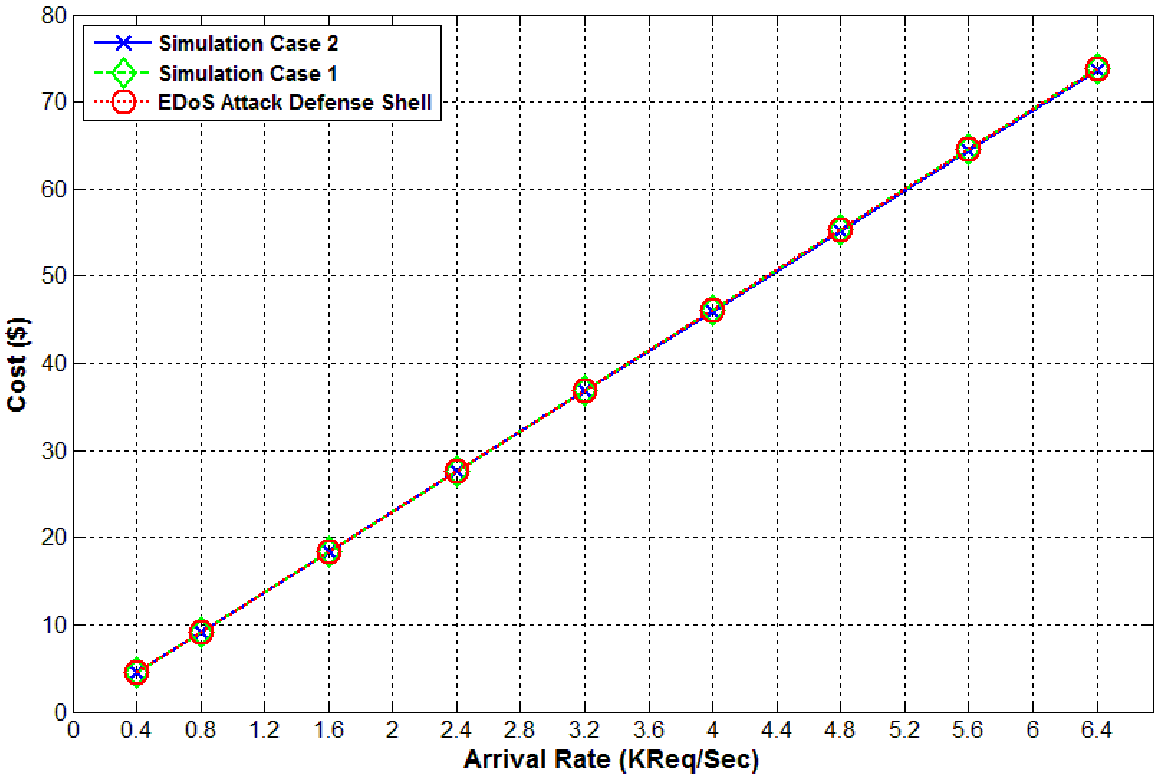


Figure 42 Cost Evaluation in the Flash Overcrowd Mode

5.2.5 reCAPTCHA Turing Test

The Maximum Requests Per Second (MRPS) greatly influences the amount of reCAPTCHA testing required from a client. Subsequently, if the legitimate clients seek to target the cloud service with Concurrent Requests Per Second (CRPS) greater than the MRPS during the ***flash overcrowd mode***, they will be asked to increment their TF level in order to avoid receiving additional reCAPTCHA Turing test. Such clients can achieve higher TF level by successfully answering the reCAPTCHA Turing tests as shown in Figure 12 until they reach the good TF level. Figure 43 shows the number of required reCAPTCHA Turing tests considering different values for both the MRPS and CRPS before reaching a good TF level. The results show that the legitimate clients will be requested to answer 5 reCAPTCHA Turing tests in average when their CRPS is larger than the MRPS. On the other hand, the requests of clients requesting the cloud service with CRPS less than or equal to MRPS will be forwarded to the cloud servers immediately without requesting those clients to answer the reCAPTCHA Turing tests.

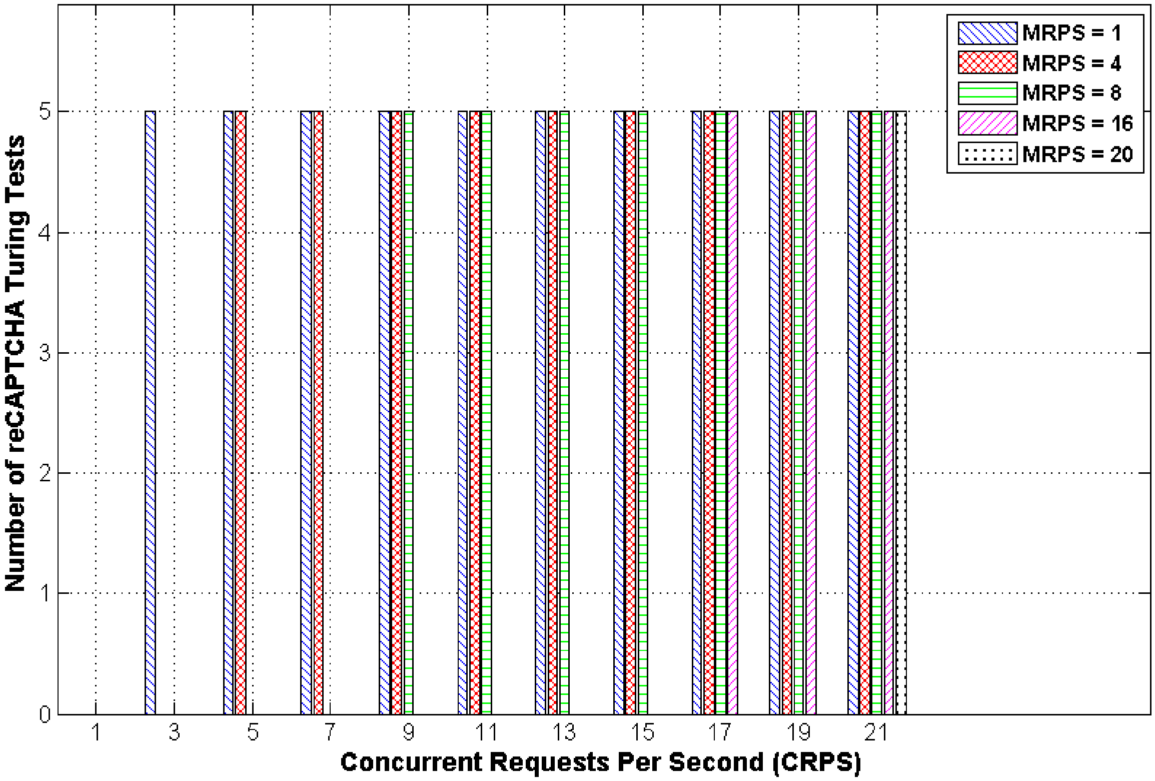


Figure 43 reCAPTCHA Turing Test Evaluation in the Flash Overcrowd Mode

5.3 Attack Mode Results - Legitimate Clients and Attackers do not belong to NAT-based Networks

In this scenario, we consider having legitimate clients and attackers targeting the cloud resources to evaluate the proposed mitigation technique under the EDoS attack. The system will start working in the ***suspicion mode*** when the current system utilization exceeds the scaling-up upper threshold. In this case, the cloud service load balancer will redirect all incoming requests to the EDoS Attack Defense Shell for a validation period. In this validation period, the **Suspicion Shell** will send a reCAPTCHA and URL redirection packets to the clients to differentiate between the ***flash overcrowd mode*** and the ***attack mode*** as explained in chapter 3. Upon detection of an attack behavior, the cloud system mode is changed to an ***attack mode*** and all incoming requests will be redirected to the **Attack Shell**.

This scenario assumes that the legitimate clients and attackers do not belong to NAT-based networks. Thus, the IP addresses for the legitimate clients and attackers will be dissimilar. We used different attack rates ranging from 400 to 6000 Req/sec to evaluate the proposed mitigation technique under the EDoS attack. However, the legitimate arrival rate is assumed to be fixed with a rate of 400 Req/sec, and each legitimate client transmits at a rate less than MRPS. We compare the EDoS Attack Defense Shell results with the results of the EDoS-Shield mitigation technique (simulation case 1), and the cloud computing system that uses of the auto scaling technique but without the use of an EDoS mitigation technique (simulation case 2). For every attack rate, the number of initial running VM instances is set to 7, 6 and 7 VM instances for the EDoS-ADS, the simulation case 1, and the simulation case 2, respectively, according to the results obtained previously in subsection 5.1.2

The EDoS-Shield mitigation technique will be vulnerable to IP address spoofing problem in such a scenario. So, we consider three cases for the EDoS-Shield mitigation technique, these cases are the EDoS-Shield optimal case, the EDoS-Shield whitelist case, and the EDoS-Shield blacklist case. The EDoS-Shield optimal case refers to the case when there is no spoofing of IP addresses. Thus, all legitimate clients' IP addresses are in the whitelist and all attackers' IP addresses are in the blacklist. The EDoS-Shield whitelist case refers to when the attackers spoof IP addresses that are already in the whitelist. Therefore, the EDoS-Shield will consider these attackers as legitimate clients. Subsequently, their traffic will access the cloud computing service. Finally, the EDoS-Shield blacklist case is used to describe the EDoS-Shield when the attackers carry out an attack using spoof IP addresses that are currently neither in the whitelist nor in the blacklist. Considering that these spoofed IP addresses were not used before to access the cloud services, these spoofed IP addresses will be added to the blacklist. Therefore, the EDoS-Shield will consider the legitimate clients with the same IP addresses as attackers. Subsequently, the legitimate clients will not be able to access the cloud service because their IP addresses are already in the blacklist.

5.3.1 Resources Utilization and Number of Allocated VM Instances

Figure 44 shows the comparison of the number of allocated VM instances for the simulation case 1, the simulation case 2, and the EDoS Attack Defense Shell (EDoS-ADS) while considering different attack arrival rates. It is clear that both of the simulation case 2 and the simulation case 1 (whitelist case) have been allocated higher number of VM instances compared to the others because they are considering the attack traffic as flash traffic. As a result, both of the simulation case 2 and the simulation case 1 (whitelist case) will auto scale to serve the incoming flash traffic as discussed in subsection 5.2.1. On the other hand, the simulation case 1 (optimal case) has used the initial number of VM instances in order to serve the legitimate traffic according to EDoS-Shield analytical model. Similarly, the simulation case 1 (blacklist case) has used the initial number of VM instances for all of the considered attack rates, and did not perform auto-scaling. This is because both of the legitimate clients and attackers were unable to access the cloud service since their IP addresses are already in the blacklist. Finally, the EDoS-ADS has used the initial number of VM instances for both cases where we consider the attack requests and where we ignore those requests originated by attackers. Thus, the proposed EDoS-ADS will completely eliminate the EDoS attack.

The evaluation of the resources CPU utilization in the ***attack mode*** for different attack arrival rates is shown in Figure 45. The CPU utilization results are identical for the case when the EDoS-ADS only considers the legitimate traffic and the case when the EDoS-ADS considering both legitimate and attack traffic. This is because both cases are using the same number of VM instances for serving the legitimate clients requests during the

attack mode. However, the CPU utilization when using the simulation case 1 (optimal case) is greater than the CPU utilization when using the EDoS-ADS since the simulation case 1 (optimal case) uses less number of VM instances than the EDoS-ADS.

Note that based on Figure 44 the simulation case 1 (whitelist case) has been allocated more VM instances than the EDoS-ADS as a result of the attack traffic. The simulation case 1 (whitelist case) considers the attack traffic as flash traffic. Subsequently, the attack traffic will be served by the cloud VM instances in such scenario. Hence, the CPU utilization when using the simulation case 1 (whitelist case) is higher than the CPU utilization when using the EDoS-ADS as evident from Figure 45. In contrast, in the simulation case 2 more VM instances are allocated than the simulation case 1 (whitelist case) that result in lowering the CPU utilization to be almost identical to the CPU utilization of the EDoS-ADS.

It should be noted that the CPU utilization when using the simulation case 1 (blacklist case) is zero because the legitimate clients requests are dropped since their IP addresses are already in the blacklist.

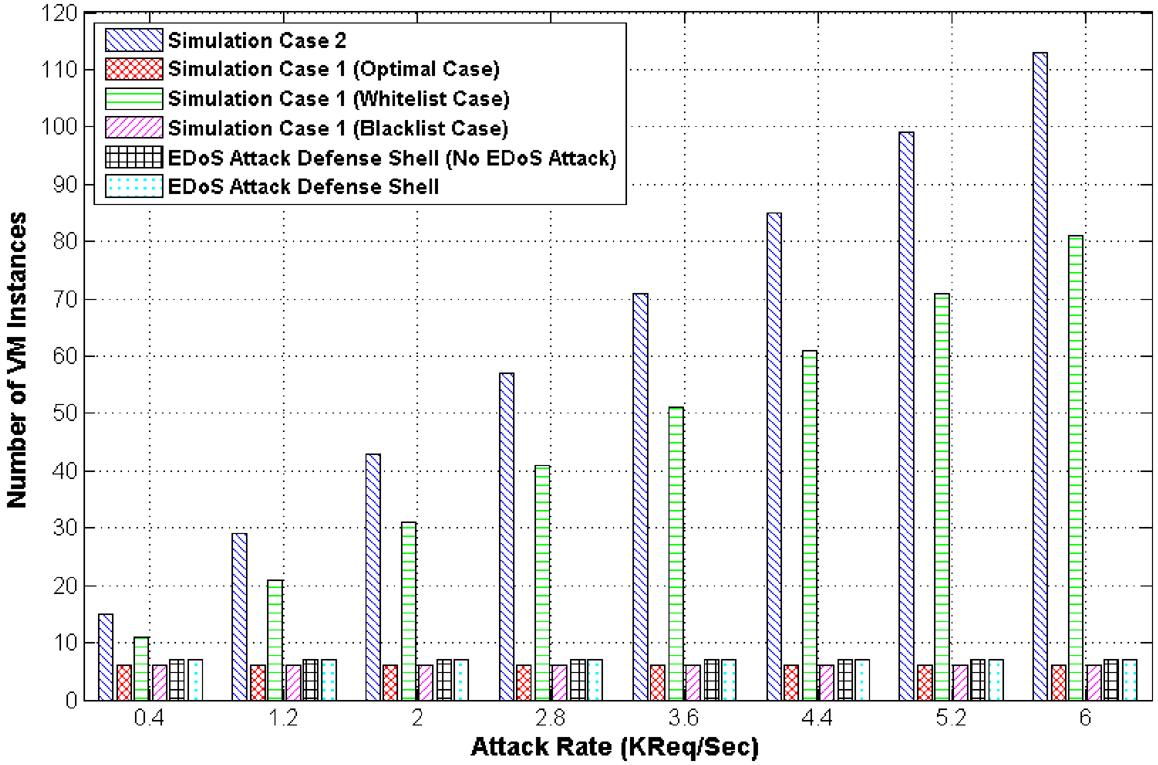


Figure 44 Evaluation of the Number of Allocated VM Instances in the Attack Mode with the Use of Different NAT-based Networks

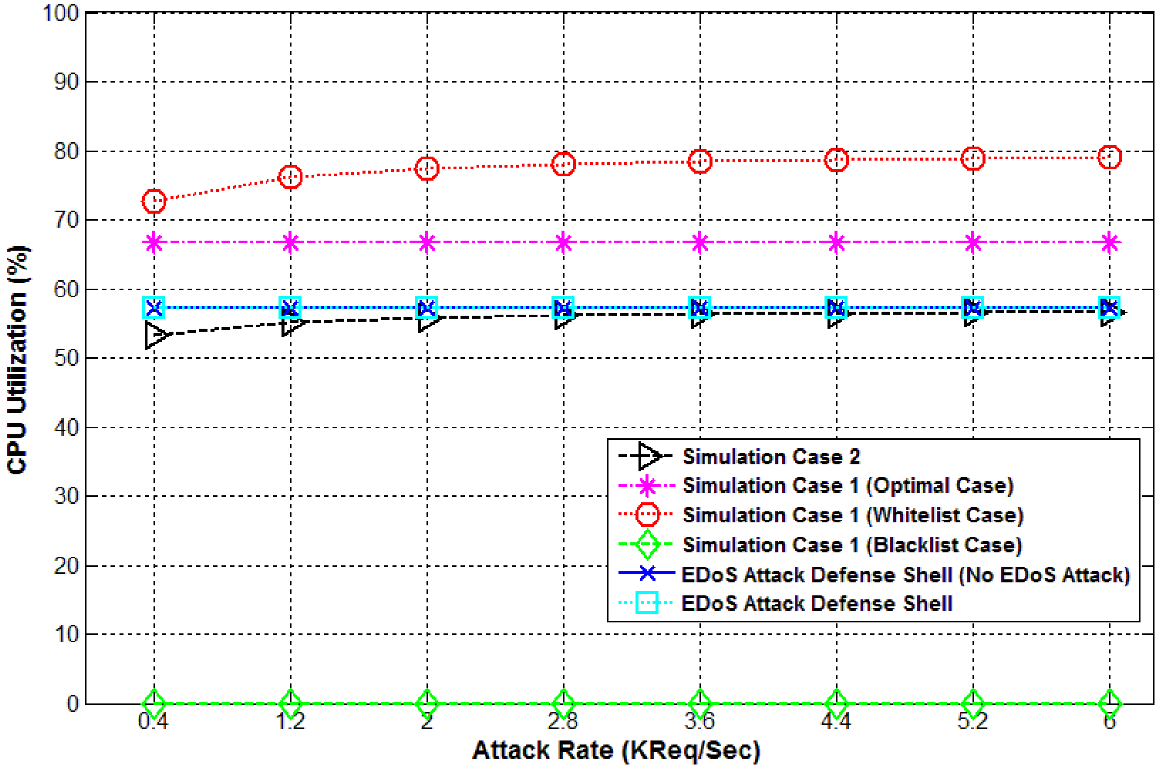


Figure 45 Resources Utilization Evaluation in the Attack Mode with the Use of Different NAT-based Networks

5.3.2 Response Time

The evaluation of the average response time in the ***attack mode*** for different attack arrival rates and for different simulation cases is shown in Figure 46. The response time results are constant for the simulation case 1 (optimal case), the case when the EDoS Attack Defense Shell (EDoS-ADS) only considers the legitimate traffic, and the case when the EDoS-ADS considers both legitimate and attack traffic. This means that these techniques have successfully eliminated the attackers' requests by dropping them; the legitimate users have not been affected by the malicious traffic. However, the response time when using the simulation case 1 (optimal case) is greater than the response time when using the EDoS-ADS due to the different number of VM instances allocated to the cloud service. As explained in subsection 5.1.2, the less number of VM instances in the simulation case 1 (optimal case) results in large queuing delay which in turn results in higher response time. In addition, the simulation case 1 (optimal case) has a much higher response time than the EDoS-ADS due to the simulation case 1 assumption of an exponential distribution for the inter arrival time of the clients requests on the cloud VM instances rather than on the cloud load balancer.

For the simulation case 1 (whitelist case) and simulation case 2, the response time increases when the attack traffic significantly increases but it does not go up considerably. This is mainly due to the auto scaling mechanism that allocates sufficient VM instances to process the additional load since both of these simulations consider the attack traffic as flash traffic. Note that based on Figure 44 the simulation case 1 (whitelist case) has been allocated more VM instances than the EDoS-ADS as a result of the attack traffic. However, the simulation case 1 (whitelist case) assumes an exponential distribution for the inter arrival time of the clients requests on the cloud VM instances rather than on the cloud load balancer. Hence, the response time when using the simulation case 1 (whitelist case) is higher than the response time when using the EDoS-ADS as evident from Figure 46. In contrast, in the simulation case 2 more VM instances are allocated than the simulation case 1 (whitelist case) that results in lowering the response time to be slightly lower than the response time of the EDoS-ADS.

Note that the response time is zero when using the simulation case 1 (blacklist case) because all of the requests are dropped due to having the legitimate clients IP addresses in the blacklist.

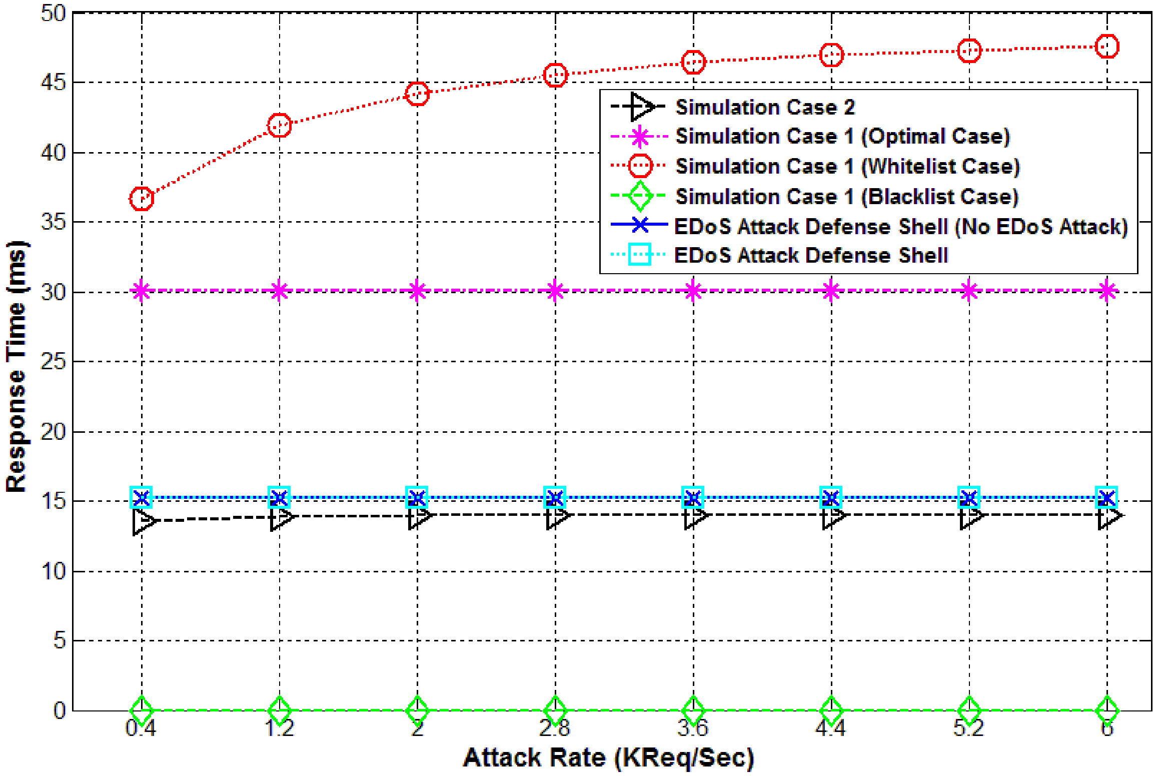


Figure 46 Response Time Evaluation in the Attack Mode with the Use of Different NAT-based Networks

5.3.3 Throughput

The throughput evaluation of the legitimate requests in the ***attack mode*** for different attack arrival rates is shown in Figure 47. The throughput results are identical for the EDoS Attack Defense Shell (EDoS-ADS) and the simulation case 2 for all considered attack arrival rates. This is due to having enough on-demand VM instances in the cloud service. In addition, the results show that the throughput of the legitimate requests was not affected by the attack arrival rate when using either the EDoS-ADS or the simulation case 2.

On the other hand, the results show slightly noticeable degradation in the throughput of the cloud computing system under simulation case 1 (optimal and whitelist cases) as reported in [64]. Finally, when using the simulation case 1 (blacklist case), there is no throughput rate to the legitimate requests because all of these requests have been dropped by the virtual firewall.

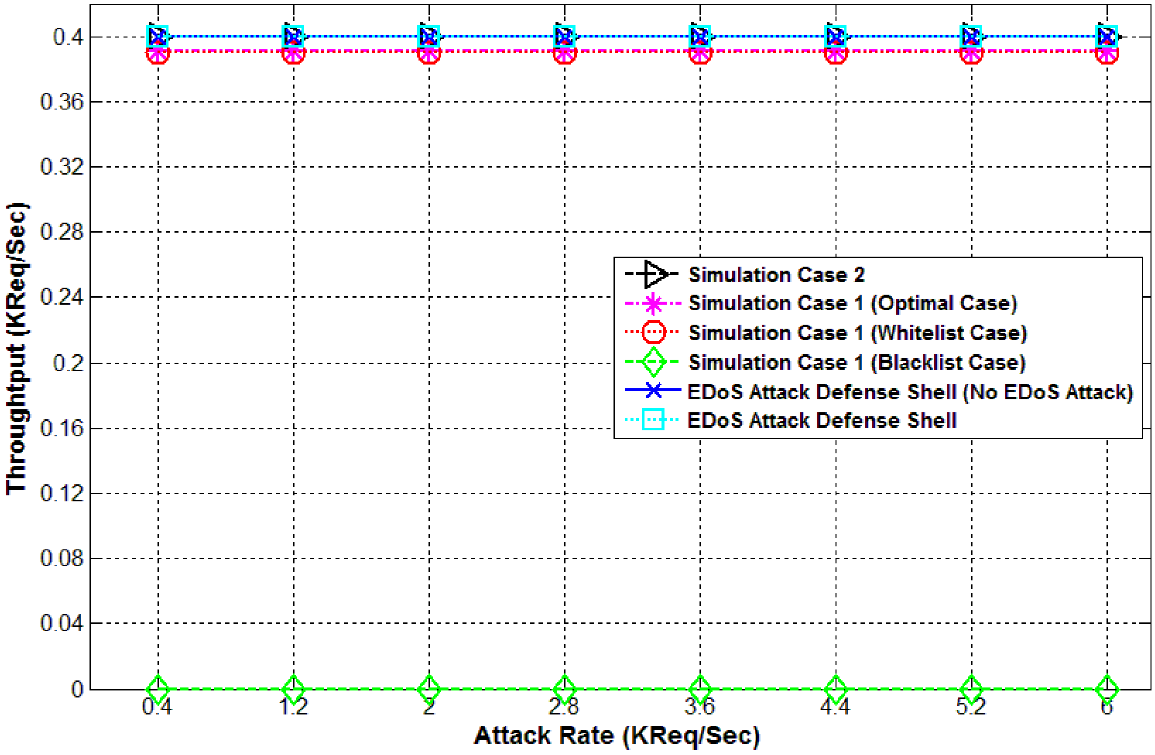


Figure 47 Throughput Evaluation of Legitimate Requests in the Attack Mode with the Use of Different NAT-based Networks

5.3.4 Cost

Figure 48 shows the cost evaluation for the EDoS Attack Defense Shell (EDoS-ADS) compared with the other techniques in the ***attack mode*** for different attack arrival rates. We considered the cost of the bandwidth allocation and the allocated VM instances assuming that the attack lasts for 10 hours [24]. The cost is calculated according to Eq. (4.8).

The costs are almost identical for the case when the EDoS-ADS only considers the legitimate traffic and the case when the EDoS-ADS considers both legitimate and attack traffic. This is because they allocate the same number of VM instances and the average CPU utilization of the cloud computing system is also the same in both cases as demonstrated in subsection 5.3.1. Moreover, the cost for the simulation case 1 (optimal case) is almost identical to the cost of the EDoS-ADS even though it allocates less VM instances when compared with the VM instances allocated in the EDoS-ADS. This is because the average CPU utilization in the simulation case 1 (optimal case) is greater than that of the EDoS-ADS.

On the other hand, the costs for the simulation case 1 (whitelist case) and the simulation case 2 are greater than the cost of the EDoS-ADS. This is because they have been allocated higher number of VM instances compared to the allocated VM instances for the EDoS-ADS since they consider the attack traffic as flash traffic. Note that the cost for the simulation case 1 (blacklist case) is almost zero because the average CPU utilization in the simulation case 1 (blacklist case) is zero too.

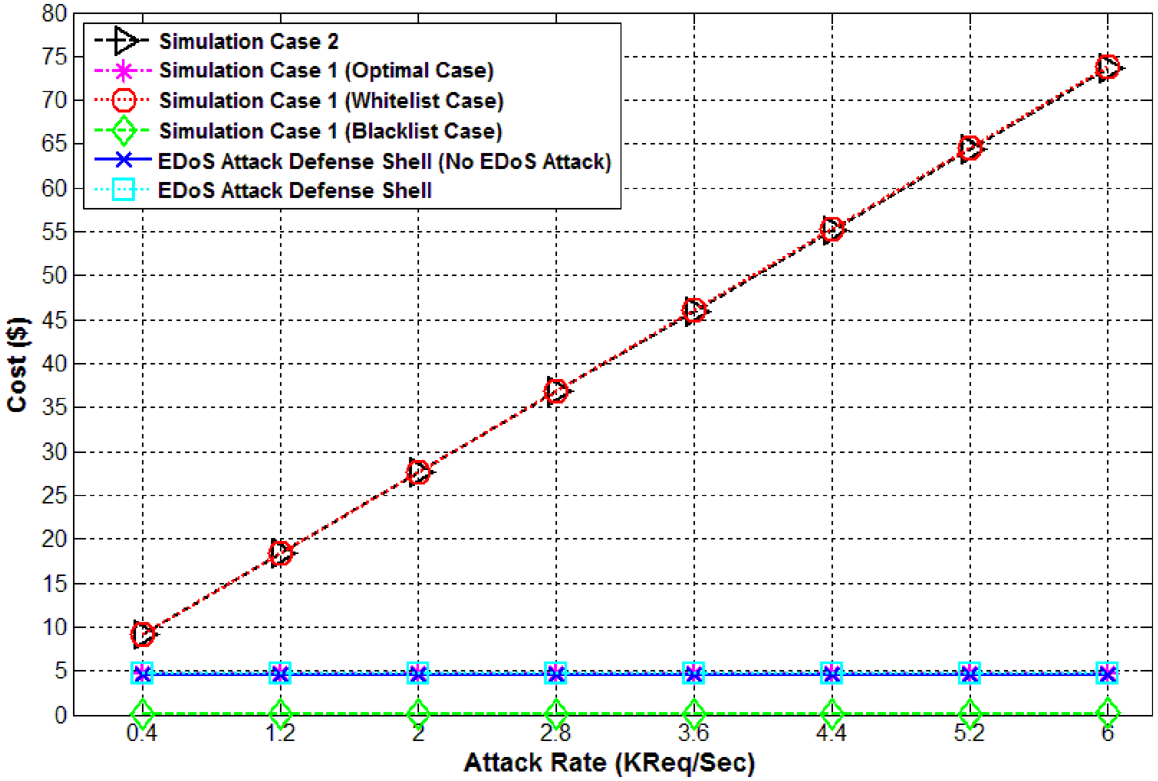


Figure 48 Cost Evaluation in the Attack Mode with the Use of Different NAT-based Networks

5.4 Attack Mode Results - Legitimate Users and Attackers belong to the Same NAT-based Network

In this scenario, we consider having legitimate clients and attackers targeting the cloud resources to evaluate the proposed mitigation technique under the EDoS attack. The system will start working in the ***suspicion mode*** when the current system utilization exceeds the scaling-up upper threshold. In this case, the cloud service load balancer will redirect all incoming requests to the EDoS Attack Defense Shell for a validation period. In this validation period, the **Suspicion Shell** will send a reCAPTCHA and URL redirection packets to the clients to differentiate between the ***flash overcrowd mode*** and the ***attack mode*** as explained in chapter 3. Upon detection of an attack behavior, the cloud system mode is changed to an ***attack mode*** and all incoming requests will be redirected to the **Attack Shell**.

This scenario assumes that the legitimate clients and attackers belong to the same NAT-based network. Thus, the IP addresses for the legitimate clients and attackers will appear to the cloud service to be the same and equal to the public IP address of the NAT router. This scenario is importantly considered because it is the typical real live scenario, and that it is more serious than the previous scenario since both the legitimate and the malicious clients share the same NAT public IP address.

The EDoS-Shield mitigation technique will be vulnerable to block an entire NAT-based network due to an EDoS attack by attackers that belong to the NAT-based network. So, we consider two cases for the EDoS-Shield mitigation technique; the EDoS-Shield whitelist case, and the EDoS-Shield blacklist case. The EDoS-Shield whitelist case refers to when the NAT public IP address is already in the whitelist. Therefore, the EDoS-Shield will consider the attackers that belong to the NAT-based network as legitimate clients and it will allow them to access the cloud service. The EDoS-Shield blacklist case is used to describe the case when the NAT public IP address was not used before to access the cloud service. Subsequently, the NAT public IP address was used by an EDoS attacker that belongs to the NAT-based network. Hence, the NAT public IP address will be added to the blacklist. Therefore, the EDoS-Shield will consider the legitimate clients that belong to the NAT-based network as attackers. Subsequently, the legitimate clients will not be able to access the cloud service because, from the cloud service point of view, their IP addresses are already in the blacklist. Note that the results for the EDoS-Shield work in this scenario are exactly the same as the results that were shown in the previous attack scenario demonstrated in subsection 5.3.

In this scenario, we generated a fixed load representing the legitimate traffic that consumes about 40% of the resources CPU utilization [106]. The arrival rate to the cloud computing environment is divided into five groups as shown in Table 4. The first group represents the legitimate clients targeting the cloud resources with requests less than what is allowed by the system (< MRPS). The second group denotes 20 legitimate clients transmit at a rate more than MRPS. The third, fourth, and fifth groups are for attackers sending requests less than what is allowed by the system, equal to that allowed by the system, and more than what is allowed by the system, respectively. These groups were chosen to study the behavior of the proposed mitigation technique when dealing with the various arrival rates of legitimate and malicious requests.

Table 4 Clients Targeting the Cloud Services in the Attack Mode with the Use of the Same NAT-based Network

Set ID	Type	Specifications	Number of Clients
1	Legitimate	Sends less requests than allowed by the system (CRPS = 1 < MRPS)	100
2	Legitimate	Sends more requests than allowed by the system (CRPS = 5 > MRPS)	20
3	Malicious	Sends less requests than allowed by the system (CRPS = 1 < MRPS)	100
4	Malicious	Sends requests equal to that allowed by the system (CRPS = 4 = MRPS)	25
5	Malicious	Sends more requests than allowed by the system (CRPS = 10 > MRPS)	10

The legitimate arrival rate is set to 200 Req/sec during the simulation with the use of 5 VM instances as an initial VM instances in the cloud service. The attack traffic is assumed to start after one minute of the simulation. We compare the EDoS Attack Defense Shell results with the results of the cloud system that uses of the auto scaling technique but without the use of an EDoS mitigation technique (simulation case 2).

5.4.1 Resources Utilization and Number of Allocated VM Instances

Figure 49 shows the comparison of the number of allocated VM instances for the simulation case 2, and the EDoS Attack Defense Shell (EDoS-ADS) during the simulation time. It is clear that the simulation case 2 has been allocated 2 additional VM instances compared to the EDoS-ADS because it is considering the attack traffic as regular traffic. As a result, the simulation case 2 will auto scale to serve the incoming traffic as discussed in subsection 5.2.1. On the other hand, the EDoS-ADS has properly identified the EDoS attack traffic as such. Accordingly, the EDoS-ADS drops that traffic, and prevents it from reaching the cloud servers. Subsequently, only the initially allocated VM instances are used to serve the legitimate incoming requests, and the EDoS-ADS does not perform auto-scaling. Thus, the proposed EDoS-ADS completely eliminates the EDoS attack.

The evaluation of the resources CPU utilization is shown in Figure 50. It is clear that the CPU utilization of the cloud servers fluctuates around 40% during the first minute of the simulation for both of the simulation case 2 and the EDoS-ADS. Note that during the first minute only the legitimate traffic is received by the cloud system, and the 40% CPU utilization represents the CPU usage for serving that traffic. Then, the resources CPU utilization immediately jumped to be around 100% for both techniques as soon as the attack began. Around minute 6, the CPU utilization for simulation case 2 returns to about 72% when the cloud system performed auto-scaling.

The cloud mode for the EDoS-ADS has been changed from the ***normal mode*** to the ***suspicion mode*** after one minute from the beginning of the simulation since the current system utilization exceeds the scaling-up upper threshold as shown in Figure 51. At this time, the cloud service load balancer redirects all incoming requests to the **Suspicion Shell**. Subsequently, the **Suspicion Shell** sends a reCAPTCHA and URL redirection packets to the clients to differentiate between the ***flash overcrowd mode*** and the ***attack mode*** as explained in chapter 3. Then, the cloud system mode is directly changed to the

attack mode due to the fast detection of the EDoS attack. Subsequently**,** all incoming requests have been redirected to the **Attack Shell**.

Accordingly, in Figure 50, during the ***attack mode*** (between minutes one and two), the second group of legitimate clients have been asked to answer the reCAPTCHA Turing test since their requests rate is greater than what is allowed by the system. These clients are delayed by 13.06 seconds on average to solve the reCAPTCHA Turing test. Accordingly, the traffic coming from the second group does not immediately reach the VM instances. Subsequently, the resources CPU utilization suddenly drops to about 20% which is associated with the CPU usage for serving the first legitimate group requests only. The two spikes between minutes one and two are due to the additional traffic contributed by the second legitimate group. The amount of additional traffic is 80 Req/sec (4 Req/sec × 20 clients without additional reCAPTCHA Turing test). The drop in these two spikes is due to the fifth request within the same second from the second legitimate group which forces reCAPTCHA Turing test to be sent to such clients.

After that (around minute two until the end of the simulation), the resources CPU utilization climbs to 40% because the arrival requests from the second group of legitimate clients becomes 5 Req/sec. This is because of the improvement of their TF due to answering the required number of reCAPTCHA Turing tests which ultimately leads to higher allowable requests per second for such clients.

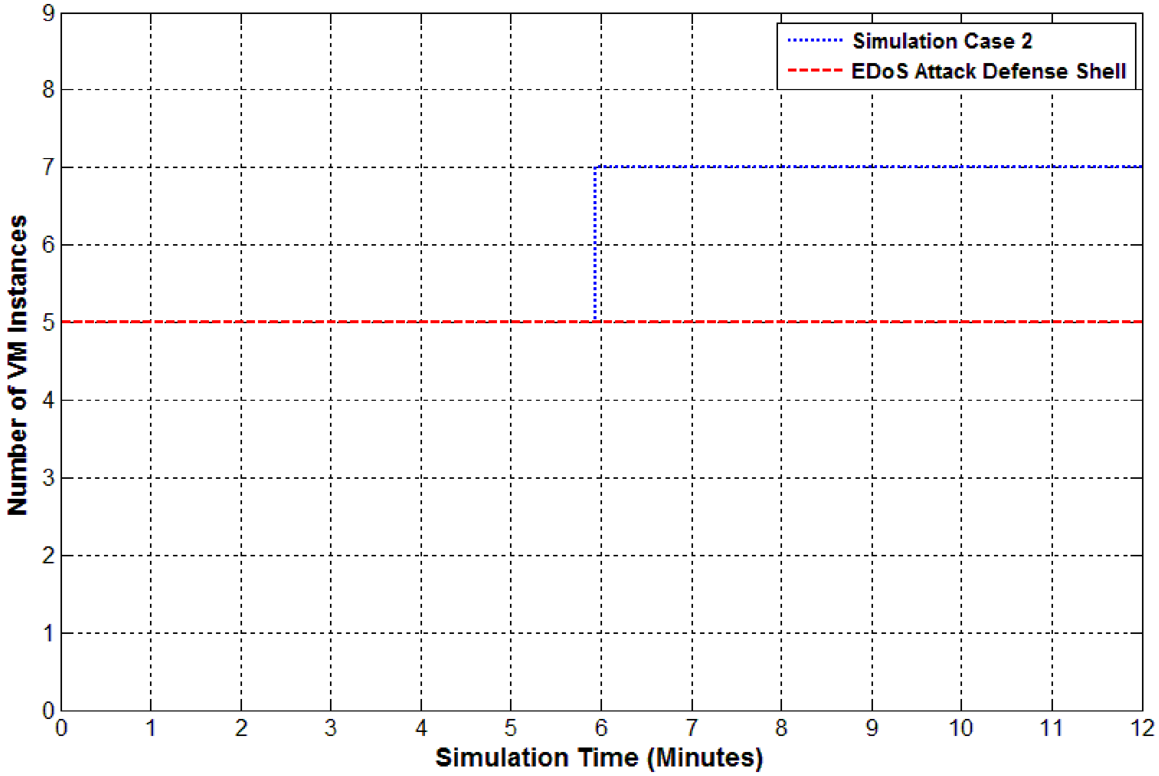


Figure 49 Evaluation of the Number of Allocated VM Instances in the Attack Mode with the Use of the Same NAT-based Network for both the Legitimate Clients and the Attackers

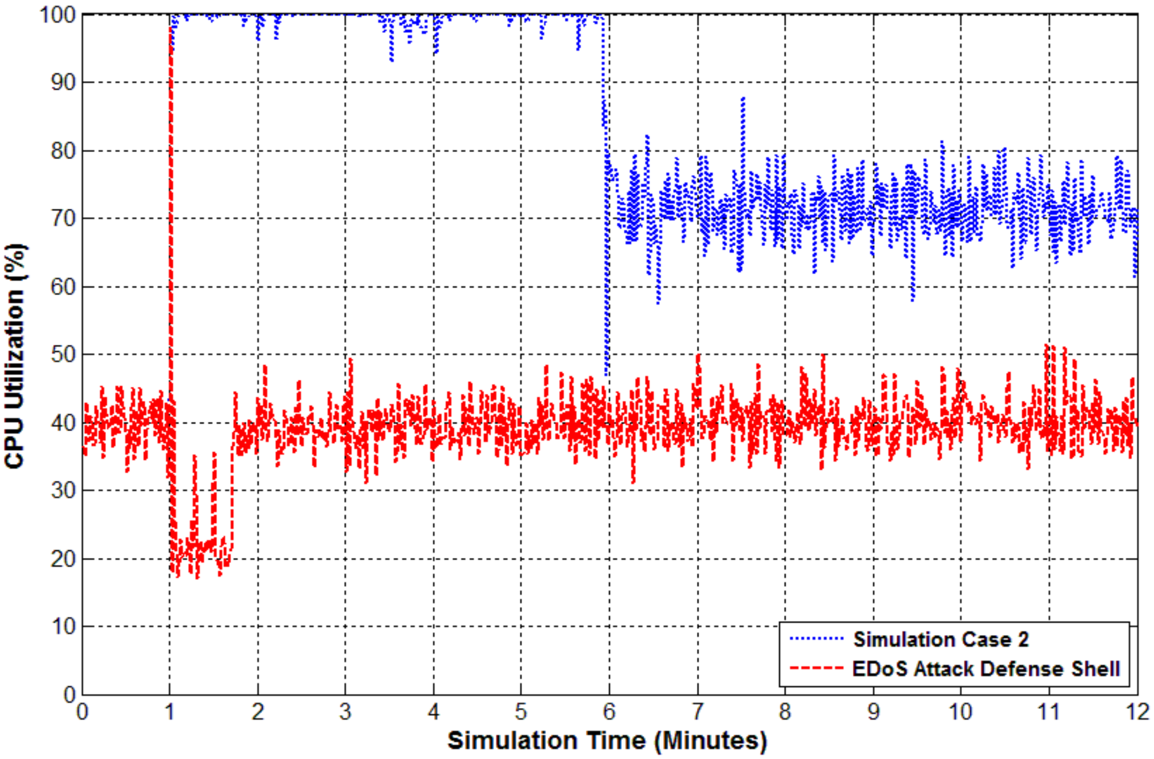


Figure 50 Resources Utilization Evaluation in the Attack Mode with the Use of the Same NAT-based Network for both the Legitimate Clients and the Attackers

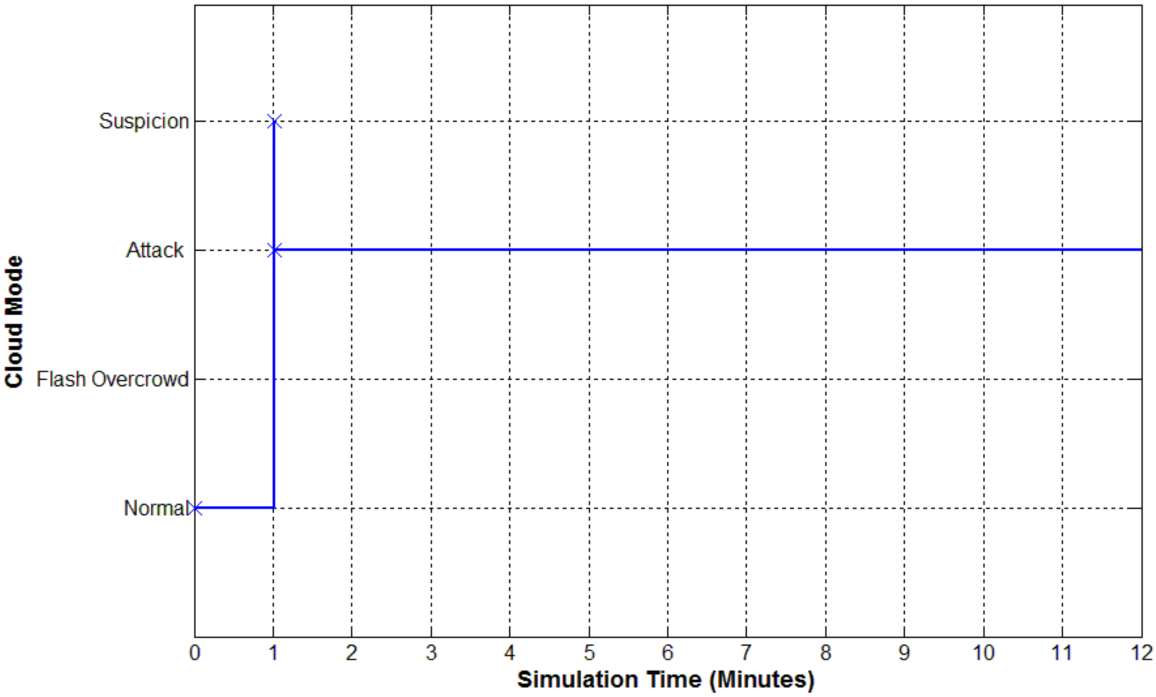


Figure 51 Cloud Mode Evaluation in the Attack Mode with the Use of the Same NAT-based Network for both the Legitimate Clients and the Attackers

5.4.2 reCAPTCHA Turing Test

The **Suspicion Shell** will send a reCAPTCHA Turing test to a client sending a new request to the cloud service, if the client's CRPS is larger than the MRPS, and the TF is either average or bad. While, the **Attack Shell** will send reCAPTCHA Turing test for those clients targeting the cloud service using the virtual IP address but with a CRPS exceeding their Allowable-RPS threshold. The Turing test will help in allowing the legitimate users to improve their TF. The improvement of users TF results in an increase in their Allowable-RPS threshold and allows them to target the cloud service with higher requests rate without a need for reCAPTCHA tests in subsequent submission of requests. The Allowable-RPS threshold is calculated based on Eq. (3.1).

Figure 52 shows the cumulative number of reCAPTCHA Turing tests sent to the legitimate clients. The requests from the first group of legitimate clients are forwarded to the cloud servers immediately without requesting those clients to answer the reCAPTCHA Turing tests. This is because they are requesting the cloud service with CRPS less than what is allowed by the system (CRPS < MRPS) during the ***suspicion mode***. In addition, their CRPS does not exceed their Allowable-RPS threshold during the ***attack mode***.

On the other hand, the results show that the second group of legitimate clients is requested to answer 3 reCAPTCHA Turing tests on average. The second group of legitimate clients has been asked to answer the first reCAPTCHA Turing test during the ***suspicion mode*** since their requests rate is greater than what is allowed by the system. The fifth request within the same second from the second legitimate group forces reCAPTCHA Turing test to be sent to such clients since the allowed MRPS is set to 4. Moreover, the second group of legitimate clients has been asked to answer the second and third reCAPTCHA Turing tests during the ***attack mode***. This is because their CRPS is larger than their Allowable-RPS threshold.

Note that the time difference between two consecutive reCAPTCHA Turing tests represents the response time overhead of answering the reCAPTCHA Turing tests. The legitimate clients of the second group are delayed by 13.06 seconds on average to solve the reCAPTCHA Turing test.

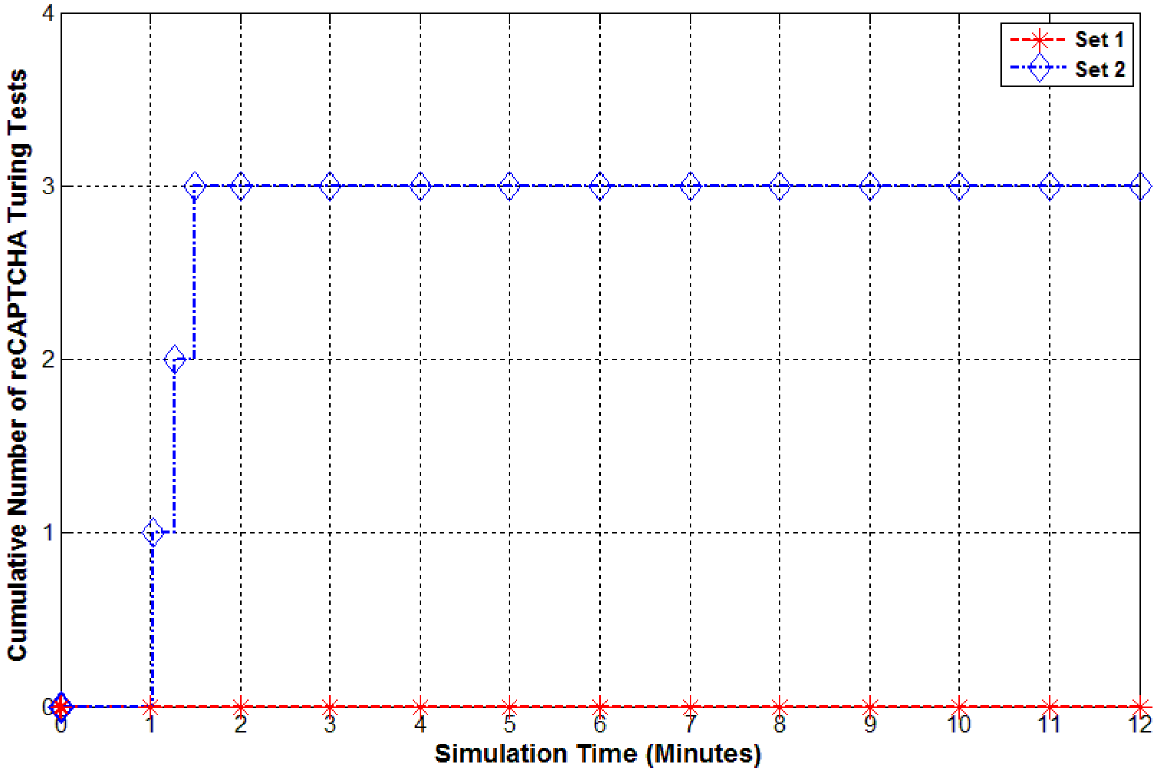


Figure 52 reCAPTCHA Turing Test Evaluation in the Attack Mode with the Use of the Same NAT-based Network for both the Legitimate Clients and the Attackers

5.4.3 Response Time

The evaluation of the response time for legitimate requests is shown in Figure 53 and Figure 54. The response time results for the simulation case 2 and the EDoS Attack Defense Shell (EDoS-ADS) are fluctuating around 12 milliseconds during the first minute of the simulation. However, the response time for the simulation case 2 increases when the attack traffic starts targeting the cloud service at which time the response time goes up considerably until it becomes 1.85 seconds. Almost at the beginning of the sixth minute of the simulation, the response time of legitimate requests for the simulation case 2 returns to be around 24 milliseconds. This is mainly due to the auto scaling mechanism that allocates additional VM instances to process the additional load since the simulation case 2 considers the attack traffic as regular traffic.

In Figure 53, during the second minute of the simulation, it is noticeable that the response time of the legitimate requests for the EDoS-ADS has three spikes due to the overhead of the reCAPTCHA Turing tests sent to the legitimate clients in the second group. After that, the response time for the EDoS-ADS returns similar to the state it was at before the start of the attack; around 12 milliseconds as clearly shown in the response time close up provided in Figure 54. Thus, the EDoS-ADS successfully eliminates affecting the legitimate traffic by the EDoS attack.

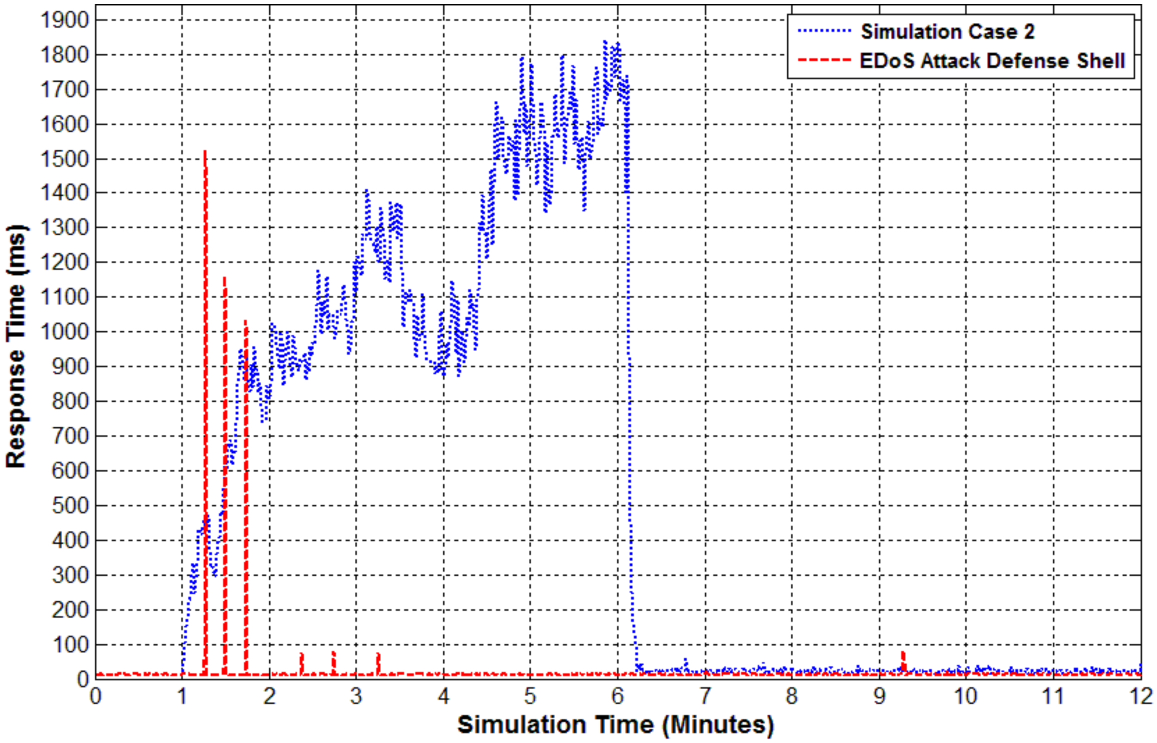


Figure 53 Response Time Evaluation in the Attack Mode with the Use of the Same NAT-based Network for both the Legitimate Clients and the Attackers

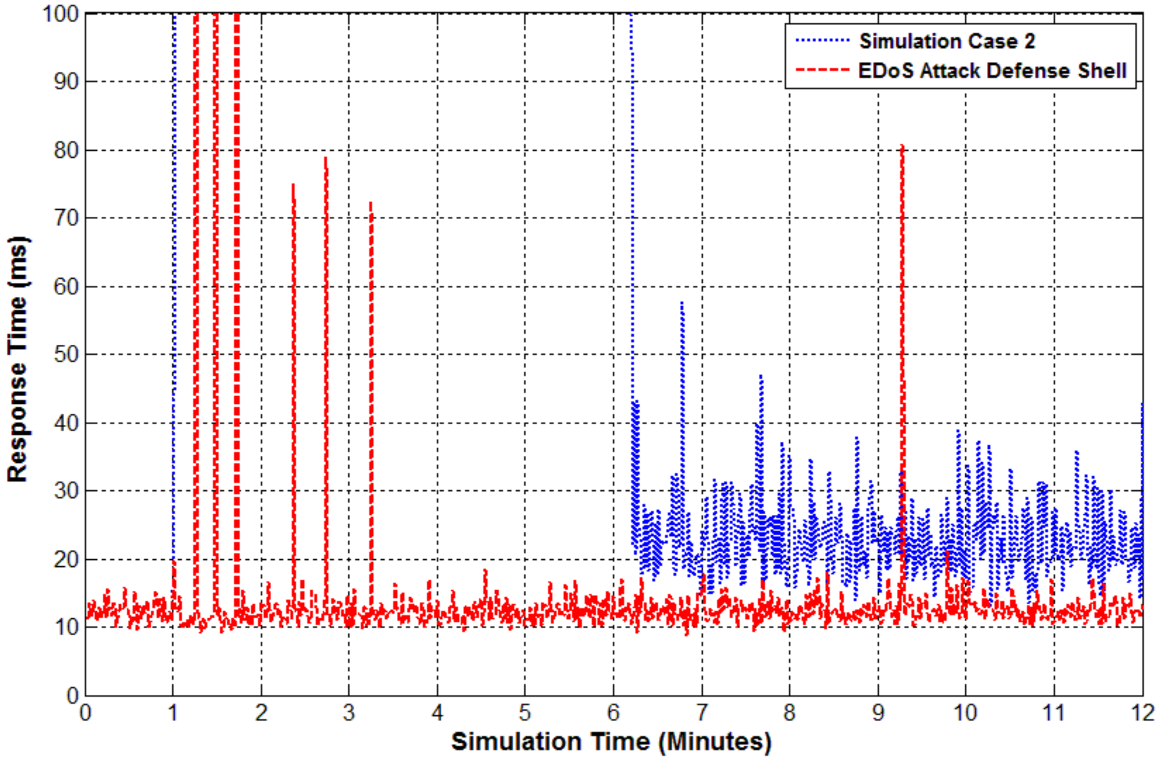


Figure 54 A Close Look at Response Time Evaluation in the Attack Mode with the Use of the Same NAT-based Network for both the Legitimate Clients and the Attackers

5.4.4 Throughput

The throughput evaluation of the legitimate requests during the simulation time is shown in Figure 55. The throughput results are identical for the EDoS Attack Defense Shell (EDoS-ADS) and the simulation case 2 during the first minute of the simulation. This is due to having enough on-demand VM instances in the cloud service. In addition, it is clear that the throughput of the legitimate requests for simulation case 2 oscillates around 200 Req/sec during the simulation even after the beginning of the EDoS attack and before it allocates additional VM instances. This is mainly due to the ability to serve the legitimate and the attack traffic using the initial number of VM instances.

For the EDoS-ADS, it can be noticed that the throughput of the legitimate requests suddenly drops to about 65 Req/sec once the load balancer starts redirecting the incoming requests to the EDoS-ADS. This is due to the small overhead of the URL redirection requests sent to the legitimate clients, and the delay associated with the response to the reCAPTCHA Turing test for the second group of legitimate clients, as mentioned in subsection 5.4.1. During the ***attack mode*** (between minutes one and two), the throughput returns to fluctuate around 100 Req/sec. This throughput is associated with the first legitimate group requests only. The two spikes between minutes one and two are due to the 100 Req/sec of the first legitimate group, and some of the requests originated from clients in the second legitimate group before violating the allowable requests rate according to their TF. After that (around minute two until the end of the simulation), the throughput climbs to 200 Req/sec because the arrival requests from each client in the second group of legitimate clients becomes 5 Req/sec. This is because of the improvement of their TF due to answering the required number of reCAPTCHA Turing test.

In summary, the overall throughput of the legitimate requests is not affected by the attack arrival rate when using either the EDoS-ADS or the simulation case 2.

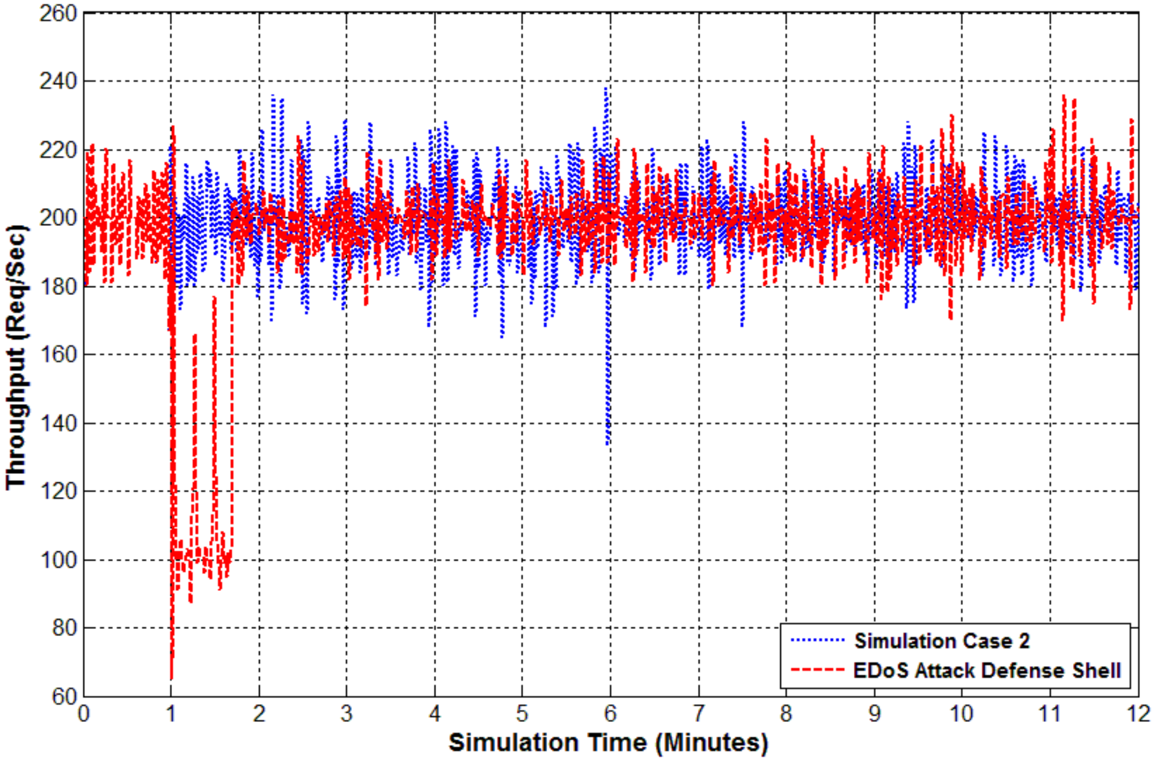


Figure 55 Throughput Evaluation of Legitimate Requests in the Attack Mode with the Use of the Same NAT-based Network for both the Legitimate Clients and the Attackers

5.4.5 Trust Factor

The Trust Factor (TF) is a value in the range of [0, 1] assigned to each client depending on the client's response to the reCAPTCHA Turing test. The **Suspicion Shell** will send a reCAPTCHA Turing test to a client sending a new request to the cloud service, if the client's CRPS is larger than the MRPS, and the TF is either average or bad. While, the **Attack Shell** will send reCAPTCHA Turing test for those clients targeting the cloud service using the virtual IP address but with a CRPS exceeding their Allowable-RPS threshold. A TF value close to 1 indicates a high trust factor and that the associated client is likely trust worthy. The TF value is initially set to 0.5 by the **Suspicion Shell** when a new request from a new client passes the EDoS Attack Defense Shell. The TF value is incremented or decremented according to the finite state machine shown in Figure 12 in subsection 3.3.

Figure 56 and Figure 57 show the corresponding average TF value for the considered clients groups. It is noted that the TF values for clients at group 1, 3, and 4 are initially set to 0.5 by the **Suspicion Shell** and they did not change during the simulation. This is due to that these clients request with a CRPS less than or equal to the MRPS threshold during the ***suspicion mode***, and their CRPS does not exceed their Allowable-RPS threshold during the ***attack mode***.

However, the average TF value for the legitimate clients in group 2 rises gradually from 0.5 to 0.51, then from 0.51 to 0.53, and then from 0.53 to 0.57, after 74, 88, and 102 seconds from the beginning of the simulation, respectively. The TF value of 0.57 allows those clients to request the cloud service with CRPS equal to 5 requests per second according to Eq. (3.1).

On the other hand, the average TF value for the third group of the malicious clients (CRPS = 10) falls substantially from 0.5 to 0.48, then from 0.48 to 0.45, then from 0.45 to 0.39, then from 0.39 to 0.27, and then from 0.27 to 0.03, at second 61. The TF value has been dropped from the average TF level to the bad TF level just within one second because those clients do not wait for the response of the cloud system, and they do not provide an answer to the reCAPTCHA Turing test sent to them. It should be noted that receiving of an HTTP web page request while waiting for an answer to the reCAPTCHA Turing test is counted as a failure in solving the Turing test. Subsequently, the TF value will be decremented according to Figure 12 in subsection 3.3.

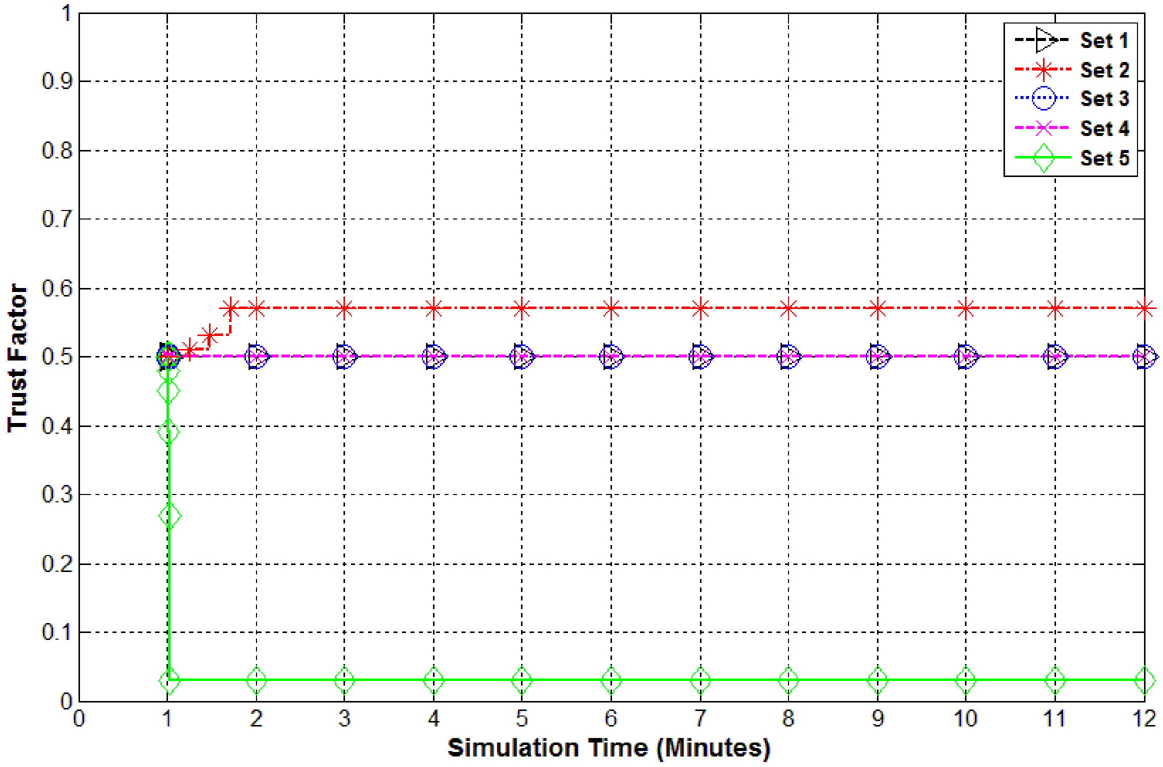


Figure 56 Trust Factor Evaluation in the Attack Mode with the Use of the Same NAT-based Network for both the Legitimate Clients and the Attackers

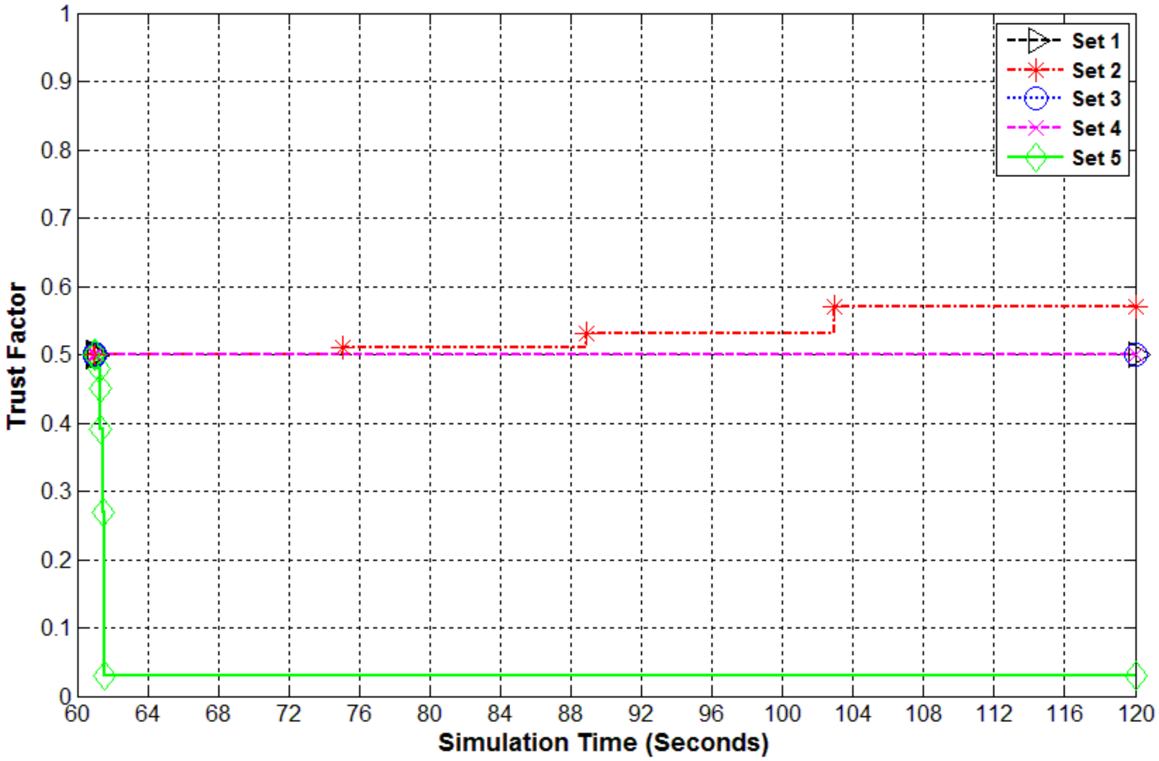


Figure 57 A Close Look at Trust Factor Evaluation in the Attack Mode with the Use of the Same NAT-based Network for both the Legitimate Clients and the Attackers

5.4.6 False Negative

Figure 58 shows the number of malicious requests attempting to access the cloud services per second, and at what point in time the EDoS Attack Defense Shell (EDoS-ADS) and the simulation case 2 succeed in blocking these. The malicious requests are generated by clients in group 3, 4, and 5 with total rate of 300 requests per second.

Once the EDoS attack begins at minute one, we can notice about 300 malicious requests attempting to access the cloud service for the EDoS-ADS and the simulation case 2. It is noted that the EDoS-ADS succeeds in blocking the malicious requests immediately. However, the simulation case 2 does not block the malicious requests and the number of malicious requests attempting to access the cloud services per second fluctuates around 300 Req/sec.

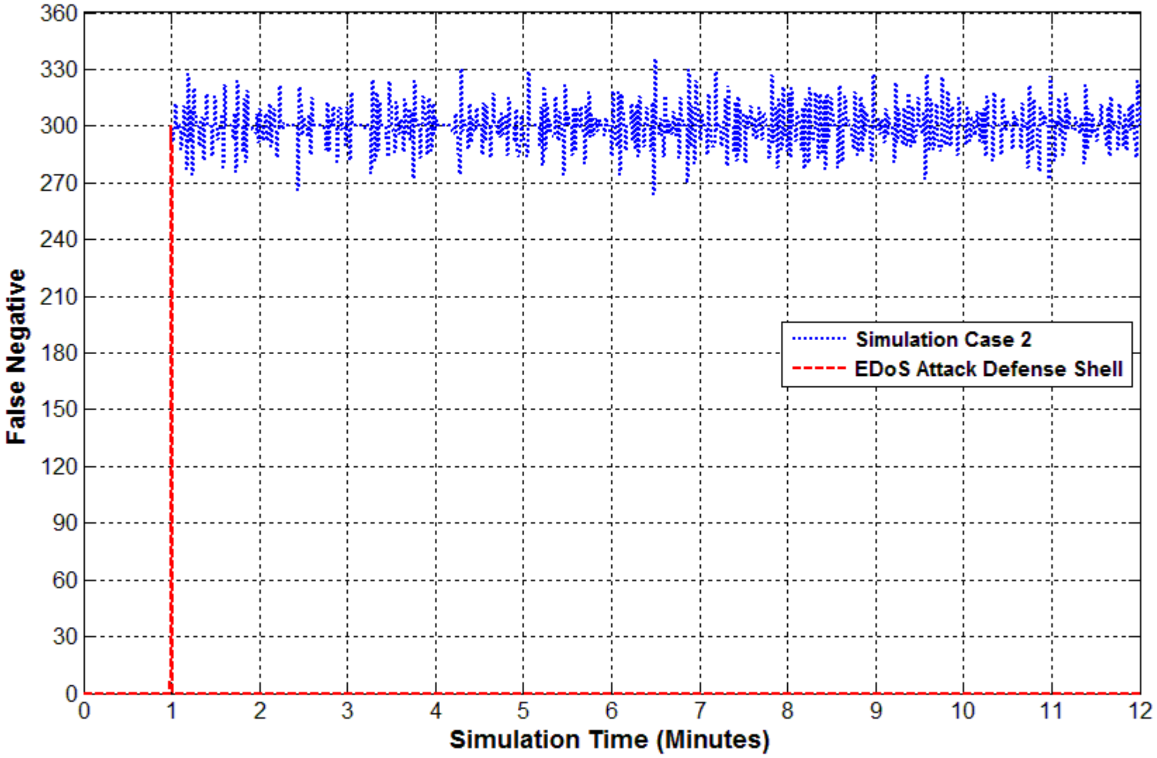


Figure 58 False Negative Evaluation in the Attack Mode with the Use of the Same NAT-based Network for both the Legitimate Clients and the Attackers

CHAPTER 6

CONCLUSION AND FUTURE WORK

This chapter summarizes the major contribution and findings in this thesis. The main objective of this research is to design and implement a mitigation technique capable of preventing or mitigating the impact of the Economic Denial of Suitability (EDoS) attack on the cloud computing environment. In addition, this chapter states the limitations of the proposed work with possible improvements as future work.

6.1 Conclusion

Cloud computing is a promising technology. However, the security of cloud computing must be investigated deeply. The EDoS attack is one of the major threats targeted towards the cloud computing environments. Thus, it needs to be considered. In this thesis, a novel solution is presented to mitigate or prevent such attacks. The mitigation technique, namely EDoS Attack Defense Shell (EDoS-ADS), is based on reactive mitigation schemes. It is only triggered when there is suspicious traffic coming to the cloud platform. Therefore, all incoming traffic is directed to the EDoS-ADS for investigating and evaluating the legitimacy of the requests.

Through the proposed technique, limited access permission for cloud services is granted to each user based on different factors such as reCAPTCHA Turing test, Uniform Resource Locator (URL) redirection technique, Trust Factor (TF), and Maximum

Requests Per Second (MRPS). Initially, the proposed technique will monitor the auto scaling feature and the auto scaling thresholds to detect if there is an EDoS attack. Once an attack behavior is detected, the cloud service will trigger a checking component for differentiating between legitimate users and automated attackers. Subsequently, the requests generated by an automated attacker will be dropped while the legitimate users' requests will be directed to the cloud servers. The proposed approach has the ability to identify the legitimacy of clients behind a Network Address Translation (NAT) router and avoid blocking an entire NAT-based network that may hosts legitimate clients from accessing the cloud servers. The effectiveness of the proposed mitigation technique is evaluated using CloudSim simulator.

In addition, we compared our proposed mitigation technique with the EDoS-Shield in three cases; the optimal case, the whitelist case, and the blacklist case. The comparison results have shown that our proposed mitigation technique is better than the EDoS-Shield in all of the considered cases in terms of performance metrics. In the whitelist case, the EDoS-Shield could still allow some attackers to launch the EDoS attack. In the blacklist case, the EDoS-Shield may unknowingly block a NAT IP address that corresponds to thousands of legitimate users due to the misbehaving of one attacker that belongs to the same NAT-based network. Overall, the proposed mitigation technique has shown promising results. The performance measures have shown the effectiveness of our proposed mitigation technique.

6.2 Future Work

The future work improvements will look into the following aspects:

1. One of the future directions for this work is to conduct an experimental implementation of the proposed mitigation technique using a test bed of a private or a public cloud and compare the obtained results with the simulation results.
2. To enhance the simulation, we could add different distributions of the attacks and consider different types of attacks other than the DDoS attacks that could lead to the EDoS attack.
3. Using a smarter method to detect if the attack has finished rather than using the Attack Timer that depends on the Attack Period Time (APT).
4. The cloud customers could be charged to some degree especially when the DDoS attack does not trigger the proposed mitigation techniques due to not exceeding the pre-defined thresholds, Low Rate EDoS (LR-EDoS) attack. So, the proposed mitigation technique needs to be improved in order to detect the LR-EDoS attack.
5. We have assumed that the attacker is incapable to solve the reCAPTCHA Turing tests and perform the URL redirection. One of the future directions for this work is to run some scenarios where the attackers are capable to solve some the Turing tests and/or able to perform the URL redirection to confirm if the EDoS Attack Defense Shell will take care of such a scenario or whether it needs further improvement.

References

[1] P. Mell and T. Grance, "The NIST definition of cloud computing," *National Institute of Standards and Technology,* vol. 53, p. 50, 2009.

[2] L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner, "A break in the clouds: towards a cloud definition," *ACM SIGCOMM Computer Communication Review,* vol. 39, pp. 50-55, 2008.

[3] C. Pettey, "Gartner identifies the top 10 strategic technologies for 2011," *Retrieved July,* vol. 1, p. 2011, 2010.

[4] J. Rivera. *Gartner Identifies the Top 10 Strategic Technology Trends for 2015*. Available: http://www.gartner.com/newsroom/id/2867917

[5] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: state-of-the-art and research challenges," *Journal of internet services and applications,* vol. 1, pp. 7-18, 2010.

[6] *Amazon EC2*. Available: http://aws.amazon.com/ec2/

[7] *Google App Engine: Platform as a Service*. Available: https://cloud.google.com/appengine/docs

[8] *Salesforce*. Available: http://www.salesforce.com/

[9] M. Ahmed, A. Chowdhury, M. Ahmed, and M. M. H. Rafee, "An advanced survey on cloud computing and state-of-the-art research issues," *IJCSI International Journal of Computer Science Issues,* vol. 9, pp. 1694-0814, 2012.

[10] M. H. Sqalli, M. Al-Saeedi, F. Binbeshr, and M. Siddiqui, "UCloud: A simulated Hybrid Cloud for a university environment," in *Cloud Networking (CLOUDNET), 2012 IEEE 1st International Conference on,* 2012, pp. 170-172.

[11] F. Gens, "New idc it cloud services survey: top benefits and challenges," *IDC exchange,* 2009.

[12] S. Mansfield-Devine, "Danger in the clouds," *Network Security,* vol. 2008, pp. 9-11, 2008.

[13] L. M. Kaufman, "Data security in the world of cloud computing," *Security & Privacy, IEEE,* vol. 7, pp. 61-64, 2009.

[14] T. Mather, S. Kumaraswamy, and S. Latif, *Cloud security and privacy: an enterprise perspective on risks and compliance*: " O'Reilly Media, Inc.", 2009.

[15] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generation computer systems,* vol. 25, pp. 599-616, 2009.

[16] S. Sudha and V. M. Viswanatham, "Addressing security and privacy issues in cloud computing," *Journal of Theoretical and Applied Information Technology,* vol. 48, pp. 708-719, 2013.

[17] M. Jensen, N. Gruschka, and N. Luttenberger, "The impact of flooding attacks on network-based services," in *Availability, Reliability and Security, 2008. ARES 08. Third International Conference on,* 2008, pp. 509-513.

[18] C. Hoff, "Cloud computing security: From DDoS (distributed denial of service) to EDoS (economic denial of sustainability)," *Blog, Retrieved November,* vol. 27, 2008.

[19] S. Yu, R. Doss, W. Zhou, and S. Guo, "A general cloud firewall framework with dynamic resource allocation," in *Communications (ICC), 2013 IEEE International Conference on*, 2013, pp. 1941-1945.

[20] M. Jensen and N. Gruschka, "Flooding Attack Issues of Web Services and Service-Oriented Architectures," in *GI Jahrestagung (1)*, 2008, pp. 117-122.

[21] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski*, et al.*, "A view of cloud computing," *Communications of the ACM,* vol. 53, pp. 50-58, 2010.

[22] *Amazon CloudWatch*. Available: http://aws.amazon.com/cloudwatch/

[23] S. H. Khor and A. Nakao, "spow: On-demand cloud-based eddos mitigation mechanism," in *HotDep (Fifth Workshop on Hot Topics in System Dependability)*, 2009.

[24] M. H. Sqalli, F. Al-Haidari, and K. Salah, "Edos-shield-a two-steps mitigation technique against edos attacks in cloud computing," in *Utility and Cloud Computing (UCC), 2011 Fourth IEEE International Conference on*, 2011, pp. 49-56.

[25] F. Al-Haidari, M. H. Sqalli, and K. Salah, "Enhanced edos-shield for mitigating edos attacks originating from spoofed ip addresses," in *Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on*, 2012, pp. 1167-1174.

[26] S. Chapade, K. Pandey, and D. Bhade, "Securing cloud servers against flooding based ddos attacks," in *Communication Systems and Network Technologies (CSNT), 2013 International Conference on*, 2013, pp. 524-528.

[27] H. Mohan, "An effective defense against distributed denial of service in grid," in *Integrated Intelligent Computing (ICIIC), 2010 First International Conference on*, 2010, pp. 84-89.

[28] A. Yaar, A. Perrig, and D. Song, "Pi: A path identification mechanism to defend against DDoS attacks," in *Security and Privacy, 2003. Proceedings. 2003 Symposium on*, 2003, pp. 93-107.

[29] M. N. Kumar, R. Korra, P. Sujatha, and M. Kumar, "Mitigation of economic distributed denial of sustainability (eddos) in cloud computing," in *International Conference on Advances in Engineering and Technology,(ICAET-2011)*, 2011.

[30] X. Wang and M. K. Reiter, "Mitigating bandwidth-exhaustion attacks using congestion puzzles," in *Proceedings of the 11th ACM conference on Computer and communications security*, 2004, pp. 257-267.

[31] V. D. Gligor, "Guaranteeing access in spite of distributed service-flooding attacks," in *Security Protocols*, 2005, pp. 80-96.

[32] M. Naresh Kumar, P. Sujatha, V. Kalva, R. Nagori, A. K. Katukojwala, and M. Kumar, "Mitigating economic denial of sustainability (edos) in cloud computing using in-cloud scrubber service," in *Computational Intelligence and Communication Networks (CICN), 2012 Fourth International Conference on*, 2012, pp. 535-539.

[33] S. VivinSandar and S. Shenai, "Economic denial of sustainability (edos) in cloud services using http and xml based ddos attacks," *International Journal of Computer Applications,* vol. 41, pp. 11-16, 2012.

[34] W. Alosaimi and K. Al-Begain, "A New Method to Mitigate the Impacts of the Economical Denial of Sustainability Attacks Against the Cloud," in *Proceedings*

of the 14th Annual Post Graduates Symposium on the convergence of Telecommunication, Networking and Broadcasting (PGNet), 2013, pp. 116-121.

[35] W. Alosaimi and K. Al-Begain, "An Enhanced Economical Denial of Sustainability Mitigation System for the Cloud," in *Next Generation Mobile Apps, Services and Technologies (NGMAST), 2013 Seventh International Conference on*, 2013, pp. 19-25.

[36] M. Masood, Z. Anwar, S. A. Raza, and M. A. Hur, "EDoS Armor: A cost effective economic denial of sustainability attack mitigation framework for e-commerce applications in cloud environments," in *Multi Topic Conference (INMIC), 2013 16th International*, 2013, pp. 37-42.

[37] Z. Baig and F. Binbeshr, "Controlled Virtual Resource Access to Mitigate Economic Denial of Sustainability (EDoS) Attacks against Cloud Infrastructures," in *Cloud Computing and Big Data (CloudCom-Asia), 2013 International Conference on*, 2013, pp. 346-353.

[38] K. Anusha, "Detection of Economic Denial of Sustainability using Time Spent on a Web Page in Cloud," in *IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)*, 2013.

[39] P. I. Hofgesang, "Methodology for preprocessing and evaluating the time spent on web pages," in *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, 2006, pp. 218-225.

[40] C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel, and M. Rajarajan, "A survey of intrusion detection techniques in cloud," *Journal of Network and Computer Applications*, vol. 36, pp. 42-57, 2013.

[41] B. Saini and G. Somani, "Index Page Based EDoS Attacks in Infrastructure Cloud," in *Recent Trends in Computer Networks and Distributed Systems Security*, ed: Springer, 2014, pp. 382-395.

[42] M. A. Saleh and A. Abdul Manaf, "A Novel Protective Framework for Defeating HTTP-Based Denial of Service and Distributed Denial of Service Attacks," *The Scientific World Journal*, vol. 2015, 2015.

[43] Z. A. Baig, S. M. Sait, and F. Binbeshr, "Controlled Access to Cloud Resources for Mitigating Economic Denial of Sustainability (EDoS) Attacks," *Computer Networks*, 2016.

[44] K. Egevang and P. Francis, "The IP network address translator (NAT)," 2070-1721, 1994.

[45] P. Srisuresh and K. Egevang, "Traditional IP network address translator (Traditional NAT)," 2070-1721, 2000.

[46] A. Muller, G. Carle, and A. Klenk, "Behavior and classification of NAT devices and implications for NAT traversal," *Network, IEEE*, vol. 22, pp. 14-19, 2008.

[47] Y. Gokcen and V. A. Foroushani, "Can we identify NAT behavior by analyzing Traffic Flows?," in *Security and Privacy Workshops (SPW), 2014 IEEE*, 2014, pp. 132-139.

[48] Y. Ishikawa, N. Yamai, K. Okayama, and M. Nakamura, "An Identification Method of PCs behind NAT Router with Proxy Authentication on HTTP Communication," in *Applications and the Internet (SAINT), 2011 IEEE/IPSJ 11th International Symposium on*, 2011, pp. 445-450.

[49] G. Maier, F. Schneider, and A. Feldmann, "NAT usage in residential broadband networks," in *Passive and Active Measurement*, 2011, pp. 32-41.

[50] H. S. Cho and Y. K. Noh, "System for preventing normal user being blocked in network address translation (NAT) based web service and method for controlling the same," ed: Google Patents, 2013.

[51] P. Srisuresh and M. Holdrege, "IP network address translator (NAT) terminology and considerations," 1999.

[52] W. G. Morein, A. Stavrou, D. L. Cook, A. D. Keromytis, V. Misra, and D. Rubenstein, "Using graphic turing tests to counter automated ddos attacks against web servers," in *Proceedings of the 10th ACM conference on Computer and communications security*, 2003, pp. 8-19.

[53] K. Bhargrava, D. Brewer, and K. Li, "A study of URL redirection indicating spam," *CEAS (July 2009)*, 2009.

[54] L. von Ahn, M. Blum, N. Hopper, and J. Langford, "The official CAPTCHA site," ed: Carnegie Mellon University, 2000.

[55] L. Von Ahn, M. Blum, N. Hopper, and J. Langford, "Captcha: Telling humans and computers apart automatically," in *Proceedings of Eurocrypt*, 2003.

[56] L. Von Ahn, M. Blum, N. J. Hopper, and J. Langford, "CAPTCHA: Using hard AI problems for security," in *Advances in Cryptology—EUROCRYPT 2003*, ed: Springer, 2003, pp. 294-311.

[57] J. Yan and A. S. El Ahmad, "Captcha security: A case study," *IEEE Security & Privacy*, pp. 22-28, 2009.

[58] M. Mehra, M. Agarwal, R. Pawar, and D. Shah, "Mitigating denial of service attack using CAPTCHA mechanism," in *Proceedings of the International Conference & Workshop on Emerging Trends in Technology*, 2011, pp. 284-287.

[59] E. Bursztein, S. Bethard, C. Fabry, J. C. Mitchell, and D. Jurafsky, "How good are humans at solving CAPTCHAs? a large scale evaluation," in *Security and Privacy (SP), 2010 IEEE Symposium on*, 2010, pp. 399-413.

[60] L. Von Ahn, B. Maurer, C. McMillen, D. Abraham, and M. Blum, "recaptcha: Human-based character recognition via web security measures," *Science*, vol. 321, pp. 1465-1468, 2008.

[61] R. Fielding and J. Reschke, "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content," 2014.

[62] *Redirects & SEO - The Complete Guide*. Available: https://audisto.com/insights/guides/31/

[63] D. Bellenger, J. Bertram, A. Budina, A. Koschel, B. Pfänder, C. Serowy, *et al.*, "Scaling in cloud environments," *Recent Researches in Computer Science*, 2011.

[64] F. Al-Haidari, M. Sqalli, and K. Salah, "Evaluation of the impact of EDoS attacks against cloud computing services," *Arabian Journal for Science and Engineering*, vol. 40, pp. 773-785, 2014.

[65] Y. Wang, K.-J. Lin, D. S. Wong, and V. Varadharajan, "The design of a rule-based and event-driven trust management framework," in *e-Business Engineering, 2007. ICEBE 2007. IEEE International Conference on*, 2007, pp. 97-104.

[66] Z. Zhou, Y. Luo, L. Guo, and L. Sun, "Assessment of P2P Trust Model Based on Fuzzy Comprehensive Evaluation," *Journal of Software*, vol. 8, pp. 2711-2714, 2013.

[67] S. Islam, K. Lee, A. Fekete, and A. Liu, "How a consumer can measure elasticity for cloud platforms," in *Proceedings of the 3rd ACM/SPEC International Conference on Performance Engineering*, 2012, pp. 85-96.

[68] H. Wu, A. N. Tantawi, and T. Yu, "A self-optimizing workload management solution for cloud applications," in *Web Services (ICWS), 2013 IEEE 20th International Conference on*, 2013, pp. 483-490.

[69] W. Cherry. *How to Find Equations for Exponential Functions*. Available: http://wcherry.math.unt.edu/math1650/exponential.pdf

[70] D. Moore, C. Shannon, D. J. Brown, G. M. Voelker, and S. Savage, "Inferring internet denial-of-service activity," *ACM Transactions on Computer Systems (TOCS)*, vol. 24, pp. 115-139, 2006.

[71] X. Bao and H. Hong, "NSFOCUS DDoS Threat Report 2013," *NSFOCUS Information Technology Co., Ltd*, 2013.

[72] R. Buyya, R. Ranjan, and R. N. Calheiros, "Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities," in *High Performance Computing & Simulation, 2009. HPCS'09. International Conference on*, 2009, pp. 1-11.

[73] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. De Rose, and R. Buyya, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software: Practice and Experience*, vol. 41, pp. 23-50, 2011.

[74] M. F. Arlitt and C. L. Williamson, "Internet web servers: Workload characterization and performance implications," *IEEE/ACM Transactions on Networking (ToN)*, vol. 5, pp. 631-645, 1997.

[75] Z. Liu, N. Niclausse, and C. Jalpa-Villanueva, "Traffic model and performance evaluation of web servers," *Performance Evaluation*, vol. 46, pp. 77-100, 2001.

[76] J. Walraevens, S. Wittevrongel, and H. Bruneel, "Performance analysis of a priority queue with session-based arrivals and its application to E-commerce web servers," *International Journal On Advances in Internet Technology*, vol. 2, pp. 46-57, 2009.

[77] H.-p. Chen and S.-C. Li, "A queueing-based model for performance management on cloud," in *Advanced Information Management and Service (IMS), 2010 6th International Conference on*, 2010, pp. 83-88.

[78] X. Nan, Y. He, and L. Guan, "Optimal resource allocation for multimedia cloud based on queuing model," in *Multimedia Signal Processing (MMSP), 2011 IEEE 13th International Workshop on*, 2011, pp. 1-6.

[79] R. N. Calheiros, R. Ranjan, and R. Buyya, "Virtual machine provisioning based on analytical performance and QoS in cloud computing environments," in *Parallel Processing (ICPP), 2011 International Conference on*, 2011, pp. 295-304.

[80] Y. Shi, X. Jiang, and K. Ye, "An energy-efficient scheme for cloud resource provisioning based on cloudsim," in *Cluster Computing (CLUSTER), 2011 IEEE International Conference on*, 2011, pp. 595-599.

[81] R. Pal and P. Hui, "Economic models for cloud service markets," in *Distributed Computing and Networking*, ed: Springer, 2012, pp. 382-396.

[82] D. Boteanu, J. M. Fernandez, J. McHugh, and J. Mullins, "Queue Management as a DoS counter-measure?," in *Information Security*, ed: Springer, 2007, pp. 263-280.

[83] Y. Wang, C. Lin, Q.-L. Li, and Y. Fang, "A queueing analysis for the denial of service (DoS) attacks in computer networks," *Computer Networks*, vol. 51, pp. 3564-3573, 2007.

[84] N. Singh, S. Ghrera, and P. Chaudhuri, "Denial of service attack: analysis of network traffic anormaly using queuing theory," *arXiv preprint arXiv:1006.2807*, 2010.

[85] W. D. Kelton and A. M. Law, *Simulation modeling and analysis*: McGraw Hill Boston, 2000.

[86] C. Sutton and M. I. Jordan, "Bayesian inference for queueing networks and modeling of internet services," *The Annals of Applied Statistics*, pp. 254-282, 2011.

[87] W. Dawoud, I. Takouna, and C. Meinel, "Elastic VM for rapid and optimum virtualized resources' allocation," in *Systems and Virtualization Management (SVM), 2011 5th International DMTF Academic Alliance Workshop on*, 2011, pp. 1-4.

[88] K. Xiong and H. Perros, "Service performance and analysis in cloud computing," in *Services-I, 2009 World Conference on*, 2009, pp. 693-700.

[89] J. Bi, Z. Zhu, R. Tian, and Q. Wang, "Dynamic provisioning modeling for virtualized multi-tier applications in cloud data center," in *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on*, 2010, pp. 370-377.

[90] M. Kihl, G. Cedersjö, A. Robertsson, and B. Aspernäs, "Performance measurements and modeling of database servers," in *Sixth International Workshop on Feedback Control Implementation and Design in Computing Systems and Networks (FeBID 2011)*, 2011.

[91] X. Shen, H. Chen, J. Dai, and W. Dai, "The finite element method for computing the stationary distribution of an SRBM in a hypercube with applications to finite buffer queueing networks," *Queueing Systems*, vol. 42, pp. 33-62, 2002.

[92] D. Gross, *Fundamentals of queueing theory*: John Wiley & Sons, 2008.

[93] J. Idziorek, "Discrete event simulation model for analysis of horizontal scaling in the cloud computing model," in *Simulation Conference (WSC), Proceedings of the 2010 Winter*, 2010, pp. 3004-3014.

[94] A. A. Scaling, D. Guide, and A. V. Aug, "1, 2010," *Amazon Web Services*, vol. 200, p. 20, 2010.

[95] J. D. Little, "A proof for the queuing formula: L= λ W," *Operations research*, vol. 9, pp. 383-387, 1961.

[96] A. Li, X. Yang, S. Kandula, and M. Zhang, "CloudCmp: comparing public cloud providers," in *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, 2010, pp. 1-14.

[97] *2015 Talkin' Cloud 100: Top Cloud Services Providers*. Available: http://talkincloud.com/tc100

[98] *The 20 Totally Most Popular Cloud Services in Today's Enterprise*.

[99] *Amazon EC2 Instance Purchasing Options*. Available: https://aws.amazon.com/ec2/purchasing-options/

[100] D. Catteddu, "Cloud Computing: benefits, risks and recommendations for information security," in *Web Application Security*, ed: Springer, 2010, pp. 17-17.

[101] G. Juve and E. Deelman, "Automating application deployment in infrastructure clouds," in *Cloud Computing Technology and Science (CloudCom), 2011 IEEE Third International Conference on*, 2011, pp. 658-665.

[102] K. Claffy, G. Miller, and K. Thompson, "The nature of the beast: recent traffic measurements from an Internet backbone," in *Proceedings of INET*, 1998, pp. 21-24.

[103] H. Liu and S. Wee, "Web server farm in the cloud: Performance evaluation and dynamic architecture," in *Cloud Computing*, ed: Springer, 2009, pp. 369-380.

[104] *Amazon EC2 Pricing*. Available: https://aws.amazon.com/ec2/pricing/

[105] *Beware HTTP Redirects for SEO and Performance*. Available: http://www.websiteoptimization.com/speed/tweak/redirect/

[106] C. Reiss, A. Tumanov, G. R. Ganger, R. H. Katz, and M. A. Kozuch, "Towards understanding heterogeneous clouds at scale: Google trace analysis," *Intel Science and Technology Center for Cloud Computing, Tech. Rep*, p. 84, 2012.

APPENDIX A: RESOURCES UTILIZATION AND NUMBER OF ALLOCATED VM INSTANCES IN THE FLASH OVERCROWD MODE

The resources utilization and the number of allocated VM instances for the simulation case 2 at rates of 0.4, 0.8, 1.6, 2.4, and 3.2 KReq/sec are shown in Figure 59, Figure 61, Figure 63, Figure 65, and Figure 67, respectively. In addition, the resources utilization and the number of allocated VM instances for the EDoS Attack Defense Shell (EDoS-ADS) at rates of 0.4, 0.8, 1.6, 2.4, and 3.2 KReq/sec are shown in Figure 60, Figure 62, Figure 64, Figure 66, and Figure 68, respectively.

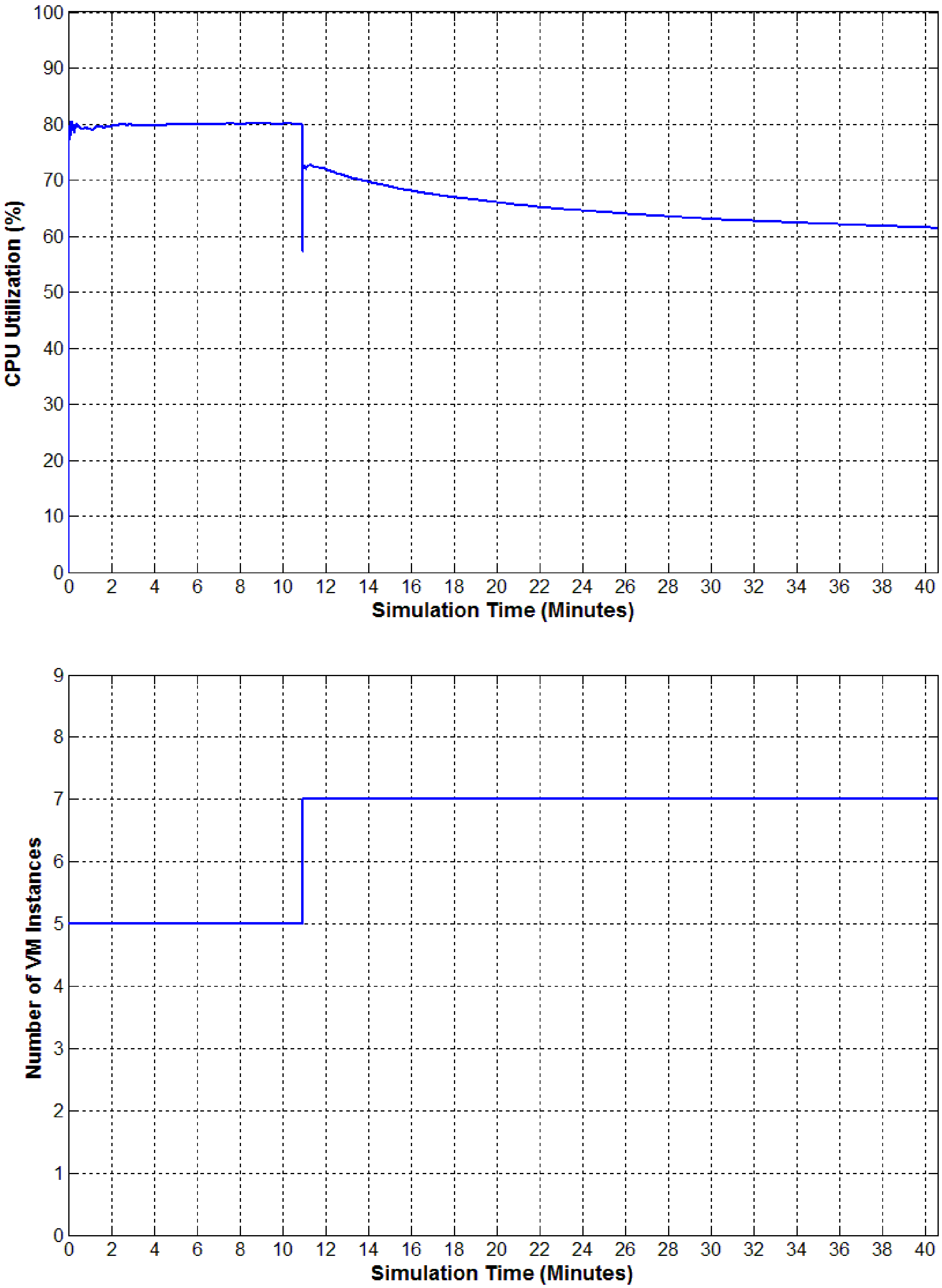


Figure 59 Resources Utilization Evaluation and the Number of Allocated VM Instances for the Simulation with Auto Scaling but Without Mitigation Technique (Simulation Case 2) at Rate of 0.4 KReq/sec in the Flash Overcrowd Mode

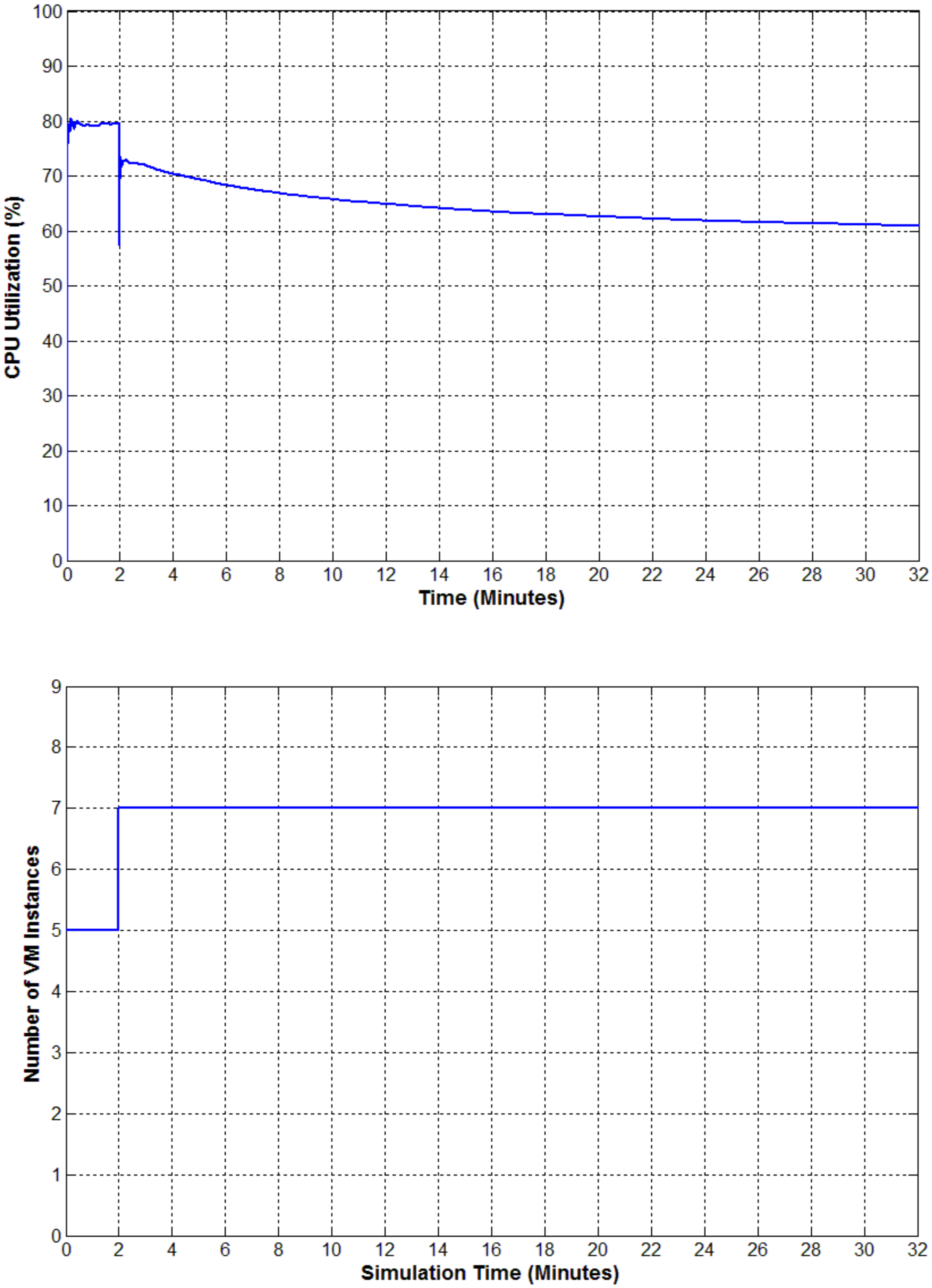


Figure 60 Resources Utilization Evaluation and the Number of Allocated VM Instances for EDoS Attack Defense Shell Simulation at Rate of 0.4 KReq/sec in the Flash Overcrowd Mode

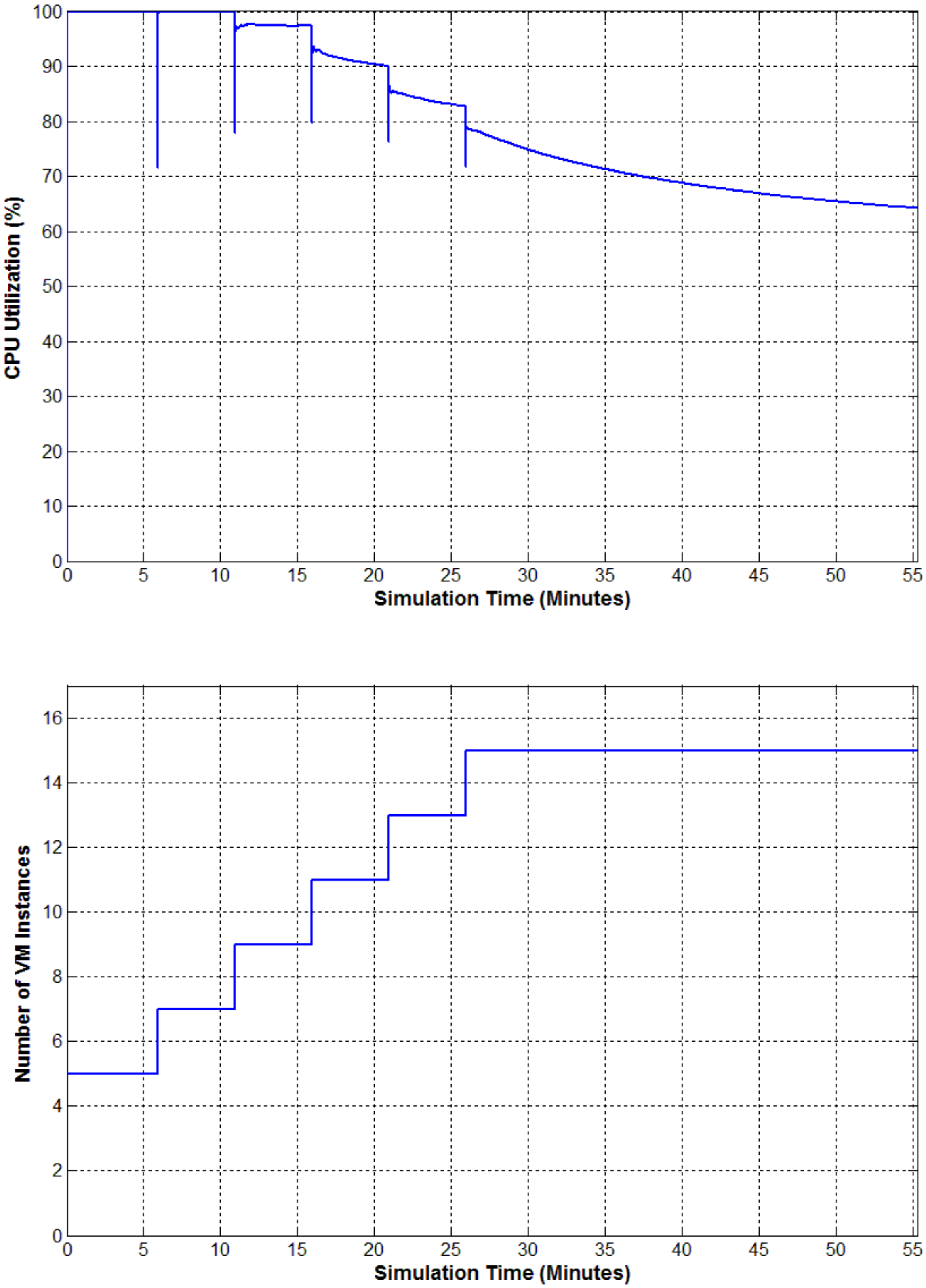


Figure 61 Resources Utilization Evaluation and the Number of Allocated VM Instances for the Simulation With Auto Scaling but Without Mitigation Technique (Simulation Case 2) at Rate of 0.8 KReq/sec in the Flash Overcrowd Mode

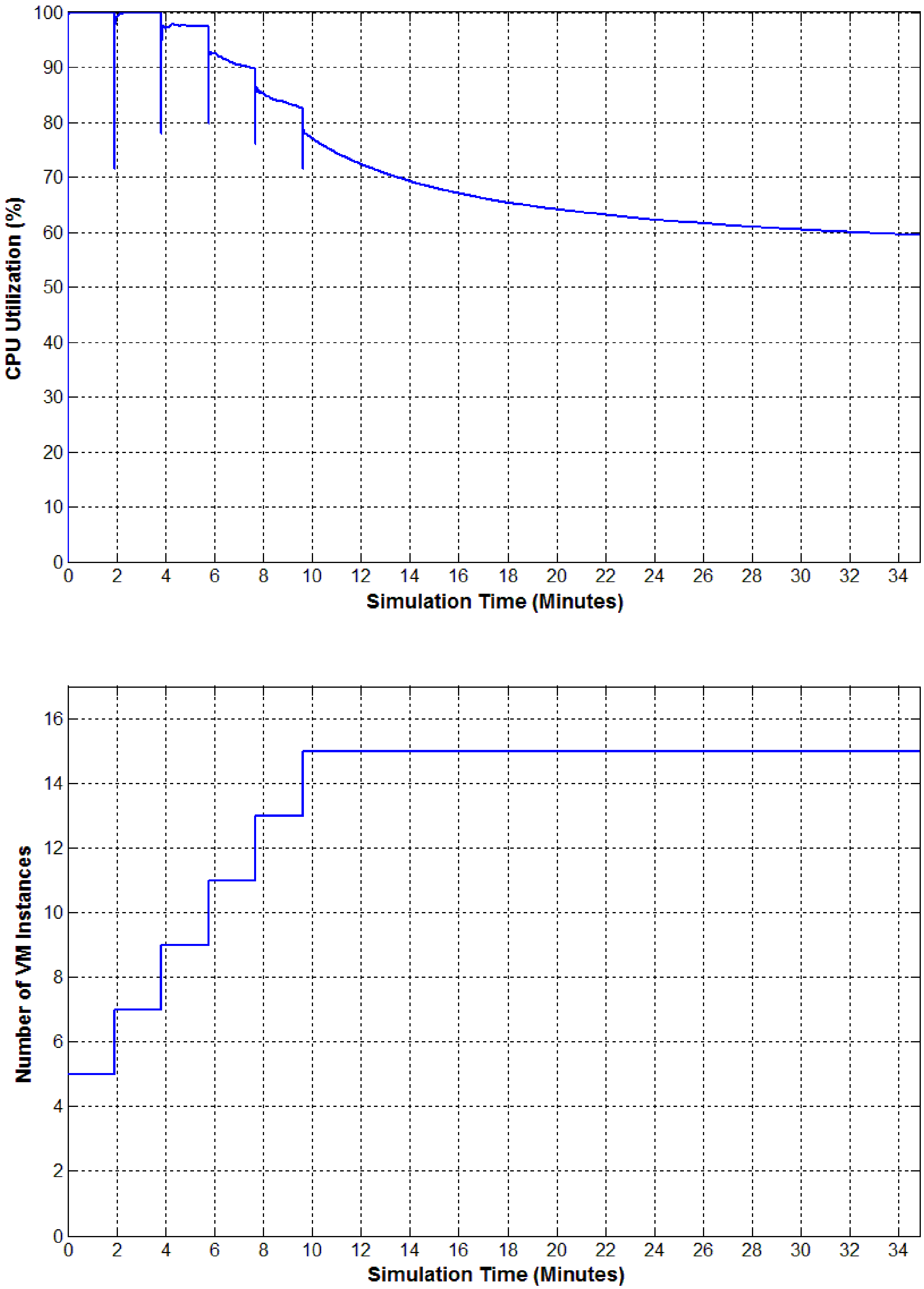


Figure 62 Resources Utilization Evaluation and the Number of Allocated VM Instances for EDoS Attack Defense Shell Simulation at Rate of 0.8 KReq/sec in the Flash Overcrowd Mode

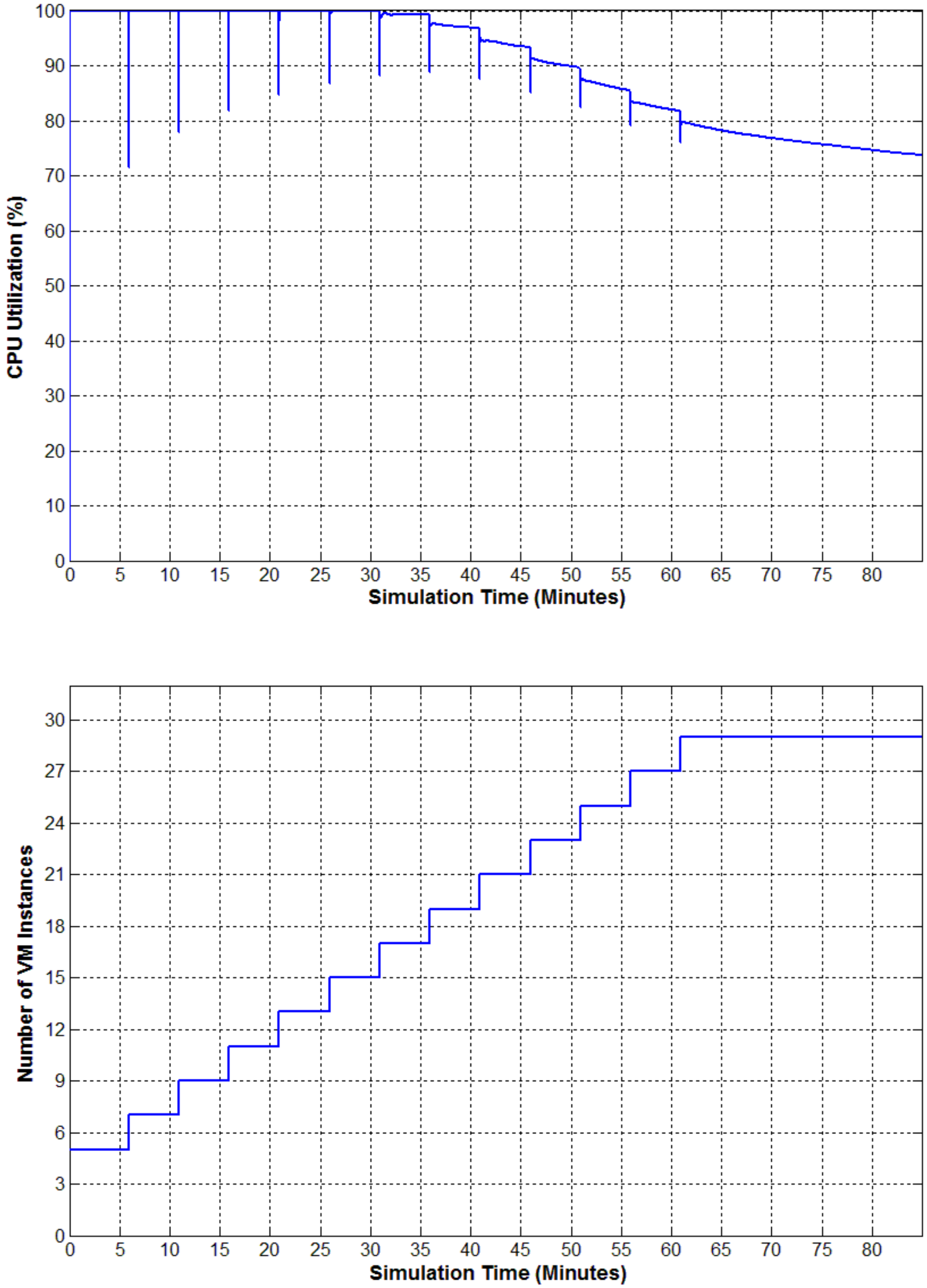


Figure 63 Resources Utilization Evaluation and the Number of Allocated VM Instances for the Simulation With Auto Scaling but Without Mitigation Technique (Simulation Case 2) at Rate of 1.6 KReq/sec in the Flash Overcrowd Mode

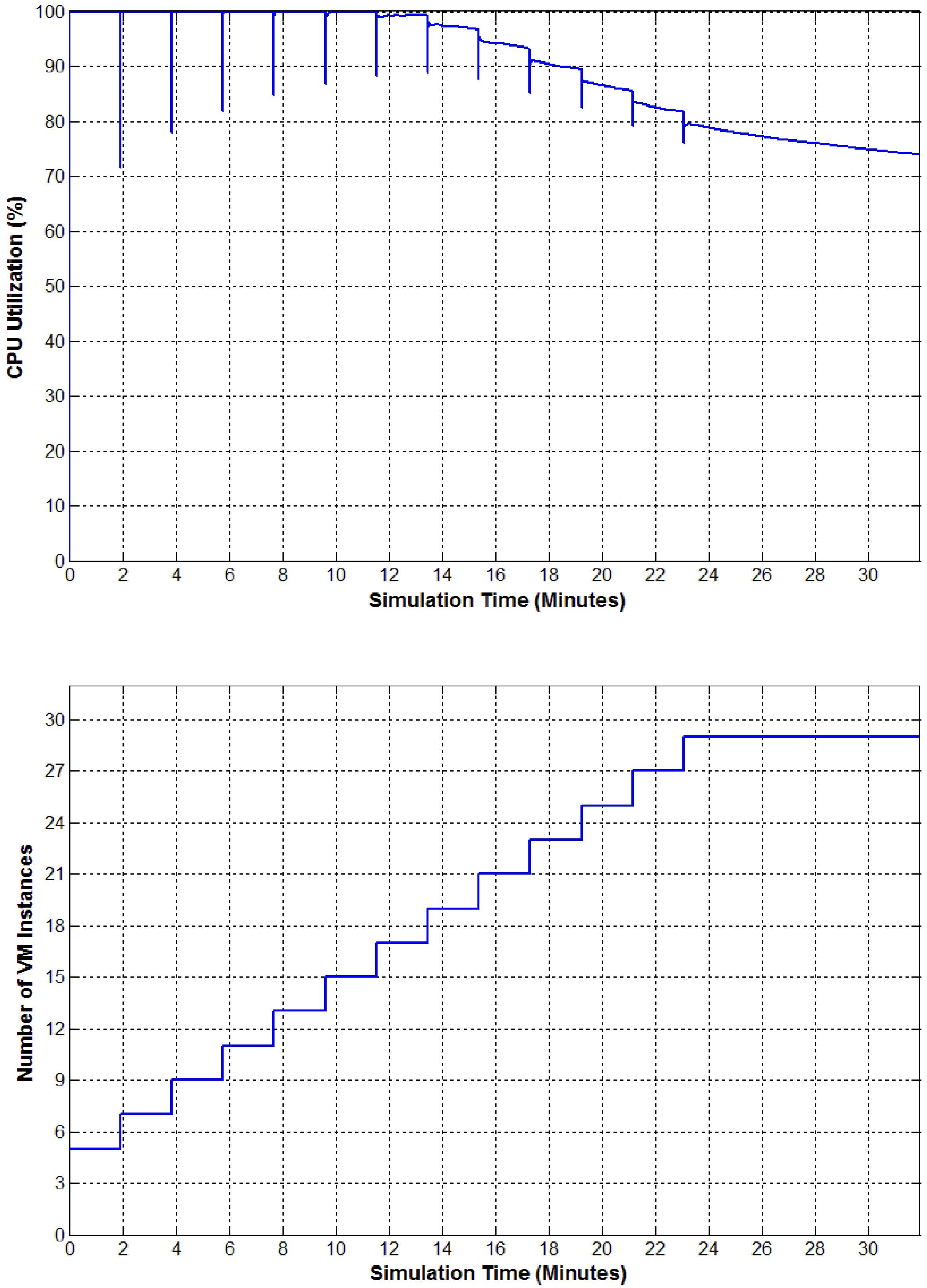


Figure 64 Resources Utilization Evaluation and the Number of Allocated VM Instances for EDoS Attack Defense Shell Simulation at Rate of 1.6 KReq/sec in the Flash Overcrowd Mode

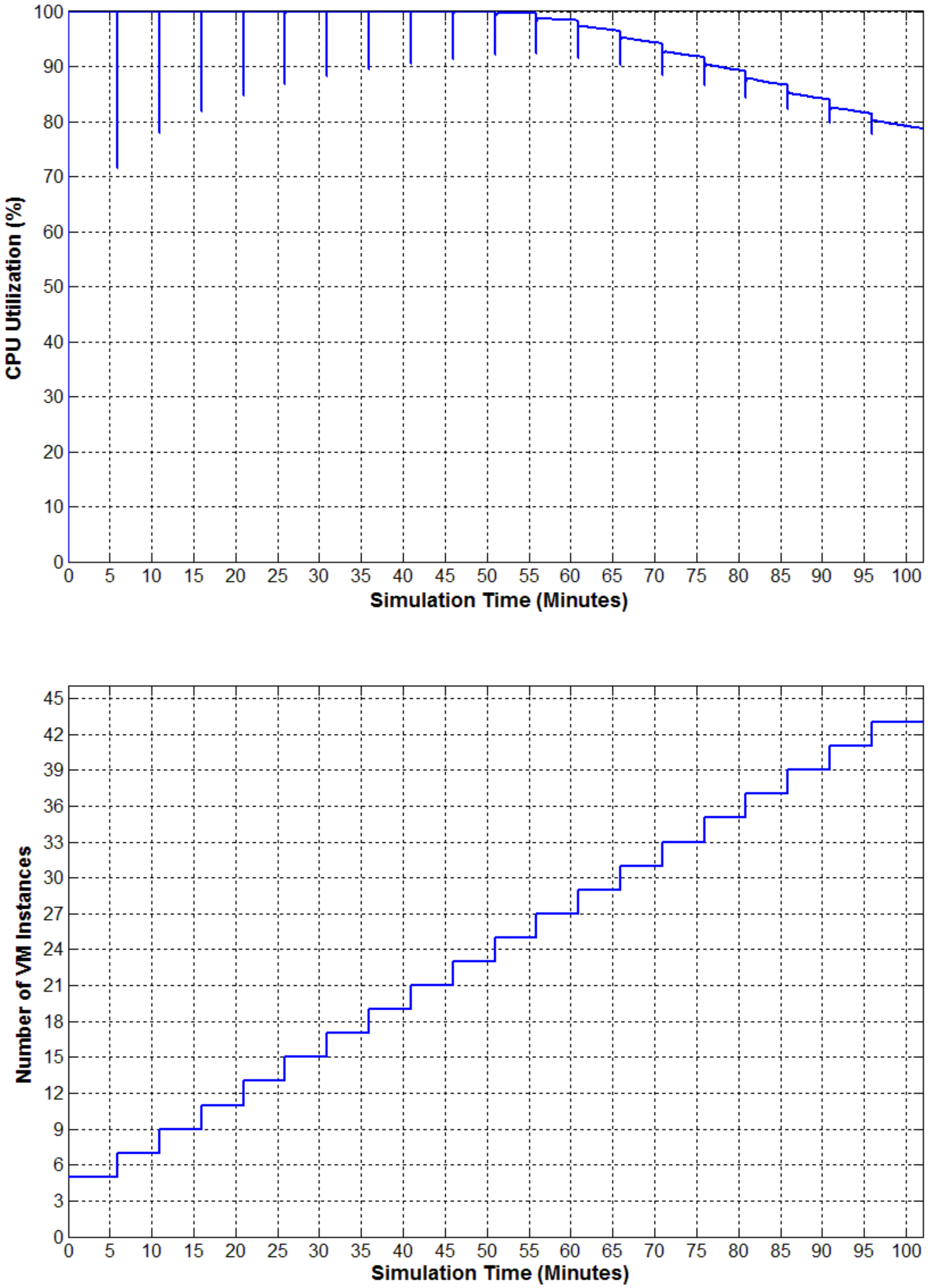


Figure 65 Resources Utilization Evaluation and the Number of Allocated VM Instances for the Simulation With Auto Scaling but Without Mitigation Technique (Simulation Case 2) at Rate of 2.4 KReq/sec in the Flash Overcrowd Mode

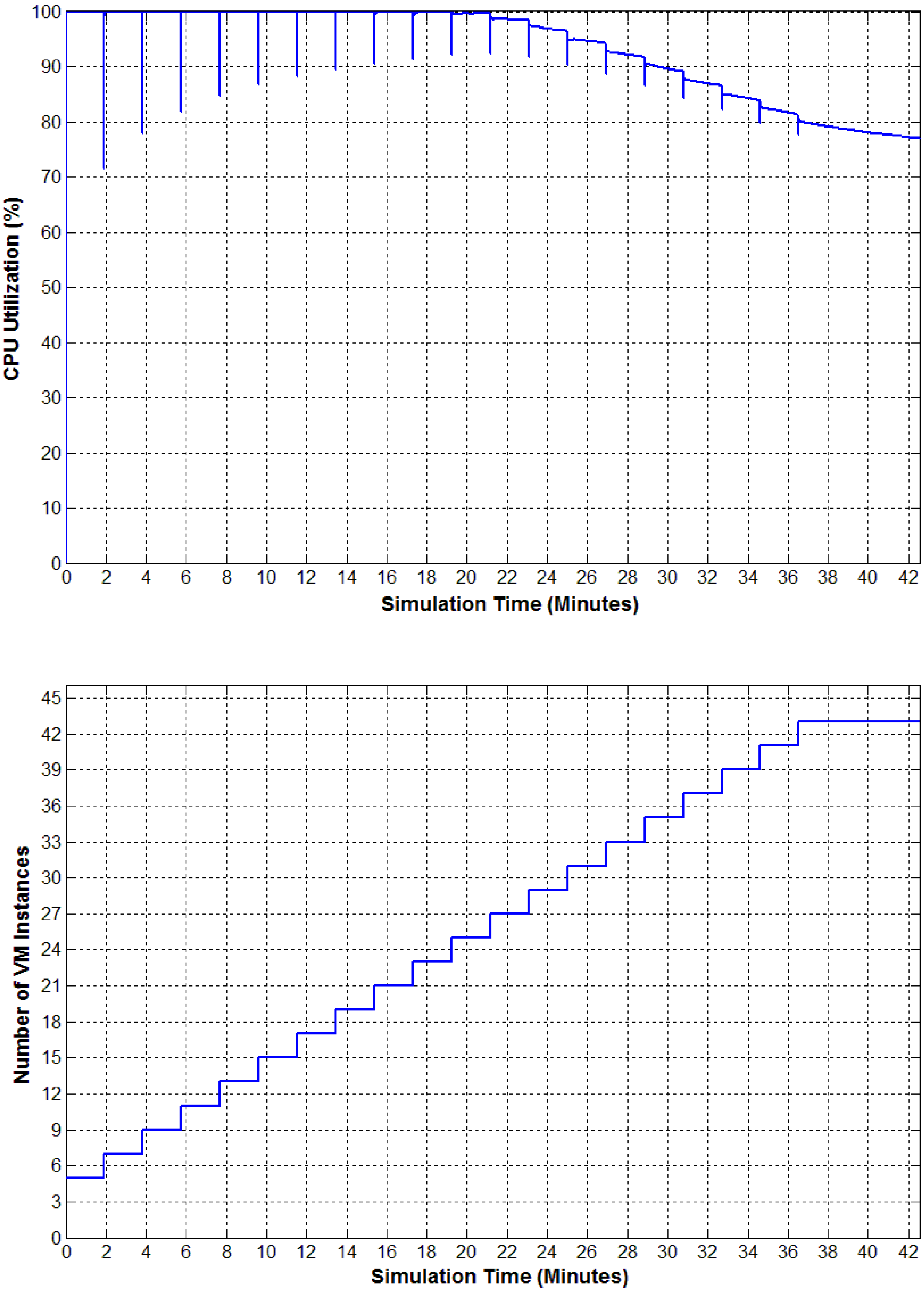


Figure 66 Resources Utilization Evaluation and the Number of Allocated VM Instances for EDoS Attack Defense Shell Simulation at Rate of 2.4 KReq/sec in the Flash Overcrowd Mode

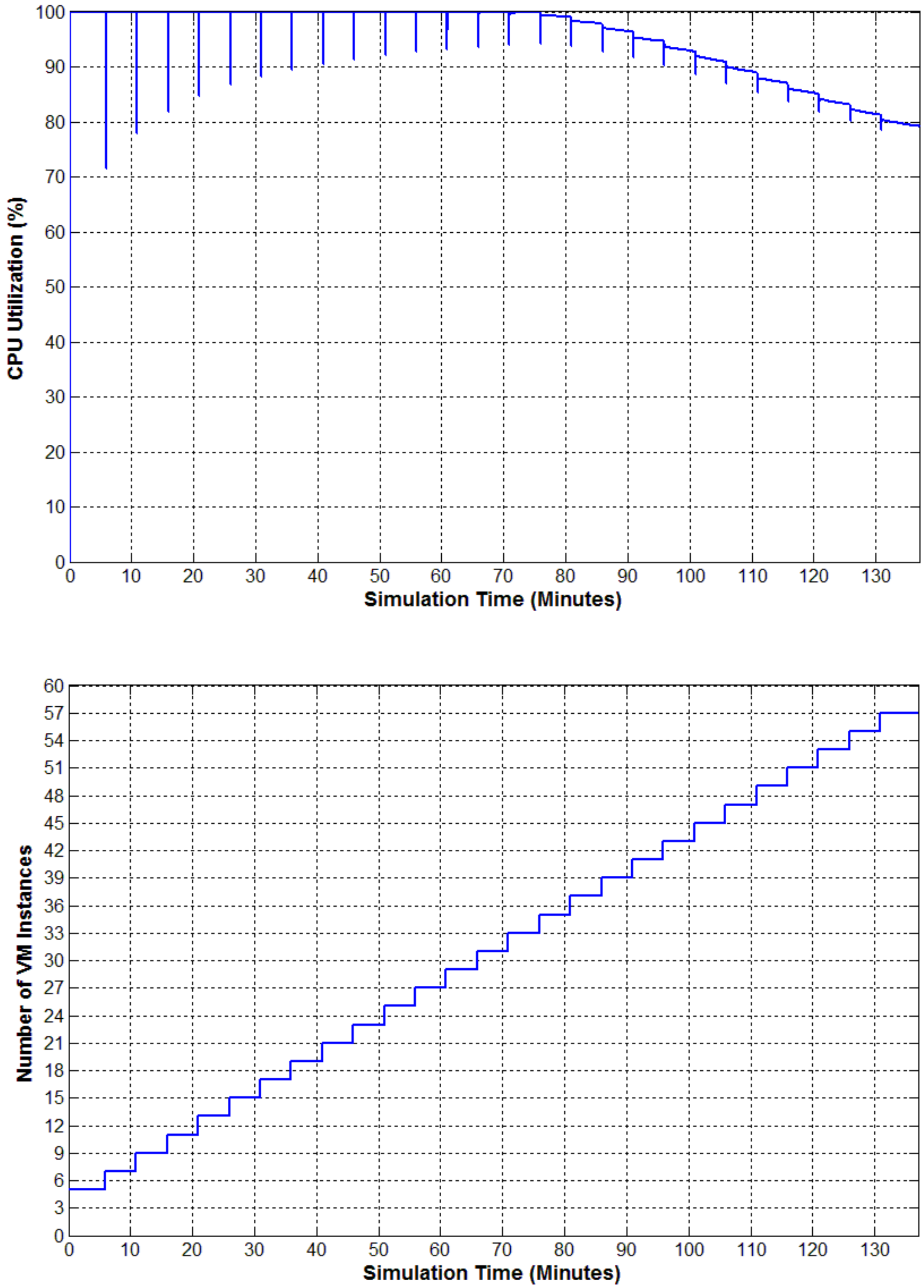


Figure 67 Resources Utilization Evaluation and the Number of Allocated VM Instances for the Simulation With Auto Scaling but Without Mitigation Technique (Simulation Case 2) at Rate of 3.2 KReq/sec in the Flash Overcrowd Mode

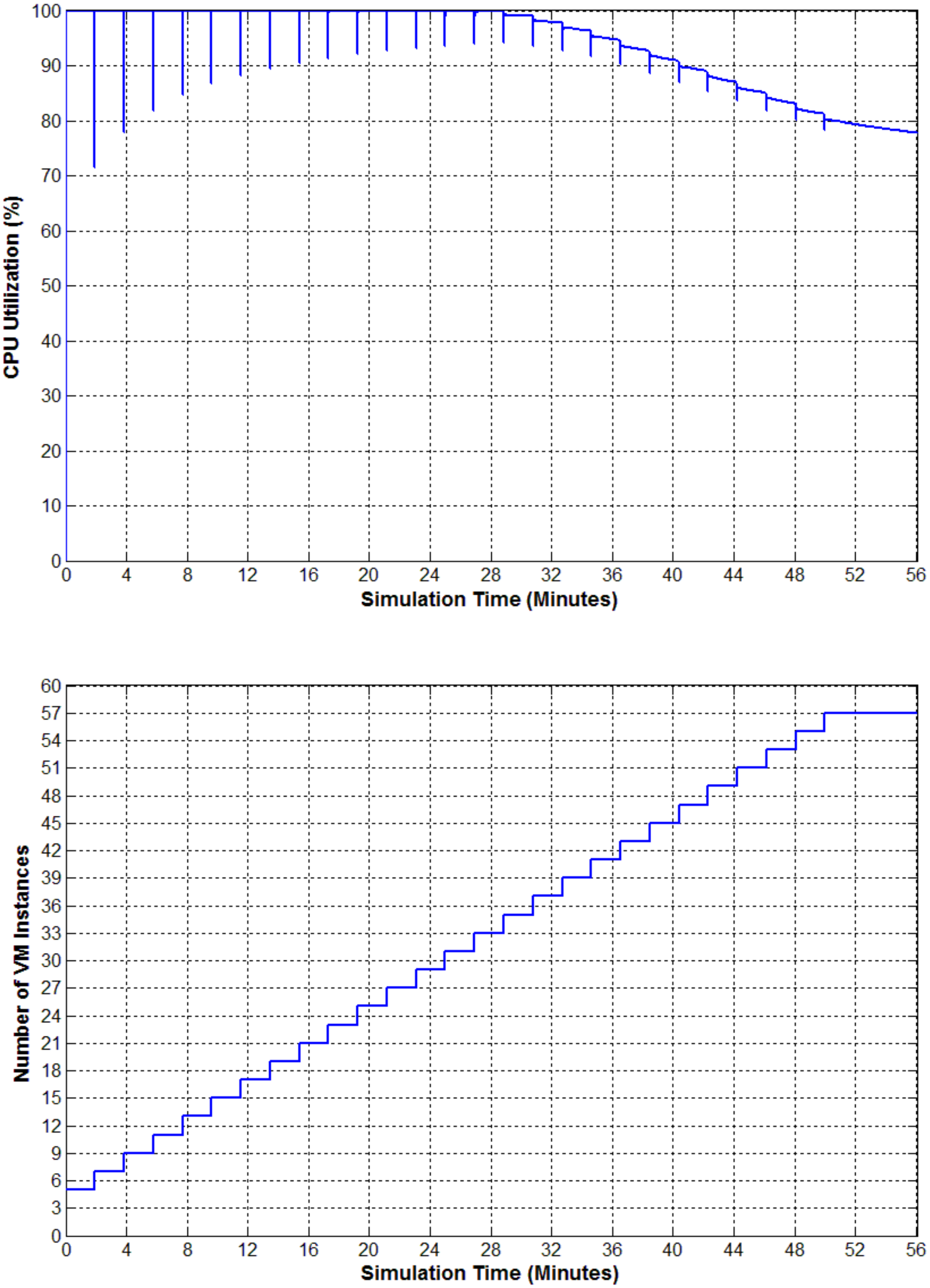


Figure 68 Resources Utilization Evaluation and the Number of Allocated VM Instances for EDoS Attack Defense Shell Simulation at Rate of 3.2 KReq/sec in the Flash Overcrowd Mode

APPENDIX B: RESPONSE TIME EVALUATION IN THE FLASH OVERCROWD MODE

The time average end-to-end response time evaluation for the simulation case 2 at rates of 0.4, 0.8, 1.6, 2.4, and 3.2 KReq/sec is shown in Figure 69, Figure 71, Figure 73, Figure 75, and Figure 77, respectively. In addition, the time average end-to-end response time evaluation for the EDoS Attack Defense Shell (EDoS-ADS) at rates of 0.4, 0.8, 1.6, 2.4, and 3.2 KReq/sec is shown in Figure 70, Figure 72, Figure 74, Figure 76, and Figure 78, respectively.

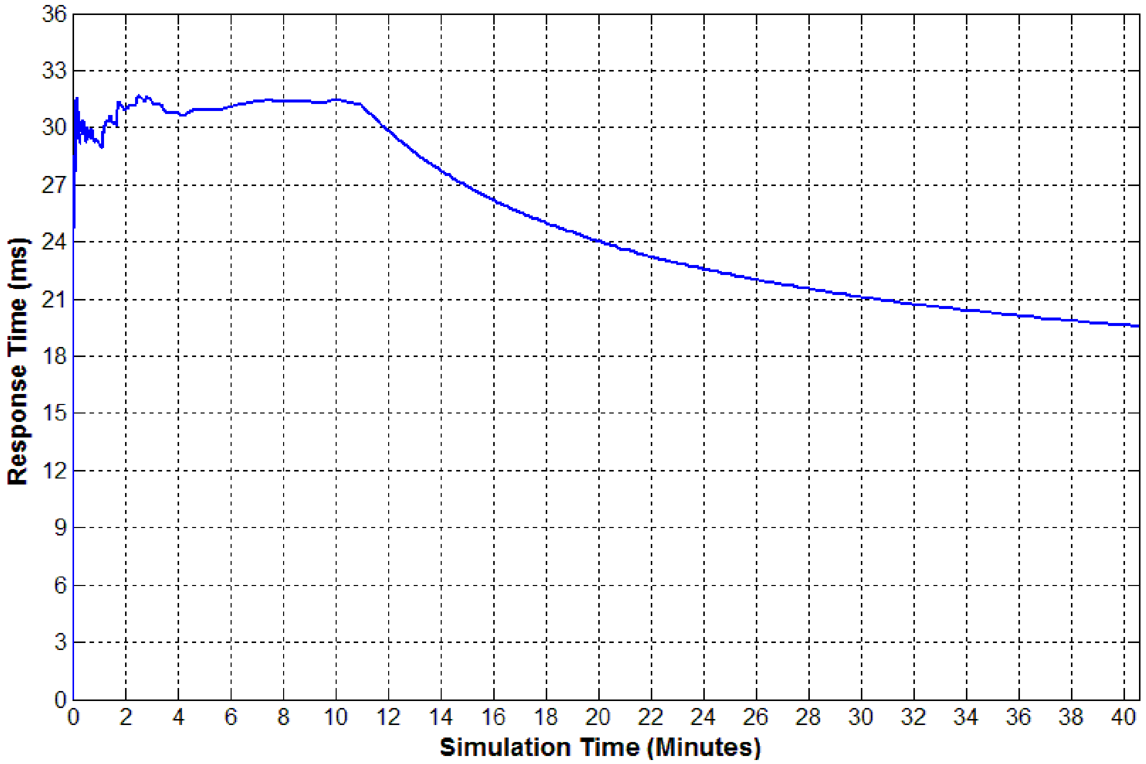


Figure 69 Response Time Evaluation for Simulation With Auto Scaling but Without Mitigation Technique (Simulation Case 2) at Rate of 0.4 KReq/sec in the Flash Overcrowd Mode

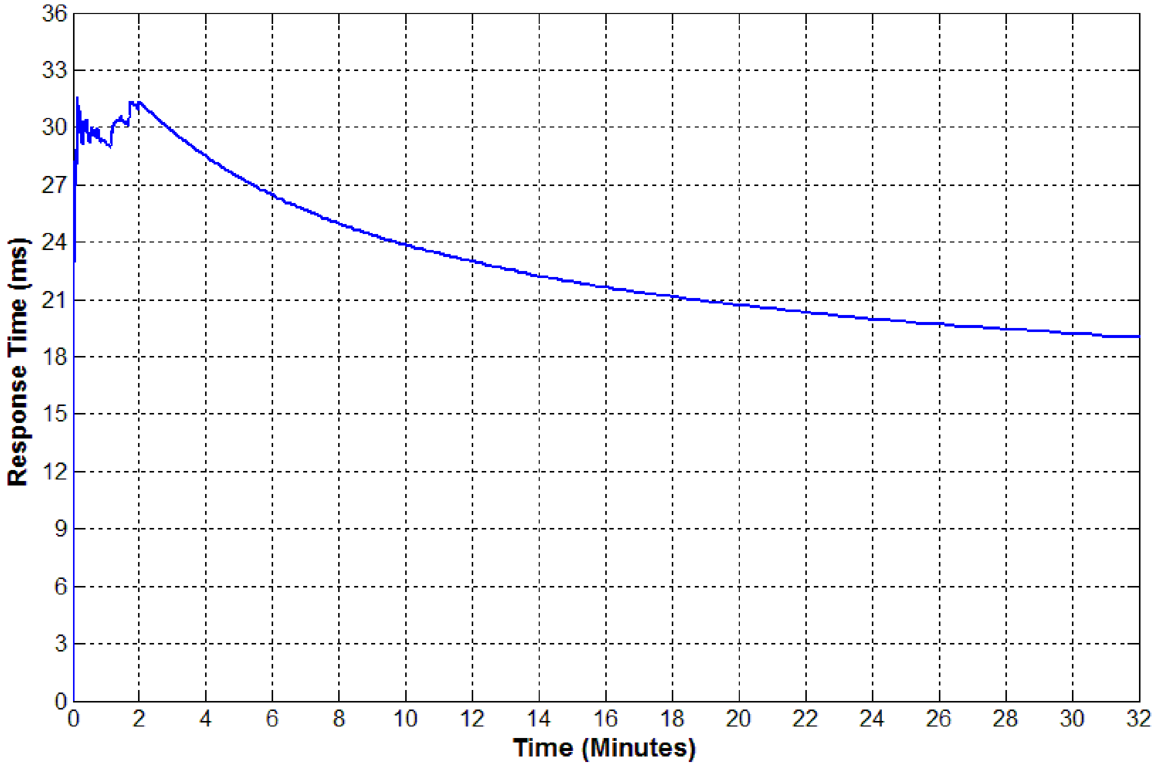


Figure 70 Response Time Evaluation for EDoS Attack Defense Shell Simulation at Rate of 0.4 KReq/sec in the Flash Overcrowd Mode

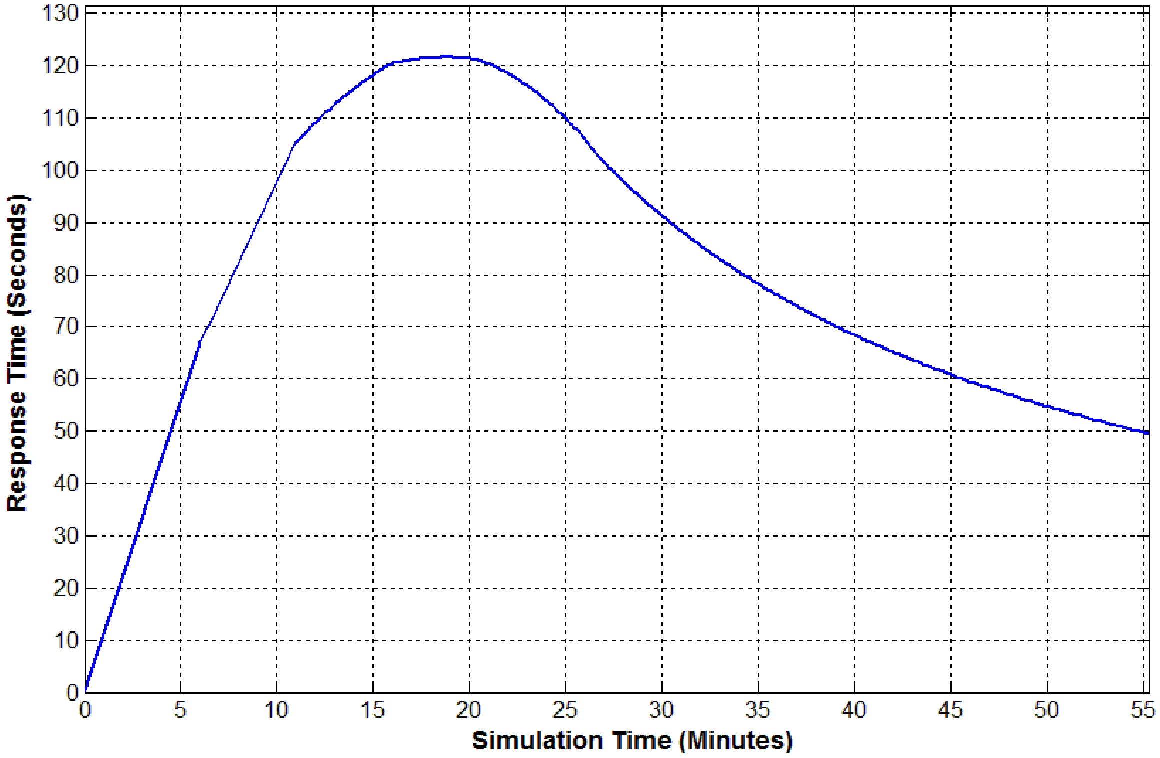


Figure 71 Response Time Evaluation for Simulation With Auto Scaling but Without Mitigation Technique (Simulation Case 2) at Rate of 0.8 KReq/sec in the Flash Overcrowd Mode

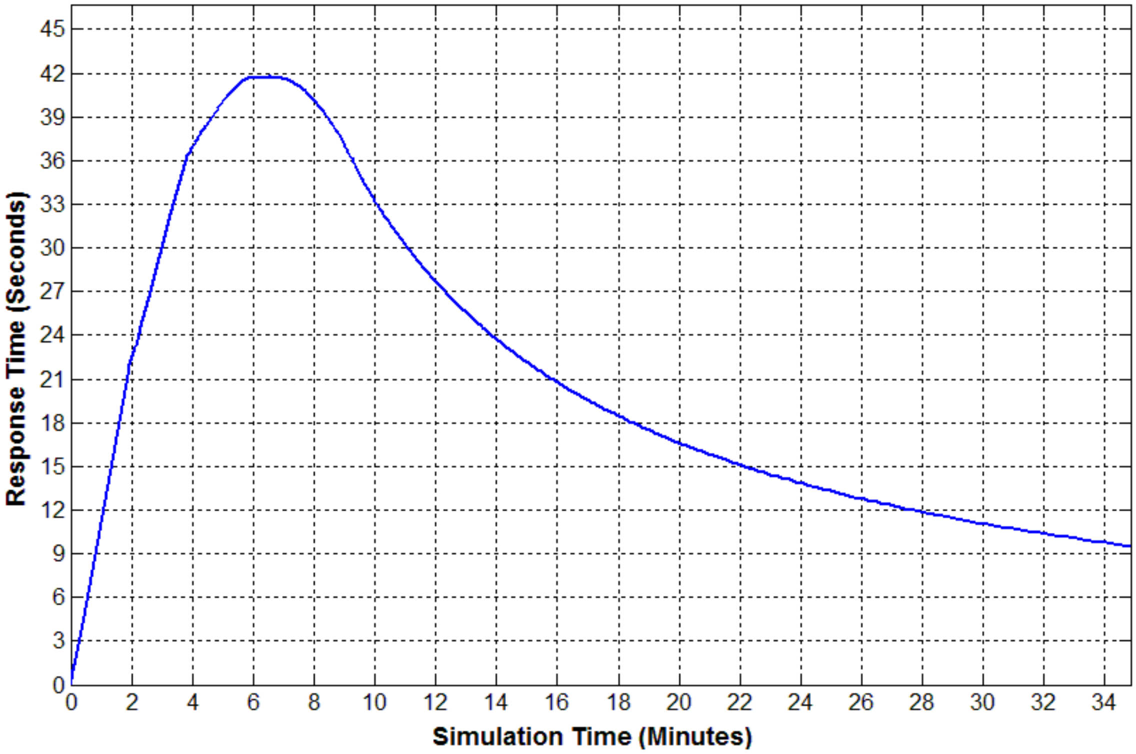


Figure 72 Response Time Evaluation for EDoS Attack Defense Shell Simulation at Rate of 0.8 KReq/sec in the Flash Overcrowd Mode

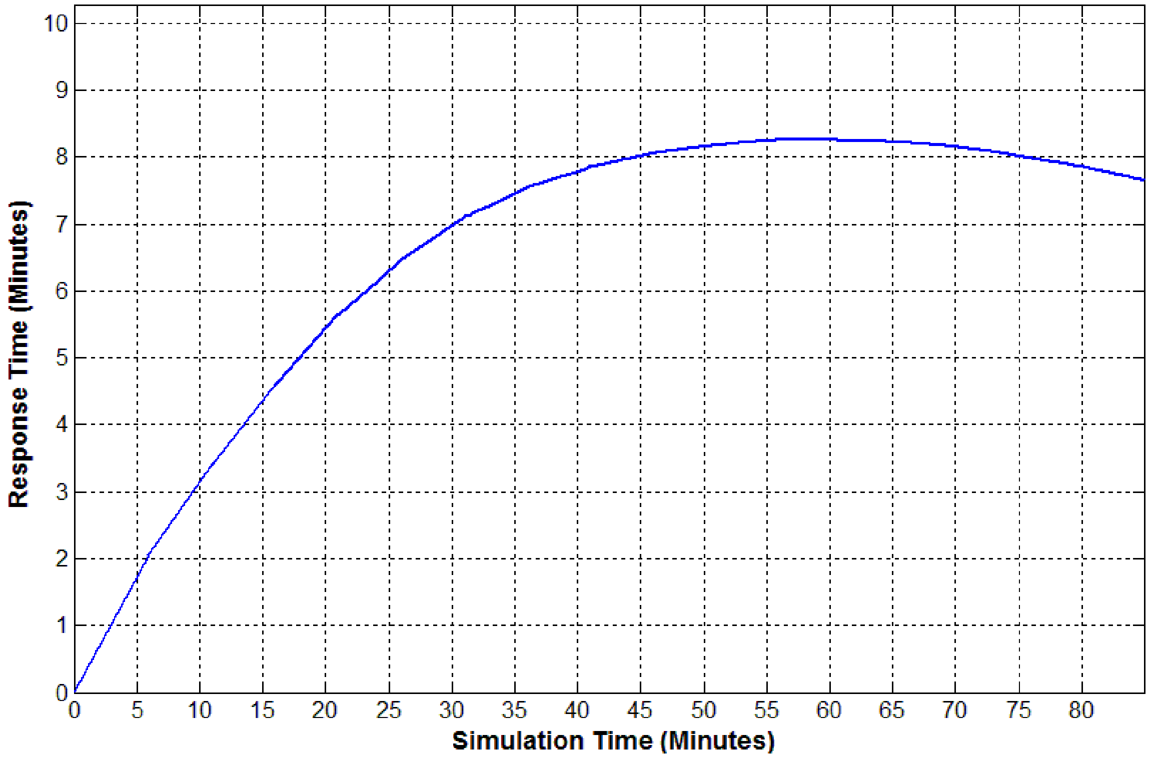


Figure 73 Response Time Evaluation for Simulation With Auto Scaling but Without Mitigation Technique (Simulation Case 2) at Rate of 1.6 KReq/sec in the Flash Overcrowd Mode

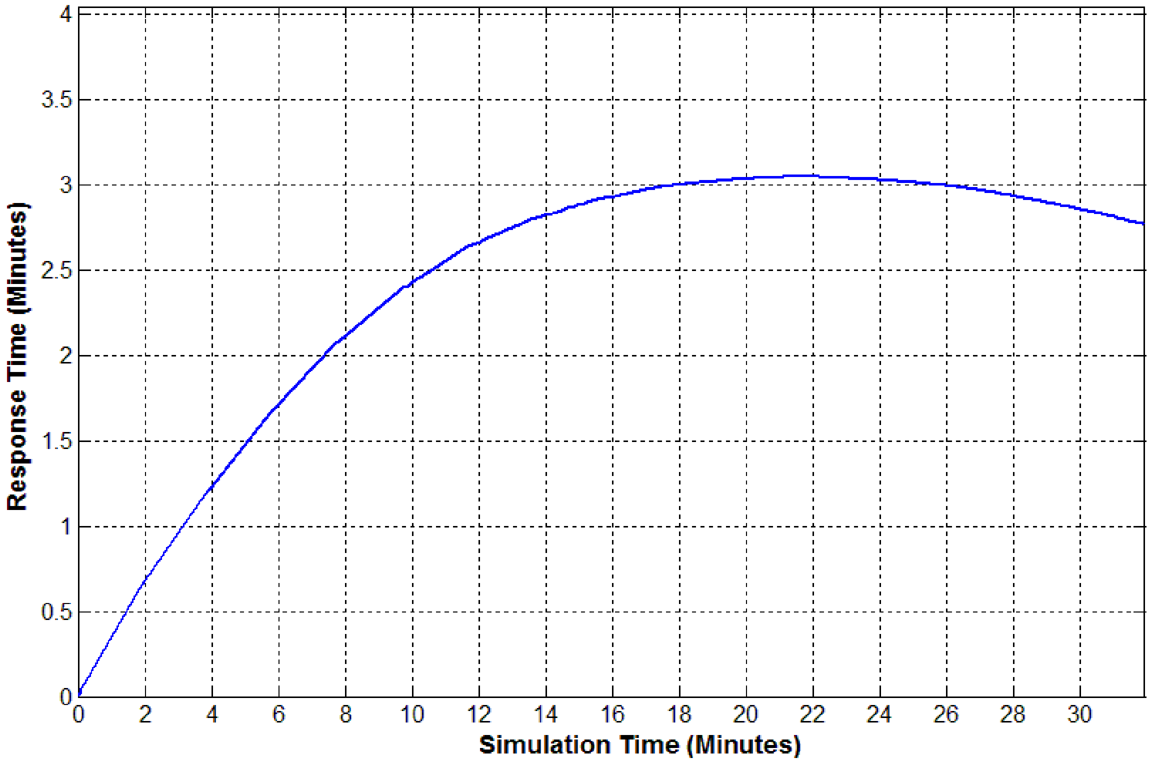


Figure 74 Response Time Evaluation for EDoS Attack Defense Shell Simulation at Rate of 1.6 KReq/sec in the Flash Overcrowd Mode

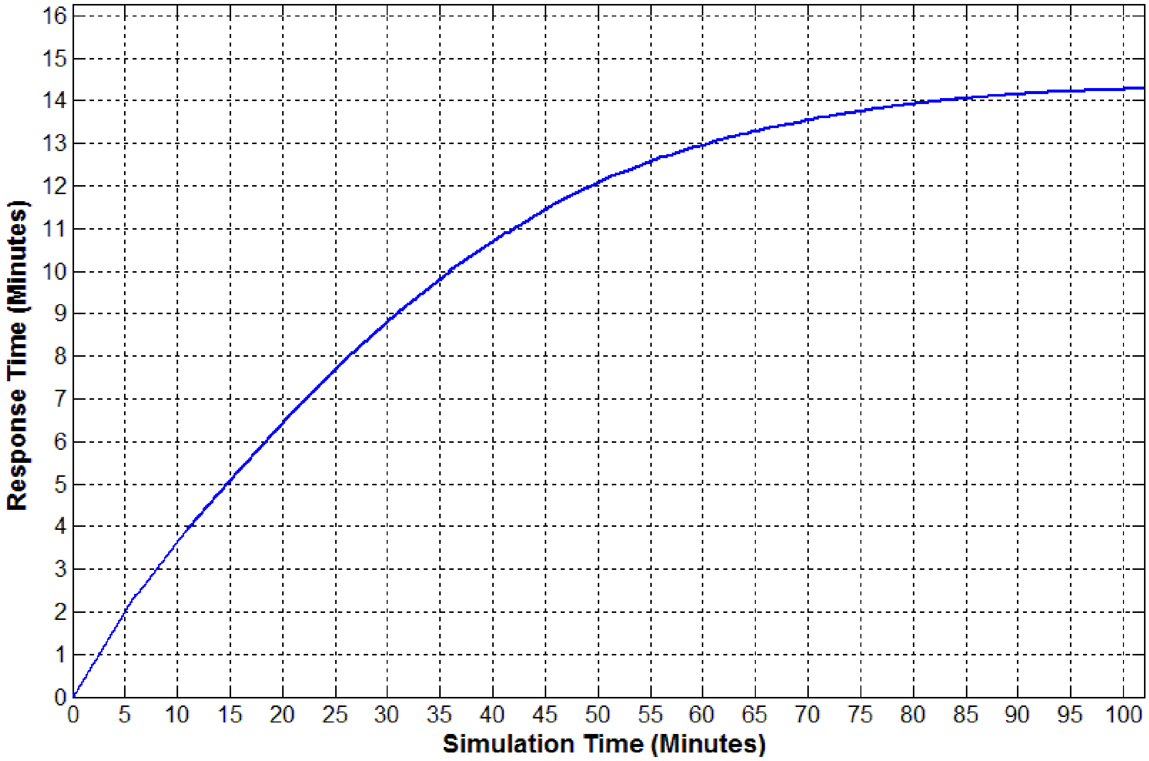


Figure 75 Response Time Evaluation for Simulation With Auto Scaling but Without Mitigation Technique (Simulation Case 2) at Rate of 2.4 KReq/sec in the Flash Overcrowd Mode

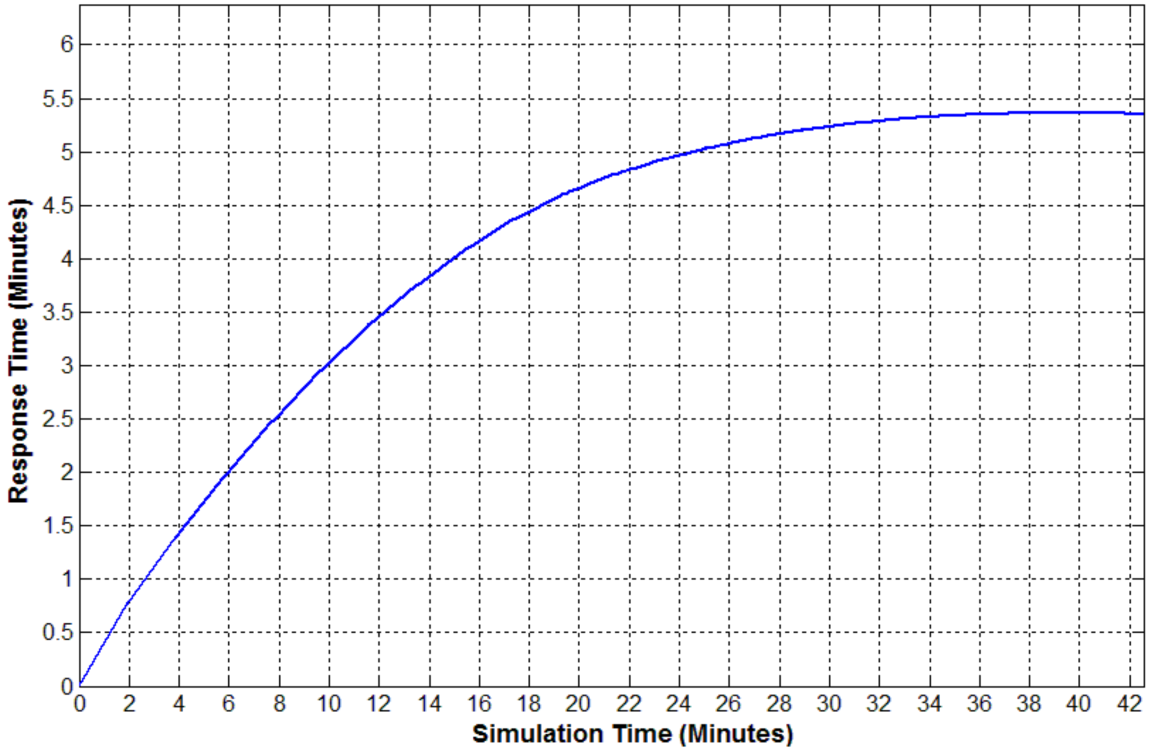


Figure 76 Response Time Evaluation for EDoS Attack Defense Shell Simulation at Rate of 2.4 KReq/sec in the Flash Overcrowd Mode

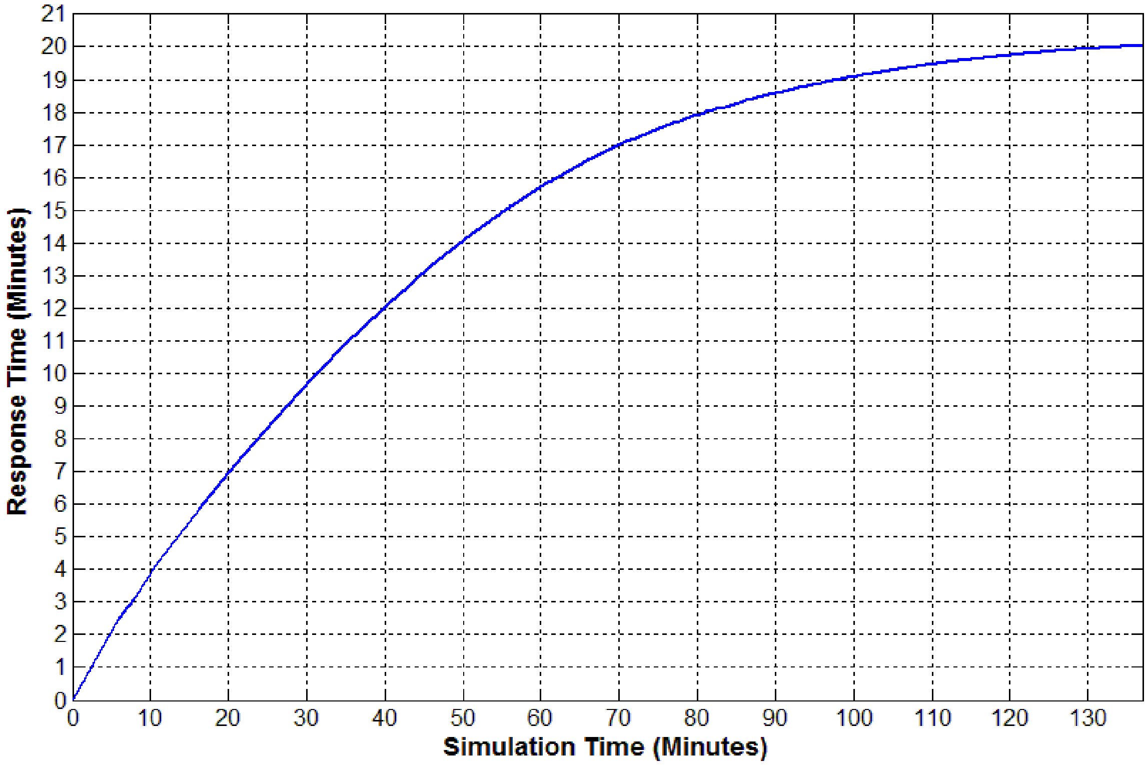


Figure 77 Response Time Evaluation for Simulation With Auto Scaling but Without Mitigation Technique (Simulation Case 2) at Rate of 3.2 KReq/sec in the Flash Overcrowd Mode

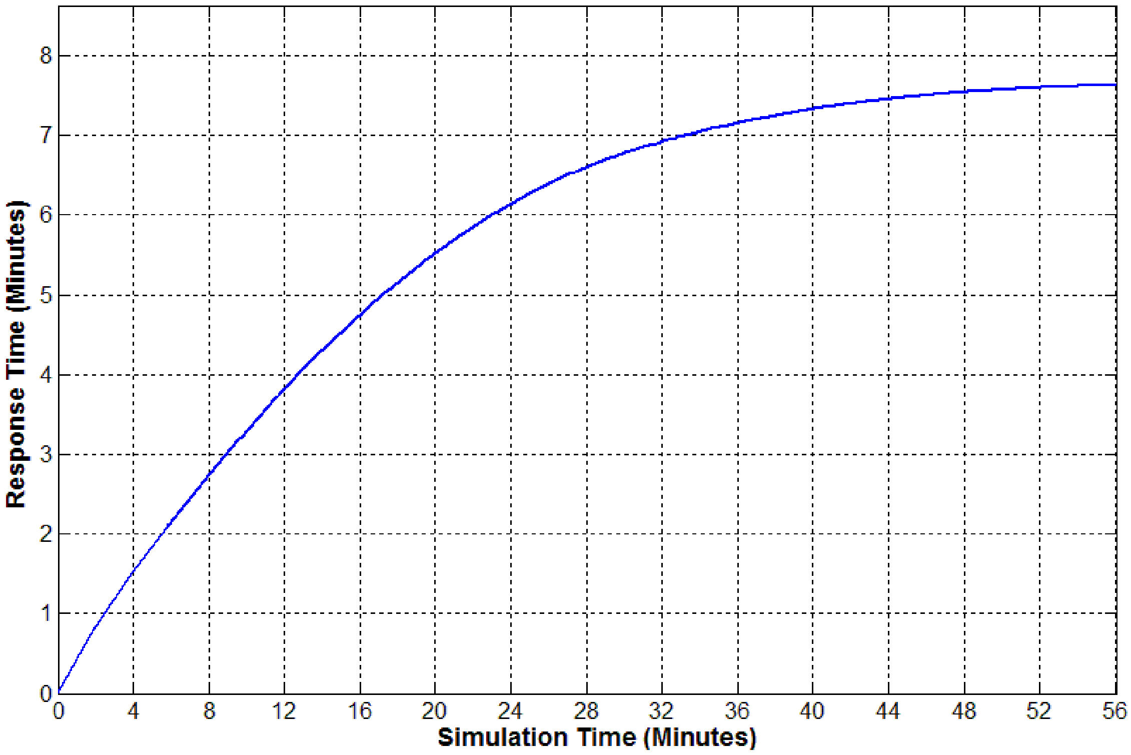


Figure 78 Response Time Evaluation for EDoS Attack Defense Shell Simulation at Rate of 3.2 KReq/sec in the Flash Overcrowd Mode

APPENDIX C: THROUGHPUT EVALUATION IN THE FLASH OVERCROWD MODE

The throughput evaluation for the simulation case 2 at rates of 0.4, 0.8, 1.6, 2.4, and 3.2 KReq/sec is shown in Figure 79, Figure 81, Figure 83, Figure 85, and Figure 87, respectively. In addition, the throughput evaluation for the EDoS Attack Defense Shell (EDoS-ADS) at rates of 0.4, 0.8, 1.6, 2.4, and 3.2 KReq/sec is shown in Figure 80, Figure 82, Figure 84, Figure 86, and Figure 88, respectively.

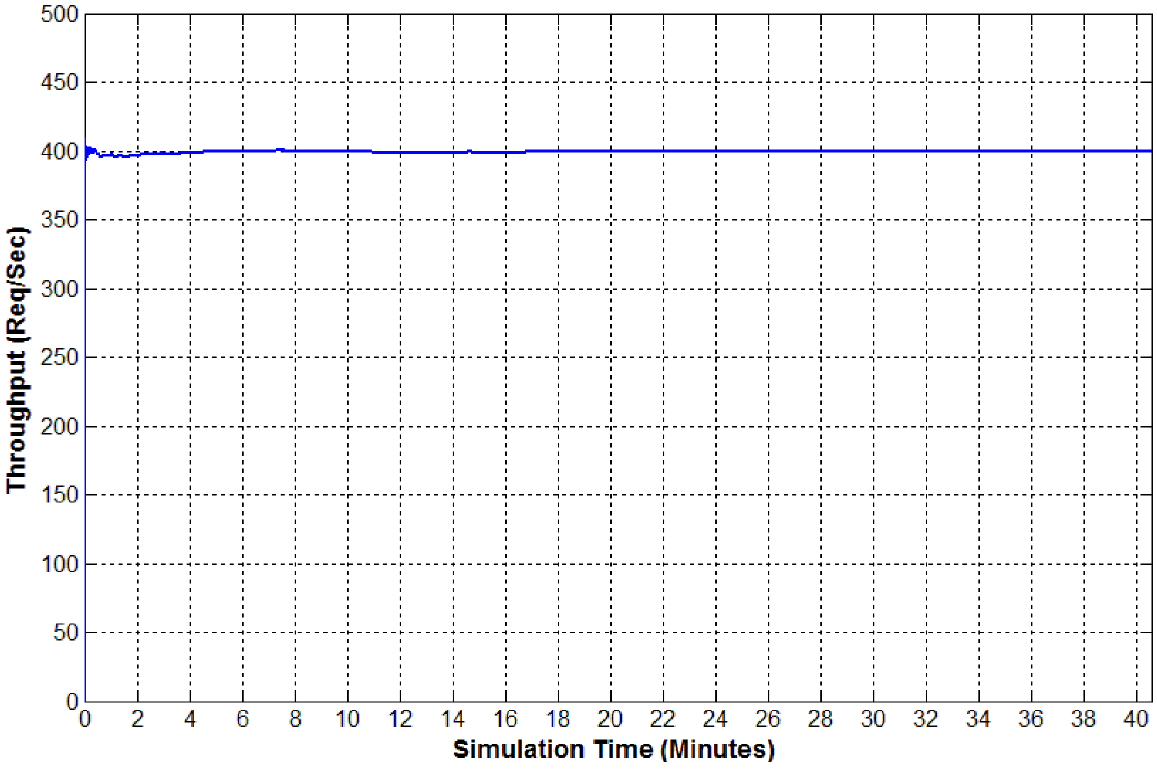


Figure 79 Throughput Evaluation for Simulation With Auto Scaling but Without Mitigation Technique (Simulation Case 2) at Rate of 0.4 KReq/sec in the Flash Overcrowd Mode

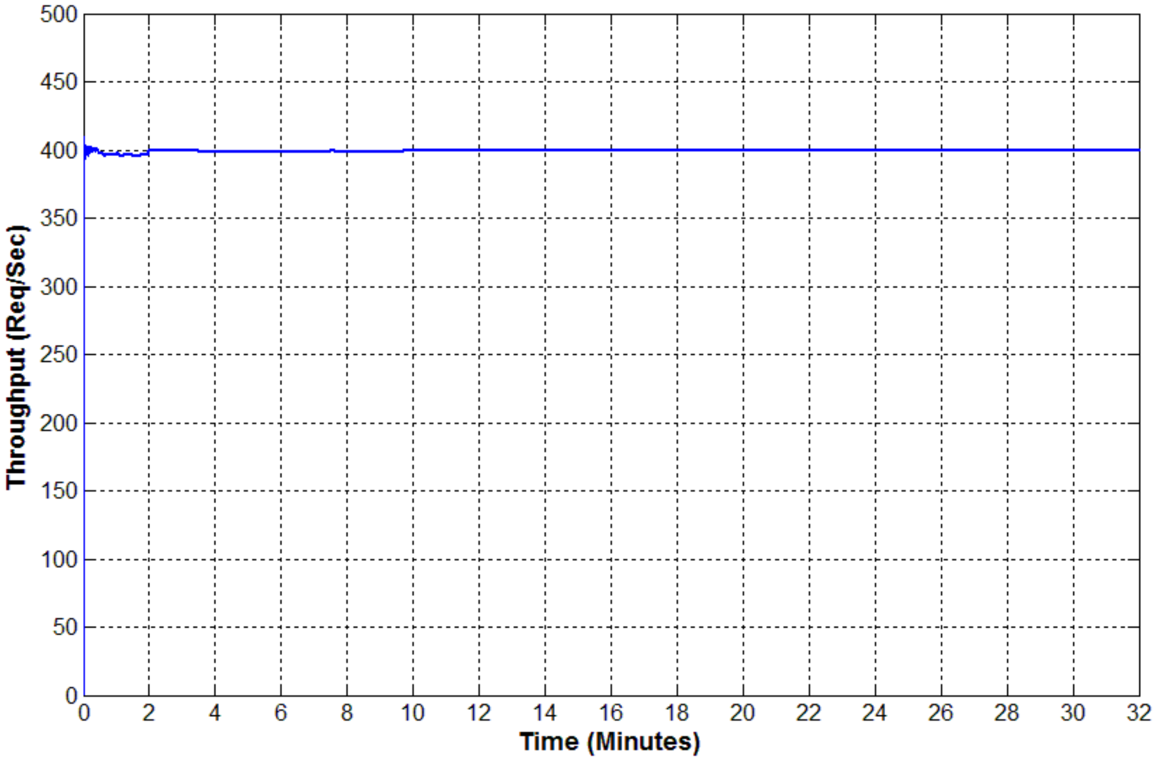


Figure 80 Throughput Evaluation for EDoS Attack Defense Shell Simulation at Rate of 0.4 KReq/sec in the Flash Overcrowd Mode

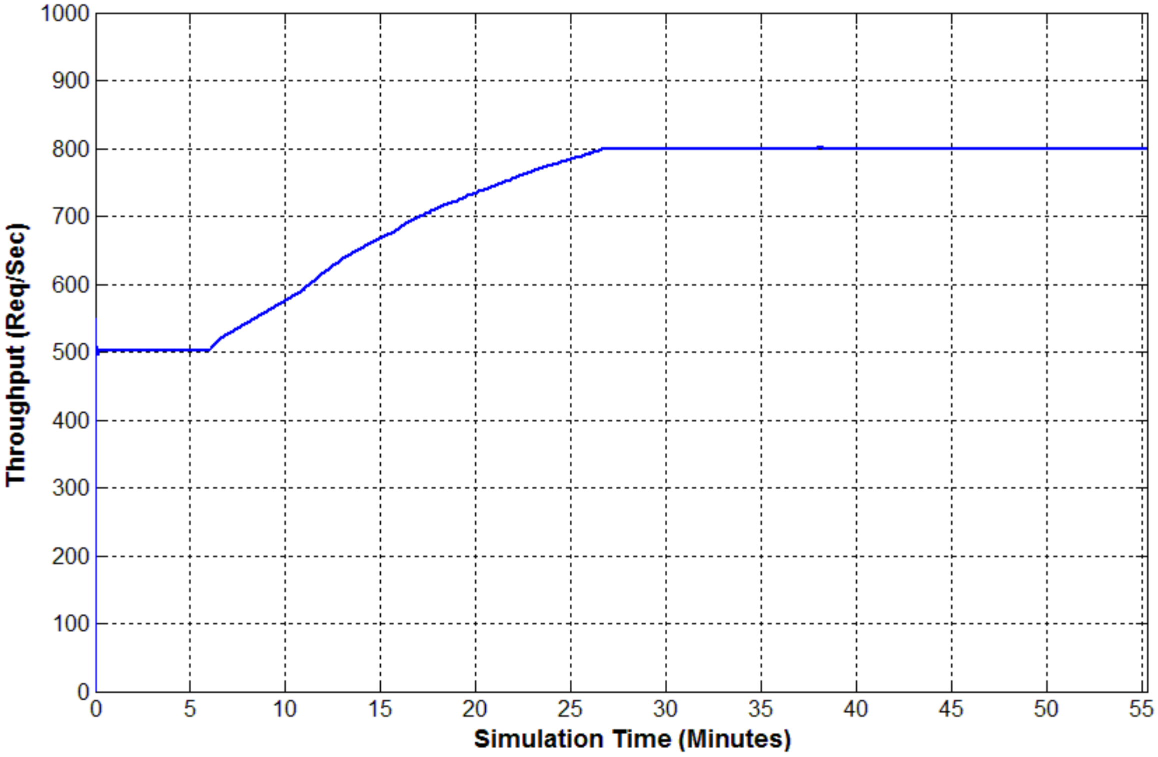


Figure 81 Throughput Evaluation for Simulation With Auto Scaling but Without Mitigation Technique (Simulation Case 2) at Rate of 0.8 KReq/sec in the Flash Overcrowd Mode

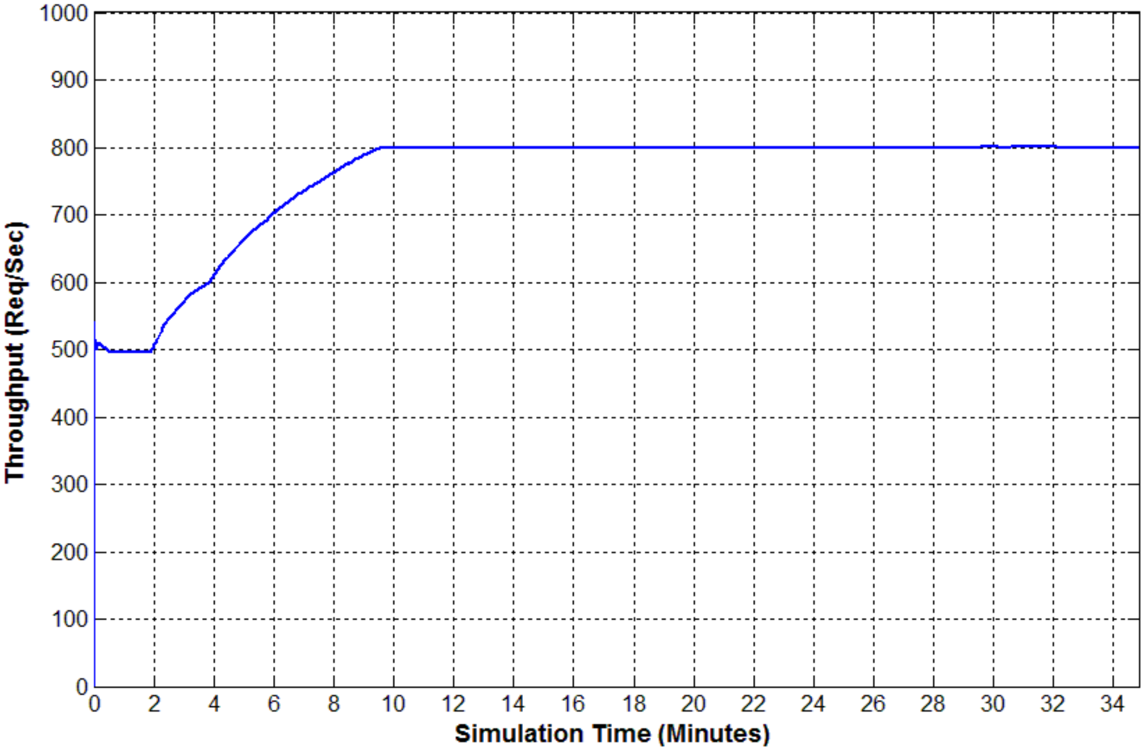


Figure 82 Throughput Evaluation for EDoS Attack Defense Shell Simulation at Rate of 0.8 KReq/sec in the Flash Overcrowd Mode

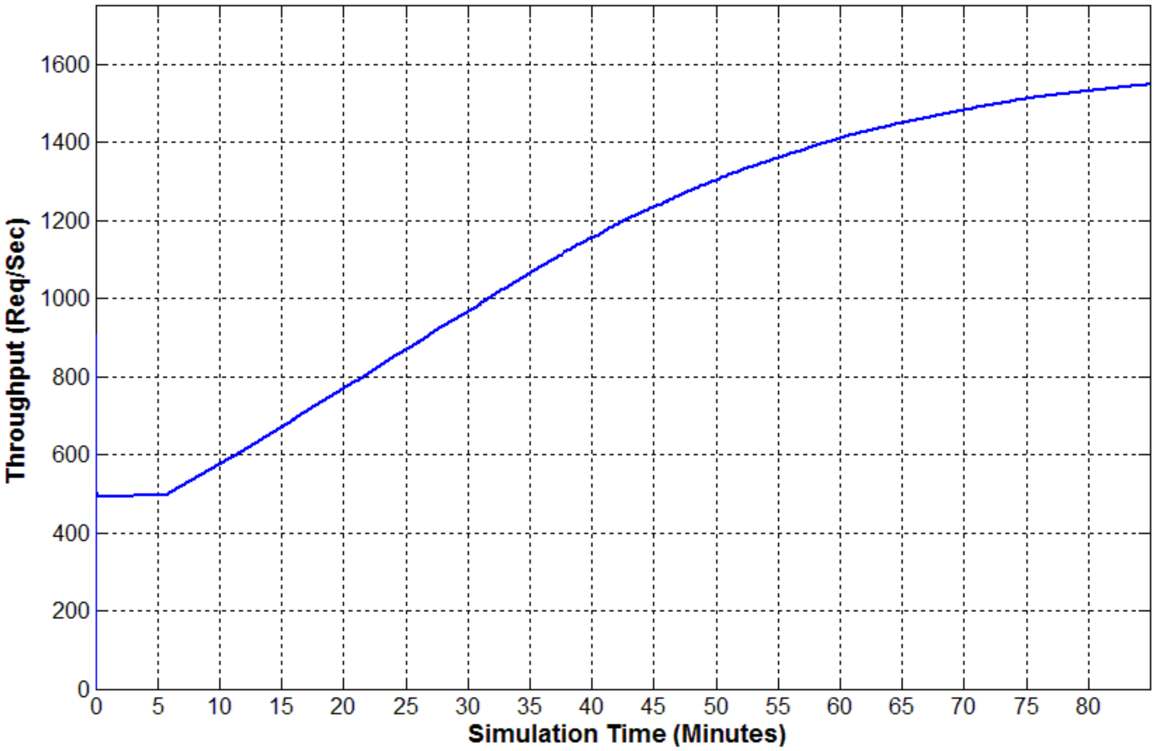


Figure 83 Throughput Evaluation for Simulation With Auto Scaling but Without Mitigation Technique (Simulation Case 2) at Rate of 1.6 KReq/sec in the Flash Overcrowd Mode

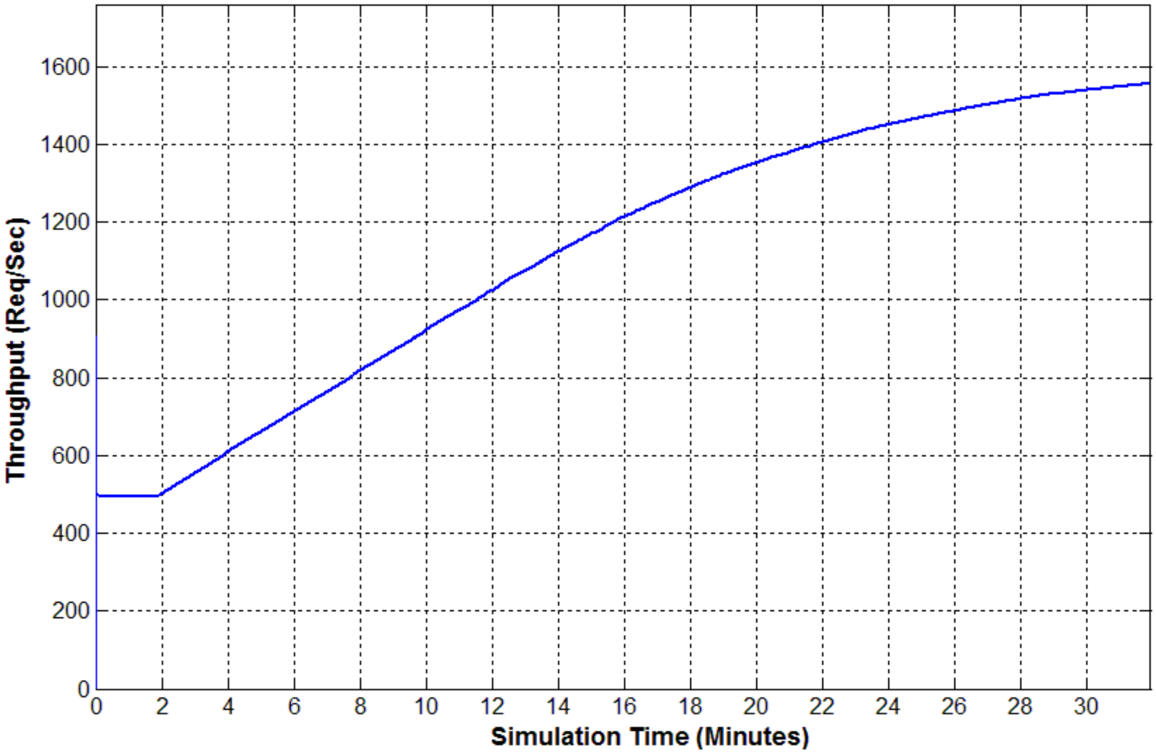


Figure 84 Throughput Evaluation for EDoS Attack Defense Shell Simulation at Rate of 1.6 KReq/sec in the Flash Overcrowd Mode

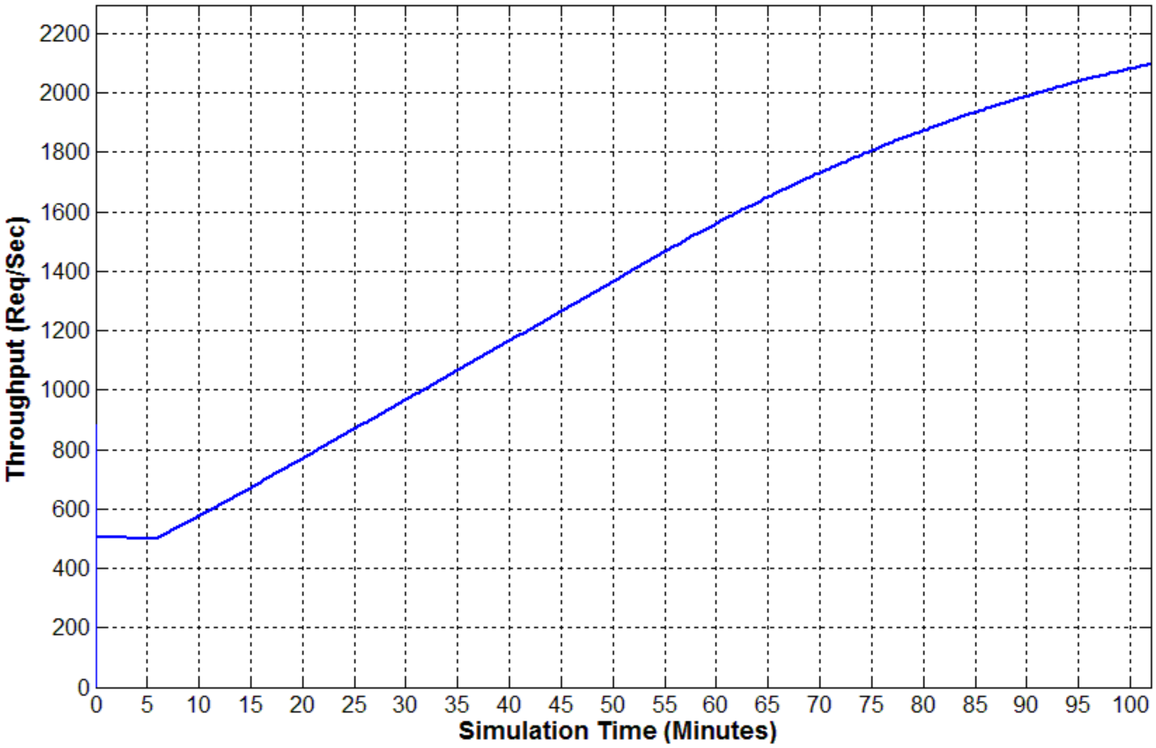


Figure 85 Throughput Evaluation for Simulation With Auto Scaling but Without Mitigation Technique (Simulation Case 2) at Rate of 2.4 KReq/sec in the Flash Overcrowd Mode

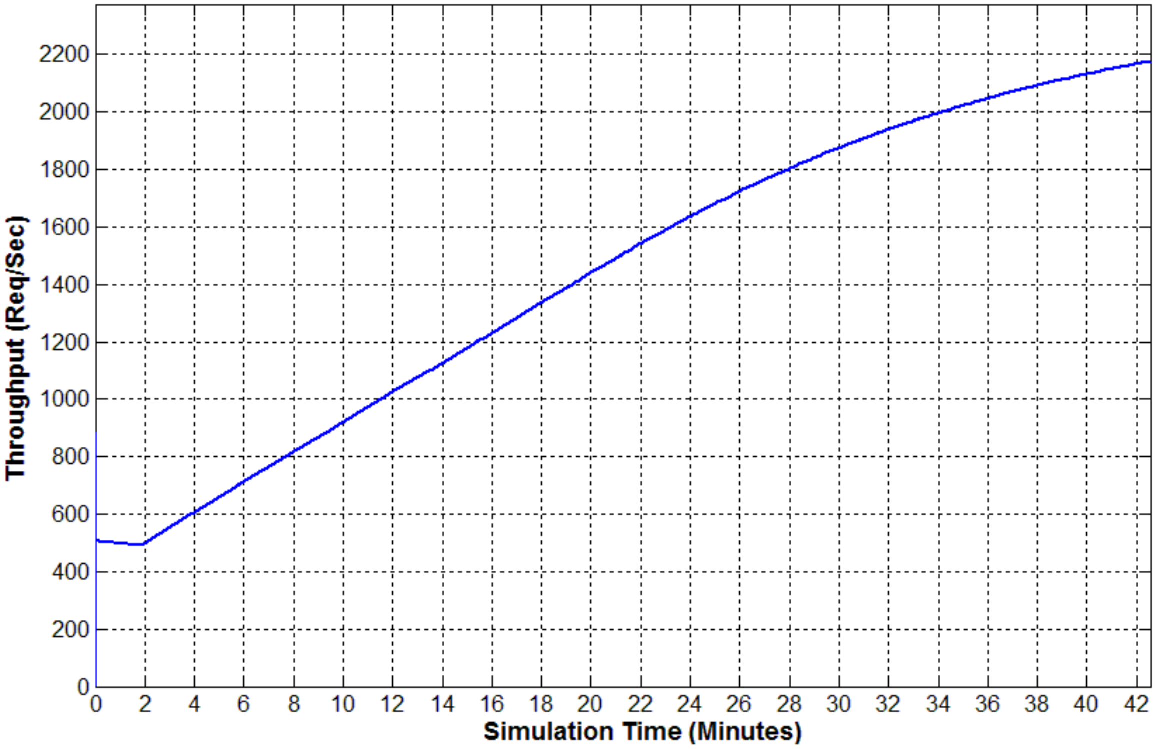


Figure 86 Throughput Evaluation for EDoS Attack Defense Shell Simulation at Rate of 2.4 KReq/sec in the Flash Overcrowd Mode

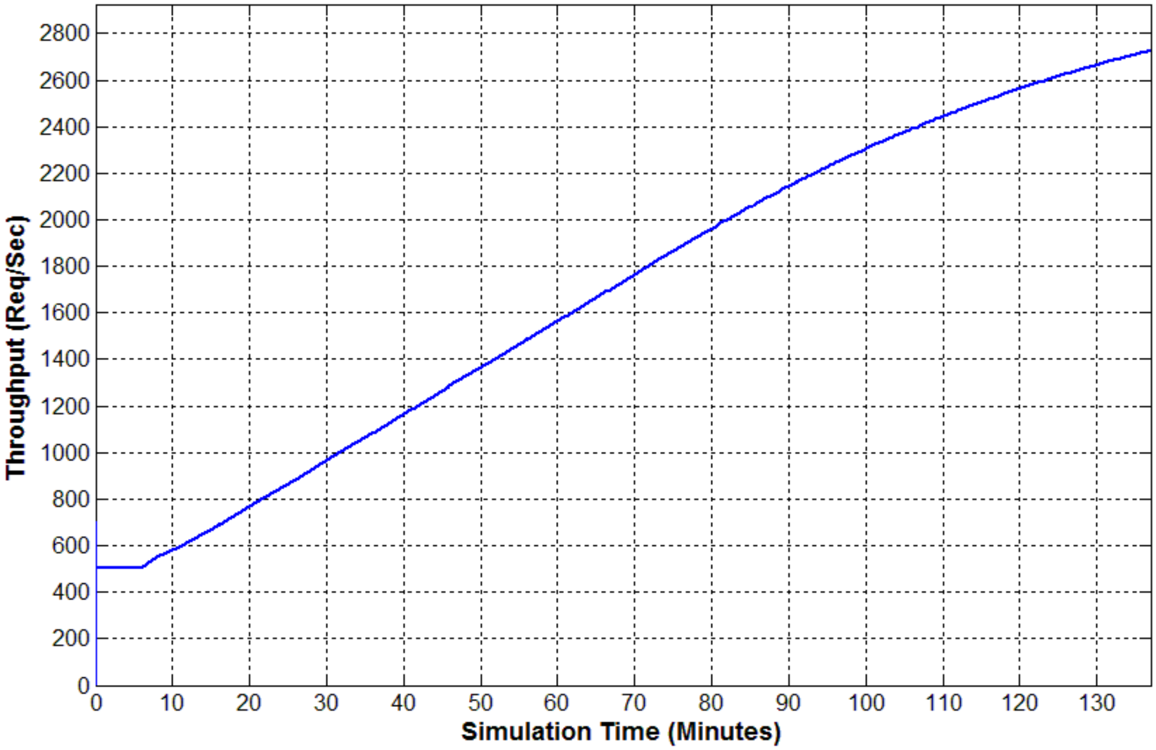


Figure 87 Throughput Evaluation for Simulation With Auto Scaling but Without Mitigation Technique (Simulation Case 2) at Rate of 3.2 KReq/sec in the Flash Overcrowd Mode

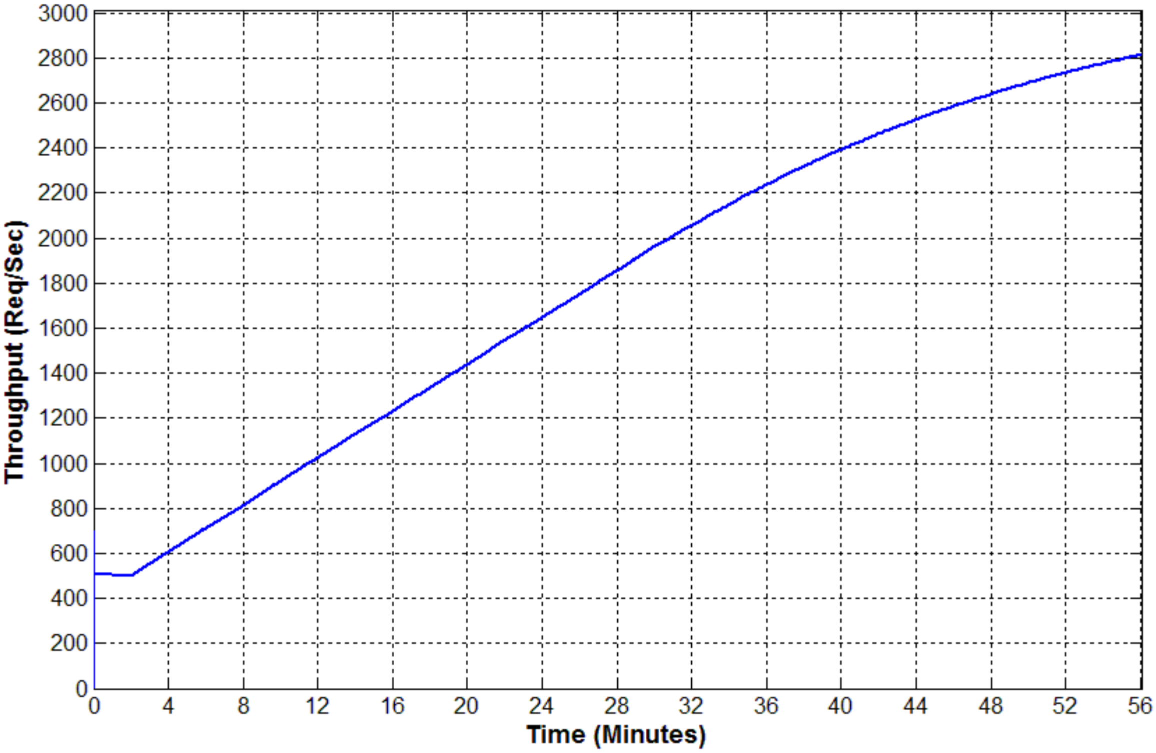


Figure 88 Throughput Evaluation for EDoS Attack Defense Shell Simulation at Rate of 3.2 KReq/sec in the Flash Overcrowd Mode

Vitae

Name : Ahmad Ibrahim Ahmad Shawahna

Nationality : Palestinian

Date of Birth : 5/6/1989

Email : ahmad.shawahna@gmail.com

Address : Jenin, Palestine

Academic Background : Computer Engineering

Educational Background : M.Sc. in Computer Engineering from King Fahd University of Petroleum and Minerals (KFUPM), Dhahran – KSA, 2013 – 2016, GPA: 3.5/4.0 (Very Good).

B.Sc. in Computer Engineering from An-Najah National University, Nablus – PS, 2007 – 2011, GPA: 80.3% (Very Good).

High School from Jenin Secondary School, Jenin – PS, 2006 – 2007, GPA: 95.7% (Excellent).

Certifications and Courses : CCNA, Cisco Academy, Hisham Hijjawi College, Nablus – PS.

Network+, Korean Palestinian IT Institute of Excellence, Nablus – PS.

CompTIA A+, Hisham Hijjawi College, Nablus – PS.

MCTS: Developing windows and web application using VS2010.Net and ASP.Net.

H2S Card, Hydrogen Sulfide certificate.

Experience to work for DOT ICT Company, Nablus – PS.

Experience : Mobile Applications Developer (Applications and Games) for Android and IOS platforms.

Excellent in developing Windows and Web applications using VS2010.Net and ASP.Net.

Programming and developing Commercial Software programs using C#.

Developing Web application using PHP, HTML, JAVA, JAVA script, and Windows Phone 7 Applications.

Design hardware-electronic and control circuits-devices using Micro Controller.

Computer, and Laptop Maintenance.

Training Work : Mobile applications developer at Art Technologies Company, Ramallah – PS.

Computer/Electronics engineer at AL-Ghanem Electronic Company, Jenin – PS.

Web application developing at Dot Learning Center, Nablus – PS.

Trainer at Intertek Consulting and Training Company, Khobar – KSA.

Sales employee at HP Company, Dammam – KSA.

Publications and Projects : EDoS Attack Defense Shell (EDoS-ADS): An Enhanced Mitigation Technique Against Economic Denial of Sustainability (EDoS) Attacks.

Efficient Energy Harvesting in Wireless Sensor Networks of Smart Grid.

Efficient RF Energy Harvesting in Wireless Sensor Network with Optimum Landmark Selection using Mobile Actuator.

Architecture and Implementation of Disease Outbreak Notification System.

Azkari, application for remembrances and supplications Prayers and Masbaha.