**International Journal of Cloud Computing and Services Science (IJ-CLOSER)**
Vol.3, No.5, October 2014, pp. 252~263
ISSN: 2089-3337

# Impact of DOS Attacks on a Private Cloud and the FAPA Solution

**Kazi Zunnurhain*, Susan Vrbsky*, Ragib Hasan****
* Department of Computer Science, University of Alabama
** Department of Computer and Information Sciences, University of Alabama at Birmingham

**Article Info**

***Article history:***

Received Jul 10th, 2014
Revised Aug 25th, 2014
Accepted Sep 26th, 2014

***Keyword:***

DDoS
Compromised
Bandwidth
Throughput
Transfer Packets

**ABSTRACT**

Cloud computing is becoming the leading IT computational technology due to the increasing acceptance of clouds as a viable solution for many different computing needs. Although it is being touted as a path for cost savings, not everyone in the computing community agrees with the advantages provided by cloud computing. While clouds can be highly productive and economical, cloud computing is vulnerable to different types of attacks, such as denial of service and distributed denial of service (DoS/DDoS) attacks. The focus of our research is to provide uninterrupted cloud services when the system is under a DoS threat from an external adversary. Several architectures have been suggested by different research groups to detect DDoS attacks. None of them considered the occurrence of virtualization in a cloud infrastructure, specially, the impact of DDoS on virtual networks and its side effects on co-resident virtual machines (siblings) or their neighbors. In this paper, we present results from experiments we conducted on our local private cloud instances to explore the amplification of external DoS attacks from a large number of botnets. All these botnets were hosted in the commercially used public cloud Amazon EC2. We measured the impact in terms of bandwidth and transfer packets for the victim machine as well as the siblings and neighbor instances. We then deployed a filtering mechanism named FAPA to successfully protect the victim machines from DoS attacks.

***Corresponding Author:***

Kazi Zunnurhain,
Department of Computer Science,
The University of Alabama,
SEC 3429, The University of Alabama, Box 870290, Tuscaloosa, AL 35487-0290.
Email: kzunnurhain@crimson.ua.edu

## 1. INTRODUCTION

The popularity of cloud computing is increasing rapidly due to its many offerings. Customers can receive computational services and resources on demand from a cloud, allowing access to be expressed as “Utility Computing". Clouds provide on-demand access to services ranging from hardware to platform to software services. According to Forbes magazine in Nov. 2012 [26], the growth of SaaS (Software as a Service) and cloud-based applications will result in a $33B global market by 2016. Forbes also predicts that in 2015, public and private clouds will be strong promoters of server growth, with a $9.4B global market. A cloud can satisfy customers' requirements, including ensuring both scalability and elasticity. Hence, cloud clients receive two advantages: (1) the avoidance of a large up-front investment, which is very attractive to small companies and startups, and (2) industries are not required to plan for their IT growth in advance with the “pay as you go" feature of a cloud system [2].

To achieve uninterrupted service from cloud computing, the most essential task for both public and private clouds is to detect malware activities and the efficient removal of these activities. According to a survey report [1], 74.6% of the service providers strongly recommend that the most vital issue for cloud computing is security assurance. In addition, according to Prolexic's Q3 2011 report (one of the first and largest companies offering DDoS mitigation) [3], 24% of all cyber-attacks were SYN flooding. The report also stated 22% of ICMP and 19% of UDP attacks were accomplished by cyber adversaries. TCP-SYN packets can clog the victim's bandwidth easily if the attacker exceeds the maximum bandwidth capacity of the network channel. The concept of SYN flooding is quite simple and straightforward. The hacker sends spoof packets with the victim's IP address as the destination and replaces its own IP address with a false source IP address, which is not easily traceable. Another report from Prolexic's [3] stated that HTTP flooding is responsible for 88.9% of DDoS attacks in Q2 2011. In our work, we focus on the SYN flooding type of DoS and explore its impact on cloud computing.

### 1.1 Motivation

- **Amplification of Attack**

The main properties of a cloud can, unfortunately, enhance a DoS attack and these properties serve to distinguish a DoS attack in a cloud from traditional DoS attacks as follows. Due to the load balancing feature in a cloud, if a single adversary sends spoof packets to a cloud server, once the server becomes overloaded it will offload its validating tasks to the nearest server [15] and [16]. The newly assigned server also eventually becomes overloaded and offloads its task to another server, thus propagating the flooding attack over the entire network. The cloud properties of elasticity and scalability provide further disadvantages during an attack. A server overloaded with enormous validation tasks will scale up to engage more of its resources and compute nodes to validate the spoof packets, continuously exhausting each server with its assigned resources. A dynamic adversary with knowledge of the targeted cloud topology and server connections can even try to compromise the complete system by using a botnet and deploy several handlers to compromise a cloud network [6].

- **Impact on Virtualization**

A cloud’s use of virtualization can also be exploited by an adversary. There exist many generic tools (nmap, hping, wget etc.) with which a cloud user can estimate the placement of virtual machines (VMs) in a cloud with a high probability [7]. Unfortunately, these tools can be used by the adversaries to impair the cloud system by launching various attacks. A generic network testing tool can successfully identify a target virtual machine on the cloud with higher likelihood and instantiate VMs co-resident to the target VM to conduct a variety of attacks [24]. There are many DDoS attack tools like Agobot, Mstream, and Trinoo [24] that can also be used against a cloud. For example, Agobot can be used as a backdoor Trojan and/or network worms by establishing an IRC channel to the remote servers. Then a client running user friendly social networking websites can be easily controlled by a single attacker to conduct highly effective and efficient DDoS attacks [7], [24]. Also a DoS attack not only compromises the target VM but costs a lot of network bandwidth and throughput for the co-resident virtual machines and neighbor instances. Obviously, in these examples of DoS attacks in the pay-as-you-go model of a cloud, users can be charged for services not requested. These observations of DoS vulnerabilities in a cloud environment motivated our work to design a strategy for DoS protection in a cloud system not only for the victim but also the coresident and neighbor virtual machines.

- **Contribution**

While currently there exist many different threats of DoS attacks, there are also different types of countermeasures that promise to protect flooding. The existing countermeasures are not cost effective in a cloud and may require an upgrade in the end systems. The objective of this research is to analyze the effects of a DoS attack from a distributed system (public cloud) to compromise local virtual instances (private cloud). Also, unique to this research, is the study of the effect of flooding from several external adversaries (bots launched from a public cloud) on sibling virtual machines and neighbor nodes, a topic which to the best of our knowledge has not been considered previously.

This paper is organized as follows. Ongoing related research in clouds for secure virtualization is presented in Section 2. In Section 3 we discuss the experimental setup we deployed for conducting the research from a public cloud (AWS EC2). This is followed by the illustration of our experimental results and analysis, and followed by a brief description of our filtering mechanism called FAPA in Section 4. This section also covers several experimental results to prove the efficiency of FAPA in terms of protection from DoS/DDoS attacks. We present conclusions in Section 5.

## 2. RELATED WORK

Clouds are vulnerable to various security issues, privacy [10], [12], data stealing, resource unavailability [11], denial of service attacks [8], [9] and many more. Virtualization can also be a victim of these types of threats in a cloud. Many approaches have been deployed or are under construction to overcome these threats. In this section we will discuss some of the approaches of cloud security focusing on secure virtualization and intrusion detection. We identify the weaknesses of each approach and discuss the feasibility of the solution for DoS or DDoS protection in a cloud environment.

One of the prominent solutions to provide security services for a cloud environment is CloudSec [20]. The major aspects of CloudSec are implementation of decentralized task scheduling, collaboration of server resources to defend attacks and utilization of a hypervisor to induce additional monitoring for virtual machines [20]. However, decentralization of tasks can cause additional cost for continuous communication among these collaborating organizations. Also in a heterogeneous cloud environment, implementation of collaboration for resource sharing requires extended Service Level Agreements (SLA) among the organizations for accountability purposes. A unique approach was proposed in Cloudsec for maintaining VM security based on using VM safe libraries on a VMware ESX cloud platform [21]. Rather than focusing on filtering attack packets, the approach was to minimize the semantic gaps between the hypervisor and the host operating system of the virtual machine. Due to the transformation of raw hardware bytes into an OS process, kernel information would require an extensive manipulation of OS global variables [21]. This kind of approach might risk the complete platform of the cloud environment due to complications. If there exists a bug to penetrate the system, then an attacker will be able to take control of the total system. In addition, the hypervisor, which allows multiple operating systems to run concurrently in a cloud, was considered completely trustworthy from the beginning. Also to be mentioned, hosting CloudSec inside the hypervisor [21] not only revokes the cloud client's transparency with the cloud provider but generates a potential dispute between them related to a security breach.

A multilayer overlay approach was introduced in [19] to defend against denial of service attacks based on IP addresses. Their principle strategy was to define a threshold for every layer of virtual hosts, and communication among them through encryption involving Message Authentication Control (MAC) [19]. However, extensive amounts of encryption (source) and decryption (destination) in a cloud environment would add additional charges to the clients, decreasing the economic benefits of a cloud. Also using thresholds instead of traffic flows might cause high false positive rates due to the dynamic nature of cloud environments. In our FAPA approach there is no encryption, reducing the cost of communication among VMs as much as possible. A Graphic Turing test [22] could be a suitable approach to defend DoS attacks for a high traffic web server to distinguish between human interaction and automated attack zombies. Using an alphanumeric image challenge might defend against a botnet [22], but a modern private cloud is also vulnerable from inside attacks which involve human interaction to a great extent.

Involvement by a hypervisor is an effective approach for secure virtualization in a cloud. In [14], the hypervisor-based architecture secured the guest OS and the running application from each other. However, their strategy prevented the creation of large numbers of VMs on a physical node, contradicting one of the primary features of cloud computing, which is its elastic nature [18], [19]. No protection or recovery plan was presented if an adversary compromises the hypervisor. Also, there was no performance evaluation for the proposed architecture [14]. To combat DDoS attacks, an ISP level solution [4] has managed to keep false positive rates as low as 2.8%. However, in their simulation they did not consider the scalable and elastic nature of the cloud which causes any DoS attack to be quickly amplified.

Virtualization is an important component of cloud computing. A virtual machine provides a guest operating system upon which the user’s software can run. CUDACS [13] considers the security of VMs by utilizing underutilized GPU cores to conduct VM monitoring efficiently. However, the extensive monitoring of guest VMs results in an overhead in memory access. The Lares architecture [25] for secure active monitoring uses virtualization that is deployable only on VMs running with Windows XP on Zen.

We have proposed a model (FAPA) [17], [18] to detect and and filter packets when DoS attacks occur. By considering different types of DoS attacks, our goal was to make the cloud more dynamic and adaptive. After conducting a number of experiments, we were able to identify some distinguishing features of traffic patterns in a cloud environment under the influence of flooding. These features were incorporated into our FAPA model to help protect against DoS attacks. We have implemented a prototype of our FAPA model. Results using this prototype have shown the ability to detect spoof packets and eliminate those packets. FAPA’s filtering of spoof packets in a progressive manner provides the cloud user an additional advantage. FAPA is employed on the client’s side, so a user is able to keep track of filtering and can reboot her desired instances after the completion of filtering. Also, disputes between providers and customers can be resolved with evidence of a traffic log. Also in another study [23] we measured the performance of FAPA with respect to false positive and false negative rates to investigate the adaptability and efficiency of FAPA being under attack from lots of bots.

FAPA can be deployed at different levels of the system, such as at the user's end. Since FAPA can run locally on top of the client's terminal, it is independent of the provider's cloud machine. There is no need to deploy any expensive packet capturing tools nor does it require any embedded digital signature inside the packets. There is no additional charge from the provider's end since the application runs in the customer's end. Because FAPA is employed on the client's side, customers have control over traffic trends, which is absent in other DoS prevention approaches. Moreover, automatic message propagation invokes the cloud server to trace the source or adversary.

## 3. METHODOLOGY

We want to study the effect of DoS attacks on a private cloud's virtual machines, the collocated VMs, and also the neighbors. In order to study the impact of DoS attacks, we perform experiments in which attackers from a commercial public cloud are deployed to attack nodes in a local private cloud. Figure 1 illustrates this botnet attack scenario. A *botnet* is a collection of internet-connected terminals whose security defenses have been breached and controlled by an unauthorized third party, e.g. an adversary. Figure 1 illustrates that the attacker from a commercial cloud has successfully taken control over all the connected nodes in a local private cloud environment.

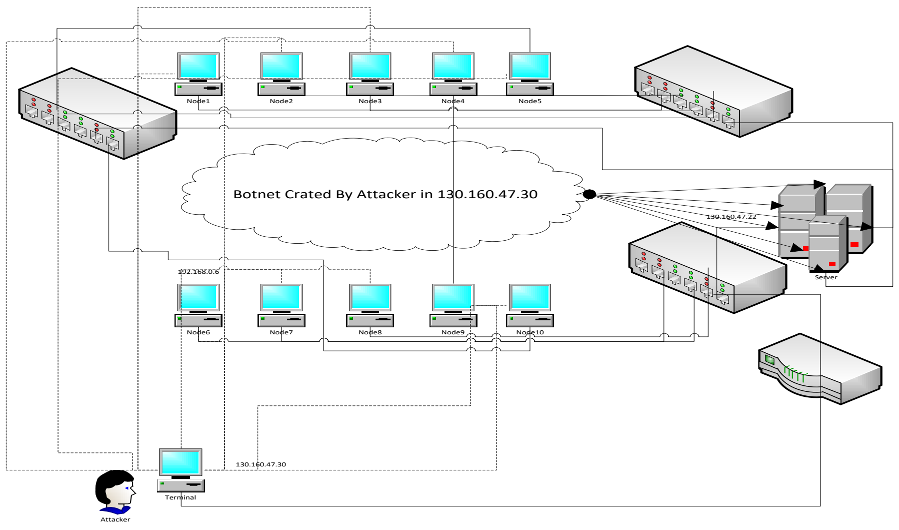

Figure 1. Botnet created by an attacker and continuous flooding from commercial cloud to local VM

We use Amazon Web Services (AWS) as the commercial cloud for our study. While there are various services provided by the Amazon Management Console, such as networking, database, storage, and app services, for this study we need light-weight compute nodes to be deployed as attackers. Amazon's EC2 (Elastic Compute Cloud) dashboard offers such type of instances. The nodes from the commercial cloud will direct a SYN packet attack towards a targeted machine in our local Eucalyptus cloud. Eucalyptus is a popular open source cloud platform that is compatible with Amazon's EC2 APIs and tools. EC2 dashboard provides various launch configurations, such as Linux, Red Hat, Ubuntu and Windows in both 32 and 64-bit platforms. As our local cloud was built on an Ubuntu platform, we chose Ubuntu. The type of instance is a micro instance, called t1.micro, and provides a small amount of consistent CPU resources. It has 1 virtual core and 2MiB, and provides 2.4 GB per second of sequential reading as well as 2.6 GB per second of sequential writing [5]. To create a new instance, a key pair is downloaded onto the user's local machine from the remote cloud. This key pair will confirm the secure SSH connection between the remote server in which the instance is running and the local virtual machine. We used the same key pair to launch local instances in our private cloud in order to establish the SSH connection with the remote Amazon instances. Flooding is then conducted from the Amazon instances towards the local VM instances.

We created several custom rules in the AWS instances for the communication between the AWS images and the local cloud virtual machines, which required changing some of the rules in the AWS default security groups. First, we changed the Custom TCP rule to allow TCP packets from all the available ports from 0 to 65568. We kept port 22 for SSH and deleted the Custom TCP rule that allows TCP through port 80 only. This was necessary because we use iperf which is a network testing tool that can generate TCP and UDP data streams. Iperf was used to send legitimate TCP packets from the client, which is a traffic measuring tool hosted in the local virtual machine of the private cloud. The server will listen only in port 5001 by default if the server listening is using iperf as well. If AWS instances are sent only through port 80 then the server will not listen to any packets. It allows the user to select her specific port number for a TCP connection. Thus, the server can listen to a specific port for TCP packets. Also keeping the other ports open widens the attack vector when TCP flooding is generated from the commercial cloud instances.

In the private cloud, the physical nodes establish SSH communication with the running local virtual instances using the same key pair that was downloaded for the AWS instances in the local terminal. VMbuilder allows launching an instance with a self-defining key pair, so before launching the local instances the images were stored in the libvirt directory provided by the VMbuilder. This directory is used as an image repository. The selected key pair was inserted into this image repository for each local VM with the batch file. When the batch files are triggered, SSH connection is established with the physical nodes on the private cloud via the key pair. Thus, the remote AWS images were able to send TCP-SYN packets to the local virtual machines. Throughout the rest of the study we will refer to the AWS instances as bots (handlers responsible for generating the attack in a botnet) and the local instances in the private cloud as compromised VMs.

## 4. EXPERIMENTAL RESULTS

We conducted several experiments considering the impact on network bandwidth and transfer packet rates of the virtual instances in order to determine the effect of an attack on the victim node and its co-located sibling nodes. We then deployed two industrially used stress-testing tools to compare the performance of FAPA in terms of mitigating spoof packets.

### 4.1 Impact on the Local VM Bandwidth after Deployment of EC2 Bots

For this experiment, we began with one bot on the EC2 commercial cloud, enabled with a TCP flooding program, in order to target a local VM. The experimental setup is illustrated in Figure 2. Custom TCP rules were declared and SSH connections were established both ways: one from the remote bot to the local terminal and another from the local terminal to the soon to be compromised private cloud VM.

After observing the bandwidth in the local VM once one bot was deployed, the number of bots was increased to observe the impact on bandwidth. Figure 2 illustrates the findings. IAM (Identity Access Management) are the bots deployed for generating attack traffic towards the victim machine and refer to the images running in the cloud.

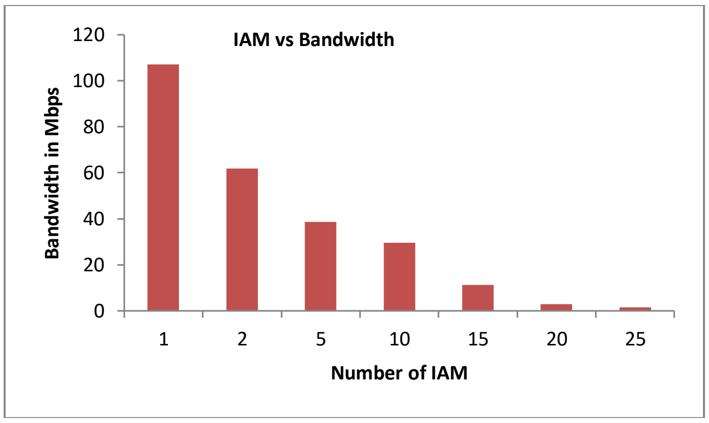

Figure 2. Bandwidth impact on the local VM

Figure 2 illustrates the bandwidth when SYN flooding begins. Once SYN flooding was conducted from 1 bot, the bandwidth of the compromised VM decreased about 45.1 Mbps by the time a second bot was ready to be deployed. We then continued to increase the number of bots. On average the bandwidth decrement was 46.9%, compromising half the current bandwidth as the number of bots increased. After deploying 20 bots there was a 74% bandwidth drop in the local VM. These results indicate that the decrease in bandwidth due to the attack also affects sibling VMs on the same node as the compromised VM. Also to be mentioned that without any bot, in the beginning the bandwidth was over 120 Mbps in the victim machine.

### 4.2 Impact on Local VM Transfer Packets after Deployment of EC2 Bots

In this experiment, the scenario and steps are similar to the previous experiment, but the network parameter for this study is the number of transfer packets. Figure 3 below illustrates the impact on the transfer packets when a VM is compromised.

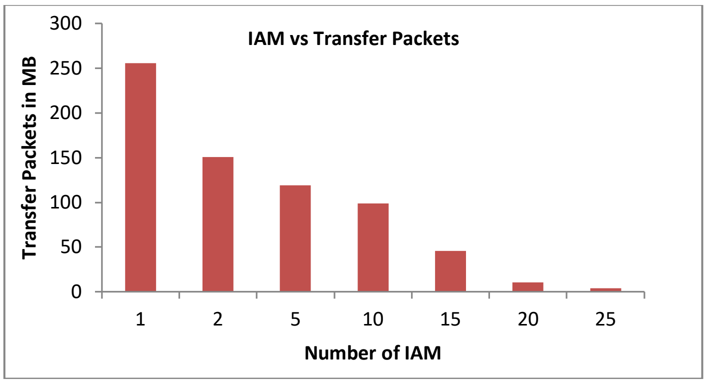

Figure 3. Transfer Packets impact on the local VM

We observed that after commencing the TCP flooding attack by 1 bot on the local instance, the transfer packets decreased by 105 MB, which is a 41.1% decrement. Again, we increased the number of bots in the botnet and as a result continued to compromise the transfer packets of the local VM. On average every attack decreased the transfer packets in the local VM about 45.5%. The deployment of 20 bots in the botnet resulted in an almost 77% decrement in transfer packets of the compromised VM. These results indicate that the decrease in transfer packets due to the attack also affects sibling VMs on the same node as the compromised VM. The transfer rates in the beginning were over 300 MB when no bot was deployed.

### 4.3 Bandwidth Fluctuation in the Local Cloud VM

For this experiment, an iperf server was installed in order to listen to all of the TCP packets that compromised the local instances in our Eucalyptus private cloud. As shown in sections 4.1 and 4.2, the local private cloud VM was forced to validate the attack packets from the bots, and eventually, the bandwidth could not serve the resources for the legitimate packets. Iperf was set to listen every 2 seconds, and provided a maximum and minimum bandwidth value available for each attack. Figure 4 below shows the results.

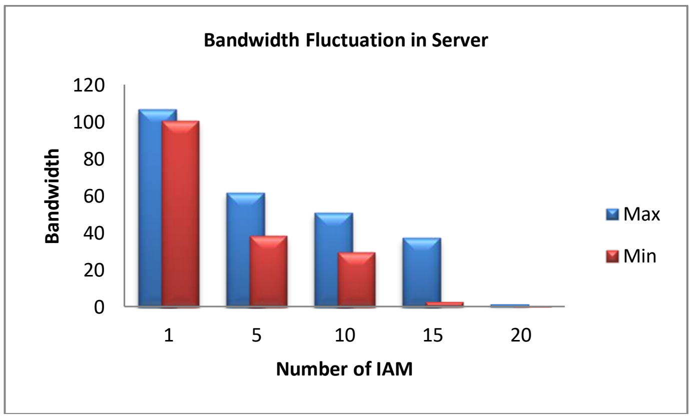

Figure 4. Bandwidth fluctuation on local VM

As the first bot was deployed, there was a difference of about 6 Mbps between the max and min bandwidth. After the deployment of an additional 4 bots, the difference increased to 32.3 Mbps. As the number of bots increased, the difference between the max and min increased rapidly. After the deployment of 15 bots, the bandwidth fluctuation difference increased to 92%. On average the difference was 54%, which is significantly different than when the first bot was deployed, which was only 6%.

### 4.4 Transfer Packet Fluctuation in the Local Cloud VM

In this experiment iperf was again used in the local VM and flooding was conducted through the remote bots. Instead of measuring the bandwidth, we measured the transfer packets sent to the local VM and calculated the percentage of fluctuation with respect to the number of remote bots as illustrated in Figure 5.

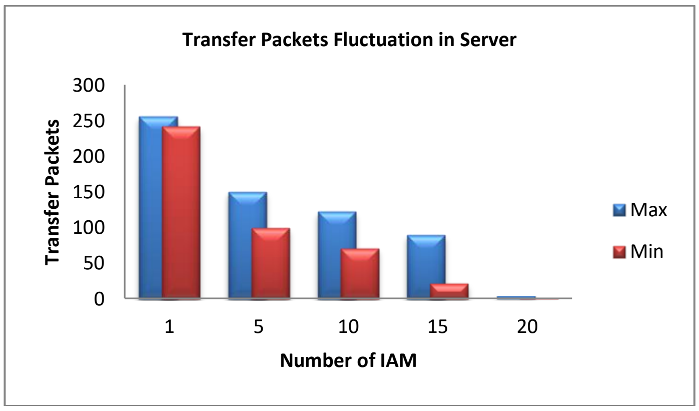

Figure 5. Transfer Packets fluctuation on local VM

With a single bot trying to flood the local VM, the difference between the maximum and the minimum number of transfer packets is about 14 MB. With 5 bots the difference increased dramatically to approximately 80 MB, which is 79.8%. The progression of bots in the botnet resulted in an increment in the maximum difference of 88.33% for 20 bots. On average the transfer packet fluctuation is 50.24%. Also notable, the increment in bots resulted in about a 40 MB increment in transfer packet fluctuation on average for the local VM.

### 4.5 Impact on Neighbor’s Bandwidth

Next, we wanted to study the impact of an attack on the neighbor VMs. To perform this experiment, we utilize two different physical nodes containing VM instances on the local private cloud, referred to as NC1 and NC2. NC1 has 3 VMs and NC2 has 4 VMs. We installed iperf on all the VMs on physical nodes NC1 and NC2 on the private cloud. The SYN flooding was conducted from botnets inside AWS and each of the bots targeted a VM existing in the second physical node NC2 of the private cloud. Thus, the iperf server was listening to legitimate TCP packets coming from the AWS micro instance (iperf client) as well as being flooded by non-legitimate TCP packets from the same micro instance (flooding program). We again increased the number of bots to observe the impact on the bandwidth of the neighbor VMs. While we refer to VMs on NC2 as sibling nodes of the compromised VM, VMs on NC1 are referred to as neighbor VMs. Figure 6 illustrates the impact on the bandwidth of the neighbor VMs in the local private cloud due to an attack from a botnet(s) inside the commercial cloud.

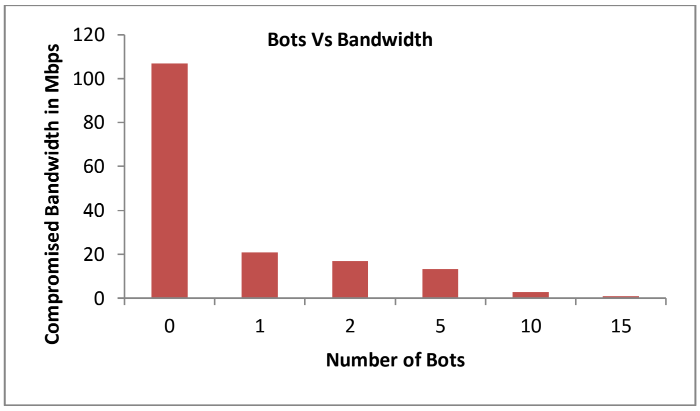

Figure 6. Impact on neighbors’ bandwidth

When the targeted VM was flooded from a single bot, almost 80% of the bandwidth of the neighbor VMs was immediately consumed. The bandwidth decreased from 107 Mbps to 21 Mbps after the first attack. We then increased the number of bots in order to observe the impact on the neighbor VMs. During the course of flooding with additional bots, the bandwidth of the neighbor VMs decreased about 21.2 Mbps. About 54%

of its bandwidth was compromised during the attack. Also, after the deployment of 15 bots the bandwidth decreased to only 651 Kbps, whereas the neighbor VMs had 107 Mbps of bandwidth before flooding. One more observation was that the bandwidth decrement was not as rapid/sharp after the deployment of the first bot. Hence, it can be inferred that most of the resources are consumed at the beginning of the attack. Unfortunately, this occurs when the system is completely unaware of any intrusions.

### 4.6 Impact on Neighbor's Transfer Packets

This experiment was conducted in the same manner as the previous experiment but we measured the impact on the neighbor machine transfer packets instead of measuring the bandwidth as illustrated in Figure 7. In this experiment we again increased the number of bots to observe the impact of flooding on the neighbor VMs in the private local cloud.

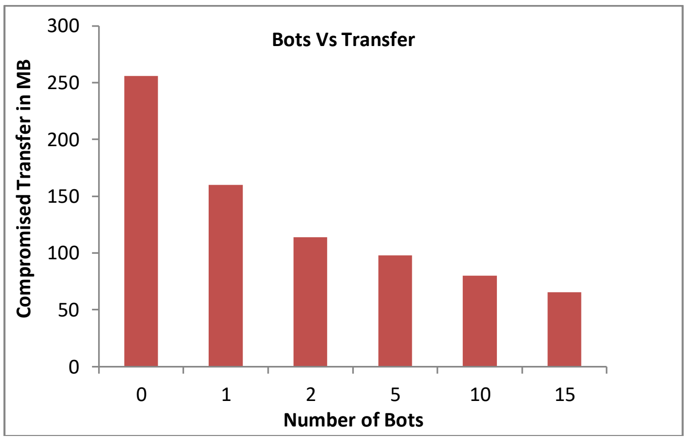

Figure 7. Impact on neighbors' transfer packets

Before flooding, the VMs were transferring 256 MB every 20 sec. Once the first bot was deployed and flooding took place, the transfer rate shrank to 160 MB, and the neighbor VMs lost about 96 MB of data. There was approximately a 38% decrement in transfers. As bots were added, the transfer rate continued to decrease, and on average there was about a 24% decrement in the number of transfer packets. Similar to the previous bandwidth experiment, the transfer decrement was not as rapid as during the first deployment of the bot. On average the transfer dropped about 24%, but the first bot alone caused about a 38% drop.

After observing all the above effects on the victim as well as neighbor virtual instances from an external DoS attack conducted by several bots, we introduced our FAPA filtering mechanism as a remedy for such a scenario. In the following section we describe the individual components of our FAPA architecture and results from experiments employing FAPA during a DoS attack.

### 4.7 FAPA

FAPA was designed by considering machine learning laws and considering the lower level networking needs required to analyze individual packets. FAPA is user centric as it is deployed in the user's terminal or virtual machine, and it is not proactive but reactive. FAPA captures raw packets and through analysis detects the behavioral pattern of traffic packets with respect to individual users in each session. These patterns invoke the server to propagate an announcement to the user and filter the packets if an intrusion is detected. In FAPA [17], separate modules have individual functionalities. The modules in FAPA are as follows: (1) Traffic Unpacking, (2) Feature Selection, (3) Comparison Checking, (4) Validity Checking, and (5) Profile Generator. FAPA begins by fetching each incoming network packet from a domain in a packet-by-packet manner. The network packet is unwrapped in the Traffic Unpacking module. In the Feature Selection module, pertinent header information is recorded and used for generating profile in Profile Generator module. These profiles identify the uniqueness of every running instance in the cloud. If spoofing takes place, then these unique features change. FAPA also utilizes a second capturing style, in which the throughput of certain packet types is recorded (a handler is defined to select packets either one by one or in a loop fashion for a bulk amount). The Comparison Checking and Validity Checking modules monitor the throughput and compare it with the previously recorded parameters. Any altering of certain bits in the packets or change in traffic behavior will generate an alarm. If any change in pattern is detected, then packets arriving from those nodes are filtered. The filtering process consists of calculating the spoof traffic, miscellaneous traffic and original traffic. If a legitimate user is compromised then the attacker will be traced back and packets coming from the actuator will be filtered. Regular services in the cloud will not be affected. In the next section we describe experiments to study the effectiveness of FAPA in filtering spoof packets during a DoS flooding attack on VM instances, and their sibling and neighbor nodes.

### 4.7.1 DoS Impact on Speed

In Sections 4.1-4.6 we observed the impact of a DoS flooding attack that we generated from the virtual instances of a commercial cloud, with respect to bandwidth and transfer packets. In this section we deploy FAPA filtering to observe the improvement in network parameters during flooding. We utilize several popular DoS attack tools to study their impact on local VM instances with respect to TX and RX rate, the amount of transmitted and received packets, respectively. Since FAPA is a reactive approach, after the first deployment of a bot, we anticipate FAPA will respond immediately with the first intrusion. The DoS attack tools we used are LOIC (Low Orbit Ion Cannon) and XOIC (Extreme Orbit Ion Cannon). LOIC is an open source DoS attack tool that can be used to check server vulnerability. An admin can easily conduct a stress test on the server by providing the IP address, specific packet type for flooding generation, port number and also a timer. LOIC can also take a URL as its parameter, by which one can compromise a web server through HTTP flooding. XOIC is a new DoS attack tool that only requires an IP address and port number to launch an attack, but it can be used to send messages to a target machine as well. Both of these DoS attack applications were used on the local virtual machines in the private cloud to check the vulnerability of our local cloud. We then deployed FAPA filtering to resolve the flooding issues.

In this experiment, we observed the impact of a DoS attack with respect to the RX and TX speeds of a virtual machine and results are shown in Figure 8. The virtual machine was attending to a legitimate workload requested from another VM. Statistical results for the five scenarios listed below include the results from FAPA filtering.

1. Measuring the traffic scenario without any workload on a VM
2. Creating a workload by deploying iperf to send legitimate packets from a VM to the target VM
3. Injecting a DoS attack from a Windows7 terminal with LOIC
4. Injecting a DoS attack from a Windows7 terminal with XOIC
5. Deploying FAPA filtering to intervene in the DoS attacks.

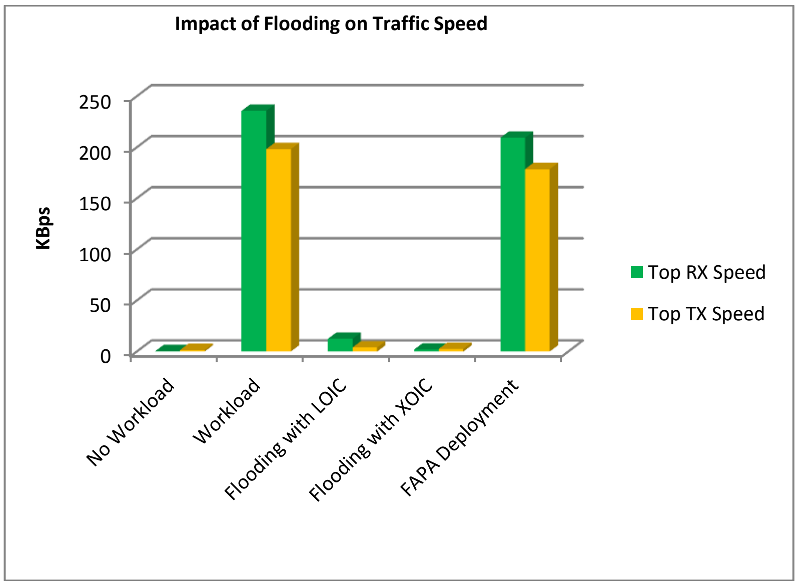

Figure 8. DoS impact on traffic speed

*Analysis-*

1. In the initial phase (No Workload) the TX and RX top speeds were very low, 1.43 and 0.42 KBps, respectively. This is expected, because without any workload or any data processing the VM was idle.
2. Once the workload was sent by iperf in 2 second intervals from another VM to the target VM, the RX and TX top speeds increased to 235 and 198 KBps, respectively.

3. LOIC was deployed to conduct TCP flooding on the target VM for 5 minutes. The RX speed dropped dramatically by 95% and TX by 98%. The dramatic nature of the decrease was surprising.
4. Then LOIC was terminated and a XOIC attack was deployed. The RX speed dropped dramatically by 97% and TX by 98%. The impact of XOIC was a bit more than the LOIC attack.
5. Finally, we deployed FAPA and recovered about 74% of RX speed and 81% of TX speed, meaning 26% of RX speed and 19% of TX speed remained compromised during this operation.

### 4.7.2 DoS Impact on Amount of Packets

In this experiment, we measured the amount of packets received and transmitted as we observed the impact of a DoS attack. Similar to the previous experiment, all of the five circumstances were considered and an analysis is provided below in Figure 9.

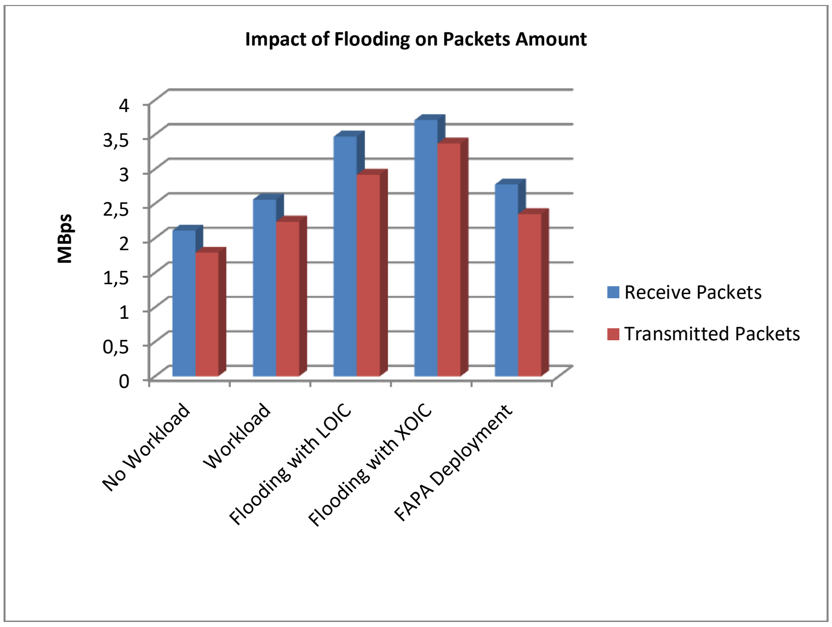

Figure 9. Packets Amount

*Analysis-*

1. Without any workload the VM was receiving about 2.11 MBps and transmitting about 1.8 MBps. This was computed as the average over 10 readings without any workload. We assumed this to be the usual cloud traffic due to message passing among the running VMs.
2. After the workload was injected from iperf, the receive amount increased about 21% and the transmitted packets increased by 25%. This is reasonable because iperf was deploying packets in KBps in a 2 second interval.
3. LOIC was deployed to commence TCP flooding. The receive packets increased by 65% and transmitted packets were increased by 63%.
4. The LOIC attack was terminated and we began the XOIC attack on the VM. The flooding attack reduced the receive packets by 75% and transmitted packets by 88%.
5. FAPA filtering recovered about 91% of the receive packets and 95% of the transmitted packets. Overall, there was a complete loss of 5% of transmitted packets and 8% of receive packets.

Hypothetically, if FAPA is deployed in all the running VMs of the cloud network, then amplification of a DoS attack will be curtailed by three-fourths. Hence, it can be stated that FAPA is efficient enough to recover 75% of the transmission and receive speed measured in kbps (Figure 8). Also, removing most of the spoofed packets generated due to the DoS attacks from LOIC and XOIC, recovered about 90% of network packets (Figure 9).

## 5. CONCLUSION

Detection against TCP flooding from only one VM is not enough for modern clouds, because an adversary can flood the system from TCP-SYN packets through a botnet. An adversary can also penetrate the system through UDP packets, HTTP packets, and fake ICMP echo requests. Through experiments in which botnets from a commercial cloud attacked our private cloud, we were able to study the impact of TCP flooding on a compromised virtual machine as well as its siblings and neighbors. We were also able to acknowledge the amplification of the DoS impact on private cloud instances with respect to colocated and remote VMs. Additional experiments were conducted to determine the performance of our FAPA model to protect against DoS attacks in a cloud. Results have demonstrated the ability of our proposed FAPA model to detect and filter packets to eliminate a DoS attack VM instances, and their sibling and neighbor nodes. The performance analysis of FAPA confirmed that the DoS impact on the victim as well as its neighbors was eliminated successfully.

FAPA could be beneficial to cloud users in terms of increasing the adaptability of cloud use and deducting additional costs. Cloud providers would not need to buy expensive tools for the elimination of DoS or DDoS, and can invest more money on providing services for the cloud customers. It is even possible for FAPA to be deployed in the border routers of the local domains. This would allow FAPA to become even more economical and effective than deploying it in all the virtual machines. Customer satisfaction as well as business goals can be achieved through transparent traffic logging and secure services of the private cloud as provided by FAPA.

## REFERENCES

[1] Bremler-Barr, Anat, and Hanoch Levy. "Spoofing prevention method." In *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, vol. 1, pp. 536-547. IEEE, 2005.

[2] Mühlbauer, Wolfgang, Anja Feldmann, Olaf Maennel, Matthew Roughan, and Steve Uhlig. "Building an AS-topology model that captures route diversity." In *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 4, pp. 195-206. ACM, 2006.

[3] Cheswick, William R., Steven M. Bellovin, and Aviel D. Rubin. *Firewalls and Internet security: repelling the wily hacker*. Addison-Wesley Longman Publishing Co., Inc., dpm.

[4] Gupta, B. B., Ramesh C. Joshi, and Manoj Misra. "Dynamic and auto responsive solution for distributed denial-of-service attacks detection in ISP network." *arXiv preprint arXiv:1204.5592* (2012).

[5] Amazon, E. C. "Amazon elastic compute cloud (Amazon EC2)." *Amazon Elastic Compute Cloud (Amazon EC2)* (2010).

[6] Litty, Lionel, and David Lie. "Manitou: a layer-below approach to fighting malware." In *Proceedings of the 1st workshop on Architectural and system support for improving software dependability*, pp. 6-11. ACM, 2006.

[7] Bedi, Harkeerat Singh, and Sajjan Shiva. "Securing cloud infrastructure against co-resident DoS attacks using game theoretic defense mechanisms." In*Proceedings of the International Conference on Advances in Computing, Communications and Informatics*, pp. 463-469. ACM, 2012.

[8] Zunnurhain, Kazi, and S. Vrbsky. "Security attacks and solutions in clouds." Poster In *Proceedings of the 1st international conference on cloud computing*. 2010.

[9] Zunnurhain, Kazi, and Susan V. Vrbsky. "Security in cloud computing." In *Proceedings of the 2011 International Conference on Security & Management*. July 2011.

[10] Hwang, Sameer Kulkareni, and Yue Hu. "Cloud security with virtualized defense and reputation-based trust mangement." In *Dependable, Autonomic and Secure Computing, 2009. DASC'09. Eighth IEEE International Conference on*, pp. 717-722. IEEE, 2009.

[11] Sabahi, Farzad. "Virtualization-level security in cloud computing." In*Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on*, pp. 250-254. IEEE, 2011.

[12] Schwarzkopf, Roland, Matthias Schmidt, Christian Strack, Simon Martin, and Bernd Freisleben. "Increasing virtual machine security in cloud environments."*Journal of Cloud Computing* 1, no. 1 (2012): 1-12.

[13] Lombardi, Flavio, and Roberto Di Pietro. "CUDACS: securing the cloud with CUDA-enabled secure virtualization." In *Information and Communications Security*, pp. 92-106. Springer Berlin Heidelberg, 2010.

[14] Sabahi, Farzad. "Secure Virtualization for Cloud Environment Using Hypervisor-based Technology." *Int. Journal of Machine Learning and Computing* 2, no. 1 (2012).

[15] Lombardi, Flavio, and Roberto Di Pietro. "Secure virtualization for cloud computing." *Journal of Network and Computer Applications* 34, no. 4 (2011): 1113-1122.

[16] Younis, MM Younis A., and K. Kifayat. "Secure cloud computing for critical infrastructure: A survey." *Liverpool John Moores University, United Kingdom, Tech. Rep* (2013).

[17] Zunnurhain, Kazi. "FAPA: a model to prevent flooding attacks in clouds." In *Proceedings of the 50th Annual Southeast Regional Conference*, pp. 395-396. ACM, 2012.

[18] Zunnurhain, Kazi, S. Vrbsky and Ragib Hasan, "FAPA: Flooding Attack Protection Architecture in Cloud System," *International Journal of Cloud Computing, Inderscience Publishers.* 2014. (Accepted not yet published)

[19] Geneiatakis, Dimitris, Georgios Portokalidis, and Angelos D. Keromytis. "A multilayer overlay network architecture for enhancing IP services availability against dos." In *Information Systems Security*, pp. 322-336. Springer Berlin Heidelberg, 2011.

[20] Ibrahim, Amani S., James Hamlyn-Harris, John Grundy, and Mohamed Almorsy. "Cloudsec: a security monitoring appliance for virtual machines in the iaas cloud model." In *Network and System Security (NSS), 2011 5th International Conference on*, pp. 113-120. IEEE, 2011.
[21] Xu, Jia, Jia Yan, Liang He, Purui Su, and Dengguo Feng. "CloudSEC: A Cloud Architecture for Composing Collaborative Security Services." In *Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on*, pp. 703-711. IEEE, 2010.
[22] Morein, William G., Angelos Stavrou, Debra L. Cook, Angelos D. Keromytis, Vishal Misra, and Dan 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*, pp. 8-19. ACM, 2003.
[23] Zunnurhain, Kazi, and S. Vrbsky, "FAPA: Performance Analysis with Victim and Sibling Virtual Machines," In *Proceedings of the 52nd annual Southeast regional conference*. ACM, 2014.
[24] Ristenpart, Thomas, Eran Tromer, Hovav Shacham, and Stefan Savage. "Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds." In *Proceedings of the 16th ACM conference on Computer and communications security*, pp. 199-212. ACM, 2009.
[25] Payne, Bryan D., Martim Carbone, Monirul Sharif, and Wenke Lee. "Lares: An architecture for secure active monitoring using virtualization." In *Security and Privacy, 2008. SP 2008. IEEE Symposium on*, pp. 233-247. IEEE, 2008.
[26] Columbus, Louis, "Cloud Computing and Enterprise Software Forecast Update, 2012." In *Forbes Mgazine*, November 8, 2012.

## BIBLIOGRAPHY OF AUTHORS

Kazi Zunnurhain received his B.Sc. Degree in Computer Science from Bangladesh University of Engineering and Technology, Dhaka, Bangladesh, in 2006. He received his M.Sc. degree in Computer Science from University of Alabama, Tuscaloosa, AL, in 2011. Now he is enrolled as a PhD student in Department of Computer Science at the University of Alabama. His research interest is security issues in cloud computing.

Susan V. Vrbsky is an Associate Professor of Computer Science at The University of Alabama. She received her Ph.D. in Computer Science from The University of Illinois, Urbana-Champaign. She received an M.S. in Computer Science from Southern Illinois University, Carbondale, IL and a B. A. from Northwestern University in Evanston, IL. She is the Advisor of the Cloud and Cluster Computing Lab. Her research interests include database systems, data management in clouds, green computing, data intensive computing and database security.

Ragib Hasan is an Assistant Professor of Computer and Information Sciences at the University of Alabama, Birmingham. He received his Ph.D. in Computer Science in October 2009 from University of Illinois, Urbana-Champaign. He is leading the SECuRE and Trustworthy Computing Lab (SECRETLab). He is also the founder of Shikkhok.com- an award winning free education platform in Bengali language and the coordinator of the Bangla Braille project, aimed at creating educational material for visually impaired children in South Asia. His research is funded by grants from the Department of Homeland Security, the office of Naval Research, a 2012 Google Faculty Research Award, and a 2012 Amazon Research Grant. His research interests include cloud security, secure location provenance, secure data provenance, data waste management.