applied system innovation

Article

Securing Infrastructure-as-a-Service Public Clouds Using Security Onion

Abdullahi Mikail and Bernardi Pranggono *

Department of Engineering and Mathematics, Sheffield Hallam University, Howard Street, Sheffield S1 1WB, UK; abdullahi.i.mikail@student.shu.ac.uk

* Correspondence: b.pranggono@shu.ac.uk

Received: 17 December 2018; Accepted: 23 January 2019; Published: 30 January 2019

Abstract: The shift to Cloud computing has brought with it its specific security challenges concerning the loss of control, trust and multi-tenancy especially in Infrastructure-as-a-Service (IaaS) Cloud model. This article focuses on the design and development of an intrusion detection system (IDS) that can handle security challenges in IaaS Cloud model using an open source IDS. We have implemented a proof-of-concept prototype on the most deployed hypervisor—VMware ESXi—and performed various real-world cyber-attacks, such as port scanning and denial of service (DoS) attacks to validate the practicality and effectiveness of our proposed IDS architecture. Based on our experimental results we found that our Security Onion-based IDS can provide the required protection in a reasonable and effective manner.

Keywords: cloud computing; intrusion detection system; public clouds

1. Introduction

Cloud computing is a model for delivering on-demand computing services over the Internet [1]. It allows individuals and businesses to run applications on shared data centers instead of running them in private data centers. These shared data centers can be accessed by connecting to the Cloud network, making the application process starts quicker and more cost effective.

Although Cloud computing is providing numerous benefits, there are some barriers hindering its complete adoption by most organizations. The main concern of these organizations is the security and privacy of their data and information [2–5]. Cloud computing is a combination of various other technologies: Virtualization, service-oriented architecture (SOA), and Web 2.0 [6]. As well as the benefits of these underlying technologies, their security limitations are also inherited [7].

The framework of Cloud computing supports distributed service-oriented models, multi-domain and multi-user administrative infrastructure; as a result, it is very prone to various security threats and vulnerabilities. Because Cloud computing is based on a distributive architecture and is a relatively new computing paradigm, there is a reservation surrounding how security at all level (i.e., network, host, application and data levels) can be achieved [5,7]. This reservation has led company executives to remark that security of information is the biggest concern with Cloud computing [8].

A standard security mechanism to protect a Cloud computing environment is by the use of intrusion detection system (IDS). An IDS monitors enterprise computing and network environment and gives alerts on suspicious activities [9]. Traditionally, most IDS are host-based (HIDS) where IDS closely monitor specific host machines in a network. Another type of IDS is the network-based (NIDS) where IDS identifies an intrusion on main network nodes. While the use of an IDS does not guarantee and cannot be considered an absolute defense against intruders, it can play a major role in the security architecture of Cloud computing.

In this article, we propose and evaluate an approach to secure customer assets (data and information) in a Cloud computing environment by implementing an open-source IDS. In particular, we study how an IDS can be positioned in an Infrastructure as a Service (IaaS) public Cloud to monitor inter-virtual machine (inter-VM) operations. The IaaS public Cloud is the most widely used Cloud model deployment. In a public Cloud, the cloud service provider (CSP) provides different services and facilities to the general public via the Internet. In an IaaS public Cloud, the Cloud user is able to deploy and run any software and operating system (OS) in the Cloud. The user is also able to deploy IDS to add a layer of security to protect its data and information. Security and privacy in public IaaS model are especially a concern due to different users can have their virtual machines (VMs) reside on the same physical machine. We also design and implement a proof-of-concept of the proposed architecture. IaaS public Cloud technologies provide networking mechanisms that are used to configure virtual switches and experiment with new functionalities, such as traffic forwarding. Consequently, we want to assess how different virtual networking techniques can be employed in the context of securing the Cloud environment.

The remainder of the article is organized as follows. Section 2 provides background information on Cloud computing and Cloud security. Section 3 discusses the related work. In Section 4, we discuss the method implemented in securing Cloud computing with emphasis on IaaS public cloud. In Section 5, the results from our experiments are discussed and evaluated in line with real-world scenarios. In Section 6, we offer our depth analysis on the experiment results. Finally, a conclusion is highlighted in Section 7.

2. Background

2.1. Virtualization

The underlying technology of Cloud computing is virtualization [10]. The initial idea behind the creation of virtualization by IBM Corporation in the 1960s was to partition large mainframe computers into several logical instances and to run them on one physical mainframe hardware (host). In recent years, virtualization has attracted much attention in enterprise computing and is in fact now proven to be a fundamental building block in today's computing. Advancements in hardware capabilities that provide very powerful multicore and high-performance computers, has brought about a potential of utilizing the hardware capabilities to the fullest. Virtualization has made it possible to run several VMs simultaneously on one physical server. Here, each VM runs in parallel to another VM. In addition, each VM has a full software stack, including an operating system (OS) that runs in parallel to other VMs, however each VM acts as a separated physical machine. Virtualization has become a core concept in modern data centers, mainly driven by the benefits of application isolation, resource sharing, fault-tolerance, portability and cost efficiency [11].

2.2. Cloud Computing

A Cloud computing architecture is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., servers, network, applications, and services) that can be readily provisioned and released with minimal management effort or interaction from service providers [12]. Cloud computing model follows a pay-per-use model where users only pay for the services they have used. Furthermore, the Cloud computing model implements virtualization techniques to provide resources to end-users efficiently. Hence, along with the dynamicity and flexibility provided by virtualization, Cloud computing is also characterized by its manageability, scalability, and availability.

According to the National Institute of Standards and Technology (NIST), Cloud computing has four deployment models: Public, private, hybrid and community [12]. Furthermore, typically there are three delivery models: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS) (see Figure 1).

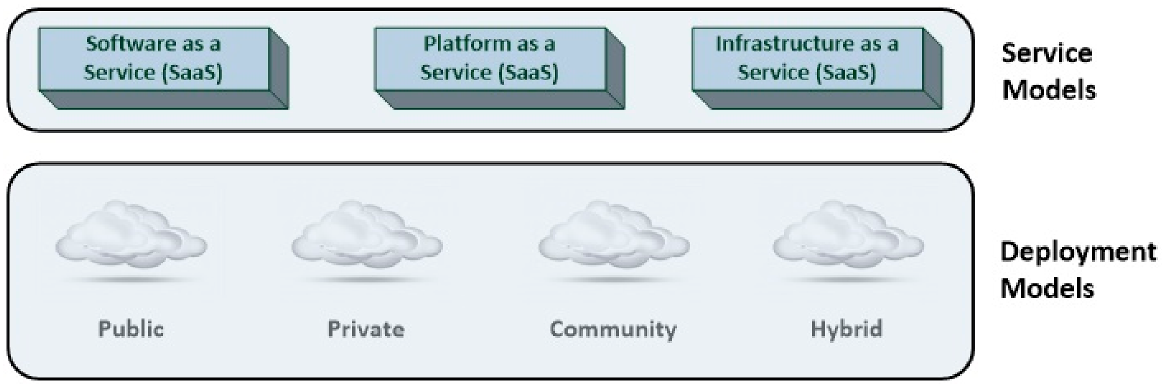

Software as a Service (SaaS)
Platform as a Service (SaaS)
Infrastructure as a Service (SaaS)
Service Models
Public
Private
Community
Hybrid
Deployment Models

Figure 1. Cloud Computing Deployment and Service Models.

2.2.1. Cloud Service Delivery Models

Software as a Service (SaaS): Is a group of remote computing services accessed through the Internet. It allows applications/software to be deployed by third-party vendors.

Platform as a Service (PaaS): Is in the middle of the service models. IaaS delivers services by providing managed platforms, which provides development tools, architecture, framework, programs and Integrated Development Environment (IDE) for the Cloud users.

Infrastructure as a Service (IaaS): Deals with computer hardware (i.e., storage, network, processor, memory, virtual servers/machines, etc.) as a service and is at the bottom of the models. In an IaaS model, Cloud users take responsibility for managing their VMs by installing a preferred OS and other software. In an IaaS Cloud model, the Cloud user is responsible for the security and management of recourses, such as storage, network, processor, memory and VMs.

Anything as a Service (XaaS): This type of service delivery model refers to a collective term that combines numerous things as 'X' as a service, where X can be anything or everything as a service. This service then becomes interchangeable in Cloud architecture. Here the Cloud systems have the ability to support substantial resources into specific personal and smaller requirements.

2.2.2. Cloud Service Deployment Models

The deployment model of a Cloud tells the nature and function of a Cloud. The four types of deployment models defined by NIST are [12]:

- **Private Cloud:** This model has to do with the operations and management of the data center of an organization (i.e., private Cloud). The Cloud infrastructure is operated solely for an organization.
- **Public Cloud:** This type of Cloud deployment model gives a true representation of Cloud hosting in which they have a strong Service Level Agreement (SLA) between the CSP and the consumer. The CSP provides different services and facilities to the general public. Multiple entities, e.g., businesses, academics or governmental organizations can own and control a public Cloud environment. This creates problems relating to where resources are located or who owns them thereby increasing the difficulty of protecting them from attacks [13].
- **Community Cloud:** This model is a collaborative effort in which resources are shared between different organizations from a specific community with similar concerns (e.g., security, jurisdiction, compliance) irrespective of where they are managed (be it internally or by a third-party) or hosted (internally or externally).
- **Hybrid Cloud:** This type of Cloud deployment model is a combination of two or more Cloud models (i.e., public, private, and community). Typically, application and data are bound together by standard and proprietary technology. Hybrid Clouds offer the advantages of other Cloud deployment models. However, it is more organized and more secure than a public Cloud.

2.2.3. Cloud Computing: The Security Viewpoint

The advent of Cloud computing has brought with it many benefits for both Cloud users and providers. Majority of IT users today use one or more public Cloud services, such as e-mail and storage. Nonetheless, many organizations and businesses despite realizing the potential for profits made available by public Clouds are still hesitant to use Cloud services due to its security vulnerability [14]. Cloud computing platforms are more vulnerable to cyber-attacks than other computing platforms due to the diverse nature of its service delivery models. These vulnerabilities could be uncovered through any of the fundamental constituents of Cloud computing i.e., network, VM, application and storage. The next subsection classifies attacks based on the fundamental elements of a Cloud network and their effects.

2.2.4. Classification of Attacks

- **Network Based Attacks:** One entry point for attackers into a Cloud system can be through the network that leads to weakening the quality of Cloud services and further putting Cloud confidentiality and privacy at risks. The most common attacks on network infrastructure in the Cloud are port scanning, Botnets, man-in-the-middle (MITM) [15], distributed denial of service (DDoS) [16], and spoofing attacks.
- **Virtual Machine Based Attack:** Several security risks occur as a result of many VM instances being hosted on a single system. The major characteristic feature of VM based attacks is the violation of data and hampering of cloud services as a result of flaws in VMs being manipulated.
- **Application Based Attack:** Applications running on Cloud platforms are typically delivered through a web browser. However, flaws in these web applications make the Cloud vulnerable to attacks [17]. Examples of application-based attacks are injections of codes into applications that run on Cloud platforms and attacks on protocols that are implemented to provide services in the Cloud.
- **Storage Based Attacks:** The Cloud storage is a high-value target for attackers because data from various users and business organizations are kept together in a Cloud environment [18]. Hence, breaching into the Cloud storage will potentially give an attacker access to a large pool of sensitive information. Examples of storage-based attacks are data scavenging and data recovery attacks.

2.2.5. Effects of Attacks on Cloud Resources

Attacks on Cloud platforms may have one or more implications that further affect the delivery of Cloud services. This section discusses and analyses these implications.

- **Account or Service Hijacking:** This type of attack can happen as a result of different aspects for example, social engineering of weak passwords/credentials. When an attacker gains access to these credentials they can carry out a variety of malicious acts, such as gaining access to sensitive data, manipulation of data and even redirecting transactions [19].
- **Malicious Manipulation of Customer Data:** The communication between the Cloud and the users is usually via the Internet through web browsers. This attack is usually carried out through the manipulation of data that are sent from the application component to the servers. Examples of these attacks are cross-site scripting (XSS), SQL injection (SQLi), direct object references, etc.
- **Denial of Service:** A malicious user may target a Cloud platform by occupying all possible resources [16]. Thereby making the system unable to satisfy other genuine users due to resources being unavailable.
- **Malicious VM Creation:** Here an attacker who creates an authentic account can create a VM image containing malicious codes, such as worms, viruses and Trojan horses, and store it in the CSP's repository [20].
- **Insecure VM Migration:** During the live migration of VMs logs of a VM's execution state that is being upheld due to a rollback implementation. These logs can become accessible and this

can give room for attackers to access data illegally, transfer a VM to an untrusted host and cause disruption or DoS as a result of creating and migrating several VMs at the same time [21,22].

- **Sniffing/Spoofing of Virtual Networks:** An attacker's VM can listen to network or even use ARP spoofing to re-address packets to and from other VMs [23].

2.2.6. Automated Cloud Security Using Intrusion Detection/Prevention

The continuous evolvement of Cloud computing architecture means that traditional firewalls are unable to secure the Cloud effectively. In disparity to traditional firewalls which secures a network by probing network packets passing through network ports, IDS monitors and analyses the headers and payloads of network packets and compares them with normal/benign traffic to detect any vulnerability exploits against a target application or system. IDS primarily try to identify patterns in known attacks and alerts the security administrator. Equally, intrusion prevention systems (IPS) is similar to IDS, though it also has the capability to reject malicious traffic and to terminate a suspicious connection.

As we know that the backbone of Cloud computing is a high-speed network, therefore this calls for the use of an automated or faster IDS/IPS. IDSs have a variety of flavors and approaches used to detect suspicious traffic (see Figure 2). A network-based intrusion detection/prevention system (NIDS/NIPS) attempts to secure all devices on a network by analyzing network traffic for signs of malicious behaviors, while a host-based intrusion detection/prevention system (HIDS/HIPS) attempts to secure one particular host [9]. HIDS/HIPS is particularly used to secure valuable or private information on a specific server. As well as HIDS/HIPS and NIDS/NIPS, hypervisor-based IDS provide another type IDS used to ensure the security of data and information in the Cloud. This type of IDS/IPS runs inside the hypervisor, making it possible to monitor and analyze communication between VMs, hypervisors, and the virtual network.

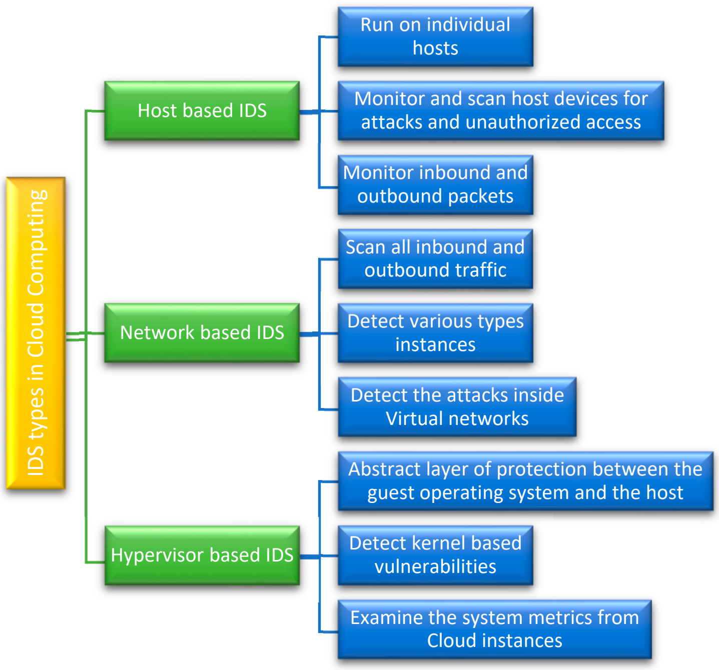

IDS types in Cloud Computing
Host based IDS
Run on individual hosts
Monitor and scan host devices for attacks and unauthorized access
Monitor inbound and outbound packets
Network based IDS
Scan all inbound and outbound traffic
Detect various types instances
Detect the attacks inside Virtual networks
Hypervisor based IDS
Abstract layer of protection between the guest operating system and the host
Detect kernel based vulnerabilities
Examine the system metrics from Cloud instances

Figure 2. Types of intrusion detection system (IDS) in Cloud Computing.

Typically, IDS have two types of detection approaches: Anomaly detection and signature detection [24]. The anomaly detection approach maintains a pool of information regarding user profile, hosts, application and connection (ports, devices, and protocols) specified by a network administrator. Anomalies are detected by comparing current activity with the rules defined by the administrator. In disparity, the signature detection approach compares signatures of activities on the system against an array of malicious signatures or threat patterns.

3. Related Work

Cloud-based intrusion detection was made possible as a result of researchers combining established techniques of malware detection with the ability to isolate VMs. Nonetheless, the IDS still carry out its duty of identifying changing patterns in the behavior of a system that might have been compromised; for example Bakshi et al., [25], proposed an approach to prevent DDoS attack in a Cloud environment. The authors implemented Snort IDS installed in a VM on VMware ESX, which sniffs all traffic either inbound or outbound over the virtual network through a virtual switch. To detect attacks, logged packets are analyzed by Snort in real time. Although this approach showed that IDS can effectively function in a Cloud computing environment, the implementation is still vulnerable to zero-day attacks as non-virtualized IDS inability to detect new attacks.

The advent of hypervisors and hypervisor-based IDS has paved the way for firewalls to be relocated into the Cloud deployment. An alarming trend in recent malware attacks has proven that knowledgeable attackers have found stealthy techniques to elude and undermine traditional firewalls. To overcome this issue, Srivastava and Giffin [26] developed a tamper-resistant virtual firewall (VMwall) which leverages the benefits of both VM isolation and traditional firewalls. VMwall makes use of Xen hypervisor for isolation of the firewall and VM introspection to correlate UDP or TCP traffic with process information [27]. The Xen hypervisor provides functionality by adding hooks that capture all network operations. A whitelist (that approves processes and connection type) is used to determine if the connection is allowed or blocked. Although the VMwall was effective at identifying and blocking network connection with minimal impact on performance, the whitelist may be vulnerable to hijack by an already established connection.

Another approach in securing the Cloud is by using prevention techniques. Jiang et al. [28] proposed an intrusion prevention system by implementing VMwatcher which methodically reconstructs the data structures (i.e., files, directories, processes and kernel modules) of a VM for analysis at the hypervisor level in an unobtrusive manner. Casting these data structures and function schematics on the hypervisor-level VM so that the VM's schematic view can be reconstructed outside, makes the reconstruction possible. Monitoring tools, such as Windows Task Manager and memory dump, can be used to display all processes in memory so as to check for differences between guest's states. Differences between the two states may indicate the presence of malware in the guest VM. Further improvement to the VMwatcher implementation is to use anti-virus software to scan the state of guest VMs from inside the hypervisor. The major shortcoming of this implementation is the assumption that the hypervisor has not been compromised already.

Our review indicates that each of the core components of Cloud computing (network, storage, application, VM) has its potential threats. However, little effort has been made to study security at service delivery (IaaS, PaaS, SaaS) levels of Cloud computing. Although existing studies, such as References [29–31], primarily investigate the potential of implementing IDS in a Cloud environment; these studies also discuss the various threats contained in each service delivery model. Evaluation of IDS based on service delivery models is important as it provides detailed information about how each service delivery model can be secured. Furthermore, we believe that one of the main challenges faced by security administrators is to identify attacks that are embedded inside normal VM operations in Cloud environments.

CSP uses VM technologies and hypervisors, such as VMware, ESXi, Xen, Microsoft virtual PC, etc., to deliver a multi-tenant Cloud environment, which may have in them potential vulnerabilities that are prone to cyber-attacks [32]. In addition, conventional techniques for protecting user data through the use of anti-viruses, firewall and endpoint encryption are becoming extinct due to the multi-threaded nature of Cloud computing architectures [33]. Hence, the target for attackers has shifted from the network layer to the application layer due to Cloud services are being delivered through the web browser. Therefore, in the remainder of this article, we explore these gaps by proposing a model that incorporates intrusion detection practices into an IaaS model public Cloud platform.

4. Methodologies

Our hypothesis is that it is possible to deploy an IDS sensor in a Cloud environment in such a way that it can detect attacks that are embedded inside normal VM operation. Furthermore, one open question is how to select a technology that makes it possible to implement and manage IDS in Cloud environments. This technology should be adaptable enough for experimentation and have adequate performance to support current Cloud topologies.

4.1. Experimental Setup

The experimental environment is a highly flexible, centralized virtual lab infrastructure, which consists of an ESXi server hosting three VMs: Security Onion (SO) [34], Kali Linux [35] and Metasploitable [36] (see Figure 3). In this setup, SO runs the IDS instance, Kali hosts the attacker and Metasploitable is the victim/target machine. ESXi 6.0 is used which is a robust, high-performance hypervisor that virtualizes hardware resources and makes them shareable by several VMs inside VMware Workstation 12 on Intel (R) Core (TM) i7, 16GB RAM and 734GB hard disk space.

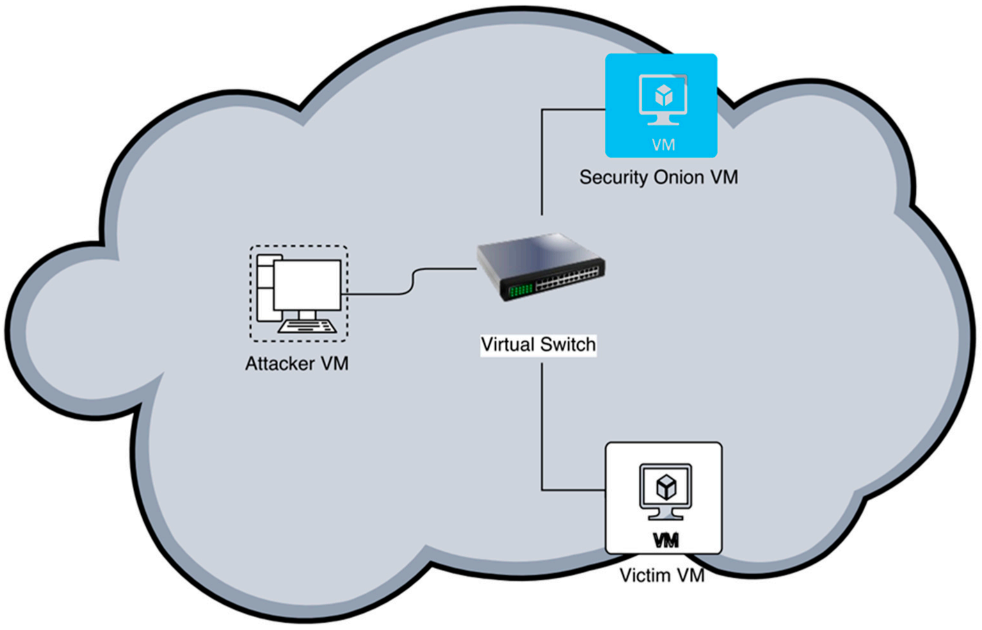


Figure 3. Experimental Setup.

To implement the IDS, SO is configured for network monitoring, intrusion detection and log management in one of our three VMs. The major factor that influenced the decision to use SO is that it comes with pre-installed security tools, such as Snort, Bro, Suricata, OSSEC, Sguil, etc., which makes network monitoring a less difficult task for security administrators. SO can be used in two ways, either proactive: When it is used to identify vulnerabilities or expiring SSL certificates, or reactive: When used for network forensics or incident response. Although SO provides visibility into a network and context around the anomalous event, it still requires commitment from a security administrator to monitor network activity, review alerts and incline to learn new things. There are other commercial solutions which offer similar tools as SO, but only a selected few contain the vast capabilities of SO in one package.

SO offers the choice of Snort or Suricata for rule-driven intrusion detection. In this work, Snort [37] is considered as it is a de-facto standard IDS with millions of downloads and is the most extensively deployed IDS technique worldwide. Snort IDS scans network traffic for identifiers or fingerprints that matched known malicious or suspicious traffic.

With full-packet capture and Snort logs, there is an overwhelming amount of data to be analyzed. Fortunately, SO integrates analysis tools, such as Sguil, Squert and ELSA. In this experiment, we use the Sguil console for data analysis. Sguil [38] provides visibility into the data that is being collected and the justification to validate the alert.

To test the effectiveness of our IDS, Metasploitable is installed in one VM to serve as a victim machine. Then attacks are simulated from Kali Linux to mimic real world scenario.

4.2. Network Configuration

For this experiment, all three VMs are interconnected via a virtual switch (vSwitch). The vSwitch is configured with two port groups: The VM port group, and the Promiscuous port group. The VM port group has its promiscuous mode set to "Reject" by default while the Promiscuous port group has its promiscuous mode set to "Accept."

4.3. Security Onion Configuration

We setup our SO as a standalone system with both server and sensor in the same location. We use production mode setup as we would like to have greater control over what we what to install. Also, we use Snort IDS engine and Emerging Threats GPL (no oinkcode required) ruleset. Oinkcode is a unique key connected with the user account given by Snort and it allows one to have more access to the latest rules and updates. Snort is an open source *libpcap*-based network intrusion detection and prevention system performing packet sniffing and logging and traffic analysis on IP networks. The current version of Snort employs different types of techniques, such as signature, protocol and anomaly-based inspection to detect and prevent cyber-attacks.

Security Onion has *pf_ring* built into it, which allows one to spin multiple Snort instances to handle a greater amount of traffic. However, this is limited by the number of CPU cores the VM has. In our case, the VM has two CPU cores, so we used the maximum amount of IDS engine processes that are allowed (i.e., 2). This means we can spin up to two instances of Snort.

There are three critical areas in the configuration and use of SO: The management of the configuration, rules file, and logs. Important file locations from which many of the tools contained in SO can be configured from are "/opt/bro" and "/etc/nsm" files. SO is only as effective as the rules that are implemented for it to use. These rules can be found in "/etc/nsm/rules" file. Important files in this location include: "local.rules", "downloaded.rules", "whitelist.rules" and "blacklist.rules" files. Also Log files of network activities, such as DHCP, DNS, SNMP, SSL and various other connection types, can be located at "/nsm/bro/current" for analysis through Sguil. For our system, we configured two NICs, NIC 1 (eth0) connected to the "VM network" and NIC 2 (eth1) for the management of the IDS.

5. Results and Discussion

This section discusses the process of penetration testing and analysis of data captured during penetration testing through the Sguil console. This testing involves all applications and services on the target system. First, we use the fping tool to determine the accessibility of the target. Next, we determine open and active services on the system by using open source Nmap [39] and traceroute tools. We further carry out a vulnerability scan on the victim system using Nessus vulnerability scanner [40]. Finally, we exploit the system using a Metasploit console.

Simultaneously, we carry out security monitoring and reporting on the platform using the Sguil console. Security onion has two software options: Sguil and Squert already preinstalled for this purpose. Classification of logged events in Sguil makes it easier for security analysts to easily review triggered alerts. For example, all forms of port scan or ping sweep are classified as Category 6. All category 6 alerts can be removed from the console hence allowing a security analyst to focus on other more dangerous alerts. However, in this experiment, all kinds of alerts are important to us. This gives us a proper view of what is monitored by our IDS.

5.1. Discovery Phase

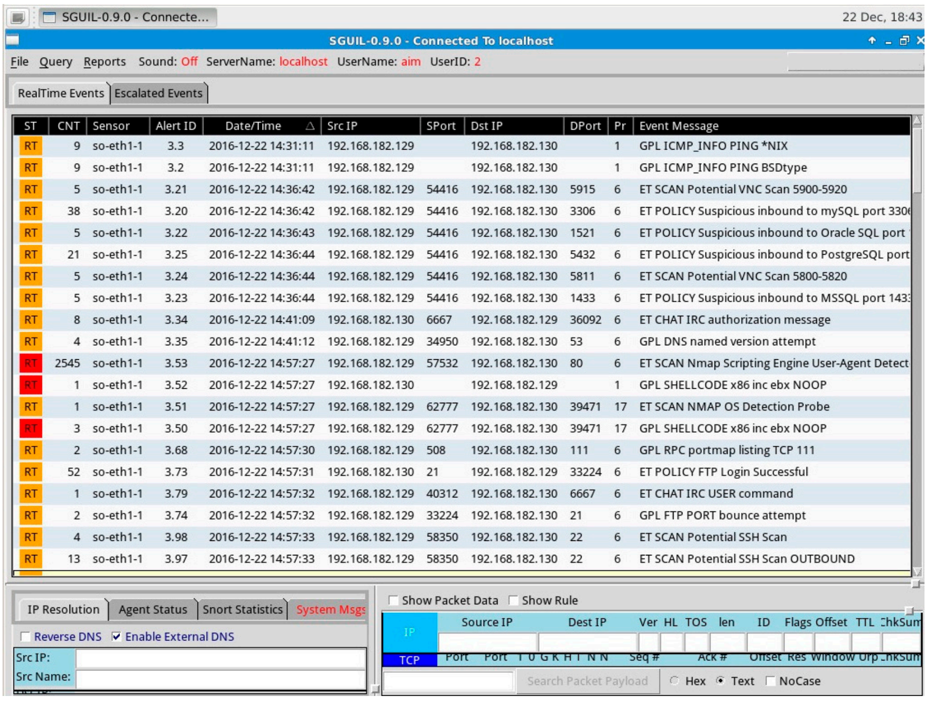
5.1.1. Intelligence Gathering and Analysis

Foot Printing: Our first assumption is that the malicious user does not know the IP address of the victim system. We employ fping tool to find out the host that are alive on the network.

Scanning: The foot printing stage discussed earlier aids a malicious user to discover the IP address of the victim. Furthermore, the malicious user can use tools, such as traceroute and ping to collect more information about the victim system. The output is given by the ping and traceroute command shows that the victim system is alive and accessible with a maximum of 30 hops. This, therefore, confirms that the victim system is active and can be reached within a range of possible distance.

After confirming that the victim system is alive, the different applications/services running on the victim system can be identified by the attacker using Nmap command. The result from Nmap scan reveals ports that are open, their service version, and the protocols types which are currently running.

During the information gathering stage, our IDS has real-time access to events, session data and packet data captured on the network, and uses the Sguil interface to analyze these events. Figure 4 shows the output of the Sguil console during the information gathering phase.


SGUIL-0.9.0 - Connected To localhost

File Query Reports Sound: Off ServerName: localhost UserName: aim UserID: 2

ST	CNT	Sensor	Alert ID	Date/Time	Src IP	SPort	Dst IP	DPort	Pr	Event Message
RT	9	so-eth1-1	3.3	2016-12-22 14:31:11	192.168.182.129		192.168.182.130		1	GPL ICMP_INFO PING *NIX
RT	9	so-eth1-1	3.2	2016-12-22 14:31:11	192.168.182.129		192.168.182.130		1	GPL ICMP_INFO PING BSDtype
RT	5	so-eth1-1	3.21	2016-12-22 14:36:42	192.168.182.129	54416	192.168.182.130	5915	6	ET SCAN Potential VNC Scan 5900-5920
RT	38	so-eth1-1	3.20	2016-12-22 14:36:42	192.168.182.129	54416	192.168.182.130	3306	6	ET POLICY Suspicious inbound to mySQL port 330
RT	5	so-eth1-1	3.22	2016-12-22 14:36:43	192.168.182.129	54416	192.168.182.130	1521	6	ET POLICY Suspicious inbound to Oracle SQL port
RT	21	so-eth1-1	3.25	2016-12-22 14:36:44	192.168.182.129	54416	192.168.182.130	5432	6	ET POLICY Suspicious inbound to PostgreSQL port
RT	5	so-eth1-1	3.24	2016-12-22 14:36:44	192.168.182.129	54416	192.168.182.130	5811	6	ET SCAN Potential VNC Scan 5800-5820
RT	5	so-eth1-1	3.23	2016-12-22 14:36:44	192.168.182.129	54416	192.168.182.130	1433	6	ET POLICY Suspicious inbound to MSSQL port 143
RT	8	so-eth1-1	3.34	2016-12-22 14:41:09	192.168.182.130	6667	192.168.182.129	36092	6	ET CHAT IRC authorization message
RT	4	so-eth1-1	3.35	2016-12-22 14:41:12	192.168.182.129	34950	192.168.182.130	53	6	GPL DNS named version attempt
RT	2545	so-eth1-1	3.53	2016-12-22 14:57:27	192.168.182.129	57532	192.168.182.130	80	6	ET SCAN Nmap Scripting Engine User-Agent Detect
RT	1	so-eth1-1	3.52	2016-12-22 14:57:27	192.168.182.130		192.168.182.129		1	GPL SHELLCODE x86 inc ebx NOOP
RT	1	so-eth1-1	3.51	2016-12-22 14:57:27	192.168.182.129	62777	192.168.182.130	39471	17	ET SCAN NMAP OS Detection Probe
RT	3	so-eth1-1	3.50	2016-12-22 14:57:27	192.168.182.129	62777	192.168.182.130	39471	17	GPL SHELLCODE x86 inc ebx NOOP
RT	2	so-eth1-1	3.68	2016-12-22 14:57:30	192.168.182.129	508	192.168.182.130	111	6	GPL RPC portmap listing TCP 111
RT	52	so-eth1-1	3.73	2016-12-22 14:57:31	192.168.182.130	21	192.168.182.129	33224	6	ET POLICY FTP Login Successful
RT	1	so-eth1-1	3.79	2016-12-22 14:57:32	192.168.182.129	40312	192.168.182.130	6667	6	ET CHAT IRC USER command
RT	2	so-eth1-1	3.74	2016-12-22 14:57:32	192.168.182.129	33224	192.168.182.130	21	6	GPL FTP PORT bounce attempt
RT	4	so-eth1-1	3.98	2016-12-22 14:57:33	192.168.182.129	58350	192.168.182.130	22	6	ET SCAN Potential SSH Scan
RT	13	so-eth1-1	3.97	2016-12-22 14:57:33	192.168.182.129	58350	192.168.182.130	22	6	ET SCAN Potential SSH Scan OUTBOUND

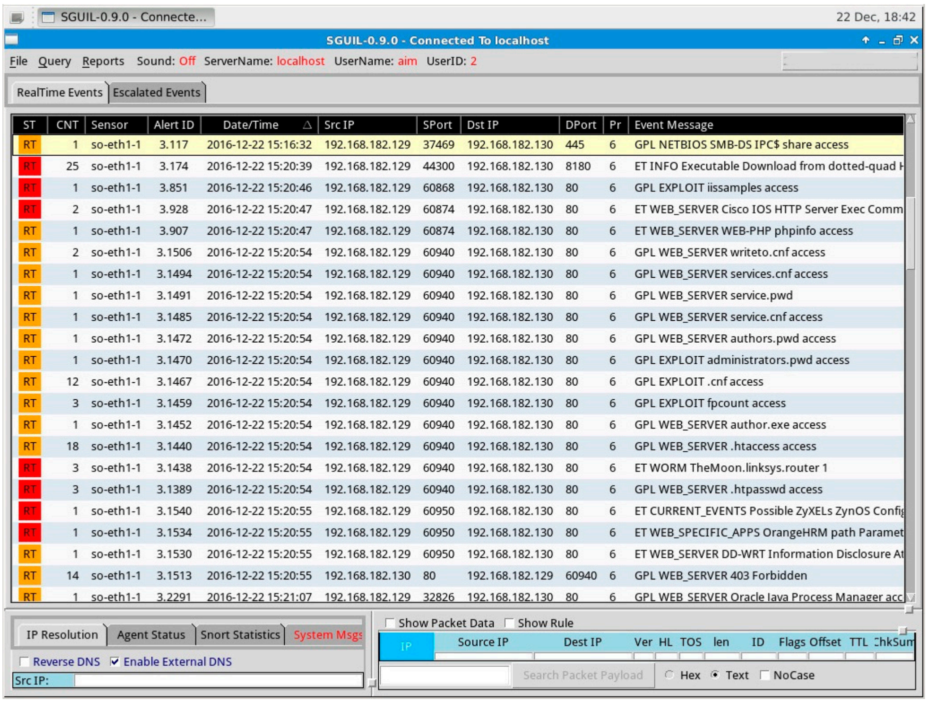
Figure 4. Sguil Console Output: Information Gathering.

Data from the Figure 4 above indicates that a variety of scans and probes are being initiated on the target IP 192.168.182.130 from the source IP 192.168.182.129. The event message column gives a security analyst a brief description of the action that triggered the alert. The first line on the console (first logged event) indicates that a ping request is being sent across the network. These tallies with the first process of information gathering we carried out. Further going down the list of logged events, we noticed that each line on the console corresponds with all of the probes and scans made during the information gathering stage. The Sguil console also shows the network interface being monitored, the source and destination ports and ID of the alert for further analysis.

Enumerating: In the enumeration stage, we make use of open port services and active connections to detect services that are not adequately implemented. This stage helps an attacker to create a blueprint of the target network, hence determining potential weaknesses. The domain name, MAC address and host name of the destination NetBIOS of the victim system are then revealed by means of *nbtscan* command.

Additional enumeration is performed using the Metasploit framework. We found the version of Samba of the victim machine. We also enumerated HTTP port 80 using the Nmap.

During the enumeration phase, our IDS continues monitoring the network and alerts for strange events to the Sguil console for further analysis by a security administrator. Figure 5 below shows the output of the Sguil console during the enumeration phase.



Sound: Off ServerName: localhost UserName: aim UserID: 2

ST	CNT	Sensor	Alert ID	Date/Time	Src IP	SPort	Dst IP	DPort	Pr	Event Message
RT	1	so-eth1-1	3.117	2016-12-22 15:16:32	192.168.182.129	37469	192.168.182.130	445	6	GPL NETBIOS SMB-DS IPC$ share access
RT	25	so-eth1-1	3.174	2016-12-22 15:20:39	192.168.182.129	44300	192.168.182.130	8180	6	ET INFO Executable Download from dotted-quad H
RT	1	so-eth1-1	3.851	2016-12-22 15:20:46	192.168.182.129	60868	192.168.182.130	80	6	GPL EXPLOIT iissamples access
RT	2	so-eth1-1	3.928	2016-12-22 15:20:47	192.168.182.129	60874	192.168.182.130	80	6	ET WEB_SERVER Cisco IOS HTTP Server Exec Comm
RT	1	so-eth1-1	3.907	2016-12-22 15:20:47	192.168.182.129	60874	192.168.182.130	80	6	ET WEB_SERVER WEB-PHP phpinfo access
RT	2	so-eth1-1	3.1506	2016-12-22 15:20:54	192.168.182.129	60940	192.168.182.130	80	6	GPL WEB_SERVER writeto.cnf access
RT	1	so-eth1-1	3.1494	2016-12-22 15:20:54	192.168.182.129	60940	192.168.182.130	80	6	GPL WEB_SERVER services.cnf access
RT	1	so-eth1-1	3.1491	2016-12-22 15:20:54	192.168.182.129	60940	192.168.182.130	80	6	GPL WEB_SERVER service.pwd
RT	1	so-eth1-1	3.1485	2016-12-22 15:20:54	192.168.182.129	60940	192.168.182.130	80	6	GPL WEB_SERVER service.cnf access
RT	1	so-eth1-1	3.1472	2016-12-22 15:20:54	192.168.182.129	60940	192.168.182.130	80	6	GPL WEB_SERVER authors.pwd access
RT	1	so-eth1-1	3.1470	2016-12-22 15:20:54	192.168.182.129	60940	192.168.182.130	80	6	GPL EXPLOIT administrators.pwd access
RT	12	so-eth1-1	3.1467	2016-12-22 15:20:54	192.168.182.129	60940	192.168.182.130	80	6	GPL EXPLOIT .cnf access
RT	3	so-eth1-1	3.1459	2016-12-22 15:20:54	192.168.182.129	60940	192.168.182.130	80	6	GPL EXPLOIT fpcount access
RT	1	so-eth1-1	3.1452	2016-12-22 15:20:54	192.168.182.129	60940	192.168.182.130	80	6	GPL WEB_SERVER author.exe access
RT	18	so-eth1-1	3.1440	2016-12-22 15:20:54	192.168.182.129	60940	192.168.182.130	80	6	GPL WEB_SERVER .htaccess access
RT	3	so-eth1-1	3.1438	2016-12-22 15:20:54	192.168.182.129	60940	192.168.182.130	80	6	ET WORM TheMoon.linksys.router 1
RT	3	so-eth1-1	3.1389	2016-12-22 15:20:54	192.168.182.129	60940	192.168.182.130	80	6	GPL WEB_SERVER .htpasswd access
RT	1	so-eth1-1	3.1540	2016-12-22 15:20:55	192.168.182.129	60950	192.168.182.130	80	6	ET CURRENT_EVENTS Possible ZyXELs ZynOS Confi
RT	1	so-eth1-1	3.1534	2016-12-22 15:20:55	192.168.182.129	60950	192.168.182.130	80	6	ET WEB_SPECIFIC_APPS OrangeHRM path Paramet
RT	1	so-eth1-1	3.1530	2016-12-22 15:20:55	192.168.182.129	60950	192.168.182.130	80	6	ET WEB_SERVER DD-WRT Information Disclosure At
RT	14	so-eth1-1	3.1513	2016-12-22 15:20:55	192.168.182.130	80	192.168.182.129	60940	6	GPL WEB_SERVER 403 Forbidden
RT	1	so-eth1-1	3.2291	2016-12-22 15:21:07	192.168.182.129	32826	192.168.182.130	80	6	GPL WEB_SERVER Oracle Java Process Manager acc

Reverse DNS [✓ Enable External DNS]
Src IP:

IP	Source IP	Dest IP	Ver	HL	TOS	len	ID	Flags	Offset	TTL	ChkSum

Hex [✓ Text] NoCase

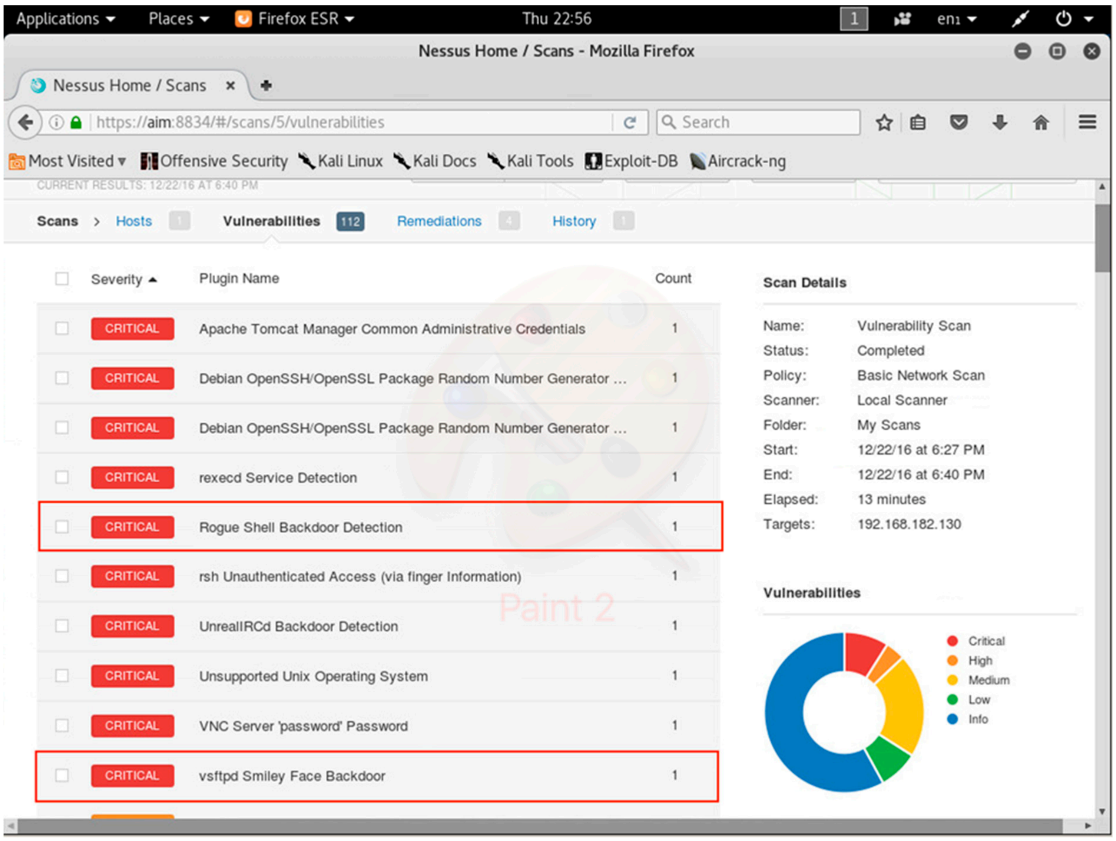
Figure 5. Sguil Console Output: Enumeration Phase.

The output of the enumeration phase showed in Figure 5 indicates in the first line of the probe on the system with the destination IP 192.168.182.130 and destination port 445 from source IP 192.168.182.129 and source port 37,469 to detect the version of Samba that is being run by the target system. Also, the output of the console indicates several probes being carried out on the HTTP port 80 of the target system; this alert corresponds to the HTTP enumeration we carried out.

5.1.2. Vulnerability Detection

After gathering useful information about the victim system and the network, the next stage in our penetration testing is to scan the victim system for vulnerabilities that can be exploited. To scan for available vulnerabilities in our victim, we configured Nessus vulnerability scanner against the victim machine. A detailed result of this vulnerability scan is shown in Figure 6 following their level of severity.

The result of the Nessus scan (see Figure 6) is strictly based on services, which are open and active, it also includes an OS and applications discovered in earlier phases. The total number of vulnerabilities found was 112, but we only focused on two vulnerabilities (rouge shell backdoor and VSFTPD smiley face backdoor) which both have a critical level of severity. Compared to other vulnerabilities that are at a critical severity level, the chance of exploiting Samba and VSFTPD is at a reasonable level due to the fact that some preconditions need to be met before the exploit can take place [41].



Severity	Plugin Name	Count
CRITICAL	Apache Tomcat Manager Common Administrative Credentials	1
CRITICAL	Debian OpenSSH/OpenSSL Package Random Number Generator ...	1
CRITICAL	Debian OpenSSH/OpenSSL Package Random Number Generator ...	1
CRITICAL	rexecd Service Detection	1
CRITICAL	Rogue Shell Backdoor Detection	1
CRITICAL	rsh Unauthenticated Access (via finger Information)	1
CRITICAL	UnrealIRCd Backdoor Detection	1
CRITICAL	Unsupported Unix Operating System	1
CRITICAL	VNC Server 'password' Password	1
CRITICAL	vsftpd Smiley Face Backdoor	1

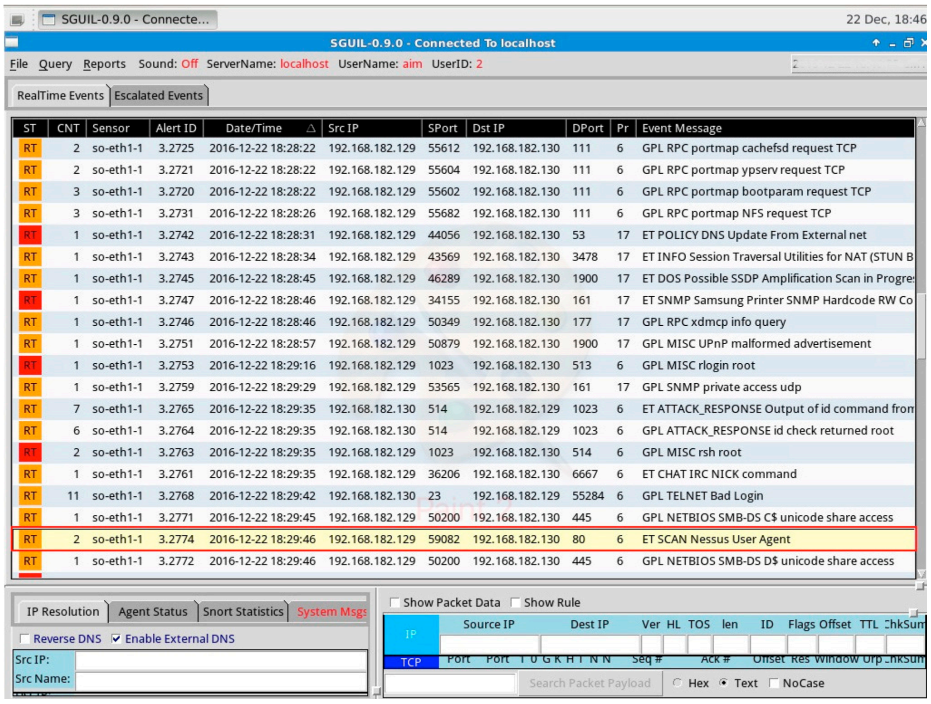
Scan Details

Name:	Vulnerability Scan
Status:	Completed
Policy:	Basic Network Scan
Scanner:	Local Scanner
Folder:	My Scans
Start:	12/22/16 at 6:27 PM
End:	12/22/16 at 6:40 PM
Elapsed:	13 minutes
Targets:	192.168.182.130

Vulnerabilities

Figure 6. Output of Nessus Vulnerability Scanner.

While running the vulnerability scan, our IDS was also monitoring the network and events were logged into the Sguil console. Figure 7 represents the output of the Sguil console while the Nessus vulnerability scanner was running.



ST	CNT	Sensor	Alert ID	Date/Time	Src IP	SPort	Dst IP	DPort	Pr	Event Message
RT	2	so-eth1-1	3.2725	2016-12-22 18:28:22	192.168.182.129	55612	192.168.182.130	111	6	GPL RPC portmap cachefsd request TCP
RT	2	so-eth1-1	3.2721	2016-12-22 18:28:22	192.168.182.129	55604	192.168.182.130	111	6	GPL RPC portmap ypserv request TCP
RT	3	so-eth1-1	3.2720	2016-12-22 18:28:22	192.168.182.129	55602	192.168.182.130	111	6	GPL RPC portmap bootparam request TCP
RT	3	so-eth1-1	3.2731	2016-12-22 18:28:26	192.168.182.129	55682	192.168.182.130	111	6	GPL RPC portmap NFS request TCP
RT	1	so-eth1-1	3.2742	2016-12-22 18:28:31	192.168.182.129	44056	192.168.182.130	53	17	ET POLICY DNS Update From External net
RT	1	so-eth1-1	3.2743	2016-12-22 18:28:34	192.168.182.129	43569	192.168.182.130	3478	17	ET INFO Session Traversal Utilities for NAT (STUN B
RT	1	so-eth1-1	3.2745	2016-12-22 18:28:45	192.168.182.129	46289	192.168.182.130	1900	17	ET DOS Possible SSDP Amplification Scan in Progre
RT	1	so-eth1-1	3.2747	2016-12-22 18:28:46	192.168.182.129	34155	192.168.182.130	161	17	ET SNMP Samsung Printer SNMP Hardcode RW Co
RT	1	so-eth1-1	3.2746	2016-12-22 18:28:46	192.168.182.129	50349	192.168.182.130	177	17	GPL RPC xdmcp info query
RT	1	so-eth1-1	3.2751	2016-12-22 18:28:57	192.168.182.129	50879	192.168.182.130	1900	17	GPL MISC UPnP malformed advertisement
RT	1	so-eth1-1	3.2753	2016-12-22 18:29:16	192.168.182.129	1023	192.168.182.130	513	6	GPL MISC rlogin root
RT	1	so-eth1-1	3.2759	2016-12-22 18:29:29	192.168.182.129	53565	192.168.182.130	161	17	GPL SNMP private access udp
RT	7	so-eth1-1	3.2765	2016-12-22 18:29:35	192.168.182.130	514	192.168.182.129	1023	6	ET ATTACK_RESPONSE Output of id command from
RT	6	so-eth1-1	3.2764	2016-12-22 18:29:35	192.168.182.130	514	192.168.182.129	1023	6	GPL ATTACK_RESPONSE id check returned root
RT	2	so-eth1-1	3.2763	2016-12-22 18:29:35	192.168.182.129	1023	192.168.182.130	514	6	GPL MISC rsh root
RT	1	so-eth1-1	3.2761	2016-12-22 18:29:35	192.168.182.129	36206	192.168.182.130	6667	6	ET CHAT IRC NICK command
RT	11	so-eth1-1	3.2768	2016-12-22 18:29:42	192.168.182.130	23	192.168.182.129	55284	6	GPL TELNET Bad Login
RT	1	so-eth1-1	3.2771	2016-12-22 18:29:45	192.168.182.129	50200	192.168.182.130	445	6	GPL NETBIOS SMB-DS C$ unicode share access
RT	2	so-eth1-1	3.2774	2016-12-22 18:29:46	192.168.182.129	59082	192.168.182.130	80	6	ET SCAN Nessus User Agent
RT	1	so-eth1-1	3.2772	2016-12-22 18:29:46	192.168.182.129	50200	192.168.182.130	445	6	GPL NETBIOS SMB-DS D$ unicode share access

IP	Source IP	Dest IP	Ver	HL	TOS	len	ID	Flags	Offset	TTL	ChkSum

Figure 7. Sguil Output: Nessus Vulnerability Scan.

During the Nessus vulnerability scan, many alerts were reported on the Sguil console, and this involves different types of port scans to detect open ports and various kinds of attacks simulated by the vulnerability scanner to detect if the target system is porous. From the console, it can be observed that although the destination IP (192.168.182.130) and source IP (192.168.182.129) remained constant, the origin and destination ports were varying. Also visible from the console is the name of the vulnerability scanner being used, i.e., Nessus User Agent.

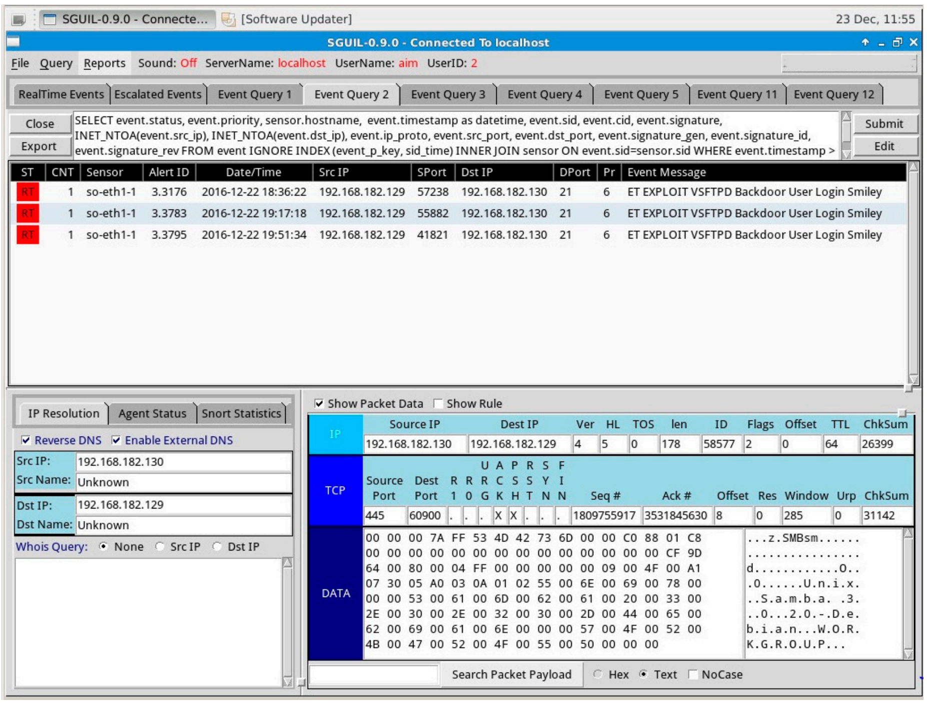
5.2. Attack Phase (Penetration Attempt)

After determining the vulnerabilities that exist in the system in the previous phases. Next, we attempt to exploit these vulnerabilities.

5.2.1. VSFTPD Smiley Face Backdoor Attack

OSs, such as Ubuntu, CentOS and Fedora, contain VSFTPD ftp server. Over the years, this service has been quite secure, but a key incident happened in July 2011 when the original version of this service was replaced with a version containing malicious codes and a backdoor. This vulnerability is existent in version 2.3.4 of VSFTPD. The VSFTPD vulnerability allows attackers to gain access to the shell of affected systems by listening on port 6200 and logging in with Username: ":)" and any password of choice. We carried out this attack by using the Netcat command to find the version of VSFTD, and we found it was version 2.3.4 which contains a backdoor. We then proceed to use username "letmein :)" and password "hi". While the first terminal was still running, we opened a second terminal and then used Netcat command to gain access to shell and root using port 6200 unlike 21 employed in the first.

During this attack, our IDS was in position and was monitoring all the events traversing the network. Figure 8 illustrates a screenshot of the alert received on the Sguil interface during the attack.


SGUIL-0.9.0 - Connecte...
[Software Updater]
23 Dec, 11:55
SGUIL-0.9.0 - Connected To localhost
File Query Reports
RealTime Events | Escalated Events | Event Query 1 | Event Query 2 | Event Query 3 | Event Query 4 | Event Query 5 | Event Query 11 | Event Query 12
Close
Export
Submit
Edit
IP Resolution | Agent Status | Snort Statistics
Search Packet Payload

Sound: Off ServerName: localhost UserName: aim UserID: 2

```
SELECT event.status, event.priority, sensor.hostname,  event.timestamp as datetime, event.sid, event.cid, event.signature,
INET_NTOA(event.src_ip), INET_NTOA(event.dst_ip), event.ip_proto, event.src_port, event.dst_port, event.signature_gen, event.signature_id,
event.signature_rev FROM event IGNORE INDEX (event_p_key, sid_time) INNER JOIN sensor ON event.sid=sensor.sid WHERE event.timestamp >
```

ST	CNT	Sensor	Alert ID	Date/Time	Src IP	SPort	Dst IP	DPort	Pr	Event Message
RT	1	so-eth1-1	3.3176	2016-12-22 18:36:22	192.168.182.129	57238	192.168.182.130	21	6	ET EXPLOIT VSFTPD Backdoor User Login Smiley
RT	1	so-eth1-1	3.3783	2016-12-22 19:17:18	192.168.182.129	55882	192.168.182.130	21	6	ET EXPLOIT VSFTPD Backdoor User Login Smiley
RT	1	so-eth1-1	3.3795	2016-12-22 19:51:34	192.168.182.129	41821	192.168.182.130	21	6	ET EXPLOIT VSFTPD Backdoor User Login Smiley

[✓ Reverse DNS] [✓ Enable External DNS]

Src IP:	192.168.182.130
Src Name:	Unknown
Dst IP:	192.168.182.129
Dst Name:	Unknown

Whois Query: [✓ None] Src IP Dst IP

[✓ Show Packet Data] Show Rule

IP	Source IP	Dest IP	Ver	HL	TOS	len	ID	Flags	Offset	TTL	ChkSum
	192.168.182.130	192.168.182.129	4	5	0	178	58577	2	0	64	26399

TCP	Source Port	Dest Port	R 1	R 0	U R G	A C K	P S H	R S T	S Y N	F I N	Seq #	Ack #	Offset	Res	Window	Urp	ChkSum
	445	60900	.	.	.	X	X	.	.	.	1809755917	3531845630	8	0	285	0	31142

DATA

```
00 00 00 7A FF 53 4D 42 73 6D 00 00 C0 88 01 C8    ...z.SMBsm......
00 00 00 00 00 00 00 00 00 00 00 00 00 00 CF 9D    ................
64 00 80 00 04 FF 00 00 00 00 00 09 00 4F 00 A1    d...........O..
07 30 05 A0 03 0A 01 02 55 00 6E 00 69 00 78 00    .0......U.n.i.x.
00 00 53 00 61 00 6D 00 62 00 61 00 20 00 33 00    ..S.a.m.b.a. .3.
2E 00 30 00 2E 00 32 00 30 00 2D 00 44 00 65 00    ..0...2.0.-.D.e.
62 00 69 00 61 00 6E 00 00 00 57 00 4F 00 52 00    b.i.a.n...W.O.R.
4B 00 47 00 52 00 4F 00 55 00 50 00 00 00          K.G.R.O.U.P...
```

Hex [✓ Text] NoCase

Figure 8. Sguil Console: VSFTD Smiley Face Attack Alert.

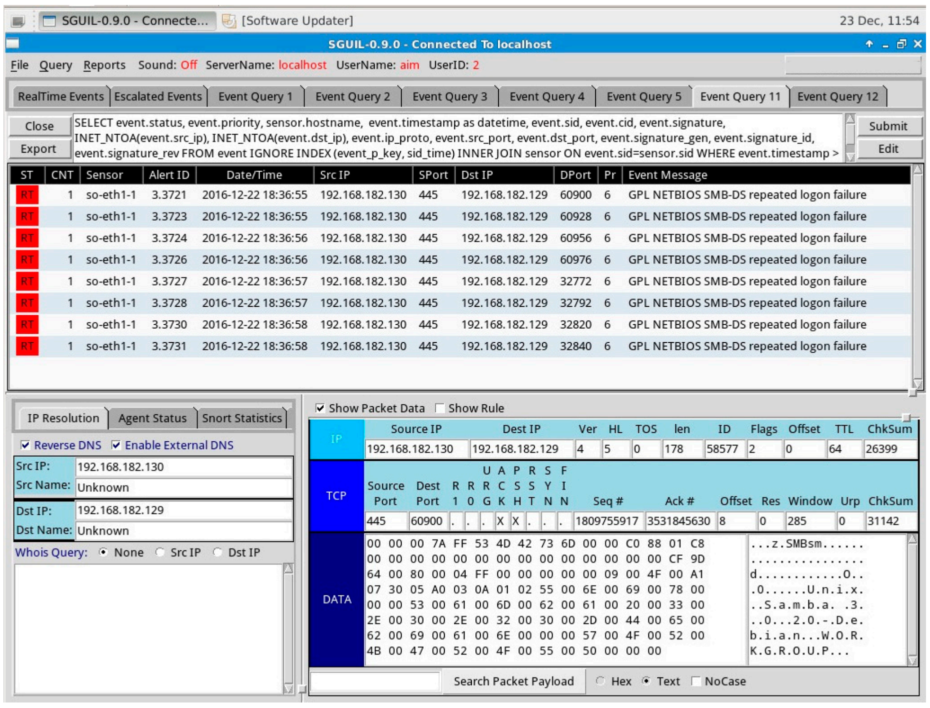
The Sguil console in Figure 8 shows that a VSFTPD backdoor attack was launched from IP address 192.168.182.129 to 192.168.182.130. The red banner in the first column indicates that alert needs immediate attention from the security administrator.

5.2.2. Samba MS-RPC Remote Shell Command Execution Attack

Versions 3.0.0 to 3.0.25rc3 of Samba contains a vulnerability that allows intruders to perform shell commands due to the failure of the software to verify and sanitize user inputs. Malicious users can exploit this vulnerability by running arbitrary shell commands through the vulnerable Samba application on the target machine.

In the previous stage of enumeration, we found out the version of samba to be 3.0.20; this means the Samba has a vulnerability. In Metasploit framework, we used "exploit/multi/samba/usermap_script" module to exploit command execution susceptibility in Samba when utilizing non-default "username map script".

Our IDS running in the background alerted us on some suspicious activities happening on the network. Figure 9 demonstrates the alerts displayed on the Sguil console during this attack. As seen on the Sguil interface in Figure 9, our IDS successfully alerts the security administrator of an attempt from source IP 192.168.182.129 to exploit the Samba application of the target machine 192.168.182.130.


SGUIL-0.9.0 - Connecte...
[Software Updater]
23 Dec, 11:54
SGUIL-0.9.0 - Connected To localhost
File Query Reports
RealTime Events | Escalated Events | Event Query 1 | Event Query 2 | Event Query 3 | Event Query 4 | Event Query 5 | Event Query 11 | Event Query 12
Close
Export
Submit
Edit
IP Resolution | Agent Status | Snort Statistics
Search Packet Payload

Sound: Off ServerName: localhost UserName: aim UserID: 2

```
SELECT event.status, event.priority, sensor.hostname, event.timestamp as datetime, event.sid, event.cid, event.signature,
INET_NTOA(event.src_ip), INET_NTOA(event.dst_ip), event.ip_proto, event.src_port, event.dst_port, event.signature_gen, event.signature_id,
event.signature_rev FROM event IGNORE INDEX (event_p_key, sid_time) INNER JOIN sensor ON event.sid=sensor.sid WHERE event.timestamp >
```

ST	CNT	Sensor	Alert ID	Date/Time	Src IP	SPort	Dst IP	DPort	Pr	Event Message
RT	1	so-eth1-1	3.3721	2016-12-22 18:36:55	192.168.182.130	445	192.168.182.129	60900	6	GPL NETBIOS SMB-DS repeated logon failure
RT	1	so-eth1-1	3.3723	2016-12-22 18:36:55	192.168.182.130	445	192.168.182.129	60928	6	GPL NETBIOS SMB-DS repeated logon failure
RT	1	so-eth1-1	3.3724	2016-12-22 18:36:56	192.168.182.130	445	192.168.182.129	60956	6	GPL NETBIOS SMB-DS repeated logon failure
RT	1	so-eth1-1	3.3726	2016-12-22 18:36:56	192.168.182.130	445	192.168.182.129	60976	6	GPL NETBIOS SMB-DS repeated logon failure
RT	1	so-eth1-1	3.3727	2016-12-22 18:36:57	192.168.182.130	445	192.168.182.129	32772	6	GPL NETBIOS SMB-DS repeated logon failure
RT	1	so-eth1-1	3.3728	2016-12-22 18:36:57	192.168.182.130	445	192.168.182.129	32792	6	GPL NETBIOS SMB-DS repeated logon failure
RT	1	so-eth1-1	3.3730	2016-12-22 18:36:58	192.168.182.130	445	192.168.182.129	32820	6	GPL NETBIOS SMB-DS repeated logon failure
RT	1	so-eth1-1	3.3731	2016-12-22 18:36:58	192.168.182.130	445	192.168.182.129	32840	6	GPL NETBIOS SMB-DS repeated logon failure

[✓] Reverse DNS [✓] Enable External DNS

Src IP:	192.168.182.130
Src Name:	Unknown
Dst IP:	192.168.182.129
Dst Name:	Unknown

Whois Query: [✓ None] Src IP Dst IP

[✓] Show Packet Data [ ] Show Rule

IP	Source IP	Dest IP	Ver	HL	TOS	len	ID	Flags	Offset	TTL	ChkSum
	192.168.182.130	192.168.182.129	4	5	0	178	58577	2	0	64	26399

TCP	Source Port	Dest Port	R 1	R 0	U R G	A C K	P S H	R S T	S Y N	F I N	Seq #	Ack #	Offset	Res	Window	Urp	ChkSum
	445	60900	.	.	.	X	X	.	.	.	1809755917	3531845630	8	0	285	0	31142

DATA

```
00 00 00 7A FF 53 4D 42 73 6D 00 00 C0 88 01 C8    ...z.SMBsm......
00 00 00 00 00 00 00 00 00 00 00 00 00 00 CF 9D    ................
64 00 80 00 04 FF 00 00 00 00 00 09 00 4F 00 A1    d...........O..
07 30 05 A0 03 0A 01 02 55 00 6E 00 69 00 78 00    .0......U.n.i.x.
00 00 53 00 61 00 6D 00 62 00 61 00 20 00 33 00    ..S.a.m.b.a. .3.
2E 00 30 00 2E 00 32 00 30 00 2D 00 44 00 65 00    ..0...2.0.-.D.e.
62 00 69 00 61 00 6E 00 00 00 57 00 4F 00 52 00    b.i.a.n...W.O.R.
4B 00 47 00 52 00 4F 00 55 00 50 00 00 00          K.G.R.O.U.P...
```

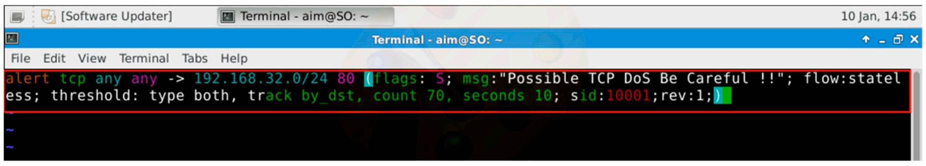
Hex [✓ Text] [ ] NoCase

Figure 9. Sguil Console: Samba Attack Alert.

5.2.3. VSFTPD Smiley Face Backdoor Attack

In this section, we make use of Hping3 tool in Kali Linux to mimic a basic DDoS attack on our Metasploitable VM. Although this DDoS attack simulation is very limited regarding attack traffic used and scale, it gives us a taste of how efficient our IDS can detect real-life Internet scale attacks.

Before we started launching the DDoS attacks, we added several rules into our Snort IDS engine to make sure it able to detect malicious traffic and generate appropriate alerts. Figure 10 shows an example of one of the rules added to the *local.rules* file.



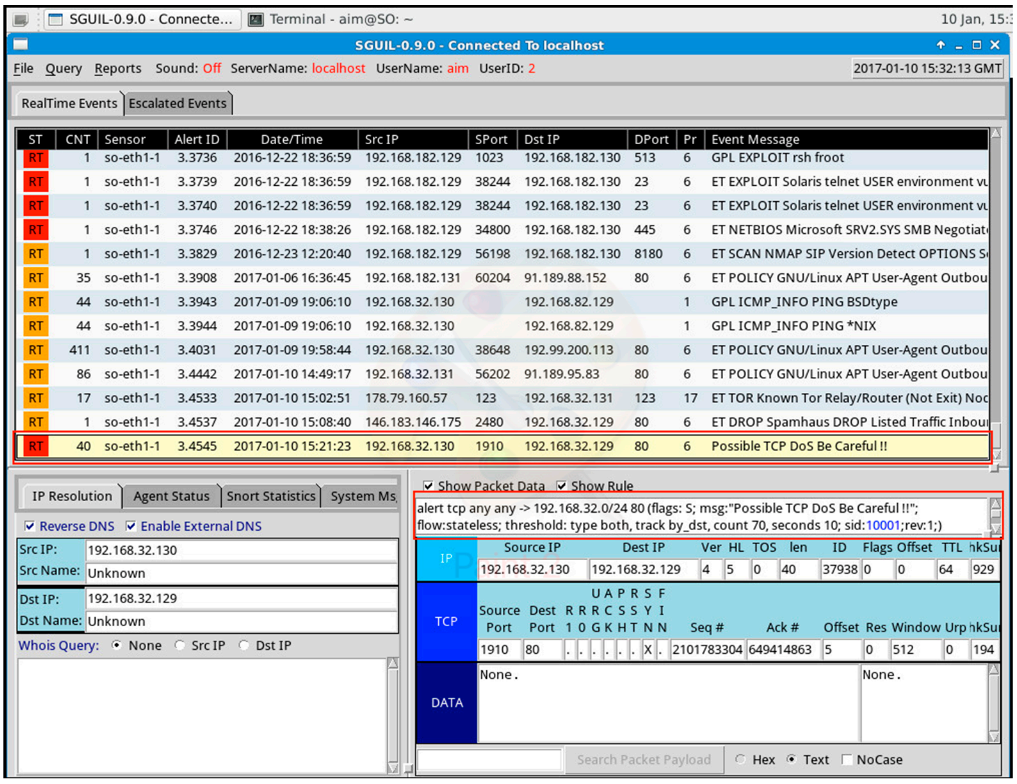
```
alert tcp any any -> 192.168.32.0/24 80 (flags: S; msg:"Possible TCP DoS Be Careful !!"; flow:statel
ess; threshold: type both, track by_dst, count 70, seconds 10; sid:10001;rev:1;)
~
~
~
```

Figure 10. Rule Added for DDoS Detection.

This means that the rule will alert if any TCP flooding is targeted at the destination network 192.168.32.0/24 and destination port *"80"* from *"any"* source network and *"any"* source port.

The *Hping3* tool in Kali Linux was used to simulate an SYN flood attack. This attack involves hitting our victim (Metasploitable VM) with a large number of SYN packets and seeing how it affects usage of the VM.

While the simulated attack running, our Sguil interface is active and alerts us of a possible DDoS attack (see Figure 11). Figure 11 shows an alert that was triggered on the Sguil console. We have highlighted the alert produced and the rule that triggered the alert. It shows a possible TCP DoS attack was launched from IP address 192.168.32.130 (IP address of our Kali Linux VM) to the destination IP address 192.168.32.129 and port 80.



Sound: Off ServerName: localhost UserName: aim UserID: 2 2017-01-10 15:32:13 GMT

ST	CNT	Sensor	Alert ID	Date/Time	Src IP	SPort	Dst IP	DPort	Pr	Event Message
RT	1	so-eth1-1	3.3736	2016-12-22 18:36:59	192.168.182.129	1023	192.168.182.130	513	6	GPL EXPLOIT rsh froot
RT	1	so-eth1-1	3.3739	2016-12-22 18:36:59	192.168.182.129	38244	192.168.182.130	23	6	ET EXPLOIT Solaris telnet USER environment vu
RT	1	so-eth1-1	3.3740	2016-12-22 18:36:59	192.168.182.129	38244	192.168.182.130	23	6	ET EXPLOIT Solaris telnet USER environment vu
RT	1	so-eth1-1	3.3746	2016-12-22 18:38:26	192.168.182.129	34800	192.168.182.130	445	6	ET NETBIOS Microsoft SRV2.SYS SMB Negotiat
RT	1	so-eth1-1	3.3829	2016-12-23 12:20:40	192.168.182.129	56198	192.168.182.130	8180	6	ET SCAN NMAP SIP Version Detect OPTIONS S
RT	35	so-eth1-1	3.3908	2017-01-06 16:36:45	192.168.182.131	60204	91.189.88.152	80	6	ET POLICY GNU/Linux APT User-Agent Outbou
RT	44	so-eth1-1	3.3943	2017-01-09 19:06:10	192.168.32.130		192.168.82.129		1	GPL ICMP_INFO PING BSDtype
RT	44	so-eth1-1	3.3944	2017-01-09 19:06:10	192.168.32.130		192.168.82.129		1	GPL ICMP_INFO PING *NIX
RT	411	so-eth1-1	3.4031	2017-01-09 19:58:44	192.168.32.130	38648	192.99.200.113	80	6	ET POLICY GNU/Linux APT User-Agent Outbou
RT	86	so-eth1-1	3.4442	2017-01-10 14:49:17	192.168.32.131	56202	91.189.95.83	80	6	ET POLICY GNU/Linux APT User-Agent Outbou
RT	17	so-eth1-1	3.4533	2017-01-10 15:02:51	178.79.160.57	123	192.168.32.131	123	17	ET TOR Known Tor Relay/Router (Not Exit) Noc
RT	1	so-eth1-1	3.4537	2017-01-10 15:08:40	146.183.146.175	2480	192.168.32.129	80	6	ET DROP Spamhaus DROP Listed Traffic Inbou
RT	40	so-eth1-1	3.4545	2017-01-10 15:21:23	192.168.32.130	1910	192.168.32.129	80	6	Possible TCP DoS Be Careful !!

[✓] Reverse DNS [✓] Enable External DNS

Src IP:	192.168.32.130
Src Name:	Unknown
Dst IP:	192.168.32.129
Dst Name:	Unknown

Whois Query: [● None] ○ Src IP ○ Dst IP

[✓] Show Packet Data [✓] Show Rule

```
alert tcp any any -> 192.168.32.0/24 80 (flags: S; msg:"Possible TCP DoS Be Careful !!";
flow:stateless; threshold: type both, track by_dst, count 70, seconds 10; sid:10001;rev:1;)
```

IP	Source IP	Dest IP	Ver	HL	TOS	len	ID	Flags	Offset	TTL	hkSu
	192.168.32.130	192.168.32.129	4	5	0	40	37938	0	0	64	929

TCP	Source Port	Dest Port	R 1	R 0	U R G	A C K	P S H	R S T	S Y N	F I N	Seq #	Ack #	Offset	Res	Window	Urp	hkSu
	1910	80	.	.	.	.	.	.	X	.	2101783304	649414863	5	0	512	0	194

DATA None. None.

○ Hex [● Text] ☐ NoCase

Figure 11. Sguil Console Output: DDoS Attack.

6. Discussion

The findings from our experiments suggest that IDS can be implemented in a Cloud-computing environment in such a way that it can detect attacks embedded inside inter-VM activities. As observed from the experiment performed on our IDS, it is realized that our proposed architecture is capable of detecting attacks aimed at a victim system. These attacks include port scanning, DoS, spoofing, malware injection and so on. Our proposed model also gives information about where an attack originates from the source port and IP address and the target of the attack i.e., destination port and IP address.

The findings made during penetration testing reveals that the vulnerabilities present in our victim system were either because of the nonexistence of system hardening, lost patches or guessable login credentials. Therefore, we recommend that OS of VMs in Cloud networks should be keept updated to their latest version which contains reinforcements to remedy the vulnerabilities we discovered. Furthermore, more rigid ones that cannot be predicted easily should replace weak login IDs.

We found that the detection capability of Snort is dependent on the number of rules enabled. This means that the more attacks will be detected if more rules are implemented. However, the downside of having a large number of rules is that there will be an equivalent increase in the number of false positive alerts as a result of rules that are intended to detect more complex attacks being triggered. We also noticed that a lot of unwanted alerts were generated. Nonetheless, these unwanted alerts could be reduced by fine-tuning and careful selection of rules. Snort rulesets are updated regularly, thereby enabling them to detect most of the current attacks.

7. Conclusions

The increase in the number of Cloud users has also brought about an increase in the number of malicious users targeting the resources available in the Cloud. Securing Cloud computing architectures from malicious users can be very complicated because it is delivered via public networks, such as the Internet. In traditional computing architectures, different types of intrusion detection techniques are used to counter malicious attacks. Although these techniques can also be implemented in the Cloud computing environment, they cannot efficiently secure Cloud infrastructures. Security of Cloud resources serves as a major bottleneck in the full adoption of Cloud computing in organizations today.

We proposed a Cloud-based IDS to help secure Cloud resources particularly in IaaS public Cloud models from cyber-attacks that are embedded inside normal VM activities. In this experiment, we have built a proof-of-concept using Security Onion and observed that well-design open-source software, such as Security Onion IDS, can effectively detect different types of cyber-attacks targeted at our victim VM (Metasploitable) in real time.

Our hypothesis is that it is possible to deploy an IDS sensor in a Cloud environment in such a way that it can detect cyber-attacks that are embedded inside normal VM operation. We have designed an architecture that follows this idea, and observed that state-of-the-art has evolved in this direction with commercial solutions from Suricata based CounterSnipe [41] and Snort IDS for Amazon EC2. Snort IDS for Amazon EC2 can be installed with support for Sourcefire's commercial vulnerability research team (VRT) rule set, which allows EC2 users to secure their Cloud with commercial support from Sourcefire. The findings from our experiments suggest that IDS can be implemented in a Cloud-computing environment in such a way that it can detect attacks embedded inside inter-VM activities.

We believe that implementing IDS inside Cloud computing architectures can have a positive effect on information technology by ensuring integrity and confidentiality of Cloud resources, and in our humble opinion, we believe that our proposed architecture is a step in that direction. We plan to enhance our work by implementing a machine learning-based IDS to decrease human intervention and reduce false alarms.

Author Contributions: Conceptualization, A.M. and B.P.; methodology, A.M. and B.P.; software, A.M.; validation, A.M.; investigation, A.M.; resources, A.M. and B.P.; writing—original draft preparation, A.M.; writing—review and editing, B.P.; visualization, A.M. and B.P.; supervision, B.P.

Funding: This research received no external funding.

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

References

1. Bohn, R.B.; Messina, J.; Fang, L.; Jin, T.; Jian, M. Nist cloud computing reference architecture. In Proceedings of the 2011 IEEE World Congress on Services (SERVICES), Washington, DC, USA, 4–9 July 2011; pp. 594–596.
2. Alashoor, T. Cloud computing: A review of security issues and solutions. *Int. J. Cloud Comput.* **2014**, *3*, 228–244. [CrossRef]
3. Xiao, Z.; Xiao, Y. Security and privacy in cloud computing. *Commun. Surv. Tutor. IEEE* **2013**, *15*, 843–859. [CrossRef]
4. Islam, S.; Ouedraogo, M.; Kalloniatis, C.; Mouratidis, H.; Gritzalis, S. Assurance of security and privacy requirements for cloud deployment models. *IEEE Trans. Cloud Comput.* **2018**, *6*, 387–400. [CrossRef]
5. Subramanian, N.; Jeyaraj, A. Recent security challenges in cloud computing. *Comput. Electr. Eng.* **2018**, *71*, 28–42. [CrossRef]

6. Hashizume, K.; Rosado, D.; Fernández-Medina, E.; Fernandez, E. An analysis of security issues for cloud computing. *J. Internet Serv. Appl.* **2013**, *4*, 1–13. [CrossRef]
7. Rosado, D.G.; Gómez, R.; Mellado, D.; Fernández-Medina, E. Security analysis in the migration to cloud environments. *Future Internet* **2012**, *4*, 469–487. [CrossRef]
8. Mather, T.; Kumaraswamy, S.; Latif, S. *Cloud Security and Privacy: An Enterprise Perspective on Risks and Compliance*; O'Reilly Media, Inc.: Sebastopol, CA, USA, 2009; p. 292.
9. Pranggono, B.; McLaughlin, K.; Yang, Y.; Sezer, S. Intrusion detection systems for critical infrastructure. In *The State of the Art in Intrusion Prevention and Detection*; Pathan, A.-S.K., Ed.; CRC Press: Boca Raton, FL, USA, 2014; pp. 115–138.
10. Pranggono, B.; Alboaneen, D.; Tianfield, H. Simulation tools for cloud computing. In *Simulation Technologies in Networking and Communications: Selecting the Best Tool for the Test*; Pathan, A.-S.K., Ed.; CRC Press: Boca Raton, FL, USA, 2015; pp. 311–335.
11. Barham, P.; Dragovic, B.; Fraser, K.; Hand, S.; Harris, T.; Ho, A.; Neugebauer, R.; Pratt, I.; Warfield, A. Xen and the art of virtualization. *Oper. Syst. Rev. (ACM)* **2003**, *37*, 164–177. [CrossRef]
12. Mell, P.; Grance, T. *The Nist Definition of Cloud Computing*; NIST Special Publication 800-145; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2011.
13. Modi, C.; Patel, D.; Borisaniya, B.; Patel, A.; Rajarajan, M. A survey on security issues and solutions at different layers of cloud computing. *J. Supercomput.* **2013**, *63*, 561–592. [CrossRef]
14. Ali, M.; Khan, S.U.; Vasilakos, A.V. Security in cloud computing: Opportunities and challenges. *Inf. Sci.* **2015**, *305*, 357–383. [CrossRef]
15. Yang, Y.; McLaughlin, K.; Littler, T.; Sezer, S.; Im, E.G.; Yao, Z.Q.; Pranggono, B.; Wang, H.F. Man-in-the-middle attack test-bed investigating cyber-security vulnerabilities in smart grid scada systems. In Proceedings of the International Conference on Sustainable Power Generation and Supply (SUPERGEN 2012), Hangzhou, China, 8–9 September 2012; pp. 1–8.
16. Asri, S.; Pranggono, B. Impact of distributed denial-of-service attack on advanced metering infrastructure. *Wirel. Pers. Commun.* **2015**, *83*, 2211–2223. [CrossRef]
17. Zarpelão, B.B.; Miani, R.S.; Kawakani, C.T.; de Alvarenga, S.C. A survey of intrusion detection in internet of things. *J. Netw. Comput. Appl.* **2017**, *84*, 25–37. [CrossRef]
18. Kaufman, L.M. Data security in the world of cloud computing. *Secur. Priv. IEEE* **2009**, *7*, 61–64. [CrossRef]
19. Cloud Security Alliance. Top Threats to Cloud Computing v1.0. Available online: https://cloudsecurityalliance.org/topthreats/csathreats.v1.0.pdf (accessed on 28 November 2018).
20. Grobauer, B.; Walloschek, T.; Stocker, E. Understanding cloud computing vulnerabilities. *IEEE Secur. Priv.* **2011**, *9*, 50–57. [CrossRef]
21. Garfinkel, T.; Rosenblum, M. When virtual is harder than real: Security challenges in virtual machine based computing environments. In Proceedings of the 10th Conference on Hot Topics in Operating Systems, Santa Fe, NM, USA, 12–15 June 2005; USENIX Association: Santa Fe, NM, USA, 2005; Volume 10, p. 20.
22. Dawoud, W.; Takouna, I.; Meinel, C. Infrastructure as a service security: Challenges and solutions. In Proceedings of the 7th International Conference on Informatics and Systems (INFOS), Cairo, Egypt, 28–30 March 2010; pp. 1–8.
23. Hanqian, W.; Yi, D.; Winer, C.; Li, Y. Network security for virtual machine in cloud computing. In Proceedings of the 5th International Conference on Computer Sciences and Convergence Information Technology, Seoul, Korea, 30 November–2 December 2010; pp. 18–21.
24. Scarfone, K.; Mell, P. *Guide to Intrusion Detection and Prevention Systems (idps) Nist 800-94*; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2007.
25. Bakshi, A.; Dujodwala, Y.B. Securing cloud from ddos attacks using intrusion detection system in virtual machine. In Proceedings of the 2010 Second International Conference on Communication Software and Networks, HongKong, China, 26–28 February 2010; pp. 260–264.
26. Srivastava, A.; Giffin, J. Tamper-resistant, application-aware blocking of malicious network connections. In Proceedings of the 11th International Symposium on Recent Advances in Intrusion Detection, Cambridge, MA, USA, 15–17 September 2008; Springer: Cambridge, MA, USA, 2008; pp. 39–58.
27. Pfoh, J.; Schneider, C.; Eckert, C. A formal model for virtual machine introspection. In Proceedings of the 1st ACM Workshop on Virtual Machine Security, Chicago, IL, USA, 10–12 March 2009; ACM: Chicago, IL, USA, 2009; pp. 1–10.

28. Jiang, X.; Wang, X.; Xu, D. Stealthy malware detection through vmm-based "out-of-the-box" semantic view reconstruction. In Proceedings of the 14th ACM Conference on Computer and Communications Security, Alexandria, VA, USA, 29 October–2 November 2007; ACM: Alexandria, VA, USA, 2007; pp. 128–138.
29. Takabi, H.; Joshi, J.B.D.; Ahn, G. Security and privacy challenges in cloud computing environments. *IEEE Secur. Priv.* **2010**, *8*, 24–31. [CrossRef]
30. Subashini, S.; Kavitha, V. A survey on security issues in service delivery models of cloud computing. *J. Netw. Comput. Appl.* **2011**, *34*, 1–11. [CrossRef]
31. Zissis, D.; Lekkas, D. Addressing cloud computing security issues. *Future Gener. Comput. Syst.* **2012**, *28*, 583–592. [CrossRef]
32. Sabahi, F. Virtualization-level security in cloud computing. In Proceedings of the 2011 IEEE 3rd International Conference on Communication Software and Networks, Xi'an, China, 27–29 May 2011; pp. 250–254.
33. Oliveira, D.A.S.D.; Crandall, J.R.; Wassermann, G.; Ye, S.; Wu, S.F.; Su, Z.; Chong, F.T. Bezoar: Automated virtual machine-based full-system recovery from control-flow hijacking attacks. In Proceedings of the NOMS 2008—2008 IEEE Network Operations and Management Symposium, Salvador Da Bahia, Brazil, 7–11 April 2008; pp. 121–128.
34. Burks, D. Security Onion: Peel Back the Layers of Your Network in Minutes. Available online: http://resources.sei.cmu.edu/asset_files/Presentation/2014_017_001_90218.pdf (accessed on 23 November 2018).
35. Offensive-Security. Kali Linux. Available online: https://www.kali.org/ (accessed on 23 November 2018).
36. Moore, H. Metasploitable 2 Exploitability Guide. Available online: https://community.rapid7.com/docs/DOC-1875 (accessed on 23 November 2018).
37. Snort. Network Intrusion Prevention and Detection System (ids/ips). Available online: http://www.snort.org (accessed on 23 November 2018).
38. Sguil. Sguil: The Analyst Console for Network Security Monitoring. Available online: http://bammv.github.io/sguil/index.html (accessed on 23 November 2018).
39. Nmap. Network Mapper. Available online: https://nmap.org/ (accessed on 23 November 2018).
40. Security, T.N. Nessus. Available online: https://www.tenable.com/ (accessed on 23 November 2018).
41. CounterSnipe. Countersnipe: Intrusion Detection and Prevention (ids/ips) Software. Available online: http://countersnipe.com/index.php/products (accessed on 23 November 2018).