EICIDS-Elastic and Internal Cloud-based Detection System

Josenilson Dias Araújo[1], Dhileane de Andrade Rodrigues[1], Leonardo Silva de Melo[1] and Zair Abdelouahab[1]

[1]Federal University of Maranhão-UFMA, Brazil
jdias @ifto.edu.br, dhully25 @gmail.com, leonardomelo91 @gmail.com, zair @dee.ufma.br

Abstract: The elasticity and abundant availability of computational resources are attractive to intruders exploit cloud vulnerabilities and were able to launch attacks against legitimate users to gain access to private and privileged information. The Intrusion Detection Systems are presented as a possible solution for protection; however, to effectively protect the cloud users, IDS should have the ability to expand rapidly by increasing or decreasing the amount or sensors, the measure of cloud resources are available, and isolating the access to infrastructure and the system levels. Protection against internal threats should also be planned, as most protection systems do not identify them correctly.

In order to solve these problems, we present the EICIDS - Elastic and Internal Cloud-based Intrusion Detection System, which monitors the internal cloud environment, entering data capture sensors on the local network of user´s VMs, and therefore, able to detect suspicious behavior of users. For this, the EICIDS uses the characteristics of virtual machines such as fast boot, fast recovery, stop or pause of VM, migrate between different hosts and execution across multiple platforms, to monitor and protect the cloud computing environment and keep up with the growth or reduction cloud, in order to save resources.

Keywords: Cloud Computing system, intrusion detection, virtual machines, elasticity, internal threats**.**

1. Introduction

Cloud computing is emerging as a new computing paradigm, proposing a new way of using computing resources such as software and hardware, through the pay per use, consumption on demand and access to these services over the Internet. This paradigm allows the consumer or client, who can be a company or a single user, purchase services from a provider and pay according to what it consumes, outsourcing software development and server administration.

Cloud computing is here to revolutionize the way of providing IT resources, freeing users of any complication with the infrastructure providing computing resources as they are needed, i.e., abstracting all the infrastructure and providing on-demand services. However, the shift to this new paradigm brings several challenges, including concern about the security of stored data and the integrity of the cloud environment. Such concern is justified by the increased number of reported incidents in the cloud [33].The survey in [33] shows an increase of incidents in the cloud over the past 5 years, where we can observe a frightening increase. Over the years, research results make evident the enormous growth of incidents in the cloud, and in 2009 met an incident-level slightly above 40 % and in 2011 these incidents almost doubled [33]. This alarming increase in the number of incidents in the cloud, according to [34], is due to the growth of the deployment models of this new approach. Concern about security is cited as the main barriers to the use of cloud computing by institutions [3], according to the research conducted by International Data Corporation IDC1 in 2009 [35]. Therefore, it is important to develop appropriate security mechanisms that can detect and respond to malicious actions that constitute an intrusion.

Protection mechanisms such as Intrusion Detection Systems (IDS) are very important to compose a secure environment. These mechanisms focus on the detection of intrusive network [4] activities. However, due to the elasticity and the distributed architecture of the cloud, the IDS need to be adapted to effectively protect the new environment.

To effectively monitor a cloud environment which increases the amount of demanded resources, the IDS must be dynamically expandable. It must follow the expansion of the cloud, by removing or redirecting users to computing resources according to consumption demand.

For this purpose, this paper proposes the EICIDS - Elastic and Internal Cloud-based Intrusion Detection System, which is based on protection of virtual machines in cloud computing environment against internal users who can use some VMs to perform malicious activities. EICIDS uses characteristics of virtualization as insulation, quick stop, fast startup, and elasticity to protect VMs from attacks that can be performed from within the cloud environment; more specifically from a compromised VM.

Monitoring of virtual machines is done by IDS sensors scattered in the cloud environment, and the instantiation of these sensors is made in each VM, where the packets passing in VMs are captured and subsequently analyzed for the identification of threats.

Thus, the entire virtual environment is monitored, while the remaining components of EICIDS reside outside the virtual environment are thus protected from possible attacks by compromised VMs.

The use of elastic and dynamic monitoring for IDS can allow us to accomplish some important tasks, such as checking for active VMs, virtual network traffic, detecting intrusions and malicious actions performed by a VM. EICIDS can save the resources of the cloud and guarantee the safety of the environment.

This paper is organized as follows. Section 2, 3 present the main concepts in cloud computing, and a summary of intrusion detection systems, respectively. Section 4 describes related works. In section 5, we present EICIDS architecture. Section 6 presents the implementation, testes, and results of EICIDS. Finally, conclusions and future work are given in section 7.

2. Cloud Computing

Cloud computing can be defined as an infrastructure for data processing in which applications, data and processing power are available remotely via internet. Cloud computing offers a new way of using IT resources, providing users with access to a set of applications and services, while it abstracts the complexities involved in delivering them. The change in location of data and programs, leaving the user's desktop to the cloud, provides a geographical shift in computing, where processing is performed remotely in data centers that provide a pool of IT resources such as processing power, storage and software to all users on demand [5].

Since the model is a new, cloud computing still has its specifications and standardized definitions, resulting in various definitions and concepts. One of the most widely accepted definitions by several researchers is provided by NIST (National Institute of Standards and Technology). It defines cloud computing as a model that allows convenient access on-demand to a set of configurable computational resources (e.g., networks, servers, storage, applications and services) that can be acquired and released with minimal management effort or interaction with the service provider [6].

Another definition is that the cloud can be seen as a distributed computing model that derives from grid computing, with respect to the provision of information on demand for multiple concurrent users and is used with virtualization of resources through services to users that need only a browser and internet connection to consume such resources [7]. Thus, virtualization has become a key element for providing a set of IT resources (storage software, processing power, etc.) as services to users who need web access only to use them [8].

Cloud computing can be conceived as a layered model where the basic structure of the cloud have the physical infrastructure consisting of servers, network equipment and operating systems. Right above is a layer of virtualization and virtual machines. Then, we have a layer formed of operating environments and development programs running on virtual machines and finally a layer of software available to end users.

NIST defines a model of cloud computing composed of five essential characteristics, three service models, and four deployment models [6].

The main characteristics of cloud computing are self - service on-demand or on-demand self-service, resource virtualization, location independence with access to Internet resources, elasticity and payment model based on consumption [6], [8]. Cloud resources are made available to users according to one of the three service models:

- **IaaS** (Infrastructure as a Service): It provides resources such as processing, storage and network connectivity through virtualization. Servers, storage systems, routers and other equipment are made available to manage a workload required by applications.
- **PaaS** (Platform as a Service) is the execution platform, distribution and application development. It refers to a level of abstraction above the IAAS in which the cloud provides an applicative programming platform. The PaaS allows programming of easily manageable applications in the Cloud.
- **SaaS** (Software as a Service): the Cloud provides applications directly to users according to their expectations.

The deployment models depend on how resources are organized and how they are made available to users can be classified as:

- Public Cloud where access is available to the general public and may be owned and managed by a private, academic, governmental, or any combination of them.
- Private cloud: Physical resources are provisioned for exclusive use by a single organization.
- Community Cloud: It is controlled by several customers that come together to form a cloud that meets their specific needs, particularly in terms of control, security and compliance.
- Hybrid Cloud: The cloud infrastructure is made up of any combination between different infrastructures (private, public or community), making it a single cloud.

With the increase in use of more computational resources, especially hardware and software in various areas, cloud computing presents itself as a good alternative for businesses by providing IT services based on usage payment. The big advantage in using cloud computing is to enable hiring new features as they become necessary, reducing costs in infrastructure and maintenance [2].

3. Intrusion Detection Systems

An IDS or intrusion detection system aims to improve security in a computer network. IDS include monitoring processes, identifying and reporting instances of suspicious or malicious activity. An IDS tries to recognize a behavior or an intrusive action to alert an administrator or automatically trigger counter-measures [9], working with the operating system or a network of computers trying to identify malicious activities. It acts as a security tool, in the same fashion as others such as antivirus, firewalls and access control systems, designed to enhance the security of information and communication systems [10].

An intrusion detection system is composed of three components that have functions to collect, analyze and display information about the system [11]. These components are:

- Sensor: It has the function of capturing data information traffic;
- Analyzer: it is the main component of the IDS. Its goal is to analyze the data obtained by the sensor, through the analysis process, then give an alert in case of intrusion;
- User Interface: It provides a structured data collected and analyzed for the administrator.

According to the classification schemes [12], IDS can use: signature-based detection and anomaly-based detection. The signature-based detection identifies patterns of known attacks for possible intrusion attempts. These signatures are formed by a set of rules characterizing the attacker, having the advantage of an almost immediate detection and preventing the occurrence of false positives. The anomaly detection is based on classifying activities outside the standard (or normal) behavior as anomalies in network traffic [10] [13], and abnormal system behavior, identifying suspicious activity by presenting deviant behavior, which could indicate the presence of malicious activities in the network. The advantage of this method is the detection of unknown threats; however, it can produce a high rate of false positives due to the unpredictable behavior of users.

IDSs can be further classified according to the data collection:

- Network based Intrusion Detection Systems (NIDS): The IDS captures the data of network traffic for analysis, looking for signatures of known attacks and anomalies in the network activities in monitored system. It is located in a spot with full visibility of network traffic monitor.
- Host-based Intrusion Detection System (HIDS): An IDS monitors the local activity of the host using the event log file for analysis. . It can use log files of operating system events or databases for analysis. The goal is to identify attacks and attempt of unauthorized access to the machine itself. In this case, the IDS is located on the machine monitor.
- Hybrid based Intrusion Detection System (HBIDS): it uses a combination of sensors to capture network traffic and read event logs from the host to analyze and detect any malicious activity. According to [16], most IDS systems analysis utilize network and host analysis.

According to the classification schemes of IDS in [4], [20] and [21], the architecture can be:

- Centralized: It consists of sensors and analyzers elements in which the main task is concentrated in a single element, thus presenting the disadvantage of a single point of failure. However, a positive point is that it provides an easy administration;
- Hierarchical: It is formed in layers where the upper layers delegate tasks to their subordinate. The IDS may be arranged into a layer of data capture, analysis and a coordinator which is responsible for the management. A failure in one layer can compromise one part of the system.
- Distributed: The IDS has components that are distributed and work in cooperation to achieve detection without the presence of a central manager.

4. Related Work

The characteristics of virtualization provide greater coverage and resilience against threats within IDS solutions for cloud computing environments. In order to adapt to the characteristics of cloud computing, some works have been proposed.

4.1 IDSaaS: Intrusion Detection System as a Service in Public Clouds

IDSaaS (Intrusion Detection System as a Service in Public Clouds): The proposal of [14] is to provide scalable and adjustable IDS to users of cloud services by providing the ability to monitor and react to attacks on several existing VMs in a virtual private network. The IDSaaS is implemented using EC2 (Elastic Compute Cloud) of Amazon web service [15]. IDSaaS creates a virtual network environment for user's services through Amazon VPC (Virtual Private Cloud) where instances of EC2 VM type are created to store and execute security components for the infrastructure level (IaaS) and letting the detection mechanisms completely controlled by users. IDSaaS is an IDS to detect subscription-based model and uses the network as a source of data collection. It is scalable, portable, on-demand, user-controlled and available through pay-per-use model. The IDS targets the level of cloud infrastructure, where its first task is to monitor and record suspicious activity in the network between virtual machines within a pre-defined virtual public cloud. The IDSaaS is built and packaged as AMI format (Amazon Machine Images) components and the VPC service is used to create two subnets. One is public where the VMs reside with IDSaaS, and the other is private where business applications are kept protected.

4.2 GCCIDS: Intrusion Detection for Grid and Cloud Computing

[16] Proposes an intrusion detection system for grid computing and cloud environments. The system is implemented at the level of cloud middleware for preventing against intrusions by the insulating characteristics of virtualization. The IDS has a distributed architecture where each node of the cloud is monitored by a part of the intrusion detection system. When an attack occurs, an alert is sent to inform other nodes in the environment. The system captures and audit information logs, and offers a middleware with secure communication between each node of the environment (grid/cloud). GCCIDS uses two detection techniques: behavior based intrusion detection that works with an artificial neural networks of type feed-forward and intrusion detection based on knowledge to identify known attacks.

4.3 Intrusion Detection System in Cloud Computing Environment

In [17], an IDS architecture is proposed an architecture to create separate instances for each user of the cloud and uses a single controller for management purposes. The IDS is composed of instances classified as "mini IDS" created by a central IDS controller. The instances are deployed between each user of the cloud service provider. According to the authors, the main advantage is the reduction in workload because it is split between multiple instances to carry the work in a better way rather than letting a single IDS for the whole cloud. Thus, whenever a user wants to access cloud services an instance of the IDS is provided by IDS Controller with the responsibility of monitoring and achieving protection.

4.4 CIDS: A Framework for Intrusion Detection in Cloud Systems

A framework for cloud-based IDS is proposed in [18]. The aim is to deal with attacks like masquerade (where intruders pose as legitimate users), Host-based attacks (which may be a consequence of masquerade attack) and Network-based attacks. CIDS summarizes intensive network IDS alerts by sending summary reports to the administrator of the cloud.

The solution works in the cloud middleware level using its mechanisms, such as the messaging system and memory insight. CIDS monitors and protects the runtime environment for users (residing in virtual machines), it also maintains its own components protected from threats that may affect the VMs, because the components of the CIDS are located outside of the virtual machines. This protection feature is made possible by isolation of VMs.

4.5 An Extensible and Virtualization-Compatible IDS Management Architecture

The work done in [19] presents an extensible IDS architecture that consists of multiple sensors embedded in virtual machines controlled by a central component that analyzes the collected results. Different IDS sensors can be activated and communicate with each other using messages of the standard IDMEF (Intrusion Detection Message Exchange Format). The architecture is based on characteristics of virtual machines, such as isolation, fast recovery in case of compromise, and uses the IDMEF standard for the exchange of messages between the sensors and the central unit management. The use of the IDMEF provides standardization in information alerts, and allows the use of different IDS sensors. Each IDS sensor component consists of sensors embedded in virtual machines. Users can control the management of the IDS directly by interacting and configuring the core components.

The structure of VM IDS management system consists of the following components:

- Virtual machines called VM IDSs that host the sensors and IDS Event Gatherer.
- The central management unit that controls the VM IDS which has four components: Gatherer Event, Event Database, Component Analysis and Remote Controller IDS.

5. EICIDS

This paper proposes an IDS architecture called EICIDS that aims to protect a cloud environment against malicious users and save cloud´s resources. The main contribution of this paper is providing a mechanism of alert e protection against internal threats conducted by malicious users inside the cloud environment. Another contribution is to save the cloud resources by inserting IDS sensor only where they are needed; for this an EICIDS component, called IDS-Pool, provides the information of where IDS sensor must be instantiated, according to user´s VMs are created and made available. EICIDS provides security for users at the cloud infrastructure level, by monitoring virtual machines that makes up the IaaS layer and also that underpins the other layers (PaaS and SaaS). To protect VM users, research results on the major threats in cloud computing done by CSA [20] were used as a basis for the design of EICIDS; specifically threats such as abuse, transgression use, sequestration of accounts services, and malicious traffic. Research on surface attacks in cloud computing performed by [21] also served as a basis for constructing the IDS. In this study, we learned that through the user interface provided by the service provider or through an unauthorized or even access abuse of the infrastructure resources, attacks can be run against services running on the provider's infrastructure in order to stop the services or gain access to information. Therefore, a potential threat in cloud computing environments is part of the running VMs; VMs may or may not perform activities considered suspicious or malicious, which depends on the profile of the account owners in the cloud provider. The proposed architecture is shown in Figure 1 and in Figure 2 we have a preview of architecture with many nodes controlled by IDS_Admin.

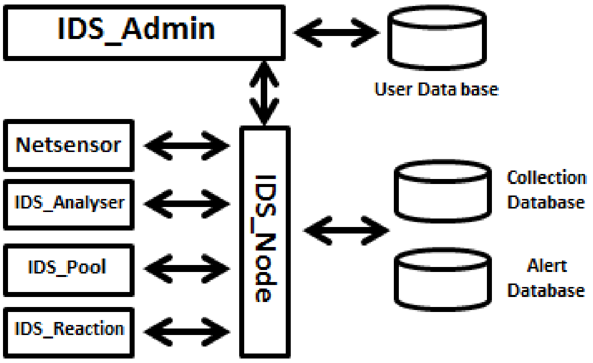


Figure 1. Architecture of the Proposed IDS (for one node)

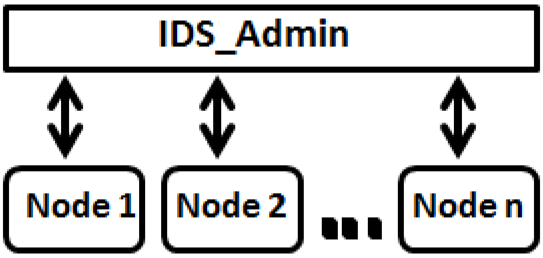


Figure 2. Architecture of IDS (with multiple nodes)

Figure 1 shows the components of EICIDS and their interactions with each other. Such components are distributed through the cloud infrastructure. Every Node of IaaS is associated with a set composed from IDS_Node, Netsensor, IDS_Analyser, IDS_Pool and IDS_Reaction. Each IDS_Node centralizes the information in its Controller and then sends it to the IDS_Admin, located in the IaaS Cloud Controller.

The IDS_Admin acts as the administrator of the full IDS derived from each Node Controller infrastructure, receiving information from each IDS_Node present in its respective Node. Figure 2 shows the implementation of IDS_Admin as an administrator of all IDS modules, which are on each node, having the function of informing administrators of the cloud about the events that occurred and the system status with regard to the maintenance of security.

5.1 Operation of EICIDS

The EICIDS combines a centralized and hierarchical structure where a component called IDS_Admin initializes the modules in each node of the cloud architecture. A local component, called IDS_Node, is instantiated on each node of the cloud and has the function of monitoring user’s virtual machines. Tracking is done by IDS sensors inserted into virtual machine, to capture Information of VLANs traffic. With this information, a local analysis is performed by signature. If a pattern of attack is found an alert is sent to the central controller and a countermeasure is done (shutdown the VM, for example). In case no attack is found, the information is forwarded to the node located in the cloud for a complete analysis by anomaly with NIDIA [21]. Instantiation of IDS sensors are made as according to the amount of VMs. The activity diagram of Figure 3 shows the activities performed by EICIDS. Initially, Netsensor captures frames of VM traffic and sends them to IDS_Analyser to perform the analysis of the IP header of each packet, by comparing it with patterns of predetermined attack (detection by signature). If an attack signature is found in a given packet header, an alert is sent to IDS_Node with some information on the attack such as source and destination addresses in order to take necessary countermeasures, and sending and alerting IDS_Admin on the event. If a pattern of attack is not found in the packet header, the analysis by anomaly is performed.

5.2 EICIDS Components

The components operations of EICIDS are presented below.

5.2.1 IDS_Admin

IDS_Admin is responsible for managing the process of intrusion detection by integrating and controlling all components of EICIDS as well as organizing, and summarizing alerts issued by various nodes of the cloud. This component maintains information about the behavior of the IDS across the cloud, shows an overview of system behavior, and provides a functional interface for the administrator of the cloud. It has also the function of identifying VMs involved in security incidents using user’s accounts database of the cloud.

5.2.2 IDS_Node

IDS_Node monitors the virtual environment by instantiating sensors and analyzers for the virtual machines. It is responsible for receiving security alerts from signature analysis and directs them to IDS_admin. At the end of the signature analysis, the IDS_Node forwards the captured packets for anomaly analysis in order to find suspicious traffic, unwanted events or patterns that indicate intrusion. According to information received by the IDS_Pool, it periodically checks whether or not the VMs are running; it may instantiate or remove sensors and analyzers of the virtual environment in order to save resources of the cloud environment (elasticity control).

5.2.3 IDS_Pool

IDS_Pool monitors the virtual environment searching for active virtual machines. When is finds a new running VM, it sends a message to IDS_Node so that EICIDS sensors are activated. Similarly, when no virtual machine is found, a message is sent to IDS_Node to disable the sensors, thus saving resources of the cloud.

5.2.4 Netsensor

The process of detection starts from this component. It has the function of capturing the packets traffic of the users VMs of the cloud. A sensor interacts with the network passively, acting as a sniffer and interfering as little as possible, for not compromising the performance of VMs and not corrupting the traffic. In addition to the function of capturing, filtering packets is done by selecting only the relevant information for analysis. Packets are stored in a database and are made available for analysis by both signature and anomaly methods. Due to cloud computing feature of resource allocation, the issue of privacy of customers should be taken into consideration because privacy laws may prevent monitoring data or resources used by cloud customers, even for purposes of inspection and security. According to [22], privacy laws may hinder the process of adding security measures to Cloud Computing. Thus, the captured data by the sensor do not go against the laws of privacy because only the information contained in the IP packet header are stored for later analysis by other components of EICIDS.

5.2.5 IDS_Reaction

This component is responsible for taking countermeasures when a malicious action is detected.

The measures range from a pause of a VM with a suspicious behavior, a shut down or a destruction of a malicious VM. Upon finding a malicious activity, the reaction component is triggered by the node agent, to undertake an appropriate action.

5.2.6 Signature Analysis

The signature analyzer is located in the virtualized environment created by EICIDS. It is responsible for analyzing the data generated by the sensor, applying detection rules, thus able to identify malicious actions of any VM before sending the information to IDS_Node. If a pattern of attack is found, an alert message is sent to IDS_Node, if a malicious action is not detected, the captured packets are also sent to IDS_Node for a more detailed analysis. Due to privacy laws, only the content of the IP packet header is analyzed, where patterns of known attacks, such as portscan attacks are targeted. Attacks based on header protocols make use of vulnerabilities of the TCP/IP stack and can be identified by the use of a packet filter or specific content rules header by the IDS.

5.2.7 Anomaly Analysis

The anomaly analyzer is located outside the virtualized environment, specifically in IDS_Node. It is responsible for a new analysis of the data generated by the sensor and forwarded by the signature component analysis. An alert is issued to the node in case of a confirmation of anomalous behavior in a VM. The analysis is based on the variation of the amount of packet types caught in the net. According to the study by Zaidi [23], and also applied in [24], the variation of the packet types captured between values defined in a lower limit and an upper limit will be used as the basis to determine the status of network traffic. Traffic is considered normal if it is below the threshold; abnormal, if greater than the upper limit, and is considered a suspect in case it is between the lower limit and upper limit.

6. Implementation, test and results

EICIDS combines the tasks of monitoring the network environment of VMs, and signature and anomaly analysis to detect internal attacks to VMs hosted in the cloud [22]. Monitoring is based on collecting information from targeted VMs traffic in promiscuous mode.

To demonstrate the operation of EICIDS, we have simulated only DoS attacks (Denial of Service), but it can also be extended to other types of attacks [1]. EICIDS is implemented using Java [25] and is represented by the following classes: Netsensor, IDS_Analyser, Start_IDS, IDS_Node, IDS_Admin, IDS_Pool and IDS_Reaction.

To capture packet frames, jpcap is used [26] to implement the functionality of the libpcap library [27]. The set of classes, methods and attributes used in EICIDS are viewed in the class diagram shown in Figure 4.

In the diagram, we can see the relationship between 1 to n IDS_Admin and IDS_Node classes, in a cloud environment. IDS_Admin is responsible for managing, maintaining and coordinating various IDS_Node. To demonstrate the functionality of EICIDS, the classes Nestsensor, IDS_Anlyser, IDS_Node, IDS_Pool, start_IDS and IDS_Admin are partially implemented.

6.1 Test Environment

To perform the tests, an environment is created in LABSAC (Laboratory of Systems and Computer Architectures) UFMA (Federal University of Maranhão). The test environment for EICIDS simulates an IaaS cloud infrastructure, based on the use of two machines connected in a network as shown in Figure 5.

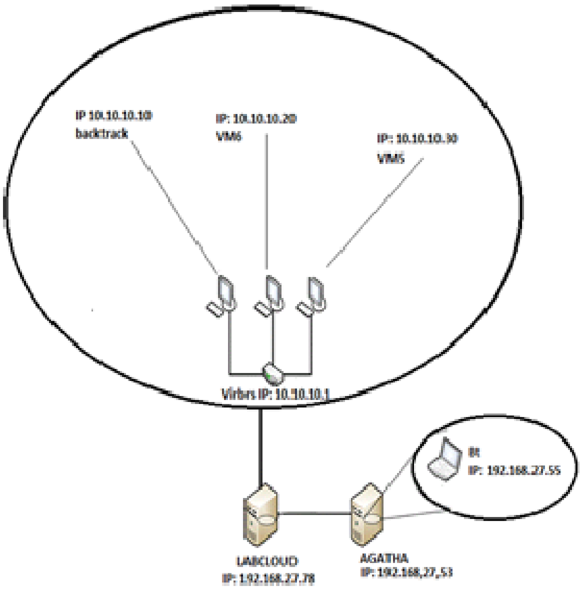


Figure5. Testing environment EICIDS

The machines that keep the operating environment are named LABCLOUD and AGATHA. The LABCLOUD host acts as the main virtualization system and has the function of Node Controller in the context of the cloud environment. It possesses installations of Linux Ubuntu Server 12.04.3 [28], KVM virtual machine monitor [29] and the libvirt virtualization management library [30] in order to permit virtualization operations similar to the ones of IaaS. The AGATHA host has the administrative VM virtualization named bt running VirtualBox virtualization manager, version 4.2 [31]. The bt VM works as the Admin module contained in AGATHA host which acts as the Cloud Controller, providing an interface for the administrator of IaaS, showing all events coming from each node of the cloud. The IDS suit any number of nodes available in the cloud, because it is deployed and distributed over the VMs. In our case above, we have only one node, represented by the network address 10.10.10.0 and gateway 10.10.10.1, achieved by the virtual network interface virbr5. Such bt VM connects directly with IDS_Node contained in LABCLOUD machine, getting the necessary alerts. If there are more nodes in the environment, all IDS_Node of each Node Controller would be connected to the VM bt to get information from all nodes.

In LABCLOUD host, representing the Node Controller, it is created a virtual network managed by KVM, segmented by virbr5 router, managing VMs backtrack, VM5 and VM6 with their IPs contained in Table 1.

Table 1. Virtual Machines Addresses IP

Virtual Machine	IP Address
Bt	192.168.27.55
Backtrack	10.10.10.10
Vm6	10.10.10.20
Vm5	10.10.10.173

In our first test, the backtrack VM is the virtual machine that suffers a masquerading attack and it is running under the administration of a malicious user that passes for a legitimate user, and performs attacks in order to test the vulnerability of the environment. The attacker runs a port scan, using the methodology of sending ICMP request with the response to all hosts on the network or sending to the broadcast address of the network, and running attacks in the context of denial of services such as Synflood and Smurf.

As "victims" of the attacks, we have two instantiated virtual machines VM6 and VM5 which are in the same network as the malicious VM, and are administered by legitimate users. However, these VMs are purposely running and showing vulnerability to attacks in order to demonstrate the functionality of EICIDS.

6.2 Tests and Results

First we initialize the administrative VM bt, running on the AGATHA host (Cloud Controller) and then the components of the IDS LABCLOUD host (Node Controller), which are the IDS_Node, the Start_IDS, backtrack VM, VM5 and VM6. We have used the backtrack VM to perform network attacks within the virtual environment, having VM5 and VM6 as the victims. The environment is organized as shown in figure 6. It shows the administration interface of LABCLOUD to view the status of VMs environment, the VM5 user interface, the initialization interface of Start_IDS, the backtrack user interface, the user interface of VM6 and

the IDS_Node interface to perform local administration of the components of EICIDS.
To do the tests, we have used the hping program [1], a network tool for sending TCP / IP packets for any target network address. With this tool, some commands are executed simulating malicious traffic against virtual machines hosted in the prepared environment.

6.2.1 Test 1: "synflooding"

We realize the first attack using the command hping3 -V -c 1000000 -d 120 -S -w 64 -p 135 -s 135 --flood -a 10.10.5.5 10.10.10.173. This command simulates an attack of type SYN Flooding, sending packets with tcp syn flag enabled, with the destination IP address of the victim (10.10.10.173) and with an invalid origin address that do not belong to the virtual network test environment. The parameters used in the command are:

- -v -- verbose;
- -c 1000000 -- sends a million packets;
- -d 120 -- data size;
- -S -- activates the SYN flag;
- -w64 -- tcp window;
- - p 135 - s 135 -- indicates the TCP port of origin and destination;
- - flood -- sends packets as fast as possible;
- -a -- masks the IP source address.

When executing the command from the VM attacker, the IDS shows a warning message on the console IDS_Admin, as illustrated in Figure 7.

6.2.2 Test 2: "probattack"

As a second test, we do a scan host or realize a "Ping of Death" which we call "probattack". This test has the following sequence of events:

1. Issue the command "# ping 10.10.10.20" to backtrack to VM6 (Figure 8);
2. The ICMP packets are captured and analyzed. The IDS takes countermeasures such as paralyzing the backtrack VM and inclusion of the current user name (malicious user) in the blacklist (Figure 9 and Figure 10);
3. A warning is issued in bt which is the system administrator. Figure 11 illustrates this situation showing the hostname of a malicious backtrack VM as suspended and the username of the malicious user inserted in the blacklist.

The following test3 and test4 are concerned with smurf attacks. We illustrate a ping for an ordinary broadcast showing a scan of hosts in the network and a ping broadcast in conjunction with IP Spoofing, which is the Smurf attack itself. Since the source IP address belongs to the virtual network there is no guarantee that such address belongs to the same machine which originated the ping. For this purpose, all broadcasts of ping are classified as suspicious and labeled as "smurf".

6.2.3 Test 3: "smurf" on broadcast ping

The third test is intended to verify the provenance of a host sending of a ping broadcast showing that this command is considered as suspicious even if it is not exactly a Smurf attack. We have the following sequence of events for this test:

1. Issue the command " ping 10.10.10.255 # - b " from backtrack for a broadcast (Figure 12). The "10.10.10.255" is the broadcast address of the network, and the option "-b " indicates the sending of ICMP packets for a broadcast.
2. The ICMP packets are captured and analyzed. The IDS takes countermeasures such as paralyzing the backtrack VM and inclusion of the current user name (malicious user) in the blacklist. (Figure 13).
3. A warning is issued to bt (system administrator). Figure 14 illustrates this situation showing the hostname of a malicious backtrack VM as suspended. In this particular case, it was not possible for the system to confirm that the IP address of the VM suspect is the same as the one in the ICMP packet header and as well as the name of the malicious user.

6.2.4 Test 4: "smurf" itself

The test refers to the characteristic of EICIDS ability to detect victims of the Smurf attack and enable the network administrator with the options: (1) migrate the VM to a quarantine zone for possible forensic analysis, (2) contact and alert the victim or (3) restart the VM.

However, EICIDS is unable to determine whether the VM related with the threat is only performing a scan of IP addresses within the virtual network, or realizing a broadcast ping or itself is a victim of a Smurf attack. For this purpose, it is necessary that the VM is moved to a safe area and put the responsibility of this VM directly into the hands of the network administrator.

Nevertheless, we have the following sequence of events for this test:

1. Issue the command " # hping3 to 10.10.10.20 10.10.10.255 -i -1 1" from backtrack for broadcast to see the answer in VM6 (Figure 15). This command uses the hping3 tool [42], which allows editing of the header and control of transmission of ICMP packets. The backtrack VM sends several ICMP echo request packets to the broadcast address (10.10.10.255), falsifying the source address of VM6 (10.10.10.20), which will be the victim of the attack, receiving scores (flooding) of ICMP echo reply packets as a response.
2. The ICMP packets are captured and analyzed. The IDS takes countermeasures such as paralyzing the backtrack VM (Figure 16) and inclusion of the current user name (malicious user) in the blacklist. (Figure 17).
3. Issue of a warning to bt VM (Figure 18) indicating the hostname of the VM (VM6) as suspended and identifying the user name of this VM (cloud user).

6.3 Evaluation of results

The tests with EICIDS in a virtual cloud computing infrastructure have showed and proved the functions of identifying attacks and taking appropriate countermeasures.

Table II presents a comparison of EICIDS with the main IDSs cloud presented in section 4. We can observe that the use of virtualization in EICIDS properties is in order to monitor the elasticity of the cloud environment, thus saving resources when required, while in other research works this property is used to increase the capacity of the IDS itself, not worrying about a possible increase in consumption of cloud resources.

7. Conclusion

In this work, a model for intrusion detection in cloud computing environments, aiming to protect users from internal threats is proposed. The proposed architecture of EICIDS been described with some detailed components operation and the methods used to detect attacks. Studies conducted in the CSA [30] in security risks in cloud computing, and specifically items of: abusive or transgressed use of cloud computing, accounts sequestration, services and traffic as well as the work of [31] on surface attacks were used as guidelines for the operation and implementation of the proposed solution. EICIDS uses characteristics of virtualization as insulation, quick stop, fast startup and elasticity to protect virtual machines against attacks that can be performed within the cloud environment, more specifically from a compromised VM. The instantiation of sensors is made for each VM, where packets are captured and subsequently analyzed for identification of threats, thus monitoring the entire virtual environment. The remaining components of EICIDS reside outside the virtual environment, and thus protected from attacks of compromised VMs. Another contribution of this work is the use of IDS_Pool component that constantly checks the virtual environment running VMs users, promoting a more efficient use of cloud resources by informing other components to instantiate sensors to capture packets when necessary thus saving resource consumption by the IDS. The results obtained from the tests have confirmed the ability of EICIDS to identify intrusive actions and then takes countermeasures. In the tests, all attacks from a malicious VM were detected by EICIDS despite that only DoS attacks were used. The proposed model allows identification of other types of attacks by implementing specific types of attacks detection mechanisms.

With the development of this work, some possibilities for future work were identified, among which we can suggest:

- The actual architecture of EICIDS is centralized (IDS_admin), this it is not tolerant to failure. We are extending EICIDS with some mechanism of fault tolerance.
- Adapt IDS_Admin module for identification of cloud users using VMs to carry out attacks;
- Provide the solution as a service, operating in the SaaS cloud layer;
- Development of a system for the generation of attack signatures by means of honeypots, working in conjunction with the EICIDS;
- Adapt EICIDS for mobile devices;

8. Acknowledgments

The authors acknowledge CAPES, CNPq and FAPEMA for financial support.

References

[1] B. Kodada, P. Gaurav, and R. Alwyn. Pais. "Protection Against DDoS and Data Modification Attack in Computational Grid Cluster Environment." International Journal of Computer Network and Information Security (IJCNIS), Vol.4, No.7, pp. 12-18, (2012).

[2] N. Jeyanthi, and N. Iyengar. "Packet resonance strategy: a spoof attack detection and prevention mechanism in cloud computing environment." International Journal of Communication Networks and Information Security (IJCNIS), Vol 4, No 3, pp. 163-173, (2012).

[3] N. Iyengar, B. Arindam, and G. Gopinath. "A Fuzzy Logic Based Defense Mechanism against Distributed Denial of Services Attack in Cloud Environment." International Journal of Communication Networks and Information Security (IJCNIS), vol.6, No.3, pp. 233-245, (2014).

[4] Movaghar and F. Sabahi, “Intrusion detection: A survey,” In Systems and Networks Communications. ICSNC'08. 3rd International Conference on, pages 23-26. IEEE, 2008.

[5] F. Sousa, L. Moreira, J. Machado, “Computação em nuvem: Conceitos, tecnologias, aplicações e desafios,” Escola Regional de Computação do Ceará, Maranhão e Puauí (ERCEMAPI) 2009 : Edufpi,” pp. 150-175 .

[6] P, Mell and T. Grance. “The NIST Definition of Cloud Computing. National Institute of Standards and Technology,” 53 (6) : 50, 2009.

[7] F. Armand, et al. “Above the clouds: A berkeley view of cloud computing,” Dept . Electrical Eng and Comput . Sciences, University of California, Berkeley, Rep. UCB / EECS, 28:13, 2009.

[8] L. Vaquero, et al. “A break in the clouds: towards a cloud definition,” ACM SIGCOMM Computer Communication Review, Vol. 39. No 1, pp 50-55, 2008.

[9] M. Laureano, C. Maziero and E. Jamhour. “Detecção de Intrusão em Máquinas Virtuais,” 5º Simpósio de Segurança em Informática – SSI. São José dos Campos, pp. 1-7, 2003.

[10] D. García, et at. "Anomaly -based network intrusion detection : Techniques, systems and challenges," in: Elsevier Computers & Security , Vol. 28 , No 1, p 18-28, 2009.

[11] J. Allen, et al. “State of the practice of intrusion detection technology,” Software Engineering Institute Carnegie Mellon University, 1999.

[12] L. Christiane. “Agentes Inteligentes para Detecção de Intrusos em Redes de Computadores,” Master Thesis, Federal University of Maranhão - UFMA , São Luís, 2002.

[13] R. Bace, P. Mell. “Intrusion Detection Systems,” NIST - National Institute of Standards and Technology . Available at: <http://www.snort.org/docs/nist-ids.pdf/>, 2001.

[14] T. Alharkan, P. Martin, "IDSaaS : Intrusion Detection System as a Service in Public Clouds," Cluster , Cloud and Grid Computing (CCGrid) , 2012 12th IEEE / ACM International Symposium on , vol in , pp.686 , 687, 13-16, May 2012 .

[15] D. Robinson.“Amazon Web Services Made Simple : Learn how Amazon EC2 , S3 , SimpleDB and SQSWeb Services Enables you to reach business goals faster,” Emereo Pty Ltd , London , pp. 20-180 2008.

[16] K. Vieira, et al. "Intrusion Detection for Grid and Cloud Computing," IT Professional , vol.12 , no.4 , pp.38 , 43 , July-Aug . 2010.

[17] S. Dhage, et al. “Intrusion Detection System in Cloud Computing Environment,” in International Conference and Workshop on Emerging Trends in Technology (ICWET 2011) - TCET , p. 235-239 , Mumbai , India . 2011.

[18] H. Kholidy, F. Baiardi, "CIDS: A Framework for Intrusion Detection in Cloud Systems," Information Technology: New Generations (ITNG), Ninth International Conference on, vol in , pp.379, 385, 16-18 April 2012.

[19] S. Roschke, C. Feng, C. Meinel. “An extensible and compatible virtualization IDS management architecture,” In: Fifth international conference on information assurance and security , pp . 130-4, 2009.

[20] Cloud Security Alliance CSA. “Top Threats to Cloud Computing v10,” Available at: http://www.cloudsecurityalliance.org/topthreats/csathreats.v1.0.pdf/. (Accessed 12 April 2012).

[21] M. Jensen and N. Gruschka . “Attack Surfaces : A Taxonomy for Attacks on Cloud Services , Cloud Computing (Cloud),” 3rd IEEE International Conference on Cloud Computing , 2010 , pp. 276 -279, 2010.

[22] D. Svantesson, R. Clarke, “Consumer and Privacy Risks in Cloud Computing,” Computer Law & Security Review 26 (4) (2010), pp. 391-397, 2010.

[23] A. Zaidi, “Recherche et des détection patterns Attaques d' dans les réseaux IP to Debits haut,” Thesis (Ph.D.) - Université d' Evry Val d' Essonne , Evry 2011 109 f .

[24] V. Thiago. “Arquitetura Multi-agentes para Detecção Distribuída de Intrusão,” Dissertation (Masters) . UFC 2012 . 102p .

[25] Oracle. “Obtenha Informações sobre a Tecnologia Java,” Available : http://www.java.com/pt_BR/about/. (Accessed 11 September), 2013.

[26] jpcap. "Jpcap - Java library for capturing and sending network packets," Available at: http://netresearch.ics.uci.edu/kfujii/jpcap/doc/. (Accessed 14 August), 2012 .

[27] Libpcap. “TCPDUMP / libpcap public repository,” Available at: http://www.tcpdump.org/. (Accessed 10 April), 2012.

[28] Canonical Ltd. "Ubuntu 12.04.3 LTS (Precise Pangolin)," Available : http://releases.ubuntu.com/precise/. (Accessed 11 September), 2013.

[29] Red Hat. "Kernel Based Virtual Machine," Available : http://www.linux-kvm.org/page/Main_Page, (Accessed 11 September) 2013.

[30] Red Hat. "Virtualization libvirt API," Available : http://libvirt.org/ . (Accessed 11 September) 2013.

[31] Oracle. "VirtualBox," Available: https://www.virtualbox.org/ . (Accessed 11 September), 2013.

[32] Hping "hping," Available : http://www.hping.org/. (Accessed 30 October), 2013.

[33] S. S. G.L.Ryan Ko, “Cloud computing vulnerability incidents: A statistical overview,” 2013.Available in: https://cloudsecurityalliance.org/download/cloudcomputing-vulnerability-incidents-a-statistical-overview/ (Accessed: 28 June 2013).

[34] C. Babcock. (2009, 7th Abril 2012). Cloud implementation to double by 2012. Available: http://www.informationweek.com/news/services/saas/214502033?queryText=cloud. (Accessed: 2 Mai). 2012.

[35] F. Gens. (2009, Feb.). “New IDC IT Cloud Services Survey: Top Benefits and Challenges”, *IDC eXchange*, Available: http://blogs.idc.com/ie/?p=730/. (Accessed: 2 Mai). 2013.

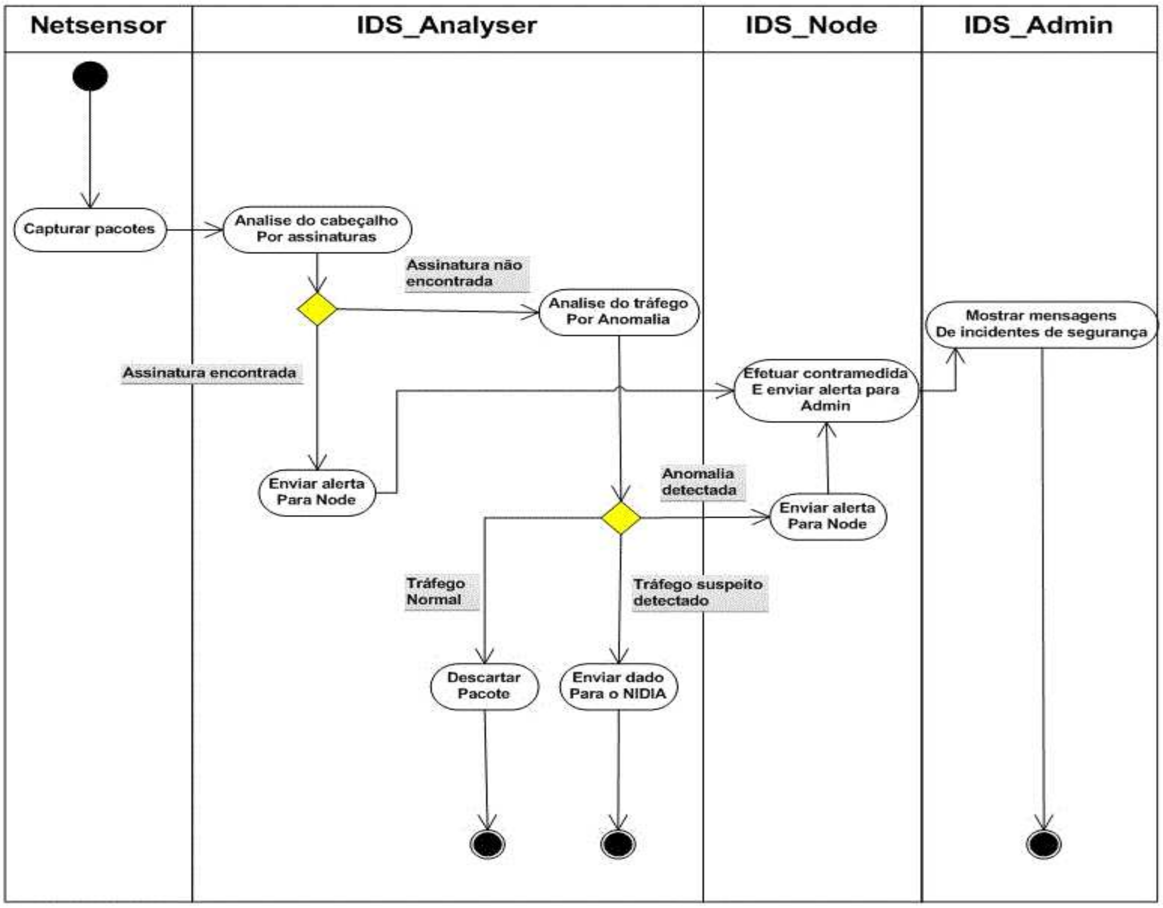

Netsensor
IDS_Analyser
IDS_Node
IDS_Admin
Capturar pacotes
Analise do cabeçalho Por assinaturas
Assinatura não encontrada
Analise do tráfego Por Anomalia
Assinatura encontrada
Mostrar mensagens De incidentes de segurança
Efetuar contramedida E enviar alerta para Admin
Enviar alerta Para Node
Anomalia detectada
Enviar alerta Para Node
Tráfego Normal
Tráfego suspeito detectado
Descartar Pacote
Enviar dado Para o NIDIA

Figure 3. IDS Activity Diagram

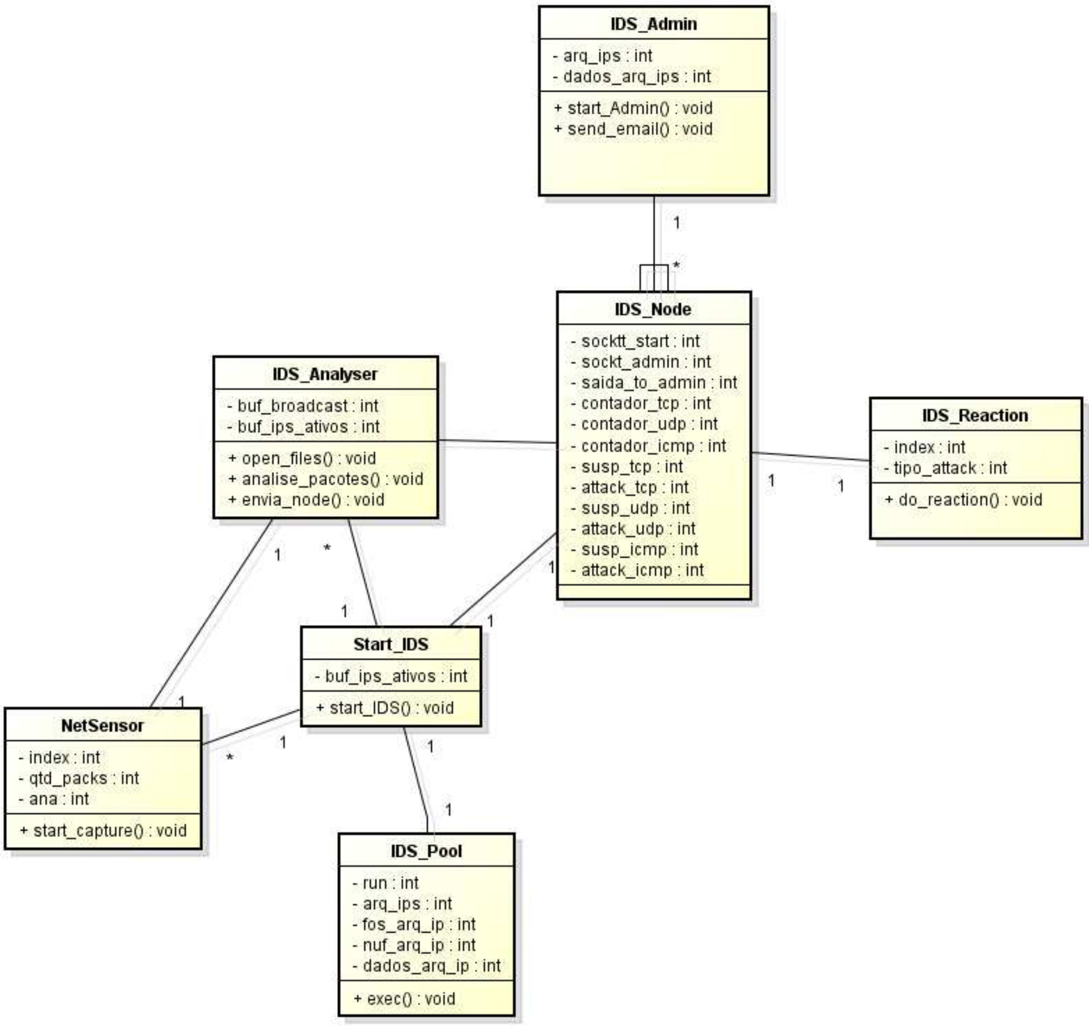
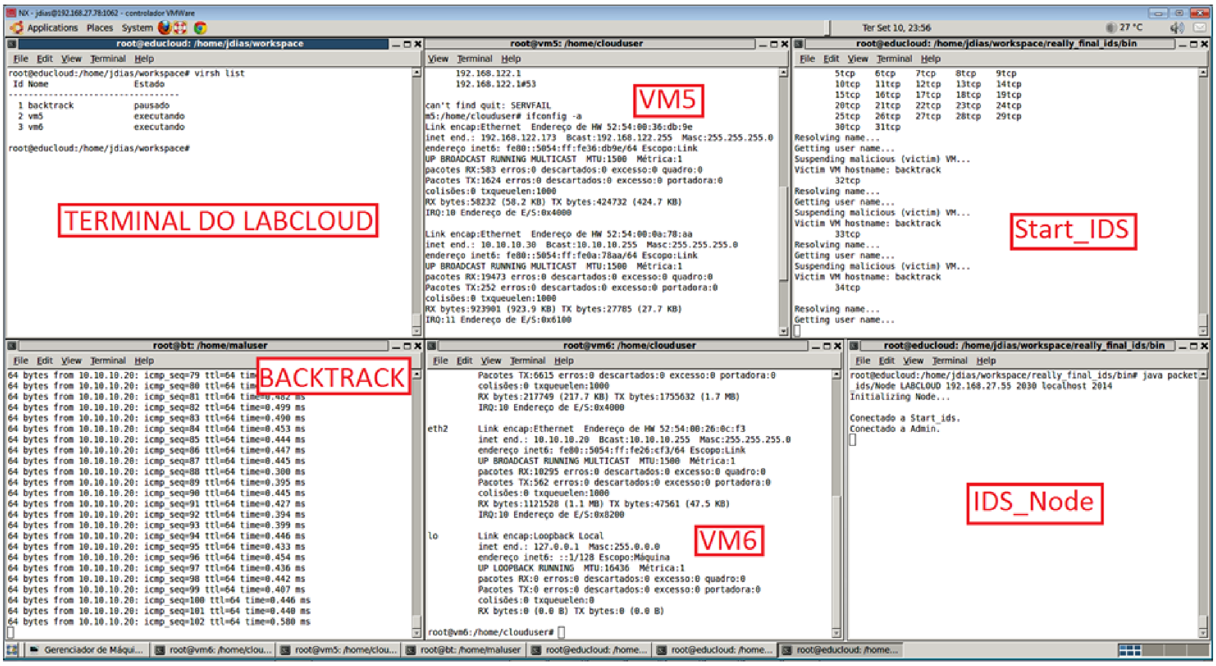


Figure 4. Class Diagram of EICIDS



```
root@educloud:/home/jdias/workspace# virsh list
 Id Nome                 Estado
----------------------------------
  1 backtrack            pausado
  2 vm5                  executando
  3 vm6                  executando

root@educloud:/home/jdias/workspace#
```

```
        192.168.122.1
        192.168.122.1#53

can't find quit: SERVFAIL
m5:/home/clouduser# ifconfig -a
Link encap:Ethernet  Endereço de HW 52:54:00:36:db:9e
inet end.: 192.168.122.173  Bcast:192.168.122.255  Masc:255.255.255.0
endereço inet6: fe80::5054:ff:fe36:db9e/64 Escopo:Link
UP BROADCAST RUNNING MULTICAST  MTU:1500  Métrica:1
pacotes RX:583 erros:0 descartados:0 excesso:0 quadro:0
Pacotes TX:1624 erros:0 descartados:0 excesso:0 portadora:0
colisões:0 txqueuelen:1000
RX bytes:58232 (58.2 KB) TX bytes:424732 (424.7 KB)
IRQ:10 Endereço de E/S:0x4000

Link encap:Ethernet  Endereço de HW 52:54:00:0a:78:aa
inet end.: 10.10.10.30  Bcast:10.10.10.255  Masc:255.255.255.0
endereço inet6: fe80::5054:ff:fe0a:78aa/64 Escopo:Link
UP BROADCAST RUNNING MULTICAST  MTU:1500  Métrica:1
pacotes RX:19473 erros:0 descartados:0 excesso:0 quadro:0
Pacotes TX:252 erros:0 descartados:0 excesso:0 portadora:0
colisões:0 txqueuelen:1000
RX bytes:923901 (923.9 KB) TX bytes:27785 (27.7 KB)
IRQ:11 Endereço de E/S:0x6100
```

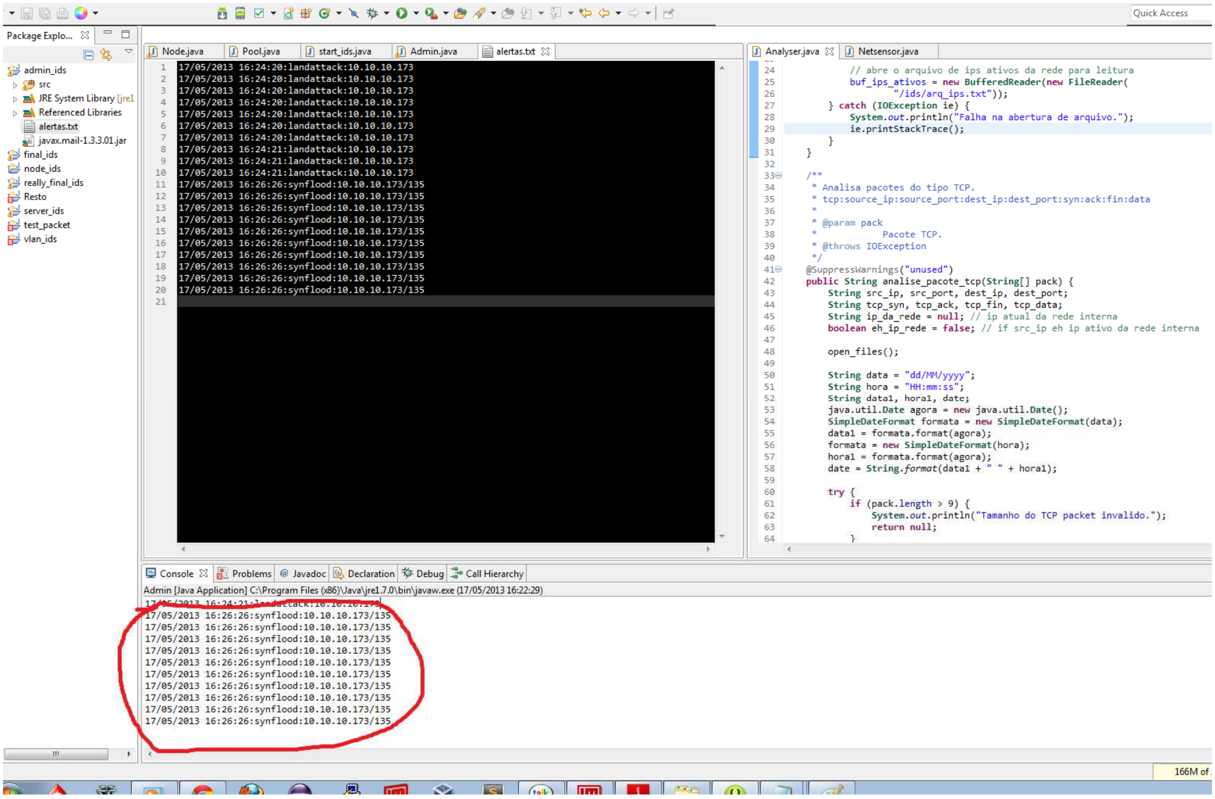
```
        5tcp    6tcp    7tcp    8tcp    9tcp
        10tcp   11tcp   12tcp   13tcp   14tcp
        15tcp   16tcp   17tcp   18tcp   19tcp
        20tcp   21tcp   22tcp   23tcp   24tcp
        25tcp   26tcp   27tcp   28tcp   29tcp
        30tcp   31tcp
Resolving name...
Getting user name...
Suspending malicious (victim) VM...
Victim VM hostname: backtrack
        32tcp
Resolving name...
Getting user name...
Suspending malicious (victim) VM...
Victim VM hostname: backtrack
        33tcp
Resolving name...
Getting user name...
Suspending malicious (victim) VM...
Victim VM hostname: backtrack
        34tcp

Resolving name...
Getting user name...
```

```
64 bytes from 10.10.10.20: icmp_seq=79 ttl=64 tim
64 bytes from 10.10.10.20: icmp_seq=80 ttl=64 tim
64 bytes from 10.10.10.20: icmp_seq=81 ttl=64 time=0.482 ms
64 bytes from 10.10.10.20: icmp_seq=82 ttl=64 time=0.499 ms
64 bytes from 10.10.10.20: icmp_seq=83 ttl=64 time=0.490 ms
64 bytes from 10.10.10.20: icmp_seq=84 ttl=64 time=0.453 ms
64 bytes from 10.10.10.20: icmp_seq=85 ttl=64 time=0.444 ms
64 bytes from 10.10.10.20: icmp_seq=86 ttl=64 time=0.447 ms
64 bytes from 10.10.10.20: icmp_seq=87 ttl=64 time=0.445 ms
64 bytes from 10.10.10.20: icmp_seq=88 ttl=64 time=0.300 ms
64 bytes from 10.10.10.20: icmp_seq=89 ttl=64 time=0.395 ms
64 bytes from 10.10.10.20: icmp_seq=90 ttl=64 time=0.445 ms
64 bytes from 10.10.10.20: icmp_seq=91 ttl=64 time=0.427 ms
64 bytes from 10.10.10.20: icmp_seq=92 ttl=64 time=0.394 ms
64 bytes from 10.10.10.20: icmp_seq=93 ttl=64 time=0.399 ms
64 bytes from 10.10.10.20: icmp_seq=94 ttl=64 time=0.446 ms
64 bytes from 10.10.10.20: icmp_seq=95 ttl=64 time=0.433 ms
64 bytes from 10.10.10.20: icmp_seq=96 ttl=64 time=0.454 ms
64 bytes from 10.10.10.20: icmp_seq=97 ttl=64 time=0.436 ms
64 bytes from 10.10.10.20: icmp_seq=98 ttl=64 time=0.442 ms
64 bytes from 10.10.10.20: icmp_seq=99 ttl=64 time=0.407 ms
64 bytes from 10.10.10.20: icmp_seq=100 ttl=64 time=0.446 ms
64 bytes from 10.10.10.20: icmp_seq=101 ttl=64 time=0.440 ms
64 bytes from 10.10.10.20: icmp_seq=102 ttl=64 time=0.580 ms
```

```
          Pacotes TX:6615 erros:0 descartados:0 excesso:0 portadora:0
          colisões:0 txqueuelen:1000
          RX bytes:217749 (217.7 KB) TX bytes:1755632 (1.7 MB)
          IRQ:10 Endereço de E/S:0x4000

eth2      Link encap:Ethernet  Endereço de HW 52:54:00:26:0c:f3
          inet end.: 10.10.10.20  Bcast:10.10.10.255  Masc:255.255.255.0
          endereço inet6: fe80::5054:ff:fe26:cf3/64 Escopo:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Métrica:1
          pacotes RX:10295 erros:0 descartados:0 excesso:0 quadro:0
          Pacotes TX:562 erros:0 descartados:0 excesso:0 portadora:0
          colisões:0 txqueuelen:1000
          RX bytes:1121528 (1.1 MB) TX bytes:47561 (47.5 KB)
          IRQ:10 Endereço de E/S:0x8200

lo        Link encap:Loopback Local
          inet end.: 127.0.0.1  Masc:255.0.0.0
          endereço inet6: ::1/128 Escopo:Máquina
          UP LOOPBACK RUNNING  MTU:16436  Métrica:1
          pacotes RX:0 erros:0 descartados:0 excesso:0 quadro:0
          Pacotes TX:0 erros:0 descartados:0 excesso:0 portadora:0
          colisões:0 txqueuelen:0
          RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)

root@vm6:/home/clouduser#
```

```
root@educloud:/home/jdias/workspace/really_final_ids/bin# java packet
_ids/Node LABCLOUD 192.168.27.55 2030 localhost 2014
Initializing Node...

Conectado a Start_ids.
Conectado a Admin.
```

Figure 6. Broad view of the LABCLOUD interface

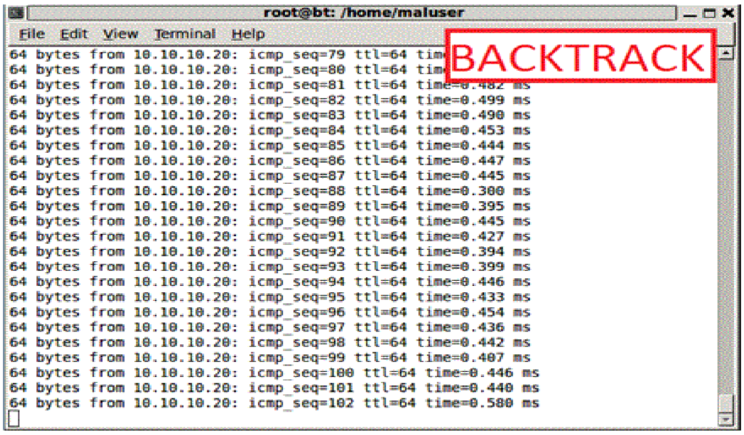


```
17/05/2013 16:24:20:landattack:10.10.10.173
17/05/2013 16:24:20:landattack:10.10.10.173
17/05/2013 16:24:20:landattack:10.10.10.173
17/05/2013 16:24:20:landattack:10.10.10.173
17/05/2013 16:24:20:landattack:10.10.10.173
17/05/2013 16:24:20:landattack:10.10.10.173
17/05/2013 16:24:20:landattack:10.10.10.173
17/05/2013 16:24:21:landattack:10.10.10.173
17/05/2013 16:24:21:landattack:10.10.10.173
17/05/2013 16:24:21:landattack:10.10.10.173
17/05/2013 16:26:26:synflood:10.10.10.173/135
17/05/2013 16:26:26:synflood:10.10.10.173/135
17/05/2013 16:26:26:synflood:10.10.10.173/135
17/05/2013 16:26:26:synflood:10.10.10.173/135
17/05/2013 16:26:26:synflood:10.10.10.173/135
17/05/2013 16:26:26:synflood:10.10.10.173/135
17/05/2013 16:26:26:synflood:10.10.10.173/135
17/05/2013 16:26:26:synflood:10.10.10.173/135
17/05/2013 16:26:26:synflood:10.10.10.173/135
17/05/2013 16:26:26:synflood:10.10.10.173/135
```

```java
            // abre o arquivo de ips ativos da rede para leitura
            buf_ips_ativos = new BufferedReader(new FileReader(
                    "/ids/arq_ips.txt"));
        } catch (IOException ie) {
            System.out.println("Falha na abertura de arquivo.");
            ie.printStackTrace();
        }
    }

    /**
     * Analisa pacotes do tipo TCP.
     * tcp:source_ip:source_port:dest_ip:dest_port:syn:ack:fin:data
     *
     * @param pack
     *            Pacote TCP.
     * @throws IOException
     */
    @SuppressWarnings("unused")
    public String analise_pacote_tcp(String[] pack) {
        String src_ip, src_port, dest_ip, dest_port;
        String tcp_syn, tcp_ack, tcp_fin, tcp_data;
        String ip_da_rede = null; // ip atual da rede interna
        boolean eh_ip_rede = false; // if src_ip eh ip ativo da rede interna

        open_files();

        String data = "dd/MM/yyyy";
        String hora = "HH:mm:ss";
        String data1, hora1, date;
        java.util.Date agora = new java.util.Date();
        SimpleDateFormat formata = new SimpleDateFormat(data);
        data1 = formata.format(agora);
        formata = new SimpleDateFormat(hora);
        hora1 = formata.format(agora);
        date = String.format(data1 + " " + hora1);

        try {
            if (pack.length > 9) {
                System.out.println("Tamanho do TCP packet invalido.");
                return null;
            }
```

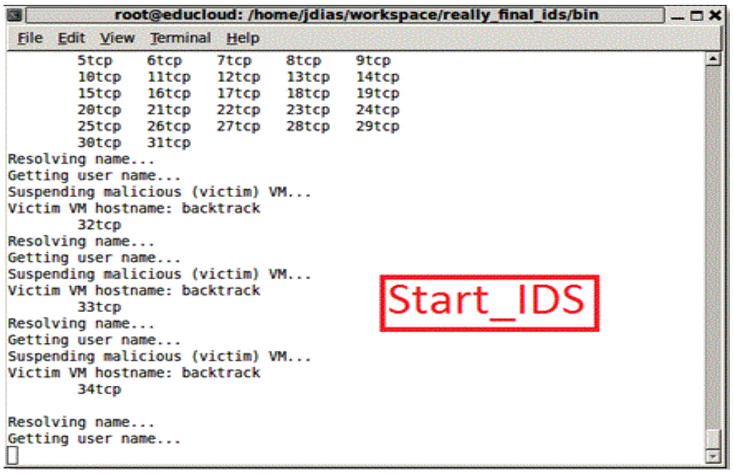
```
Admin [Java Application] C:\Program Files (x86)\Java\jre1.7.0\bin\javaw.exe (17/05/2013 16:22:29)
17/05/2013 16:24:21:landattack:10.10.10.173
17/05/2013 16:26:26:synflood:10.10.10.173/135
17/05/2013 16:26:26:synflood:10.10.10.173/135
17/05/2013 16:26:26:synflood:10.10.10.173/135
17/05/2013 16:26:26:synflood:10.10.10.173/135
17/05/2013 16:26:26:synflood:10.10.10.173/135
17/05/2013 16:26:26:synflood:10.10.10.173/135
17/05/2013 16:26:26:synflood:10.10.10.173/135
17/05/2013 16:26:26:synflood:10.10.10.173/135
17/05/2013 16:26:26:synflood:10.10.10.173/135
17/05/2013 16:26:26:synflood:10.10.10.173/135
```

Figure 7. Syn Flood Attack Detected and displayed by IDS_Admin



```
64 bytes from 10.10.10.20: icmp_seq=79 ttl=64 tim
64 bytes from 10.10.10.20: icmp_seq=80 ttl=64 tim
64 bytes from 10.10.10.20: icmp_seq=81 ttl=64 time=0.482 ms
64 bytes from 10.10.10.20: icmp_seq=82 ttl=64 time=0.499 ms
64 bytes from 10.10.10.20: icmp_seq=83 ttl=64 time=0.490 ms
64 bytes from 10.10.10.20: icmp_seq=84 ttl=64 time=0.453 ms
64 bytes from 10.10.10.20: icmp_seq=85 ttl=64 time=0.444 ms
64 bytes from 10.10.10.20: icmp_seq=86 ttl=64 time=0.447 ms
64 bytes from 10.10.10.20: icmp_seq=87 ttl=64 time=0.445 ms
64 bytes from 10.10.10.20: icmp_seq=88 ttl=64 time=0.300 ms
64 bytes from 10.10.10.20: icmp_seq=89 ttl=64 time=0.395 ms
64 bytes from 10.10.10.20: icmp_seq=90 ttl=64 time=0.445 ms
64 bytes from 10.10.10.20: icmp_seq=91 ttl=64 time=0.427 ms
64 bytes from 10.10.10.20: icmp_seq=92 ttl=64 time=0.394 ms
64 bytes from 10.10.10.20: icmp_seq=93 ttl=64 time=0.399 ms
64 bytes from 10.10.10.20: icmp_seq=94 ttl=64 time=0.446 ms
64 bytes from 10.10.10.20: icmp_seq=95 ttl=64 time=0.433 ms
64 bytes from 10.10.10.20: icmp_seq=96 ttl=64 time=0.454 ms
64 bytes from 10.10.10.20: icmp_seq=97 ttl=64 time=0.436 ms
64 bytes from 10.10.10.20: icmp_seq=98 ttl=64 time=0.442 ms
64 bytes from 10.10.10.20: icmp_seq=99 ttl=64 time=0.407 ms
64 bytes from 10.10.10.20: icmp_seq=100 ttl=64 time=0.446 ms
64 bytes from 10.10.10.20: icmp_seq=101 ttl=64 time=0.440 ms
64 bytes from 10.10.10.20: icmp_seq=102 ttl=64 time=0.580 ms
```

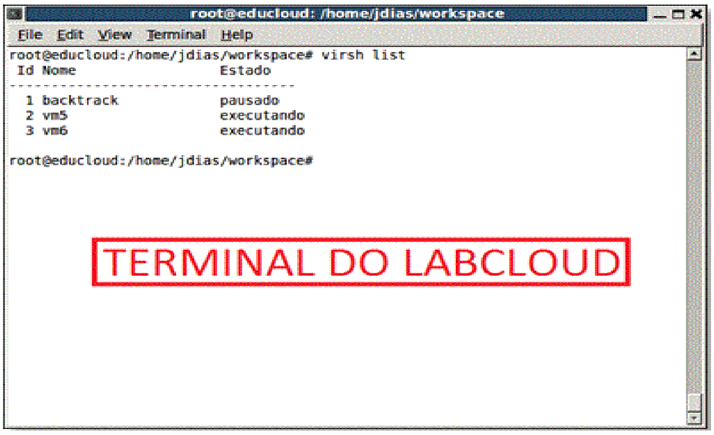
Figure 8. Interface backtrack when issuing the ping command to VM6 (10.10.10.20)



root@educloud: /home/jdias/workspace/really_final_ids/bin
File Edit View Terminal Help
Start_IDS

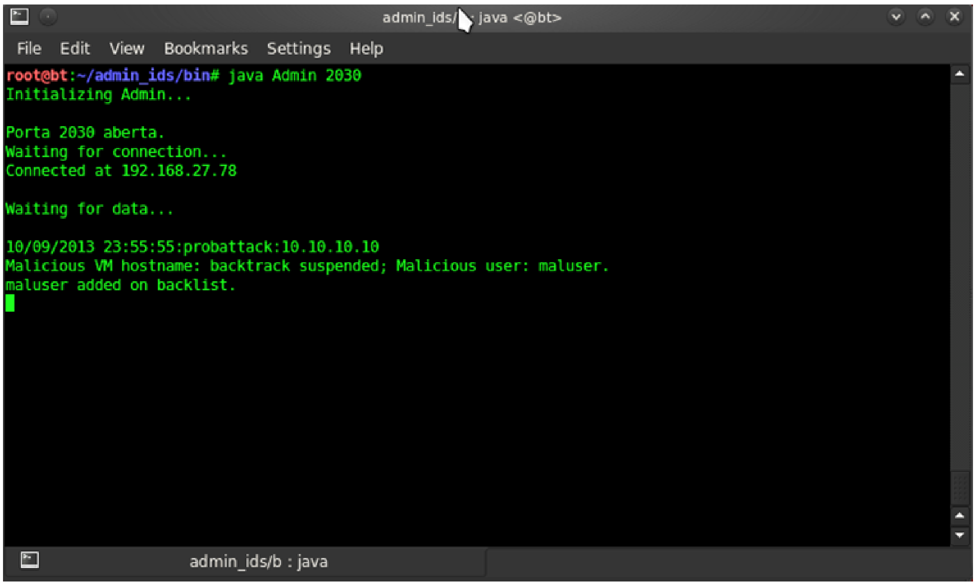
```
        5tcp    6tcp    7tcp    8tcp    9tcp
        10tcp   11tcp   12tcp   13tcp   14tcp
        15tcp   16tcp   17tcp   18tcp   19tcp
        20tcp   21tcp   22tcp   23tcp   24tcp
        25tcp   26tcp   27tcp   28tcp   29tcp
        30tcp   31tcp
Resolving name...
Getting user name...
Suspending malicious (victim) VM...
Victim VM hostname: backtrack
        32tcp
Resolving name...
Getting user name...
Suspending malicious (victim) VM...
Victim VM hostname: backtrack
        33tcp
Resolving name...
Getting user name...
Suspending malicious (victim) VM...
Victim VM hostname: backtrack
        34tcp

Resolving name...
Getting user name...
```

Figure 9. Start_IDS interface displaying countermeasures


root@educloud: /home/jdias/workspace
File Edit View Terminal Help

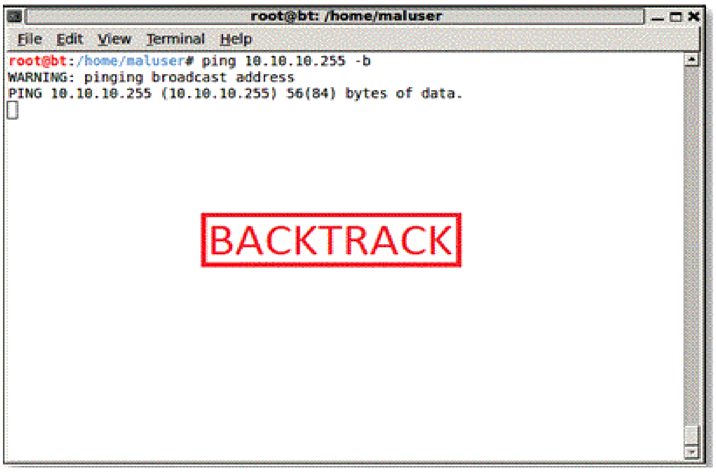
```
root@educloud:/home/jdias/workspace# virsh list
 Id Nome                     Estado
----------------------------------
  1 backtrack                pausado
  2 vm5                      executando
  3 vm6                      executando

root@educloud:/home/jdias/workspace#
```

Figure 10. LABCLOUD terminal interface showing the "paused" status in backtrack VM



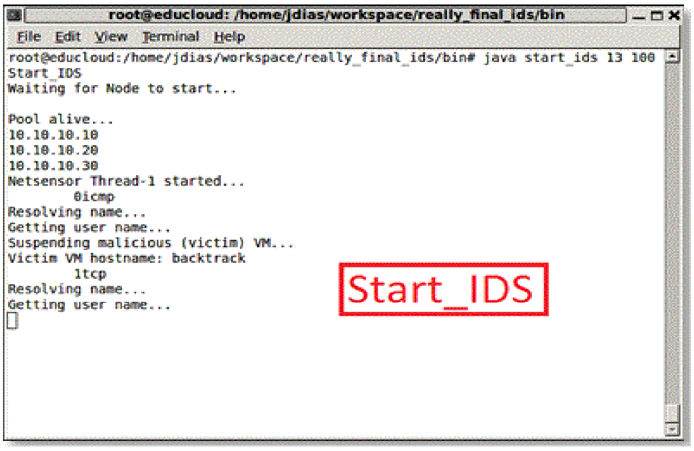
```
root@bt:~/admin_ids/bin# java Admin 2030
Initializing Admin...

Porta 2030 aberta.
Waiting for connection...
Connected at 192.168.27.78

Waiting for data...

10/09/2013 23:55:55:probattack:10.10.10.10
Malicious VM hostname: backtrack suspended; Malicious user: maluser.
maluser added on backlist.
```

Figure 11. Administrative interface Bt VM at the time of the suspected attack



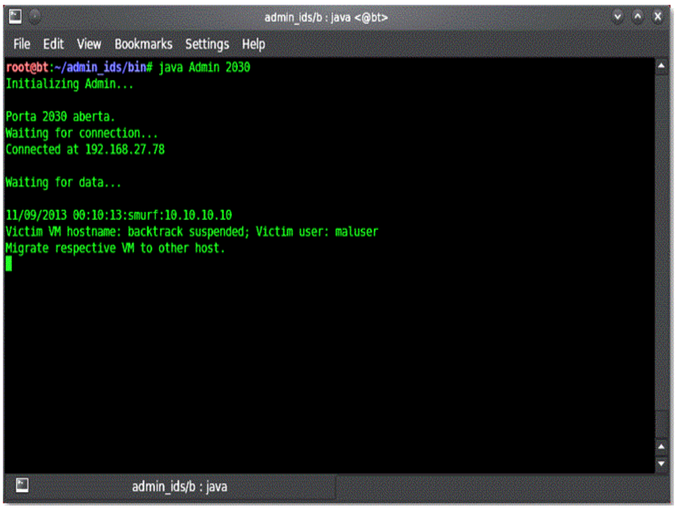
```
root@bt:/home/maluser# ping 10.10.10.255 -b
WARNING: pinging broadcast address
PING 10.10.10.255 (10.10.10.255) 56(84) bytes of data.
```

Figure 12. Interface backtrack issuing ping to broadcast


root@educloud: /home/jdias/workspace/really_final_ids/bin
File Edit View Terminal Help
Start_IDS

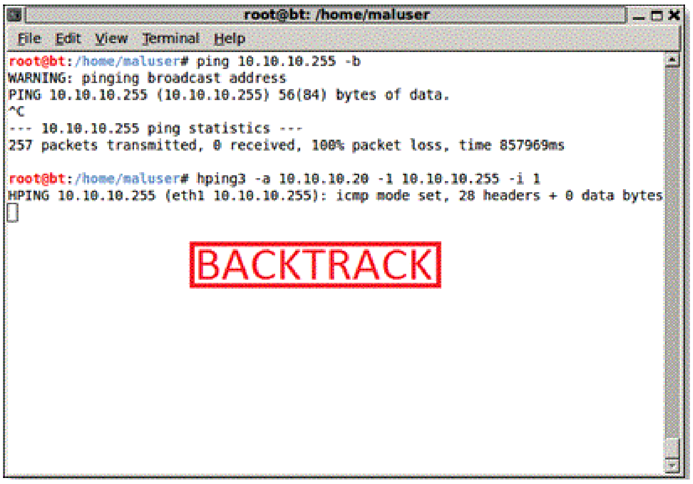
```
root@educloud:/home/jdias/workspace/really_final_ids/bin# java start_ids 13 100
Start_IDS
Waiting for Node to start...

Pool alive...
10.10.10.10
10.10.10.20
10.10.10.30
Netsensor Thread-1 started...
        0icmp
Resolving name...
Getting user name...
Suspending malicious (victim) VM...
Victim VM hostname: backtrack
        1tcp
Resolving name...
Getting user name...
```

Figure 13. Start_IDS interface displaying the status of alerts and Reaction in case of attack



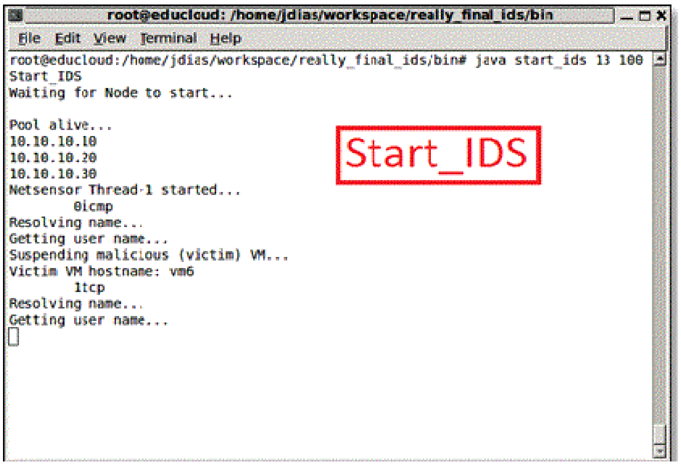
```
root@bt:~/admin_ids/bin# java Admin 2030
Initializing Admin...

Porta 2030 aberta.
Waiting for connection...
Connected at 192.168.27.78

Waiting for data...

11/09/2013 00:10:13:smurf:10.10.10.10
Victim VM hostname: backtrack suspended; Victim user: maluser
Migrate respective VM to other host.
```

Figure 14. Administrative interface of bt VM on receipt of a suspicious smurf attack alert



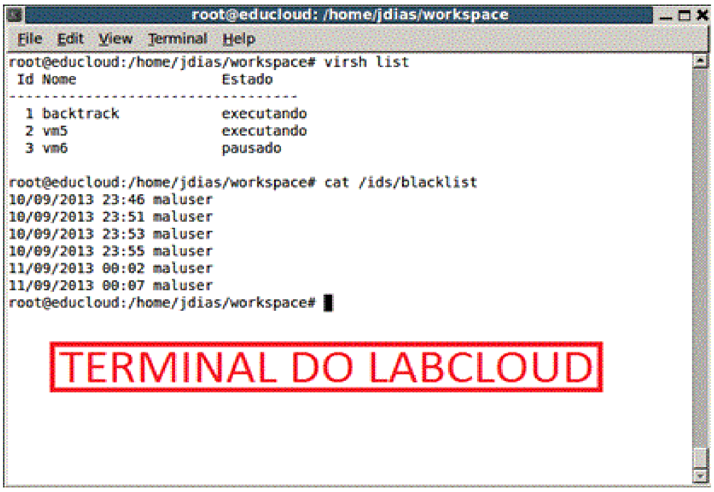
```
root@bt:/home/maluser# ping 10.10.10.255 -b
WARNING: pinging broadcast address
PING 10.10.10.255 (10.10.10.255) 56(84) bytes of data.
^C
--- 10.10.10.255 ping statistics ---
257 packets transmitted, 0 received, 100% packet loss, time 857969ms

root@bt:/home/maluser# hping3 -a 10.10.10.20 -1 10.10.10.255 -i 1
HPING 10.10.10.255 (eth1 10.10.10.255): icmp mode set, 28 headers + 0 data bytes
```

Figure 15. Backtrack interface, sending a flood of ICMP packets to broadcast with spoofed source IP


root@educloud: /home/jdias/workspace/really_final_ids/bin
File Edit View Terminal Help
Start_IDS

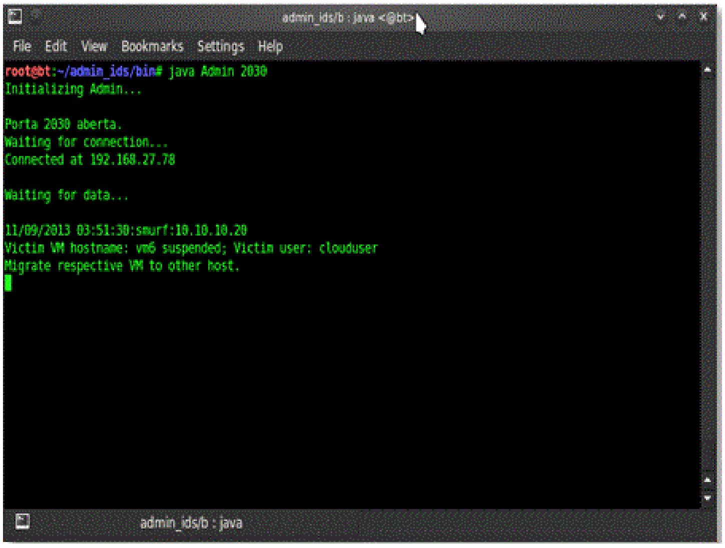
```
root@educloud:/home/jdias/workspace/really_final_ids/bin# java start_ids 13 100
Start_IDS
Waiting for Node to start...

Pool alive...
10.10.10.10
10.10.10.20
10.10.10.30
Netsensor Thread-1 started...
        0icmp
Resolving name...
Getting user name...
Suspending malicious (victim) VM...
Victim VM hostname: vm6
        1tcp
Resolving name...
Getting user name...
```

Figure 16. Start_IDS identifying the victim VM (VM6) and displaying the status of the component Reaction running countermeasures


root@educloud: /home/jdias/workspace
File Edit View Terminal Help
TERMINAL DO LABCLOUD

```
root@educloud:/home/jdias/workspace# virsh list
 Id Nome                 Estado
----------------------------------
  1 backtrack            executando
  2 vm5                  executando
  3 vm6                  pausado

root@educloud:/home/jdias/workspace# cat /ids/blacklist
10/09/2013 23:46 maluser
10/09/2013 23:51 maluser
10/09/2013 23:53 maluser
10/09/2013 23:55 maluser
11/09/2013 00:02 maluser
11/09/2013 00:07 maluser
root@educloud:/home/jdias/workspace# 
```

Figure17. LABCLOUD terminal interface showing the "paused" status of VM6 after executing the countermeasure



```
root@bt:~/admin_ids/bin# java Admin 2030
Initializing Admin...

Porta 2030 aberta.
Waiting for connection...
Connected at 192.168.27.78

Waiting for data...

11/09/2013 03:51:30:smurf:10.10.10.20
Victim VM hostname: vm6 suspended; Victim user: clouduser
Migrate respective VM to other host.
```

Figure 18. Administrative interface bt VM on receipt of the smurf attack alert

Table II. Comparison of IDS -based VM to Cloud Computing

IDS	Architecture		IDS Protection Mode			Capture Packets on the VM	Location			Use of Elasticity	
	Center.	Distribut.	By Node	By VM	By Service		Physic. Node	VM	Physical Node and VM	IDS Increase Capacity	Save Cloud´s Resources
GCCIDS[16]		X	X				X				
Intrusion Detection System in Cloud[17]	X				X		X			X	
IDS VM [19]	X			X		X		X		X	
IDSaaS [14]	X				X	X		X		X	
CIDS [18]		X	X	X		X			X	X	
Proposed IDS	X		X	X		X			X	X	X