Internet Intrusion Detection System Service in a Cloud

Amirreza Zarrabi, Alireza Zarrabi

**Department of Computer and Communication Systems, Faculty of Engineering,
University Putra Malaysia, 43400 UPM Serdang, Selangor, Malaysia**

Abstract

Intrusion Detection Systems (IDS) have been used widely to detect malicious behaviors in network communication and hosts. It is defined as a computer network system to collect information on a number of key points, and analyze this information to see whether there are violations of network security policy behavior and signs of attack.

IDS aroused the concern of users as an important computer network security technology. In recent times, with the advent of Cloud Computing, the concepts of Software as a Service (SaaS), where vendors provide key software products as services over the internet that can be accessed by users to perform complex tasks, have become increasingly popular. Cloud Computing is a method to increase the capacity or add capabilities dynamically without investing in new infrastructure, training new personnel, or licensing new software.

We introduce a Cloud Intrusion Detection System Services (CIDSS) which is developed based on Cloud Computing and can make up for the deficiency of traditional intrusion detection, and proved to be great scalable. CIDSS can be utilized to overcome the critical challenge of keeping the client secure from cyber attacks while benefit the features which are presented by Cloud Computing technology.

Keywords: *Intrusion Detection System, Cloud Computing, Software as a Service.*

1. Introduction

The last century, computer turned out to be an inseparable part of daily human life. For recent years and with the invent of the Internet, it has been deployed for communication and accessing data. However, currently people rely on the Internet to satisfy their demands utilizing its services, which can be defined as some computing function, rather than accessing the mass data from the Internet. Along with the proposal of the Cloud Computing concepts, a new paradigm of software development and deployment of resources has emerged. It is possible to get rid of the great amount of the spending for fixed assets, such as expensive network servers and software.

At present the safety of commonly used technologies such as message encryption, firewalls protect the network and can be used as a first line of defense, but only these technologies is not enough. Intrusion Detection Systems (IDS) has been proposed for years as an efficient security measure and is nowadays widely deployed for securing critical IT-Infrastructures [1]. Many commercial and open source implementations have emerged and been widely used in practice for identifying malicious behaviors against protected hosts or network environments. They can offer security measures by investigating configurations, logs, network traffic, and user actions to identify typical attack behavior [2].

In classical enterprise settings, an IDS is normally deployed on dedicated hardware at the edge of the defended networking infrastructure or run on individual hosts on the network, in order to protect respective network or host from external attacks [3]. Today small and medium companies are increasingly realizing that simply by tapping into the Cloud they can gain fast access to best business applications, without training new personnel, or licensing new software. IDS is not an exception to this tread and the interests for embedding IDS to a Cloud environment is undeniable.

In this paper, we introduce Cloud Intrusion Detection System Service (CIDSS) which is built around the software-as-a-service (SaaS) model for providing security to any Cloud based user. The CIDSS architecture is proposed which consists of light weight IDS agents integrated inside the protected network and a central detection engine unit. The concept of grouping is introduces for the flexible integration of IDS agents in to multiple network segments. Virtual Private Network (VPN) is utilized as a means of grouping and information exchange facility. A standardized interface is designed to provide a view of result reports for users.

The remainder of this paper is organized as follow: In section 2, basic concepts of IDS are discussed. The intrusion detection methods are detailed in section 3. In section 4, an overview of Cloud Computing model is presented. The architecture of CIDSS is described in section 5. In the last section, future work and conclusion are presented.

2. Intrusion Detection System

Intrusion detection systems are software or hardware systems that automate the process of monitoring the events occurring in a computer system or network, analyzing them for malicious activities or policy violations and produces reports to a management station.

An IDS is composed of several components [4]:

- Sensors which generate security events.
- Console to monitor events and alerts and control the sensors.
- Central Engine that records events logged by the sensors in a database and uses a system of rules to generate alerts from security events received.

Based on the protected objective or the information source, IDS can be classified into Host-based Intrusion Detection System and Network-based Intrusion Detection System [1], [5].

2.1 Host-Based Intrusion Detection System

Host-based Intrusion Detection System was the first type of intrusion detection software to be designed, with the original target system being the mainframe computer where outside interaction was infrequent [6]. Host-based IDSs operate on information collected from within an individual computer system. A Host-based IDS monitors the inbound and outbound packets from the computer system only and would alert the user or administrator if suspicious activity is detected[1]. Besides the benefits acquired when utilizing this model of IDS, there are some disadvantages, which discourage deploying Host-based IDS:

- Host-based IDSs are harder to manage, as information must be configured and managed for every host monitored.
- Since the information sources and the analysis engines for Host-based IDSs reside on the host targeted by attacks, the IDS may be attacked and disabled as part of the attack.
- Host-based IDSs use the computing resources of the hosts they are monitoring, therefore inflicting a performance cost on the monitored systems.

2.2 Network-Based Intrusion Detection System

Network-based Intrusion Detection Systems focus more greatly on the network than a specific host. Network-based IDS detects attacks by capturing and analyzing network packets. Listening on a network segment or switch, one network-based IDS can monitor the network traffic affecting multiple hosts that are connected to the network segment, thereby protecting those hosts.

Network-based IDSs often consist of a set of single purpose sensors placed at various points in a network. These units monitor network traffic, performing local analysis of that traffic and reporting attacks to a central management console. As the sensors are limited to running the IDS, they can be more easily secured against attacks, e.g. run the IDS sensors in stealth mode [5], [7].

The architecture of the Network-based IDS would eliminate the disadvantages which are mentioned for Host-based IDS. There is no need to configure and manage every host as one IDS sensor in a network segment could take the responsibility of all analysis. Attacks to a specific host in a network would not affect IDS and securing the IDS sensor is simpler. Network-based IDS would utilize dedicated resources for its functionalities which is isolated from any host in the network. Therefore, it does not inflict a performance cost on the monitored systems.

3. Intrusion Detection Methods

There are two primary approaches for analyzing events to detect attacks: Misuse Detection Approach and Anomaly Detection Approach. Misuse detection, in which the analysis targets something known to be attack pattern, is the technique used by most commercial systems. Anomaly detection, in which the analysis looks for abnormal patterns of activity, has been the subject of a great deal of research. Anomaly detection is used in limited form by a number of IDSs. The most effective IDSs use mostly misuse detection approaches.

3.1 Anomaly Detection Approach

Anomaly detectors identify abnormal unusual behavior on a host or network. They function on the assumption that attacks are different from legitimate activity and can therefore be detected by systems that identify these differences. Anomaly detectors construct profiles representing normal behavior of users, hosts, or network

[1] Host-based IDSs could utilize operating system audit trails and system logs for system state monitoring, e.g. It can detect which program accesses what resources.

connections. These profiles are constructed from historical data collected over a period of normal operation. The detectors then collect event data and use a variety of measures to determine when monitored activity deviates from the normal routine.

Anomaly detection approaches often require extensive training in order to characterize normal behavior patterns. Unfortunately, the IDSs based on anomaly detection often produce a large number of false alarms, as normal patterns of user and system behavior can vary wildly. Modern day enterprise network environments amplify this disadvantage due to the massive amounts of dynamic and diverse data that needs to be analyzed. Despite this shortcoming, researchers assert that IDSs based on anomaly detection are able to detect new attack forms.

3.2 Misuse Detection Approach

Misuse detectors analyze system activity, looking for events or sets of events that match a predefined pattern of events that describe a known attack. As the patterns corresponding to known attacks are called signatures, misuse detection is sometimes called signature-based detection. The most common form of misuse detection used in commercial products specifies each pattern of events corresponding to an attack as a separate signature. However, there are more sophisticated approaches to doing misuse detection that can leverage a single signature to detect groups of attacks. There are some advantages which motivate utilizing misuse detection approach:

- Misuse detectors are very effective at detecting attacks without generating an overwhelming number of false alarms.
- Misuse detector approach does not require extensive training in order to detect attacks.

As misuse detectors use the signature databases for attack detection, they can only detect those attacks they know about and their signature are present in the database. Therefore, they must be constantly updated with signatures of new attacks.

4. Cloud Computing

Many practitioners in the commercial and academic spheres have attempted to define exactly what “Cloud Computing” is and what unique characteristics it presents [8]. The National Institute of Standards and Technology (NIST) defines Cloud Computing as “. . . a pay-per-use model for enabling available, convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction [9].”
The Cloud architecture can be divided in three layers: The System Layer, The Platform Layer, and The Application Layer. The underlying hardware is not considered as a part of the Cloud, but as the fundamental basis and is operated by the Cloud providers [1], [10].

- The System Layer: The system layer is the lowest layer in the Cloud architecture and includes virtualized hosts and networks. Service model delivered at this layer is referred to as Infrastructure-as-a-Service (IaaS). IaaS completely changes the way developers deploy their applications. Instead of spending big with their own data centers or managed hosting companies or services and then hiring operations staff to get it going, they can just get a virtual server running in minutes and pay only for the resources they use.
- The Platform Layer: The platform layer is the second layer in the architecture and includes virtualized operating systems as well as runtimes and APIs. Service model delivered at this layer is referred to as Platform as a Service (PaaS). This offers an integrated set of developer environment that a developer can tap to build their applications without having any clue about what is going on underneath the service.
- The Application Layer: The application layer is the top level of the architecture and provides virtual applications. Service model delivered at this layer is referred to as Software as a Service (SaaS). Applications are remotely hosted by the application or service provider and made available to customers on demand, over the Internet.

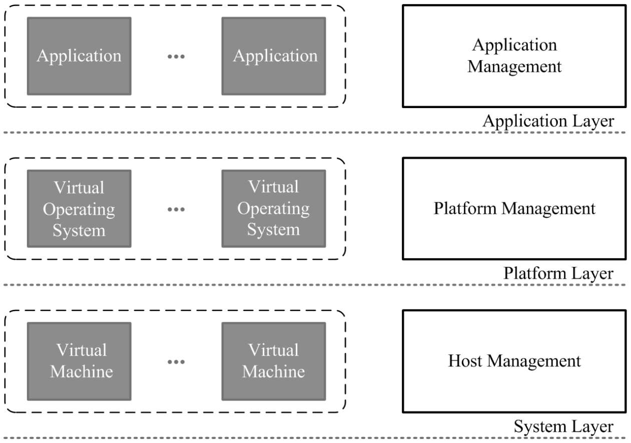

Fig. 1 Cloud Computing Architecture.

The Cloud providers need a certain kind of management on each layer to simplify the configuration of the Cloud infrastructure. The management components create the related virtual components on their specific layer. Figure 1 depicts the layered structure of Cloud Computing architecture.

In the layered architecture services of a higher layer can be composed from services of the underlying layer [11]. A core middleware manages physical resources and the virtual machines (VMs) deployed on top of them. Cloud development environments are built on top of infrastructure services to offer application development and deployment capabilities. Once deployed in the Cloud, these applications can be consumed by end users.

There are some features applied to a Cloud Computing which motivate users to migrate to Clouds and deploy its services, among those are:

- Self-Service: Consumers of Cloud Computing services expect on-demand, nearly instant access to resources. To support this expectation, clouds must allow self-service access so that customers can request, customize, pay, and use services without intervention of human operators [12].
- Elasticity: Cloud Computing gives the illusion of infinite computing resources available on demand. Therefore users expect clouds to rapidly provide resources in any quantity at any time. In particular, it is expected that the additional resources can be (a) provisioned, possibly automatically, when an application load increases and (b) released when load decreases.
- Massive scalability: Although organizations might have hundreds or thousands of systems, Cloud Computing provides the ability to scale to tens of thousands of systems, as well as the ability to massively scale bandwidth and storage space [13].

5. Cloud Intrusion Detection System Service

Developing IDS mechanism in the Cloud environment is highly motivated for both Cloud users and Cloud providers. There are some factors which instruct user for this tendency. Apart from the normal features which are obtained when using a service provided by a Cloud such as fast access to best business applications, eliminating the rule of trained new personnel, or licensing new software, some aspect of IDS in Cloud would be tempting for this migration.

Accessing to Network-based IDS on the Cloud to protect a network gives the possibility of exploiting different type of IDS detection methods on a single segment based on user demands almost instantly. It gives the illusion of a Network-based IDS which try to protect all segments of a network which are communicating over the internet with each other. It uses dedicated resources on the cloud for IDS functionalities which are isolated from any host on user network.

5.1 Cloud Intrusion Detection Service Architecture

We introduce a Cloud Intrusion Detection System Service (CIDSS) to overcome the critical challenge of keeping the client secure from cyber attacks. It is designed based on software as a service model for security of any Cloud based user. The CIDSS is composed of three components: Intrusion Detection Service Agent, Cloud Computer Service Component (CCSC), and Intrusion Detection Service Component (IDSC). Figure 2 represents each component in the proposed structure.

- Intrusion Detection Service Agent: The agent is a light weight, single purpose equipment - dedicated hardware or software - integrated inside the user network to collect necessary information. According to the location of the agent, the CIDSS could protect a segment of the network or the whole network. Agents are grouped based on rule-sets and thresholds or network traffic to improve the service efficiency and protection flexibility.
- Cloud Computer Service Component: The CCSC collects messages from agents. It formats all messages and send them to the IDSC according to grouping constrains defined for messages. A secure connection path should be established by CCSC to absorb information gathered by agents otherwise the system behavior could be tainted by external intrusion.
- Intrusion Detection Service Component: The IDSC is responsible for intrusion detection. There are four sub components playing major rule in IDSC.

  Collector: Collector is responsible for reading all information received by CCSC, selecting items of interest, and forwarding them to the appropriate analysis engine.

  Analysis Engine: Analysis engine is a sophisticated decision and pattern matching mechanism. It analysis data came from the collector and matches it to known patterns of activity stored in the signature database. It identifies malicious behavior and generates alerts through the event publisher. Event *Publisher:* It is a standardized interface to provide a unified view of result report for users as the analysis engine could be an independent process which can be implemented using any IDS, e.g. Snort. Intrusion Detection Message Exchange Format (IDMEF) [14] would be used as standard representation of IDS alerts.

IDS Controller: It is responsible for remote configuration and control of all agent groups. It has access to IDSC configuration for fine tuning its operation based on user demands.

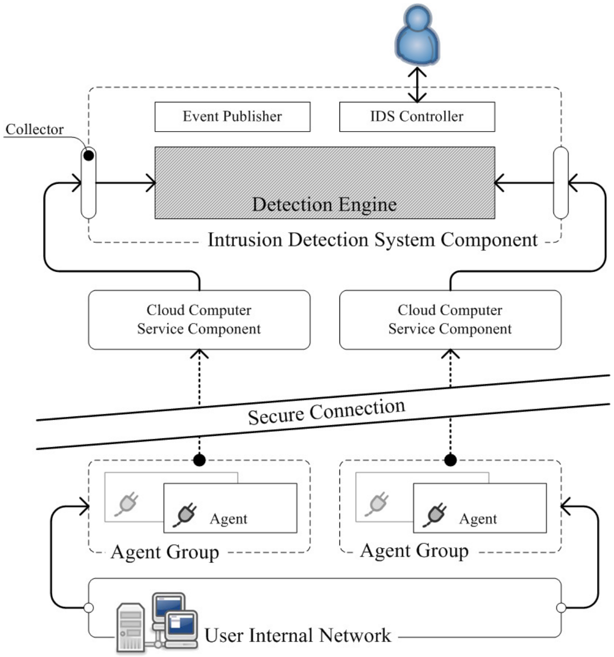

Fig. 2 Cloud Intrusion Detection System Architecture.

5.2 Cloud Intrusion Detection Service Requirements

There are a number of challenges that must be considered when implementing CIDSS. Some of these challenges are inherent in what an IDS does and others are simply part of the way that a network is configured.

Normally, the construction of a network would include switches and routers. The collision domains which is defined as the extend to which a signal can be propagated inside a network would be separated by these devices [15]. An agent needs to be able to look at all of the traffic on a protected network segment even on different collision domains. There are a number of ways to achieve this goal including: adding a hub inline at a choke point, connecting a network Test Access Port (TAP) to allow passive access to all the traffics, using Switched Port Analyzer (SPAN) port on the switch being monitored.

High-speed networks are a problem for any IDS solutions. A normal IDS would do the processing of every packets on the network. The speed of the network could increase ahead of the processing power available to pull every packet off the wire and process it [16]. An agent does not demand high resources as it would not scrutinize the network traffic and limits itself to rudimentary processing to capture packet of interest. Nevertheless, the only restriction on agent deployment is disparity between the traffic of network segment and the connection speed with the Cloud. CIDSS mostly targets the Small and Medium Businesses (SMB) as users and they do not typically have very high networking bandwidth as soon as they become available. Anyhow, agent grouping would provide the flexibility to cope with this issue. Three basic rules are applied when grouping the agents, listed in Table 1. According to the rules, the agent utilized to protect whole user network from internet attacked should be in a separate group while the agents deployed to protect two small segments of user network with similar rule-sets could be in one group if their traffics are negligible.

Table 1: Agent Grouping Basic Rules

Network Segment Configuration	***Group***
Segments with different rule-set	Separate Group
Segments with similar rule-set	Similar Group
Segments with high traffics	Separate Group

The security of agents is a preliminary factor while implementing and embedding one inside the user network. They are operating in stealth mode and a distinct connection form user network internet connection would be employed for communication with the cloud. Even though it is not considered as a requirement for agents operation but single purpose internet connection could enforce more security as the respective firewall could be configure more strictly for a single service.

5.3 Capturing Interested Packets

Agent should do the early filtering to reduce the load on the Analysis engine and on the overall system, as the process of sending packets from the agent to the CCSC can often be avoided. Considering an internet connection with n Mbps bandwidth, at most a channel with 2n Mbps (upstream plus downstream) bandwidth should be sniffed by an agent. For almost real time IDS, the captured packets should be discarded to 50% of total packets in the worst case. Otherwise, detection delay should be introduced. Some unique feature of network traffics can be exploited to this end. Refer to Figure 3 for internal structure of agent.

Since a fraction of packets is only subject to header analysis, this could be performed by agent and only packets prone to intrusion effects would be transferred [17]. Packets with no payload which do not show any indication of intrusion are simply discarded.

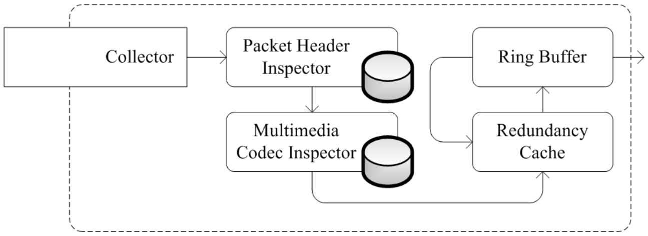

Fig. 3 Agent Internal Structure.

According to the researches 30% of all incoming traffic and 60% of all outgoing traffic is redundant by measures [18]. Therefore, agent and the respective CCSC should cooperate to eliminate the redundant data in packets. Caches are place at both agent and CCSC, which are used to hold the most recent packets. The cache at agent end would replace the redundant data inside the packet into light weight tokens and passes the encoded, smaller packet to CCSC, where the original packet would be reconstructed. The technique developed by Manber for finding similar files in a huge file system utilizing Robin fingerprint would be adopted to implement a network traffic tunable duplication detection strategy [19]. Packet losses should be prevented in order to preserve the required consistency between caches.

Multimedia data which constitutes most of the traffic in the internet communications could be inspected by agent. Multimedia traffic can be uniquely identified by certain characteristics, e.g. the header in an AVI file, and once the traffic has been recognized the remaining packets for the stream containing the multimedia content can bypass the analysis engine [20].

Ring buffer is a method for adapting the packet stream speed in a way that prevent packet loss [21]. The latency introduced by ring buffers is not particularly important for a CIDSS operation where the latency does only affect detection delay and no packet forwarding happens as no prevention is intended by CIDSS. The ring buffer should tune the performance of redundancy cache. In the best case it would be disabled if the detection latency is negligible.

5.4 Secure Connection and Group Management

As the CIDSS intends to protect the user network from the cyber attacks, the agent should sniff all internet inbound and outbound traffics. A whole user network can be protected by assigning an agent group. Similarly, different network segments of a user network which are connected to each other using a VPN over the internet can be protected by assigning a single agent group for all segments. Therefore, independent to the network structure, the user network can be protected against cyber attacks using a central IDS in a cloud. Figure 4 demonstrates sample agent configurations.

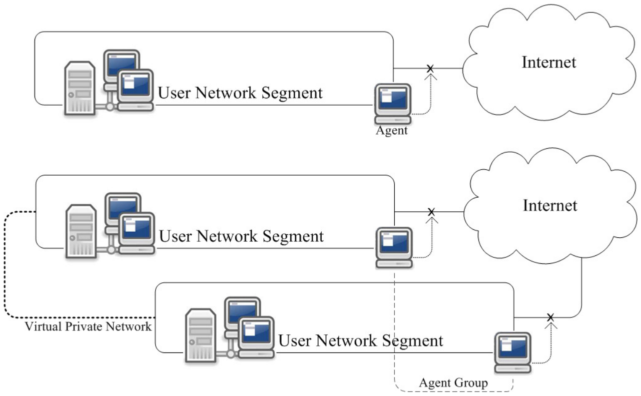

Fig. 4 Sample Agent Configuration in User Local Network.

All of the agents in a single agent group would communicate with a single instance of CCSC. Therefore, similar analysis engine an ultimately same rule-sets would be applied for all agents in an agent group.

5.4 Implementation

The Proof-of-Concept is implemented as a first step in the direction of a mature CIDSS. All components are implemented in a local test bed to evaluate the concept of the proposed structure.

The service would allocate two VMs for CCSC and IDSC components of the CIDSS. A CCSC would be allocated for each agent group. CCSCs would access a database containing all issued credentials while the service ordered by users. These credentials are used by CCSCs while the VPN is established among a agent group members and CCSC to check the validity of a user.

There is single IDSC VM for each user irrespective to the total number of agent groups. It would be equipped with a normal Network-based IDS installation which is working in promiscuous mode on a virtual NIC. All IDSC of all users would access a single database containing preconfigured ready-to-used rule-sets which can be requested by users when a service is requested or can be configured using the agent.

Three steps should be proved to be feasible in the practical situation:

- Capturing packets from the protecting network.
- Secure communication with remote node.
- Detection of Intrusion on the remote node.

Table 2: Virtual Network Interfaces Configurations

	NICs	*NICs Information*	
VM One	1	IP (eth0)	192.168.0.10
		Subnet	255.255.255.0
		Network	192.168.0.0
VM Two	3	(eth0) / (tap0) Bridge	
		IP (eth1)	192.168.1.10
		Subnet	255.255.255.0
		Network	192.168.1.0
VM Three	3	IP (eth0)	192.168.1.11
		Subnet	255.255.255.0
		Network	192.168.1.0
		(eth1) / (tap0) Bridge	
VM Four	1	IP (eth0)	192.168.2.11
		Subnet	255.255.255.0
		Network	192.168.2.0

The testing environment consists of four VMs. The specifications for each VMs is represented in Table 2. According to the table three networks would be deployed for this implementation 192.168.0.0, 192.168.1.0 and 192.168.2.0. ethX are VMware virtual network interfaces which are configured using static IP addresses in Linux operating system. tapX are the Linux Kernel virtual tap interface used by VPN software to interfere with normal multiple layers of ISO OSI model. Using tapX the VPN software could simulate the data-link and physical layer of the communication channel as a virtual medium over the internet. In the remaining part of this section the steps taken is listed:

1. Install a VMware workstation on a PC.
2. Create four VMs with Linux as operating systems.
3. Configure the virtual network using VMware Virtual Network Editor to comply with the information represented in Table 2.
4. Install OpenVPN on VM two and VM three and create the tapX interfaces.
5. Configure the OpenVPN client and server on VM two and VM three.
6. Activate bridging between eth0 and tap0 on VM two and between eth1 and tap0 on VM three.
7. Disable Linux kernel MAC table.
8. Install Snort on VM four.
9. Configure Snort to monitor traffic in promiscuous mode on eth0 on VM four.

Finally to test the proposed structure the attack should be initiated on VM one and the Snort should be capable of generating the intended attack results.

6. Future Work and Conclusion

In this paper we introduce IDS as a Service in a cloud to protect user network. It exploits some characteristics in network traffics that make it possible to extract the required data from the user network for evaluation. This architecture is intended to be scalable by allowing users to tab into different type of IDSCs simultaneously to combine the features of different product for more reliable IDS solution. Different segments of the user network on remote settings could be monitored by same infrastructure which results in ease of deployment of IDS solutions in dynamic environments. The concept is proved to be practical in local network by implementing a simplified version of the proposed architecture.

The implemented Proof-of-Concept is just a first step in the direction of a complex CIDSS. An interesting future topic is the implementation of the fully functional agent on the real internet testbed and cloud infrastructure. To practically apply the deployment, performance and scalability issues need to be considered as the next step. For this purpose, a comparison of the performance between private IDS and our solution is an interesting task.

References

[1] S. Roschke, F. Cheng, and C. Meinel, “Intrusion detection in the cloud,” in 2009 Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing. IEEE, 2009, pp. 729–734.

[2] K. Vieira, A. Schulter, C. Westphall, and C. Westphall, “Intrusion detection for grid and cloud computing,” It Professional, vol. 12, no. 4, pp. 38–43, 2010.

[3] C. Mazzariello, R. Bifulco, and R. Canonico, “Integrating a network ids into an open source cloud computing environment,” 2010.

[4] V. Marinova-Boncheva, “A short survey of intrusion detection systems,” Problems of Engineering Cybernetics and Robotics, vol. 58, pp. 23–30, 2007.

[5] R. Bace and P. Mell, Intrusion detection systems. US Dept. of Commerce, Technology Administration, National Institute of Standards and Technology, 2001.
[6] H. Debar, M. Dacier, and A. Wespi, “Towards a taxonomy of intrusion-detection systems,” Computer Networks, vol. 31, no. 8, pp. 805–822, 1999.
[7] A. Lockhart, Network security hacks, ser. Hacks series. O’Reilly, 2007.
[8] R. Buyya, J. Broberg, and A. Goscinski, Cloud Computing Principles and Paradigms. Wiley, 2011, vol. 81.
[9] P. Mell and T. Grance, “The nist definition of cloud computing,” National Institute of Standards and Technology, vol. 53, no. 6, 2009.
[10] S. Subashini and V. Kavitha, “A survey on security issues in service delivery models of cloud computing,” Journal of Network and Computer Applications, vol. 34, no. 1, pp. 1–11, 2011.
[11] L. Youseff, M. Butrico, and D. Da Silva, “Toward a unified ontology of cloud computing,” in Grid Computing Environments Workshop, 2008. GCE’08. IEEE, 2008, pp. 1–10.
[12] F. Gagliardi, B. Jones, F. Grey, M. Bégin, and M. Heikkurinen, “Building an infrastructure for scientific grid computing: status and goals of the egee project,” Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, vol. 363, no. 1833, p. 1729, 2005.
[13] W. Xin, H. Ting-lei, and L. Xiao-yu, “Research on the intrusion detection mechanism based on cloud computing,” in Intelligent Computing and Integrated Systems (ICISS), 2010 International Conference on. IEEE, pp. 125–128.
[14] H. Debar, D. Curry, and B. Feinstein, “The intrusion detection message exchange format (idmef),” 2007.
[15] G. Tomsho, Guide to Networking Essentials. Course Technology Ptr, 2011.
[16] (2011) Network-Based Intrusion Detection Systems in the Small/Midsize Business. [Online]. Available: http://danielowen.com/NIDS
[17] I. Charitakis, K. Anagnostakis, and E. Markatos, “An active traffic splitter architecture for intrusion detection,” in Modeling, Analysis and Simulation of Computer Telecommunications Systems, 2003. MASCOTS 2003. 11th IEEE/ACM International Symposium on. IEEE, 2003, pp. 238–241.
[18] N. Spring and D. Wetherall, “A protocol-independent technique for eliminating redundant network traffic,” ACM SIGCOMM Computer Communication Review, vol. 30, no. 4, pp. 87–95, 2000.
[19] U. Manber et al., “Finding similar files in a large file system,” in Proceedings of the USENIX winter 1994 technical conference. San Fransisco, CA, USA, 1994, pp. 1–10.
[20] O. Marques and P. Baillargeon, “A multimedia traffic classification scheme for intrusion detection systems,” in Information Technology and Applications, 2005. ICITA 2005. Third International Conference on, vol. 2. IEEE, 2005, pp. 496–501.
[21] (2011) Lossless Gigabit Remote Packet Capture With Linux. [Online]. Available: http://staff.washington.edu/corey/gulp/

Amirreza Zarrabi is master’s degree student in the University Putra Malaysia. His research interests are in computer architecture, device modeling, embedded systems, operating systems, system security, web services, and distributed computing. During his thesis, he was working on distributed memory, file management, and process migration in Linux operating system. Currently he is working on robust transport protocol for dynamic high-speed networks and dynamic servers as a research engineer in an international investment solution provider company in Kuala Lumpur.

Alireza Zarrabi has received his master's degree in information technology from University Malaya in 2010. His area of interest consists of information security, security auditing, vulnerability scanning, penetration testing, operating systems, distributed systems, and cluster computing. During his master's degree he worked on Steganography and currently he is working on cluster computing in enterprise environments as a research engineer in an international investment solution provider company in Kuala Lumpur.