Signature Based Semantic Intrusion Detection System on Cloud

S. Sangeetha, B. Gayathri devi, R. Ramya, M.K. Dharani and P. Sathya

Abstract Now a days, many enterprise applications are using cloud platform. Security is the most sensitive issue in cloud platform. Intrusion detection System is used to protect the Virtual machine from threats. This paper proposes Application level Signature based Semantic Intrusion Detection System, which concentrates on the application level to detect application specific attacks. A packet sniffer is placed between cloud user and Virtual cloud provider. The packets of various protocols are captured by packet sniffer and dispatch it to its corresponding parser. The parser translates a sequence of packets into protocol messages and dispatches the packet to the corresponding state machine which consists of message parsing grammar. The message parsing grammar analyses the messages and checks with the semantic rules. If any signature does not matches with the rule-base and found to be malicious. The IDS interpreter generates alert to the cloud provider. The Signature based semantic Intrusion Detection System reduces the false alarm rate. So, the accuracy of the detection rate gets increased.

Keywords Signature based detection · Network based intrusion detection system · HTTP · FTP · HTML attacks

1 Introduction

Cloud Computing is an emerging technology in the recent years. It provides basic services such as Software as a Service, Platform as a Service, Infrastructure as a service, Security as a Service etc. While providing these services, there exist several issues such as data issues [1, 2]: Data Lock-in, Data Transfer Bottlenecks, Traffic Management, Reputation sharing, security issue: availability of Service, Data

S. Sangeetha (✉) · B. Gayathri devi · R. Ramya · M.K. Dharani · P. Sathya
Department of Computer Science and Engineering, Faculty of Engineering,
Avinashilingam University, Coimbatore, India
e-mail: Visual.sangi@gmail.com

J.K. Mandal et al. (eds.), *Information Systems Design and Intelligent Applications*,
Advances in Intelligent Systems and Computing 339,
DOI 10.1007/978-81-322-2250-7_66

Security, performance issues and energy issue since cloud services are stipulated through Internet. In this, cloud security has gained more importance over the past years with the increase in the number of threats targeting network for information and misuse. Cloud virtualizations runs through standard protocols and are vulnerable to intruders for cyber attacks [3]. Intrusion Detection System plays vital tool for cloud security to prevent outside attacks rather than inside attacks. Attacks are of two types: (1) Active Attacks—likely to change the content. (2) Passive attacks —does not change the content but monitors/listens [4, 5]. The existing IDS works in the network layer which makes the intruder to intrude at application level. Some of the authorized port such as HTTP (80) is kept always open for providing the web services to the cloud user. Other protocols at application layer, such as FTP (21) is not opened always but can be opened when needed. The attack that goes undetected by network layer can be detected by Application level IDS. The efficient IDS technique is incorporated in cloud infrastructure to predict these attacks which works at application level. The classification of IDS and its detection techniques are discussed in the following chapter.

2 Classification of IDS

The Intrusion Detection System has been classified based on the following factors (i) Based on environment and (ii) based on detection techniques. Figure 1 shows the Classification of IDS [1, 6].

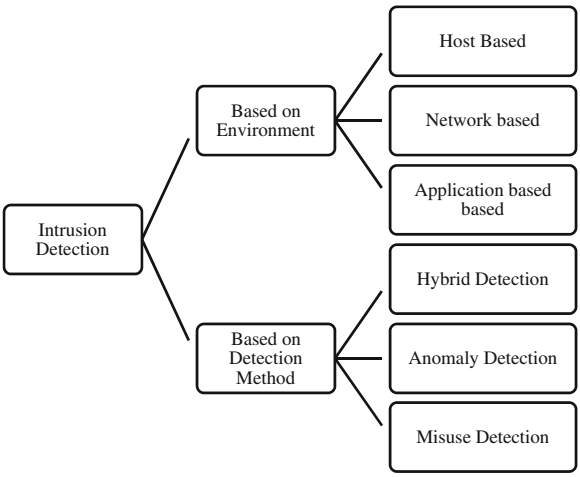

Fig. 1 Classification of IDS

Table 1 Comparison of host based IDS and network based IDS

	HIDS	NIDS
Pros	No extra hardware required	Can monitor multiple system at a time
Cons	Need to install on each machine	It helps only for detecting external intrusions
Deployment	On virtual machine	In virtual cloud provider

2.1 Based on the Environment

IDS has been classified into 3 types [7].

Network based IDS (**NIDS**): It monitors the network traffic and analyses the network for any maliciousness.
Host based IDS (**HIDS**): It monitors the activity of a system and detects the intrusive behavior through monitoring and analyzing log file.
Application based IDS: It analyses the particular application for vulnerability.

The Table 1 shows the comparison of Host based IDS and Network Based IDS.

2.2 Based on Detection Method

Signature based Detection: This method uses signature based pattern matching by comparing the captured patterns with the existing historical data in knowledge base to detect intrusions. It is used to detect known attacks.

Anomaly Detection: In this method, the legitimate users behavior are collected over a period of time and the statistical test will applied on observed behavior to detect any abnormalities. It is used to detect unknown attacks. Apart from statistical test, machine learning based and data mining technique can be used to detect anomalies.

Hybrid Detection: This method enhances the detection rate by combining both misuse and anomaly detection. The misuse detection detects only known attacks and the unknown attacks not detected by the misuse will be detected by anomaly method.

3 Semantic Versus Non-semantic

The packets transferred between cloud users and servers are captured and analyzed for any intrusion. Based on the detection techniques discussed, a signature based intrusion detection system is built to analyze the packets that are expected to be delivered to the network service or application. The intrusions can be detected either semantically or non-semantically.

3.1 Non-semantics Based IDS

Non-semantics based IDS hunted for the patterns in the input traffic and if the pattern matches with some predefined pattern then a intrusion alert will be displayed. An intelligent hacker may intrude the system by simply not using the patterns hunted by the non-semantics based IDS. So, a non-semantics based intrusion detection system fails miserably.

3.2 Semantic Based IDS

A semantics based IDS will define a rule such that the occurrence of some sort of patterns in the network traffic definitely indicates a malicious activity. So there are no false positives (false alarm of an attack) in semantic based IDS. Moreover, the time taken for detecting an attempt is very lesser than non-semantic based IDS since the search space is reduced.

4 Architecture of Cloud IDS

The packets transferred between cloud users and servers are captured by packet sniffer and analyzed for any maliciousness by Cloud IDS Engine [8]. The architecture of the Cloud IDS is shown in Fig. 2. Ethereal is used for protocol analyzer/packet capturing. The Protocol analyzer recognizes the protocol type and dispatches the packet to the corresponding state machine. A protocol analyzer will need to parse messages according to a protocol-specific message format and it reduce the number of false positive and false negative. This needs parsing to be done incrementally, since application-layer messages can be split among several packets. Correct parsing state must be maintained between packets, else partial messages will be analyzed incorrectly. The messages are then analyzed by the message parsing grammar. Semantic Classification tree is constructed by analyzing the specification of the protocol [9]. The specification gives the Rules and the individual patterns which will be matched in the corresponding fields of the protocol [10]. The tree is formed in the top-down format. As each node on the path from the root to a leaf node checks with the input, if any signature does not matches with the rule base then it raises alerts to the cloud IDS Interpreter which in turn alerts the Virtual Cloud Provider. The traffic is continuously monitored and analyzed for any malicious behavior and is reported to the administrator.

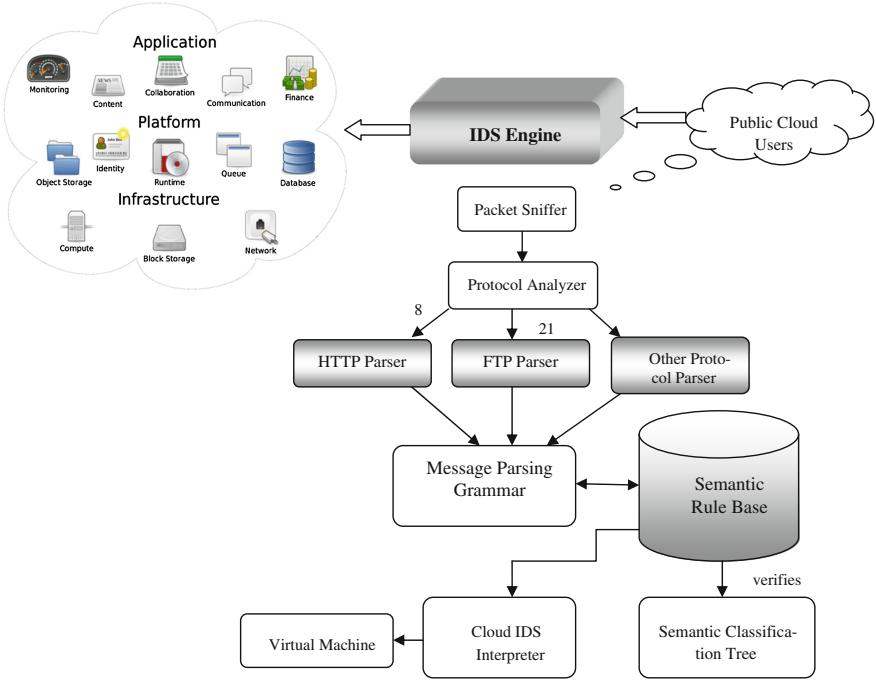

Fig. 2 Architecture of cloud IDS

5 Vulnerabilities at Application Level Protocol

The attackers make the system to be compromised by hacking the protocols such as HTTP, FTP etc.… at application layer. The semantic Classification tree has been generated for the following vulnerabilities if any malicious code matches with the semantic rule base then it raises alerts to the cloud IDS Interpreter which in turn generates alert to the Virtual Machine.

5.1 HTTP Vulnerabilities

The HTTP protocol is application level protocol and is based on the pattern of request/response [11]. A client establishes a connection with a server and sends a request to the server in the form of a request method, URI, and version, followed by a MIME-like message containing request modifiers, client information, and possible body content. The server responds with a status line, including the message's protocol version and a success or error code, followed by a MIME-like message containing server information, entity meta information, and possible body content. Most HTTP communication is initiated by a cloud user agent and consists of a

request to be applied to a resource on server. The HTTP Grammar is constructed. Based on this Grammar, Semantic classification tree is generated and analyses the message for any intrusions.

The following table gives an overview of various HTTP attacks and the extent to which the intruders can compromise the system by gaining information about the system. The HTTP Request syntax: Method/URI/Version. The method can be GET, POST etc. The attacks can be analyzed by looking at the GET method and the corresponding response. The signatures by which the intruders can compromise the system through HTTP Request is shown in Table 2. The HTTP responses are identified by the status codes associated with it and are shown in Table 3.

Table 2 Vulnerabilities at HTTP request

Attack	Impact of the attack
NASM attack	The attacker can make a standard HTTP request that contains 'nasm' in the URI to compiling a variety of sources on various platforms into executable binary files
XTERM	An attacker uses a "xterm" command to open an interactive session then uses that session to move a root kit to the system
CHSH	The attacker can make a standard HTTP request that contains '/bin/chsh' in the URI to change the shell of a user present on the host
"ID"	HTTP request containg '/usr/bin/id' in the URL can return sensitive information on groups and users present on the host
"NETCAT"	An attacker uses a "netcat" command to gain elevated privileges on the server by using netcat to open another connection
"∼" Requests	Using '∼' attackers determine a valid user on the target system. Once an attacker has a valid username, they may try guessing passwords, or brute forcing until they get a valid password
"gcc"	Using 'gcc' in the URI, the attacker can compile a program needed for other attacks on the system or install a binary program of his choosing
"ps"	Using '/bin/ps' in the URI, the attackers can check for various services running on a system to exploit or for the presence of security software, such as host IDS or monitoring scripts
"CHOWN"	Using '/bin/chown' in the URI, attacker can change file permissions of files present on the host
"+" Requests	An attacker or worm will copy cmd.exe to a file inside the web root. Once this file is copied, an attacker has full control over the windows machine
"Requests"	This character shows that there is someone trying a SQL injection attack against the software. Allows an attacker to insert SQL commands into the script
"uname"	Using 'uname' in the URI, attacker gains information on the host operating system

Table 3 Vulnerabilities at HTTP response

Attack	Impact of the attack
OK 200	The request is satisfied
Created 201	The request is success, but the textual part of the response line indicates the URI by which the newly created document should be known
Accepted 202	The request has been accepted for processing, but not completed
Partial information 203	The meta information returned from the server is not a definitive set of object, but is from a private web
No response 204	Server has received the request but no response and the client stay
Error 4xx, 5xx	The 4xx—the client seems to have erred, and the 5xx codes—the server seems to have erred. It is impossible to distinguish these cases
Bad request 400	The request has bad syntax or inherently impossible to be satisfied
Unauthorized 401	This message gives a specification of acceptable authorization schemes
Forbidden 403	The request is prohibited for something. Authorization will not help
Not found 404	The server has not found the URI specified
Internal Error 500	The server encountered an unexpected condition which prevented it from fulfilling the request
Not implemented 501	The server does not support the required facility
Service temporarily overloaded 502	The server cannot process the request due to a high load (whether HTTP servicing or other requests)
Gateway timeout 503	This is equivalent to internal error 500, this shows that the response from the other service has not returned within a time
Redirection 3xx	It indicates action to be taken by the client in order to fulfill the request
Found 302	The data requested actually resides under a different URL, the redirection may be altered on occasion as for "Forward"

5.2 FTP Vulnerabilities

FTP URL Syntax (By RFC 1738)	ftp://user:password@host:port/path
USER	A user name (user id) on the host
PASSWORD	The password corresponding to the user name
HOST	The fully qualified domain name of a network host, or its IP address
PORT	The port number 21

Table 4 shows the vulnerabilities at FTP.

Table 4 Vulnerabilities at FTP

Attack	Impact of the attack
FTP tar attack	This event is generated when an attempt is made to access roots home directory in an ftp session
FTP CWD ~ root attempt	This event is generated when an attempt to abuse an FTP servers functionality and configuration weaknesses is attempted

Table 5 Scripting vulnerabilities

Attack	Impact of the attack
Cross site scripting	The attacker insert malicious script or HTML into a web page, that redirects the user to its own website
SQL injection attack	An intruder to send crafted user name and/or password field that will modify the SQL query. Eg) 'or 1 = 1
Denial of service attack	It denies normal access to legitimate users. This attack can also be in the form of an infinite loop that gets executed in the client's browser [15]
Brute force attack	A brute force attack is a method of defeating a cryptographic scheme by trying a large number of possibilities

5.3 Scripting Vulnerabilities

There are several applications vulnerable to the HTML scripting attacks. The scripting attacks can be found in HyperText Markup Language (HTML). The malicious HTML-based content will be embedded by the attackers within cloud user web requests. Many of the browsers are enabled in default and has the capability to interpret the scripts embedded within HTML content. The attacker can successfully inject the script embedded in HTML, it will be executed by Cloud user [12]. By executing the injected malicious code, the attacker can modify the content or can hack the username and password. The code can be written in any scripting languages. Scripting tags which are used to insert malicious content are <SCRIPT>, <OBJECT>, <APPLET> and <EMBED>. Cross Site Scripting (CSS), SQL Injection, Denial of Service and Brute force are the most common of all attacks in HTML [11] shown in Table 5.

The vulnerabilities mentioned above can be detected by IDS engine. The semantic classification tree will be generated for these vulnerabilities by analyzing the features of request and its corresponding response sequence. The signature that does not matches with the semantic rule base checks for any maliciousness and report to the virtual cloud provider [13, 14]. The complete semantics of the HTTP, FTP transferred (request-response) between cloud user and cloud provider are maintained. The IDS engine checks the pattern of each incoming packet individually for attack.

6 Conclusion

The proposed architecture of cloud IDS at application level protects the vulnerabilities at levels since it protects the below layers. The semantic Classification tree plays vital role by parsing the various protocol message to the IDS Interpreter and has an efficient memory usage since the amount of memory needed for working of the IDS depends on the rule size. The signatures and rules are updated automatically and can be expandable with more semantic parameters. The false alarm rate gets reduced by using Signature based Intrusion Detection System. Hence the vulnerabilities can be detected more accurately.

References

1. Modi, C., Patel, D., Borisaniya, B., Patel, H., Patel, A., Rajarajan, M.: A survey of intrusion detection techniques in cloud. J. Netw. Comput. Appl. **36**, 42–57 (2013)
2. Reddy, V.K., Rao, B.T., Reddy, L.S.S., Kiran, P.S.: Research issues in cloud computing. Glob. J. Comput. Sci. Technol. **11**(11), 1–8 (2011)
3. Subashini, S., Kavitha, V.: A survey on security issues in service delivery models of cloud computing. J. Netw. Comput. Appl. **34**, 1–11 (2011)
4. Qaisar, S., Khawaja, K.F.: Cloud computing: network/security threats and countermeasures. Interdisc. J. Contemp. Res. Bus. **3**(9), 1323–1329 (2012)
5. Oktay, U., Sahingoz, O.K.: Attack types and intrusion detection systems in cloud computing. In: Proceedings of the 6th International Information Security and Cryptology Conference, Sept 2013
6. Scarfone, K., Mell, P.: Guide to intrusion detection and prevention systems (IDPS). NIST special publication, vol. 800, p. 94 (2007)
7. Mazzariello, C., Bifulco, R., Canonoco, R.: Integrating a network IDS into an open source cloud computing. In: Proceedings of the 6th International Conference on Information Assurance and Security (IAS), pp. 265–70 (2010)
8. Roschke, S., Feng, C., Meinel, C.: An extensible and virtualization compatible IDS management architecture. In: Proceedings of the 5th International Conference on Information Assurance and Security, pp. 130–140 (2009)
9. Abbes, T., Bouhoula, A., Rusinowitch, M.: Protocol analysis in intrusion detection using decision tree. In: Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'04) IEEE (2004)
10. Sangeetha S., Vaidehi V.: Fuzzy aided application layer semantic intrusion detection system-FASIDS. In: Proceedings of International Journal of Network Security and its Application (IJNSA April 2010), vol. 2, pp. 39–56 (2010)
11. Bellamy Jr, W.: TCP Port 80—HyperText transfer protocol (HTTP) header exploitation. Cgisecurity.com (2002)
12. Hallaraker, O., Vigna, G.: Detecting malicious javaScript code in mozilla. In: Proceedings of the 10th International Conference on Engineering of Complex Computer Systems (ICECCS 2005), pp. 85–94 (2005)
13. Krugel, C., Toth, T.: Using decision trees to improve signature-based intrusion detection. In: Proceedings of the 6th International Workshop on the Recent Advances in Intrusion Detection (RAID'2003), LNCS, vol. 2820, pp. 173–191 (2003)

14. Vieira, K., Schuler, A., westphall, C.: Intrusion detection techniques in grid and cloud computing environment. In: Proceedings of the IEEE IT Professional Magazine (2010)
15. Bakshi, A.,yogesh, B.: Securing cloud from DDoS attacks using intrusion detection system in virtual machine. In: Proceedings of the 2nd International Conference on Communication Software and Networks, pp. 260–264. (2010)