International Research Journal of Advanced Engineering and Science
ISSN (Online): 2455-9024

A Hybrid Model of Intrusion Detection System in a Cloud Computing Environment

Ogbomo-Odikayor.I.F.[1], Anigbogu. S.O.[2], Edebeatu Dom[3], Anigbogu. G.N.[4]
[1]Department of Physical Science, Benson Idahosa University, Benin City, Nigeria
[2, 3]Department of Computer Science, NnamdiAzikiwe University, Awka, Nigeria
[4]Department of Computer Science, Nwafor Orizu College of Education, Nsugle-Onitsha
Email address: {[1]faithodikayor, [2]dranigbogu, [3]domedebeatu, [4]anigbogugloria} @yahoo.com

Abstract*—Intrusion is one of the important issues in all network especially in cloud computing where all the services are served using internet. Intrusion is the process of entering into a network without proper authentication. Intrusions are detected using intrusion detection systems. In the existing system, the audited Data has to traverse a long path from its origin to the intrusion detection system (IDS) and in the process can potentially be destroyed or modified by an attacker. The system can also become vulnerable when connected to the internet. This study seek to develop an intrusion detection model that can detect violations, using the IP(Internet protocol)/MAC (Media access control) address at the point of entering the network of a cloud based system. The methodology adopted for this work is Object-Oriented Hypermedia Method (OOHDM) and the programming language used are PHP, JavaScript, CSS and MySQL. The new system brings about a new method of detecting intruders by the combined use of IP/MAC address*.

***Keywords*—** *Intrusion,detection, Internet, protocol, Media, access, control, computing*.

I. INTRODUCTION

An intrusion detection system is an application used for monitoring the network and protecting it from intruder. With the rapid progress in the internet based technology, new application areas for computer network have emerged. All of these application areas made the network an attractive target for abuse and a big vulnerability for the community. As internet is expanding with the society, new security issues like viruses and worms are imported. Intruders use different techniques like cracking of password, detecting unencrypted text to cause vulnerabilities to the system. Hence, an enhanced form of security is needed for the users to secure their system from the intruders. In cloud computing, computing resources such as data and software are managed by service providers on the internet and are provided to users and their devices in an on-demand manner. The data that is being stored in the cloud has to be highly secured. The data has to be secured from various vulnerabilities such as hacking, intrusions etc. Intrusion is one of the important issues in all the networks, especially in Cloud computing where all the services are provided via Internet. The term intrusion can be defined as the process of entering into a network without proper authentication.

1.1 Types of intrusion detection system in cloud environment

Network Based (Network IDS)

Network based Intrusion Detection System (NIDS) is an intrusion detection system that tries to detect malicious activity such as DoS attacks, port scans etc. by monitoring network traffic. The information collected from network is compared with known attacks for intrusion detection or intruders by comparing current behaviour with already observed behaviour in real time.

Host Based (HIDS)

Host based intrusion detection attempts to identify unauthorized, illicit, and anomalous behavior on a specific device. HIDS generally involves an agent installed on each system, monitoring and alerting on local operating system (OS) and application activity. The installed agent uses a combination of signatures, rules, and heuristics to identify unauthorized activity.

Distributed Intrusion Detection System (DIDS)

A Distributed IDS (DIDS) consists of several IDS (E.g. HIDS, NIDS etc.) over a large network, all of which communicate with each other, or with a central server that enables network monitoring. The intrusion detection components collect the system information and convert it into a standardized form to be passed to the central analyzer. Central analyzer is a machine that aggregates information from multiple IDS and analyzes the same.

Hypervisor-based Intrusion Detection System

A Hypervisor-based intrusion detection system is an intrusion detection system specifically designed for hypervisors. Hypervisor is a platform to run VMs (Virtual machines). This type of IDS allows user to monitor and analyze communications between VMs and hypervisor, and within the hypervisor based virtual network. Hypervisor based IDS is one of the important techniques, specifically in Cloud computing, to detect intrusion in virtual environment (Gupta et al., 2012).

1.2 Intrusion Detection Techniques

Signature based IDS

Signatures are created by vendors based on potential attacks and attacks that have taken place in the past. In other words, signature based IDS will monitor packets on the network and compare them against a database of signatures or attributes from known malicious threats. This is similar to the way most antivirus software detects malware. Any packets arriving into the network are compared to the set of downloaded signatures comparing these for any attacks. A limitation of signature-based detectors is that they can only detect those attacks they know about. They must also be constantly updated with signatures of new attacks.

Ogbomo-Odikayor.I.F., Anigbogu. S.O., Edebeatu Dom, and Anigbogu. G.N., "A hybrid model of intrusion detection system in a cloud computing environment," *International Research Journal of Advanced Engineering and Science*, Volume 3, Issue 3, pp. 194-200, 2018.

Anomaly based IDS

The anomaly intrusion detection is based on defining the network behaviour. Instead of looking for matches, anomaly intrusion detection looks for behavior that is suspicious (Pfleeger et al., 2003).Anomaly-based IDS attempt to characterize normal operation, and try to detect any deviation from normal behavior (Stillerman et al., 1999).The good thing about this type of system is that it can detect new attacks. It does not need to rely on signatures.

II. Related Works

In the earlier works on intrusion detection system, Patel et al., (2013) has proposed a model for intrusion detection system which combines the concept of autonomic computing, fuzzy logic, and ontology and risk management. Patel et al., (2013) also proposed the characteristics of the intrusion detection system for cloud environment. The method or algorithm for such proposal was not proposed by the author. The implementation of the proposed concept was the future scope of the paper.

Kleber et al. (2010) have proposed a Hybrid Intrusion Detection System for Cloud and Grid environment. This system can detect any kind of attack but the efficiency of detection is reduced. This system cannot be deployed into a real time distributed environment, as the system cannot synchronize well with the other Intrusion Detection Systems in the network and the architecture and working of both models is completely different.

Tupakula et al. (2011) have proposed a Hybrid Intrusion Detection System for Infrastructure as a Service Cloud. This system cannot handle large scale, dynamic, multithread and data processing environment. Since the system has been proposed for Infrastructure as a Service Cloud, the synchro-nization character is not applicable to the system. Software as a Service and Platform as a Service are the other two services of cloud, which has not been considered by the authors.

Kholidy et al. (2012) have proposed a framework for Intrusion Detection in Cloud Systems. This framework does not detect the intrusion in a faster manner; thereby affecting the efficiency of system. This system can partially only handle large scale, dynamic data which is another drawback of the system. The authors did not narrate the scope for implementing the algorithm for the private cloud environment.

Xinwang et al. (2010) has proposed and developed an Intrusion Detection System for cloud with the central management approach. The developed model is not scalable. The efficiency of the system gets reduced when the system is scaled. The implementation of the developed system is quite complicated and difficult to manage.

Hwang et al. (2007) proposed a hybrid system that combines a signature-based IDS with an anomaly detection system in a cascade structure, achieving twice the detection accuracy of IDS only system.

Roschke et al. (2009) have proposed an integration solution for central IDS management that can combine and integrate various renowned IDS sensors and output reports on a single interface. The authors have proposed an effective cloud IDS management architecture, which could be monitored and administered by the cloud user. They have provided a central IDS management system based on different sensors using the intrusion detection message exchange format (IDMEF) standard for communication between different IDS sensors.

Irfan and Hussain, (2011) proposed an IDS service at cloud middleware layer model, which has an audit system designed to cover attacks that NIDS and HIDS cannot detect. The authors have tested their IDS prototype with the help of simulation and found its performance satisfactory for real-time implementation in a cloud environment. Although the security policies compliance check for cloud service provider and their reporting procedures to cloud users was not discussed.

Kleber and schulter, (2010) presented a framework of IDS for Cloud computing network that could reduce the impact of attacks. The implementation results indicate that the proposed system could resist DoS attack. Moreover, by comparison, the proposed cooperative IDS system only increases little computation effort compared with pure Snort based IDS that can prevent the system from single point of failure attack.

Narwane and Vaikol, (2012) proposed a system to detect intrusions in the cloud computing using Behavior-based approach and knowledge-based approach. If first approach is unable to detect the data, second approach again verifies the data and compare it with the signatures within the database. The proposed system can have very low false positive alarm.

Amirreza and Alireza, (2012) introduces 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.

This work explored the combination of internet protocol address (IP address) and Media Access Control (MAC) address to deal with some of the drawbacks in the existing intrusion detection system as identified in the literature reviewed.

III. System Methodology, Design and Implementation

3.0. Methodology

The software development methodology adopted for this work is Object Oriented Hypermedia Design Method (OOHDM). This methodology is adopted because it uses abstraction and composition mechanisms in an object oriented framework to, on one hand, allow a concise description of complex information items, and on the other hand, allow the specification of complex navigation patterns and interface transformations.

3.1. System Design

3.2.1. Specification of User Interaction Diagram

The user interaction diagram (UID) is a diagrammatic tool that captures and specifies the interaction between the users and the system (Dayanand et al., 2012). And its purpose is "to represent the interaction between the user and the system. UID is a tool used mainly to support the communication between the designer and users in requirements gathering. The user

Ogbomo-Odikayor.I.F., Anigbogu. S.O., Edebeatu Dom, and Anigbogu. G.N., "A hybrid model of intrusion detection system in a cloud computing environment," *International Research Journal of Advanced Engineering and Science*, Volume 3, Issue 3, pp. 194-200, 2018.

International Research Journal of Advanced Engineering and Science
ISSN (Online): 2455-9024

interaction diagram (UID) of the system describes only the exchange of information between the system and the user as shown in figure 1.

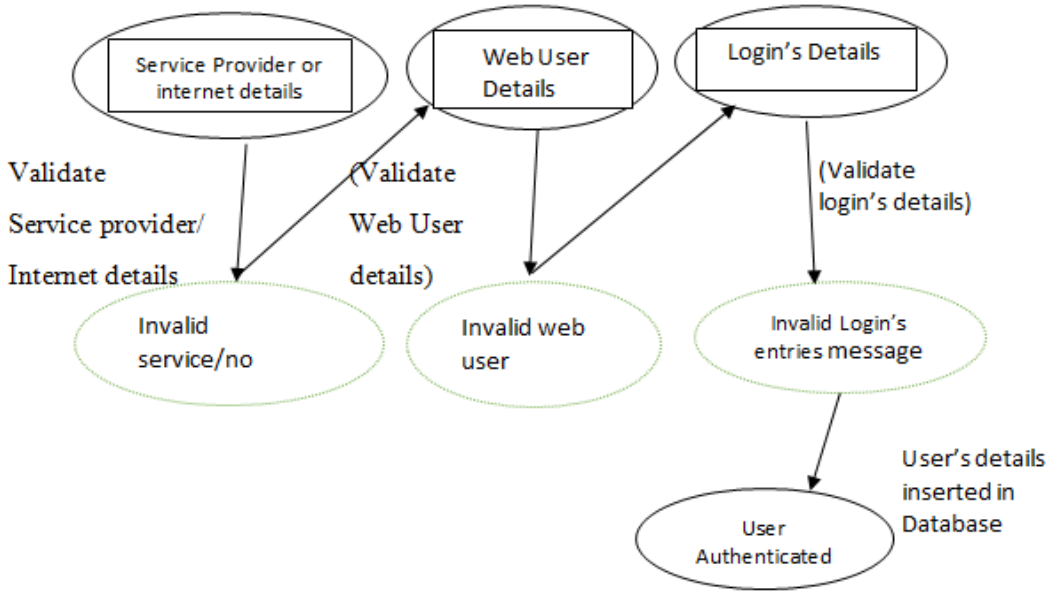

Fig. 1. User interaction diagram.

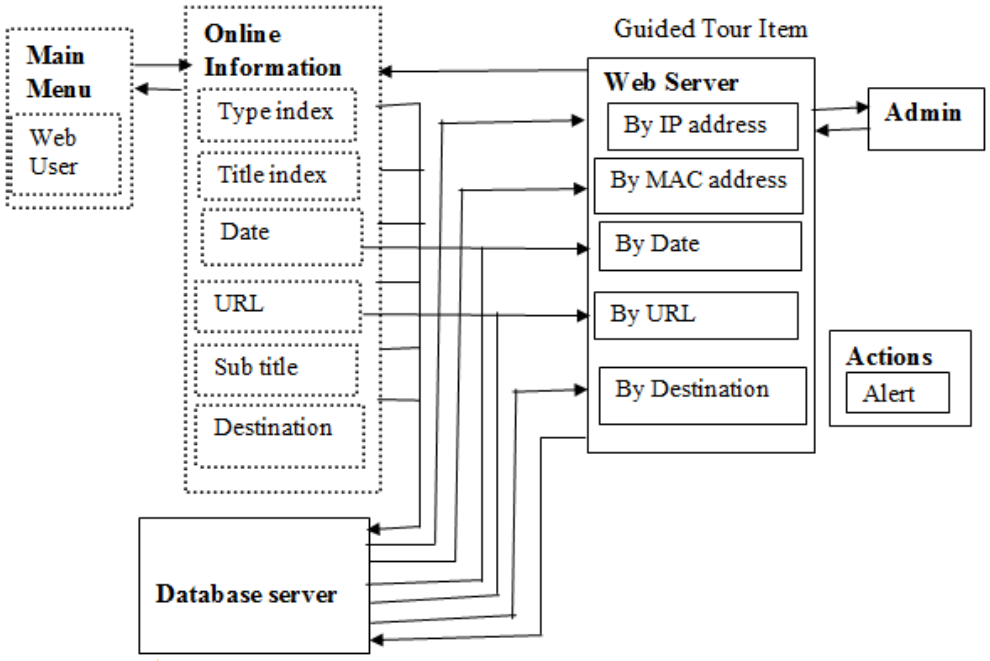

Fig. 2. Navigation context schema.

Specification of the Conceptual Schema

The starting point of the design process is the specification of a conceptual model. The target of this model is the enumeration of the object classes of the domain, their attributes and their relationships. The output of this stage is the Conceptual Class Schema, a graph of classes with the notation specified in Unified model language (UML). The Conceptual Class Schema shows the underlying structure of the information that is going to be presented to the users, independently of the presentation form. This schema has more relationship to the way the data will be stored than to the way it will be presented.

Navigational Design

Here we describe the navigational structure of a hypermedia application in terms of navigational contexts, which are induced from navigation classes such as Nodes, Links, Indices, and Guided Tours. Navigational contexts and classes take into account the types of intended users, and the set of tasks they are to perform using the system. Nodes in OOHDM represent logical "windows" (or views) on

Ogbomo-Odikayor.I.F., Anigbogu. S.O., Edebeatu Dom, and Anigbogu. G.N., "A hybrid model of intrusion detection system in a cloud computing environment," *International Research Journal of Advanced Engineering and Science*, Volume 3, Issue 3, pp. 194-200, 2018.

conceptual classes defined during domain analysis. Different navigational models may be built for the same conceptual schema to express different views on the same domain. Links are derived from conceptual relationships. And defining the navigational semantics in terms of nodes and links we can reason about the navigational space (i.e. the set of nodes the user can interact with) independently of the conceptual model.

Navigational Context Schema

Navigational Context Schema as illustrated in figure 2 shows how navigable objects are clustered together and navigated. Dashed boxes show access structures (indexes) while boxes inside Class Web Server indicate possible context in which a packet can be accessed.

Abstract Interface Design

Abstract Interface View (ADV) are objects that have a state and an interface, where the interface can be exercised through regular functional or procedural calls or input and output events as illustrated in figure 3. ADVs are abstract in that they only represent the interface and the state, and not the implementation. ADVs are usually used to represent interfaces between two different media such as a user, a network or a device such as a timer or an interface between two or more Abstract Data Objects (ADOs) where ADOs are ADVs that do not support events.

The abstract interface model is built by defining perceptible objects in terms of interface classes. These classes are defined as aggregations of primitives ones (such as text fields, buttons, etc.) or other interface classes. Interface objects provide navigational objects with a perceptible appearance. The behavior of the interface is given by specifying how to handle external and user-generated events and how communication takes place between the interface and navigational objects. Once the navigational structure has been defined, it is made perceptible to the user through the application interface, which is done in this step by defining an abstract interface model. This means defining which interface objects the user will perceive, and in particular, the way in which different navigational objects will appear again, it also defines which interface objects will activate navigation, the way in which multimedia interface objects will be synchronized as well as which interface transformation will take place. Therefore, to specify the way the user will perceive the navigational objects, OOHDM uses the Abstract Data View (ADV) design approach. ADV allow defining the interface appearance of navigational objects and of other useful interface objects (such as menus bars, buttons and menus).

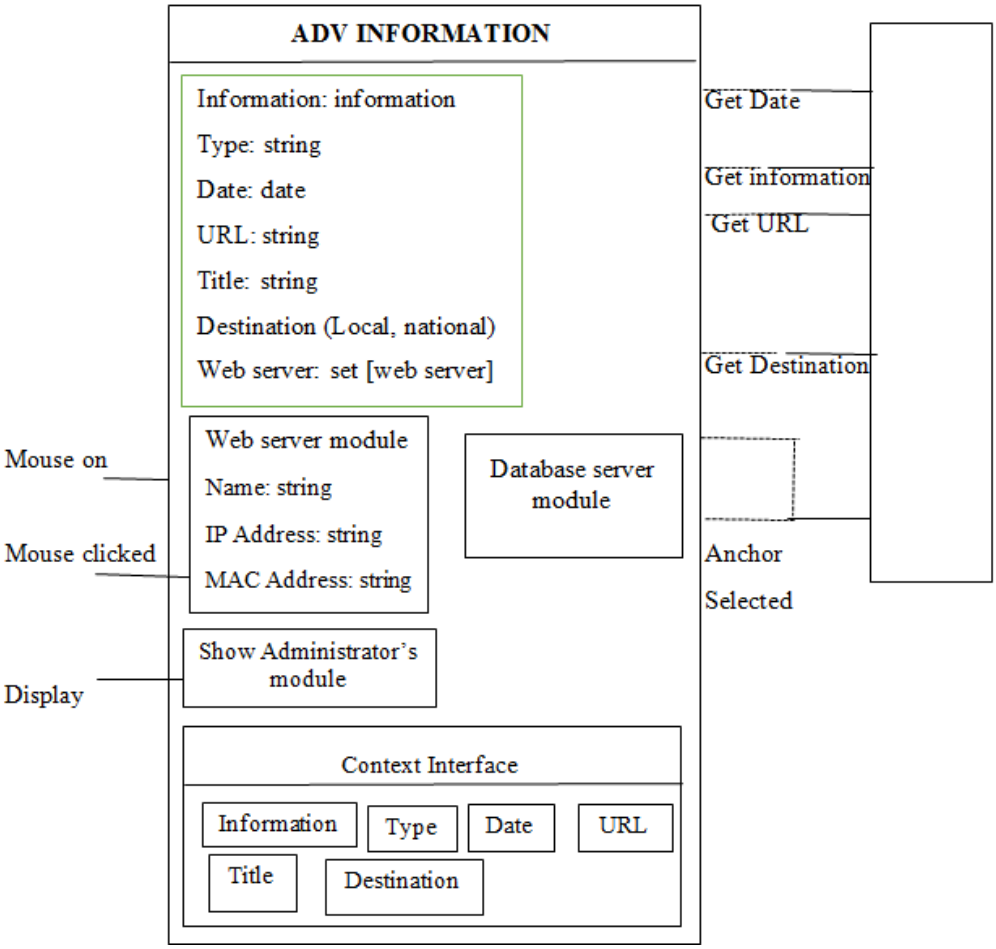

Fig. 3. Configuration diagram for ADV- Information.

Ogbomo-Odikayor.I.F., Anigbogu. S.O., Edebeatu Dom, and Anigbogu. G.N., "A hybrid model of intrusion detection system in a cloud computing environment," *International Research Journal of Advanced Engineering and Science*, Volume 3, Issue 3, pp. 194-200, 2018.

International Research Journal of Advanced Engineering and Science
ISSN (Online): 2455-9024

Dashed lines are required services while full lines are provided services. Attributes *Information, type, date, title*, act as static interfaces objects. That is, they do not react to external events. However nested Abstract Data Views *Database server module, Administrator's module, Web server module* exhibits a user perceivable behavior. For example when the user clicks on *Show Administrator's module* the ADV Administrator's module is displayed. Context interface is in turn composed of nested ADV's implementing anchors for context navigation.

ADV (Abstract Data Views) Charts

ADV (Abstract Data Views) charts provide a visual schema for the specification of the dynamic aspects of the user interface. They contain one or more states and transitions as well as may contain attributes and other ADV´s to describe their behavior. They are an extension of State charts and Object charts supporting nesting of states and ADVs. Nesting of states expresses behavioural nesting while nesting of ADVs is the expression of structural nesting. As in State charts, the different states are represented with boxes with rounded corners and the transitions between states with arrows from one state to another. The event-field specifies the event that will fire the transition (e.g. mouse click).This dynamic description is specified using ADV-Charts that extend state charts to user interface specification. We can specify the behavior of each nested ADV using different state chart or use a concise specification as shown in figure 3 .For example, ADV "Information" can react to external events Mouse on, Mouse clicked and Display, while node *information* provides *Get Date*, *Get information*, Get *URL*, *Get Destination* and *Anchor Selected*. When both ADV and node attributes have the same name and type, we can omit specifying services like "Get Date" because it can be unambiguously understood by implementers. Note that Data server module does not trigger navigation, as they just implement local interface behavior. In figure 3 some interface objects (such as *Database Server module* may have embedded anchors triggering navigation. In this case, the owner(*node information*) is sent the message *Anchor Selected* with corresponding anchor as argument and the interface objects removed from the perception context("Information" in state "off") as shown in figure 4.

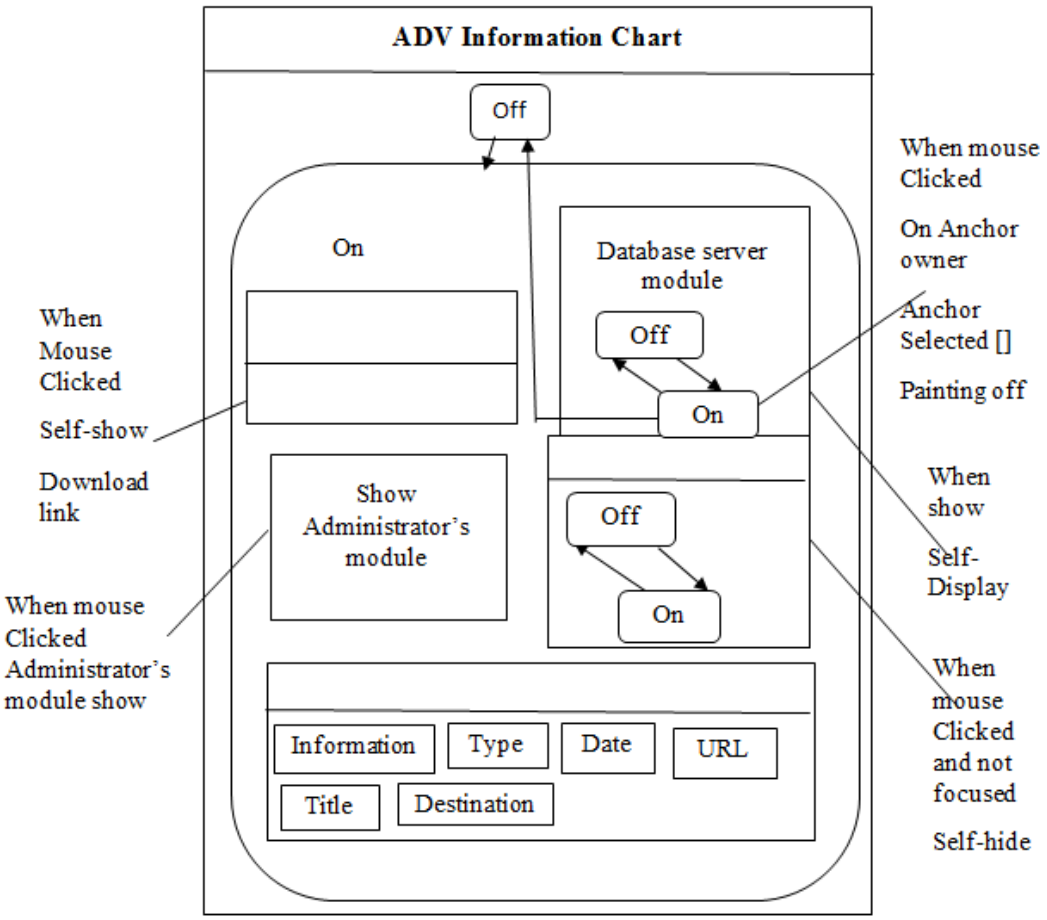

Fig. 4. ADV Chart for ADV Information chart.

Ogbomo-Odikayor.I.F., Anigbogu. S.O., Edebeatu Dom, and Anigbogu. G.N., "A hybrid model of intrusion detection system in a cloud computing environment," *International Research Journal of Advanced Engineering and Science*, Volume 3, Issue 3, pp. 194-200, 2018.

The Model View of the New System

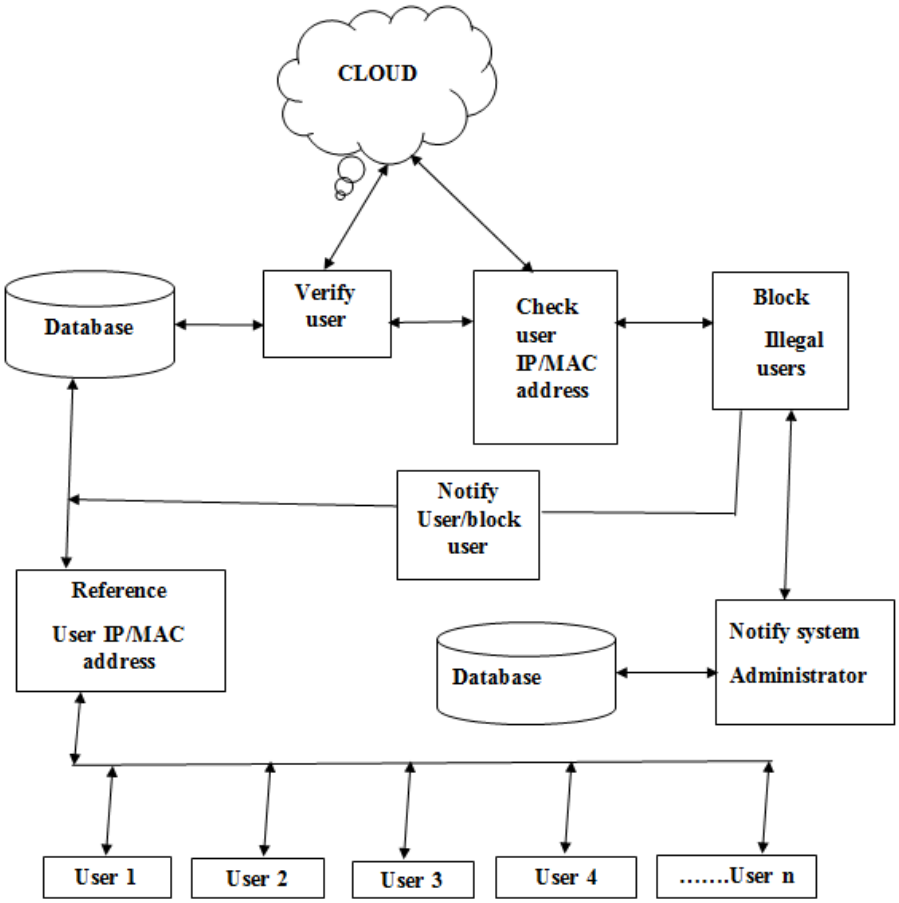

Fig. 5. Model view of the new system.

In the model view of the new system, as shown in figure 5; user sends request through a web browser commonly called browser. So many request may be sent. The request is passed to the system where they are verified and checked for the right IP (internet protocol)/MAC (Media access control) address at the point of entering the network before being stored in the database server. The database server performs the services of receiving request, storing the IP/MAC address of systems that connect to the network and referencing users IP/MAC address. The database server module is created using WampServer. WampServer is a windows web development environment. It allows you to create web applications with Apache, PHP and MySQL database. If intrusions are detected, the illegal users are blocked and the system administrator as well as the users are notified inform of an alert.

3.3 Implementation

Computer system is made up of several units that are put together to work as one in order to achieve a common goal. The system requirements for implementation of the new system include: The Hardware and Software

Hardware Requirements

Hardware requirements for the project which is designed to run on a standalone computer and/or on server are as follows:

a. Processor- Pentium III or Pentium ® Dual core 2.0GHz or higher
b. Memory- 3.00GB of Random Access Memory (RAM). The physical total size of the software is 2.0MB and the database with samples user details is 1.5KB and is expected to increase when opened for commercial use.
c. A minimum of 1.0GB of free hard disk space.
d. A network Interface Card(NIC) or Wireless Local Area Network(WLAN) for network connectivity.
e. A standard Video Graphic Array (VGA)
f. A monitor for display.
g. Keyboard and mouse for input and pointing
h. CD-ROM drive or external CD-ROM drive

Software Requirements

The software requirements for the developed software include:

a. Operating system (OS) – Windows 8 or higher version, Unbutu
b. Apache Server 2.2.17 or higher
c. MySQL5.5.8 or higher
d. PHP 5.3.5 Interpreter or higher
e. Good Web browser software.
f. Reliable access to the interne

System Testing

In testing, we check both verification (the software complies with the requirements) and validation (the software has been written correctly and effectively). The software is also tested against its analysis specification.

There are different types of tests adopted at different stages. They include:

i) Unit Testing – testing each class or unit of the software interface.
ii) Integration Testing – testing done during the combination of various class and various modules for compatibility check.
iii) Module/Sub-System Testing – Testing done when on a module before integration.
iv) System Testing – testing after the combination of the various modules or subsystems to produce the required software.
v) Acceptance Testing – In this stage, people who work in an academic environment that uses a private cloud with full internet access were invited to do the acceptance testing.

Two different software testing techniques were adopted as a systematic testing approach these include:

i) White Box Testing – This technique focused on the program control structures which involved close examination of procedure derail. Program statements, internal data structure, loop, logical paths and logical statements are tested. White box testing helped us to test the quality of the construction of the software.

ii) Black Box Testing –This technique tested the quality of the performance of the software and is conducted at the software interface. It tested the functionality of the system.

The aim of the two test technique conducted was to ensure that the software has the following attributes, namely, Completeness, Correctness, Reliability and possibility of maintenance.

IV. Result And Conclusion

The system developed has introduced a new method of detecting intruders by the combined use of IP (Internet protocol)/MAC (Media Access Control) address and can effectively protect the computer systems against intrusions at the point of entering the network.

There has been a great need over the years for an intrusion detection model due to the widespread proliferation of computer networks which has resulted in the increase of attacks on information systems. These attacks are used for illegally gaining access to unauthorized information, misuse of information or to reduce the availability of information to authorized users. These attacks vary widely and can affect different security requirements. Thus the need for complete protection of organizational computing resources has been driving the attention of people towards intrusion detection system. This work has modeled the solutions to deal with the intricate problems of illegal intrusion of network in a cloud computing environment

References

[1] Amirreza Zarrabi and Alireza Zarrabi, "Internet intrusion detection system service in a cloud," *IJCSI International Journal of Computer Science Issues*, vol. 9, issue 5, no 2, pp. 308-315, 2012.
[2] Dayanand Ingle and Dr. B. B. Meshram, "Hybrid analysis and design model for building web information system," *IJCSI International Journal of Computer Science Issues*, vol. 9, issue 4, no 3, pp. 516-535, 2012.
[3] S. Gupta, S. Horrow, and A. Sardana, "A hybrid intrusion detection architecture for defense against DDoS attacks in cloud environment," *Contemporary Computing Communications in Computer and Information Science*, vol. 306, pp. 498-499, 2012.
[4] K. Hwang, M. Cai, Y. Chen, S. Member, and M. Qin, "Hybrid intrusion detection with weighted signature generation over anomalous internet episodes," *IEEE Transactions on Dependable and Secure Computing*, vol. 4, issue 1, pp. 1-15, 2007.
[5] Irfan Gul and M. Hussain, "Distributed cloud intrusion detection model," *International Journal of Advanced Science and Technology*, vol. 34, pp. 71-82, 2011.
[6] V. Kleber, A. Schulter, C. Westphall, and C. Westphall, "Intrusion detection for grid and cloud computing," *IEEE Journal: IT Professional*, vol. 12, issue 4, pp. 38-43, 2010.
[7] Hisham A. Kholidy and F. Baiardi, "CIDS: A framework for intrusion detection in cloud systems," *Proceedings of 9th IEEE International Conference on Information Technology-New Generations*, pp. 379–85, 2012.
[8] C. Pfleeger and S. Pfleeger, *Security in Computing*, Prentice Hall, 2003.
[9] A. Patel, M. Taghavi, K. Bakhtiyari, Celestino J, Junior, "An intrusion detection and prevention system in cloud computing: A systematic review," *Journal of Network and Computer Applications*, vol. 36, issue 1, pp. 25–41, 2013.
[10] Roschke Sebastian, Cheng Feng, and Christoph Meinel, "Intrusion detection in the cloud," *Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing*, 2009.
[11] U. Tupakula, V. Varadharaja, and N. Akku, "Intrusion detection techniques for infrastructure as a service cloud," *Proceedings of 9th IEEE International Conference on Dependable, Autonomic and Secure Computing*, pp. 744–751, 2011.
[12] M. Stillerman, C. Morceau, and M. Stillman, "Intrusion detection for distributed application," *Communication of the ACM*, vol. 42, issue 7, pp. 62-69, 1999.
[13] Shimonski, What You Need to Know About Intrusion Detection Systems, 2002. http://www.windowsecurity.com/articles-tutorials/intrusion_detection/What_You_Need_to_Know_About_Intrusion_Detection_Systems.html. Retrieved 12th Februarys 2014.
[14] Matthew, Ernst and Young, 2015.
[15] SANS Boston: Intrusion Detection FAQ: What is Intrusion Detection? http://www.sans.org/security-resources/idfaq/what_is_id.php. Retrieved 3rd April 2015.
[16] S. V. Narwane and S. L. Vaikol, "Intrusion Detection System in Cloud Computing Environment," *International Conference on Advances in Communication and Computing Technologies (ICACACT) Proceedings published by International Journal of Computer Applications (IJCA)*, 2012.
[17] W. Xin, H. Ting-Lei, and L. Xiao-Yu, "Research on the intrusion detection mechanism based on cloud computing," *Proceedings of International Conference on Intelligent Computing and Integrated Systems*, Guilin, pp. 125–128, 2010.