IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 1 Issue 3, May 2014.
www.ijiset.com

ISSN 2348 – 7968

Defense-in-Depth Intrusion Detection Framework for Virtual Network Systems

R.Rohini[1], M. Kamarajan[2]
[1]R.V.S College of Engineering and Technology, Dindigul, rrohini07@gmail.com.
[2]R.V.S College of Engineering and Technology, Dindigul.

***Abstract*— Cloud security is most significant problems that possess drawn a lot of analysis along with growth hard work within past few years. Specially, attackers can easily examine vulnerabilities of a cloud system along with skimp on exclusive products to be able to use more large-scale Distributed Denial-of-Service (DDoS). DDoS problems generally contain early point measures for instance multistep exploitation, low-frequency vulnerability checking, along with limiting identified somewhat insecure exclusive products because zombies, lastly DDoS problems over the sacrificed zombies. Inside the cloud system, especially the actual Infrastructure-as-a-Service (IaaS) clouds, the actual prognosis associated with zombie exploration problems is exceedingly hard. This is because cloud users might deploy somewhat insecure purposes on the exclusive products. To counteract somewhat insecure exclusive products coming from a staying sacrificed within the cloud; we propose multiphase distributed vulnerability detection, measurement, along with countermeasure variety mechanism known as NICE, which can be constructed in invasion graph-based analytical designs along with reconfigurable exclusive network-based countermeasures. The proposed framework leverages Open Flow network programming APIs to build a monitor and control plane over distributed programmable virtual switches to significantly improve attack detection and mitigate attack consequences. The device along with safety measures critiques display the actual performance along with performance on the suggested answer. In this particular document we suggested Autonomous realtor intended for Intrusion prognosis system.**

I. INTRODUCTION

Network and security management has to assure uninterrupted access to the communication infrastructure. With growing networks and increasing amount of transported data, it gets more and more complicated to supervise the operation of the communication systems. Sometimes computer networks are not well protected against attacks from the outside, so additional surveillance may be necessary. But even well protected networks need surveillance. A lot of these networks are threatened from the inside. Intrusion Detection Systems (IDSs) [1][2]help securing these networks. This paper focuses on a tool for visualizing and detecting anomalies of the traffic structure. Several distributed Denial of Service [3] attacks have shown the necessity of better protecting computers and networks connected to the Internet. Due to widely available attack tools, attacks of this kind can be carried out by persons without in-depth knowledge of the attacked system. Insufficient protected Open University networks are an example for networks that need additional surveillance. These networks often include vulnerable computers and offer high bandwidth connections to the Internet. These features are the reason why attackers are interested in these networks. The machines in these networks are not the goal of the attacks. Normally, they do not contain interesting information for the attacker, but they are suitable for scanning other networks and starting (for example) Denial of Service attacks.

The Internet is increasingly important as the vehicle for global electronic commerce. Many organize- tins also use Internet TCP/IP protocols to build intra-networks (intranets) to share and disseminate internal information [4][5]. A large scale attack on these networks can cripple important world-wide Internet operations. The Internet Worm of 1988 caused the Internet to be unavailable for about _vet days Seven years later, there is no system to detect or an- laze such a problem on an Internet-wide scale[6]. The development of a secure infrastructure to defend the Internet and other networks is a major challenge. In this paper, we present the design of the Graph-based Intrusion Detection System (Girds).

Girds' design goal is to analyze network activity on TCP/IP networks with up to several thousand hosts. Its primary function is to detect and anal- lyre large-scale attacks, although it also has the ca- ability of detecting intrusions on individual hosts. Girds aggregates network activity of interest into Activity graphs, which are evaluated and possibly re- ported to a system security ocher (SSO)[7][8]. The hiker- archival architecture of Grids allows it to scale to large networks. Grads is being designed and built by the authors using formal consensus decision-making and a well-documented software process. We have completed the Grids design and have almost _knishes building a prototype.

This paper is organized as follows. Brier describes related work on intrusion detection systems and motivates the need for Grids'. In the simple Grids' detection algorithm is described, followed by a more detailed discussion in has a treat- meant of the hierarchical approach to scalability and discusses how the hierarchy is managed. Outlines the policy language covers some limitations of Girds. Finally, presents conclusions and discusses future work.

Network security is a complicated subject, historically only tackled by well-trained and experienced experts. However, as more and more people become ``wired", an increasing number of people need to understand the basics of security in a networked world.

II. RELATED WORK

In [8] presents overview of cloud computing. Cloud Computing refers to both the applications delivered as services over the Internet and the hardware and systems software in the datacenters that provide those services. The services themselves have long been referred to as Software as a Service (Seas), so we use that term. The datacenter hardware and software is what we will call a Cloud. When a Cloud is made available in a pay-as-you-go manner to the public, we call it a Public Cloud; the service being sold is Utility Computing. Current examples of public Utility Computing include Amazon Web Services, Google Apennine, and Microsoft Azure. We use the term Private Cloud to refer to internal datacenters of a business or other organization that are not made available to the public.

A new symbolic model checker [] describes the results of a joint project between Carnegie Mellon University (CMU) and Institute per la Ricers Sciatica e Technologic (IRST) whose goal is the development of a new symbolic model checker.1 the new model checker, called NUSMV, is designed to be a well structured, open, _exiles and documented platform for model checking. To be usable in technology transfer projects, NUSMV was designed to be very robust, easy to modify, and close to the standards required by industry. NUSMV is the result of the reengineering and reimplementation of the CMU SMV symbolic model checker. With respect to CMU SMV, NUSMV has been upgraded along three dimensions.

Paul Ammann et al [3] proposed Scalable, Graph-Based Network Vulnerability Analysis
The basic observation behind this paper is that attack graphs can easily be far too large to be practical. Paper provides some support for our position on this: in a scaling exercise with 5 hosts, 8 exploits, and the vulnerabilities associated with those exploits, Mums reportedly Took 2 hours to execute, with most of that time spent on graph manipulation. The resulting attack graph had 5948 nodes and 68364 edges. The state space in that example was represented with 229 bits. By contrast, to encode such a problem with the methods Presented in this paper, we need, at most, 229 nodes, one for each bit in the state representation. Each of these nodes must be able to store a constant amount of information about however many exploits can change the value of that particular node from 'false' to 'true'. It is clear that our structure is dramatically smaller, even for this relatively limited, from a real world perspective, example.

Hassan et al [4] describes security and privacy challenges in cloud computing Environments. Nayot Poolsappasit et al [9] proposed Dynamic Security Risk Management Using Bayesian Attack Graphs. Security risk assessment and mitigation are two vital processes that need to be executed to maintain a productive IT infrastructure. On one hand, models such as attack graphs and attack trees have been proposed to assess the cause-consequence relationships between various network states, while on the other hand, different decision problems have been explored to identify the minimum-cost hardening measures. However, these risk models do not help reason about the causal dependencies between network states. Further, the optimization formulations ignore the issue of resource availability while analyzing a risk model. In this paper, we propose a risk management framework using Bayesian networks that enable a system administrator to quantify the chances of network compromise at various levels. We show how to use this information to develop a security mitigation and management plan. In contrast to other similar models, this risk model lends itself to dynamic analysis during the deployed phase of the network. A multi-objective optimization platform provides the administrator with all trade-off information required to make decisions in a resource constrained environment.

In [15] the alert Correlation Algorithm Based on Attack Graph was proposed. As modern attacks are getting more sophisticated and the number of sensors and network nodes grows, the problem of false positives and alert analysis becomes more difficult to solve. Alert correlation was proposed to analyze alerts and to decrease false positives. Knowledge about the target system or environment is usually necessary for efficient alert correlation. For representing the environment information as well as potential exploits, the existing vulnerabilities and their Attack Graph (AG) is used. It is useful for networks to generate an AG and to organize certain vulnerabilities in a reasonable way. In this paper, we design a correlation algorithm based on AGs that is capable of detecting multiple attack scenarios for forensic analysis. It can be parameterized to adjust the robustness and accuracy. A formal model of the algorithm is presented and an implementation is tested to analyze the different parameters on a real set of alerts from a local network.

Intrusion Detection Systems (IDS) have been proposed for years as an efficient security measure and is nowadays widely deployed for securing critical IT-Infrastructures. The problem of false positive alerts is a well known problem for many IDS implementations. Suboptimal patterns or insufficient thresholds for pattern-based and anomaly-based IDS approaches are the main reasons for a huge number of false-positive alerts. By deploying the IDS sensors in a distributed environment, the number of false positive alerts increases as a single event may be detected and reported multiple times by different involved sensors. The popular solution to address this problem is correlation and clustering of relative or similar alerts. Modern algorithms for alert correlation are using environment information, e.g., attack graphs (AG),to improve their performance. The approach proposed in correlates IDS alerts in a memory-efficient way by using an AG and a matching function.

The remaining involving document is actually structured as follows: Section two provides the actual connected operates. Method versions are generally defined throughout Section 3, Section 4 describes system design and implementation. The proposed protection measurement, minimization, in addition to countermeasures are generally introduced throughout Section 5 in addition to Section 6 examines AAFID in terms of network efficiency in addition to protection. Finally, Section 7 describes future work and operates to ends this particular document

III. AAFID:- ATONOMOUS AGENT FOR INTRUSION DETECTION

The actual AAFID attack detection method keen on obtaining feedback by the used in true situations, throughout true sites, and experiencing true problems and complications. The primary aim within the improvement of AAFID is to be able to put it to use and consider the two structures along with the setup, and supply feedback that allows us to distinguish weak points and alternatives intended for development. The actual AAFID structures allows data being obtained by many sources, therefore having the ability to combine the very best features of regular host-based and network-based IDSs.

AAFID catches and inspects suspicious impair traffic with no interrupting consumers purposes and impair companies. AAFID can certainly improve episode detection probability and improve resiliency to help VM exploitation episode with no interrupting existing normal impair companies. AAFID utilizes any fresh episode chart method intended for episode detection and reduction through correlating episode conduct as well as suggests useful countermeasures. AAFID optimizes the setup in impair machines to reduce learning resource consumption.

NICE: Network Invasion detection and Countermeasure variety throughout virtual circle methods (NICE) to ascertain any defense-in-depth attack detection platform. Regarding greater episode detection, NICE includes episode chart analytical methods to the attack detection operations. The structure of NICE isn't going to mean to increase the existing attack detection algorithms; indeed, NICE utilizes any reconfigurable virtual marketing approach to diagnose and table the makes an attempt to help bargain VMs, therefore preventing zombie VMs. NICE come to a decision whether or not to place any VM throughout circle check up talk about.

1. **Threat model**

The actual attacker's main goal should be to exploit somewhat insecure VMs and bargain these people while zombies. The defense design targets on virtual-network-based episode detection and reconfiguration methods to improve resiliency to help zombie research. The function isn't going to entail host-based IDS and isn't going to tackle how to deal with encrypted traffic intended for episode detections.

2. **Attack graph model**

The actual episode chart gives details of almost all known vulnerabilities within the method along with the on the web connectivity information; all of us get a complete picture of existing safety situation from the method, in which you can estimate the probable provocations and problems through correlating diagnosed functions or perhaps routines.

To characterize the episode and the consequence of these kinds of activities, all of us prolong the notation of MulVAL reasoning episode chart while Predicament Episode Chart (SAG). Regarding correlating the alerts, all of us talk about the method described throughout [15] and establish a whole new Warn Connection Chart (ACG) to help place alerts throughout ACG for their individual nodes throughout SAG. To keep an eye on episode development, all of us track the source and getaway IP address intended for episode routines.

NICE is made up of 2 principal stages

Step 1: A light in weight mirroring-based circle attack detection agent (NICE-A) in every single impair server to help catch and review impair traffic. A NICE-A periodically scans the virtual method vulnerabilities within a impair server to ascertain Predicament Episode Chart (SAGs).

Step 2: When any VM penetrates check up talk about, Strong Package Assessment (DPI) is used, and/or virtual circle reconfigurations may be started for the examining it VM to make the potential episode behaviors prominent.

3. NICE-A

The NICE-A is a Network-based Invasion Discovery System agent. Dom0 or perhaps DomU may be the impair server It scans the traffic going through Linux bridges in which command the many traffic involving VMs and in/out in the actual impair machines. The traffic generated in the VMs around the mirrored application fill is going to be mirrored with a certain slot with a certain fill employing COURSE, RSPAN, or perhaps ERSPAN procedures.

4. VM Profiling

One major factor that counts toward a VM profile is its connectivity with other VMs. Any kind of VM that may be associated with much more variety of machines is much more vital compared to a single associated with a lot fewer VMs since the effect of bargain of an hugely connected VM can cause much more deterioration. Furthermore necessary may be the familiarity with companies working with a VM to be able to examine the authenticity of alerts concerning in which VM. The opponent incorporates the use of port-scanning method to do a rigorous study of the circle to take into consideration available plug-ins in virtually any VM. Thus info on virtually any available plug-ins with a VM along with the background of opened plug-ins has an important role throughout identifying just how somewhat insecure the VM is. Each one of these aspects merged will form the VM page. VM users are generally looked after in a database and consist of comprehensive info on vulnerabilities, warn, and traffic.

Episode chart power generator. Whilst bringing in the episode chart, every diagnosed susceptibility is combined with the matching VM gain access to within the database.

AAFID- The actual warn relating to the VM is going to be documented within the VM page database.

Network controller: The actual traffic patterns relating to the VM are based on several tuples (source MAC tackle, getaway MAC tackle, resource IP tackle, getaway IP tackle, protocol). We could possess traffic style, in which packets emanate from the sole IP and are brought to many getaway IP address, and vice versa.

5. Attack Analyzer

The actual SAG contains 3 stages: Details accumulating, episode chart development, and potential exploit journey analysis. With this particular information, episode walkways may be modeled employing SAG. Each node within the episode chart shows a exploit with the opponent. Each journey by a first node with a goal node shows an effective episode. In summary, NICE episode chart is created using the pursuing information: Impair method information is obtained in the node controller (i. elizabeth., Dom0 throughout XenServer). The info contains the volume of VMs within the impair server, working companies in every single VM, and VM's Exclusive User interface (VIF) information.

. Exclusive circle topology and construction information is obtained in the circle controller, consisting of virtual circle topology, number on the web connectivity, VM on the web connectivity, every VM's IP tackle, MAC tackle, slot information, and traffic circulation information.

. Vulnerability information is generated through each on demand vulnerability scanning and regular sexual penetration screening while using the well-known vulnerability directories, for example wide open Supplier vulnerability Data source (OSVDB), Popular Vulnerabilities and Exposures Listing (CVE), and NIST Countrywide vulnerability Data source (NVD)

Advantages of AAFID

Autonomous agent intended for Invasion detection method (AAFID) makes use of the incident graph design to help execute incident detection and conjecture. The actual suggested alternative investigates how you can use the programmability of application switches-based methods to improve detection precision and beat unwilling recipient exploitation stages of collaborative problems.To improve detection precision, host-based IDS alternatives.Network command and episode analysis.Reduce the risk from the impair method by currently being used and over used through internal and external attackers.The AAFID structures allows data being obtained by many sources, therefore having the ability to combine the very best features of regular host-based and network-based IDSs. It obviously also let to develop IDSs which have been much more resistant to help insertion and evasion problems when compared with existing architectures, while no tests happen to be executed to compliment this kind of declare.

Network Controller

The actual circle controller is an extremely important component to compliment the programmable marketing power to comprehend the virtual circle reconfiguration function determined by Wide open Flow protocol. It is applied for the reason that borders move intended for VMs to handle traffic throughout and from VMs. The actual conversation involving impair machines (i. elizabeth., actual servers) is dealt with through actual Wide open Flow-capable Swap (OFS).

In NICE all of us use the OVS and OFS circle controller allowing the impair method to line safety or selection guidelines in an incorporated and comprehensive fashion. The actual circle controller is in-charge of amassing circle information of existing wide open Flow circle and offers suggestions for the episode analyzer to make episode graphs. Network controller is additionally to blame for utilizing the countermeasure by episode analyzer According to VM Safety Listing (VSI).

AAFID

An autonomous agent is a software agent that performs a certain security monitoring function at a host. We term the agents as *autonomous* because they are independently-running entities. Agents may or may not need data produced by other agents to perform their work, but they are still considered to be autonomous. Additionally, agents may receive high-level control commands from other entities. This high level control does not interfere with our definition of agent autonomy. An agent may perform a single very specific function, or may perform more compound activities.

Agents are independently-running entities; they can be added and removed from a system without altering other components, therefore without having to restart the IDS. Furthermore, agents may provide mechanisms for reconfiguring themselves without having to restart. Additionally, agents can be tested on their own before introducing them into a more complex environment. An agent may also be part of a group of agents that perform different simple functions but that can exchange information and derive more complex results than any one of them may be able to obtain on their own.

The ability to start and stop agents independently of each other in the systems that are being monitored adds the possibility of reconfiguring the IDS (or parts of it) without having to restart it. If we need to start collecting a new type of data or monitoring for a new kind of attacks, the appropriate agents can be started without disturbing the ones that are already running. Similarly, agents can be stopped or reconfigured without having to restart the whole IDS. If an agent collects network information related to the host where it is running, we reduce the possibility of being subject to insertion and evasion attacks by reducing the number of mismatched assumptions that can be made. Additionally, using agents as data collection and analysis entities provides the following desirable features:

- An agent can be programmed arbitrarily; it may obtain it's data through a audit trail, by probing the machine exactly where it really is managing, by catching packets from the system, or even through any suitable resource. Thus, IDS built through a collection of agents can cross punch the more common limitations in between host-based in addition to system based IDSs.
- Agents might be quit in addition to began without disturbing other IDS, agents might be upgraded to be able to new versions, in addition to provided that their additional software remains the same, various other components will not need to perhaps know that your real estate agent has become improved.
- If agents are generally executed seeing that separated operations on a host, each agent might be implemented in the programming language which is ideal well-matched with the task that it has to perform.

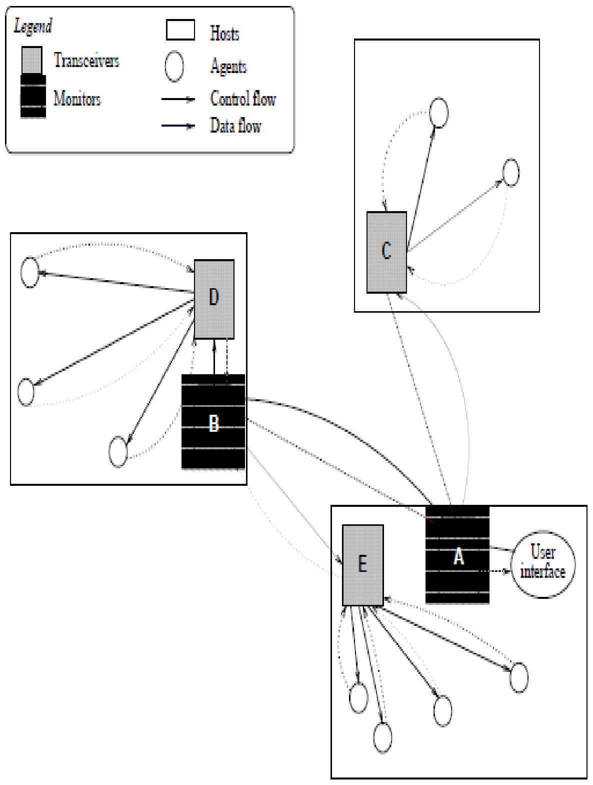


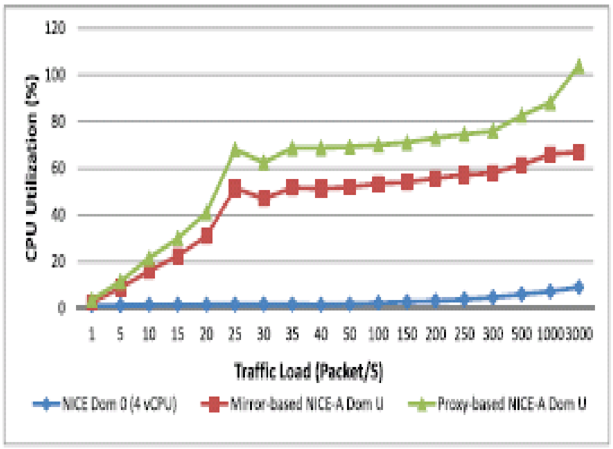
Fig: 1 Architecture of AAFID

The AAFID program can be distributed more than several serves inside a multilevel. Each service provider can simply consist of several agencies that can observe concerning useful instances taking place within service provider. Stockbrokers may use filtration to accumulate truth in a very system-independent approach. The many agencies inside a system file their own results to some single transceiver. Transceivers are per-host people that can supervise the procedure of all of the agencies managing of the service provider. Many people can to start off, stop in addition to write-up setting requires that you agencies. They will also carry out truth lessen in regards to the truth been recently given from the agencies. Your transceivers file their own brings about several screens. Each observes operates the procedure regarding a number of transceivers. Window screens gain access to network-wide truth; for that reason they'll carry out higher-level outcomes in addition to detect intrusions that can consist of a number of serves. Window screens can be organized inside a hierarchical trend in a way that a monitor may well consequently are accountable to a higher-level monitor. Also, a transceiver may perhaps report to many monitor to produce redundancy as well as resistance around the discouragement regarding on the list of displays. Eventually, a monitor is liable intended for supplying files in addition to acquiring get requires for a program.

IV. PERFORMANCE ANALYSIS

The overall performance to provide help with the amount of site visitors AAFID are designed for one cloud server along with make use of the examination metric in order to scale up to a large cloud system. In the real cloud system, traffic planning is required to run AAFID, that's over and above the particular extent in this paper. Due to the place constraint, we will certainly look into your research regarding multiple cloud clusters in the future. To show the particular feasibility individuals solution, evaluation studies have been done about several virtualization methods. Most of us evaluated AAFID depending on Dom0 along with DomU implementations together with mirroring-based along with proxy based attack detection agents. In looking mirror based IDS circumstances, we established two virtual networks in each cloud server: NICE-A can be linked with the particular keeping track of circle. Visitors for the regular circle can be mirrored for the keeping track of circle employing Traded Port Analyzer (SPAN) approach. Inside the proxy-based IDS solution, NICE-A interfaces a two VMs and the site visitors experiences NICE-A. Additionally, we have implemented the particular AAFID with Dom0 along with it removes the traffic function with mirroring along with proxy-based solutions.

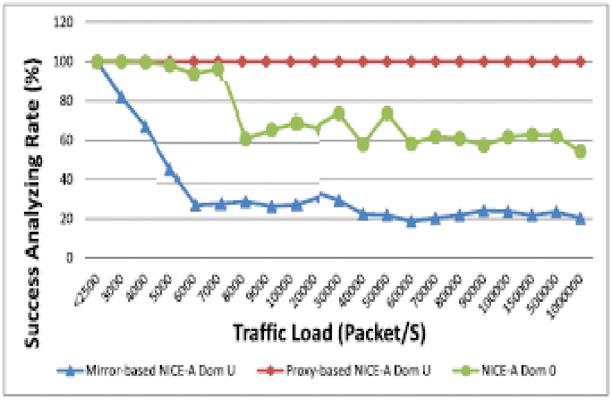
AAFID managing inside Dom0 is actually more effective given it may sniff the traffic entirely on the virtual bridge. Even so, inside DomU, the visitors must be cloned on the VM’s VIF, triggering overhead. When the IDS is actually managing inside Intrusion Prevention System (IPS) manner, its needs to intercept every one of the visitors and also perform package examining, that takes in far more technique means when compared with IDS manner. To demonstrate performance evaluations, many of us utilized four metrics that is CPU usage, community volume, adviser control volume, and also transmission wait. Many of us performed the evaluation on cloud servers along with Intel quad-core Xeon a couple of. 4-GHz CPU and also 32-G memory.



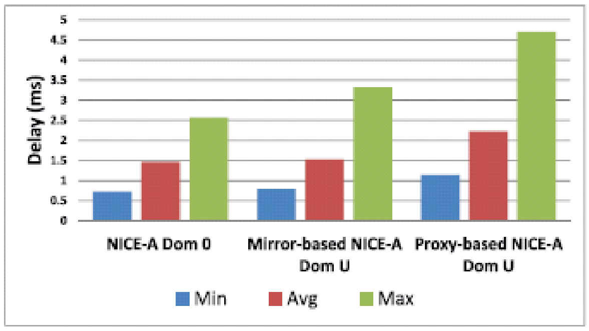
CPU utilization of AAFID

We used packet generator to mimic real traffic in the cloud system as shown in Fig. 3, this targeted traffic heap, such as box mailing speed, boosts from 1 to help 3, 000 packets per second. The particular effectiveness in Dom0 consumes fewer PROCESSOR as well as the IPS method consumes the maximum PROCESSOR means. It might be witnessed anytime this box price grows to to help 3, 000 packets per second, this PROCESSOR utilization of IPS in DomU grows to its issue, while the IDS method in DomU only takes up concerning 68 percent. Fig. 3, signifies this effectiveness regarding AAFID with regard to percentage regarding productively reviewed packets, i. electronic., the number of this reviewed packets separated through the total number regarding packets been given. The larger that importance will be, additional packets that real estate agent are designed for. It might be witnessed from your result of which IPS real estate agent proves 100 % effectiveness mainly because each box grabbed through the IPS will be cached from the detection agent buffer. Nevertheless, 100 % accomplishment analyzing price regarding IPS is at the expense of this analyzing hold off. For additional two types of agents, the detection agent agent doesn't store this grabbed packets as well as, no hold off will be introduced. However, all experience packet drop when traffic load will be big.

The communication holds up while using the method below distinct AAFID is introduced. We generated 100 consecutive standard packets while using the speed of 1 packet every minute to try the end-to-end hold up regarding two VMs in contrast by using AAFID running throughout mirroring along with proxy settings throughout DomU along with GOOD running throughout Dom0. Many of us document the little, average, along with greatest communication hold up inside marketplace analysis analyze. Results demonstrate which the holdup regarding proxy-based AAFID may be the best mainly because each and every packet features to pass through that. Mirror-based AAFID on DomU along with AAFID on Dom0 do not have recognizable dissimilarities inside hold up. To sum up, the AAFID on Dom0 along with Mirror-based AAFID on DomU possess superior overall performance with regard to hold up.



AAFID- success analyzing rate.



Network communication delay of AAFID

From this analysis, we expected to prove the proposed thus reaching our own objective protecting mechanism-based software defined networking approach which involves multiphase intrusion detections. ”This studies confirm which for just a small-scale cloud system, our own technique is successful. This performance examination includes two parts. First security performance evaluation, it signifies that our approach defines the structure stability ambitions: To prevent susceptible VMs coming from staying expose in addition to take action inside a smaller amount of embarrassing in addition to affordable way. Second, CPU in estimate to throughput performance evaluation, it demonstrates the limits using proposed solution in terms of network throughputs according to software program buttons in addition to CPU usage when running detection applications in Dom 0 in addition to Dom You. This performance results produce all of us the standard with the presented computer hardware build in addition to demonstrates the amount visitors can be handled by using a individual detection area. To level up to and including information center-level IDS, the decentralized technique must be invented, that is scheduled in our upcoming analysis.

V. CONCLUSION

The event generating system has to be improved. The concepts are interesting, but additional work is needed to optimize the process. The system is still early work. It is possible to automatically detect anomalies in the communication structure of a surveyed network, but the goal of detecting a large number of different attacks is not yet reached. The fact that some of the attacks could be discovered by the system without any knowledge on the used attack techniques encourages us to further research. It is easy to see that this approach to intrusion detection only is appropriate for intruders causing and significant traffic in the supervised network. The proposed framework leverages Open Flow network programming APIs to build a monitor and control plane over distributed programmable virtual switches to significantly improve attack detection and mitigate attack consequences. The system and security evaluations demonstrate the efficiency and effectiveness of the proposed solution. The autonomous agent for intrusion detection system proposed to detect and mitigate collaborative attacks in the cloud virtual networking environment. NICE utilizes the attack graph model to conduct attack detection and prediction.

REFERENCES

[1] Could Security Alliance, “Top Threats to Cloud Computing v1.0,” https://cloudsecurityalliance.org/topthreats/csathreats. v1.0.pdf, Mar. 2010.

[2] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, “A View of Cloud Computing,” ACM Comm., vol. 53, no. 4, pp. 50-58, Apr. 2010.

[3] B. Joshi, A. Vijayan, and B. Joshi, “Securing Cloud Computing Environment Against DDoS Attacks,” Proc. IEEE Int’l Conf. Computer Comm. and Informatics (ICCCI ’12), Jan. 2012.

[4] H. Taka, J.B. Joshi, and G. An, “Security and Privacy Challenges in Cloud Computing Environments,” IEEE Security and Privacy, vol. 8, no. 6, pp. 24-31, Dec. 2010.

[5] “Open switch Project,” http://openvswitch.org, May 2012.

[6] Z. Dean, P. Chen, F. Sanchez, Y. Dong, M. Stephenson, and J. Barker, “Detecting Spam Zombies by Monitoring Outgoing Messages,” IEEE Trans. Dependable and Secure Computing, vol. 9, no. 2, pp. 198-210, Apr. 2012.

[7] G. Gu, P. Pores, V. Yegneswaran, M. Fong, and W. Lee, “Pothunter: Detecting Malware Infection through IDS-driven Dialog Correlation,” Proc. 16th USENIX Security Seep. (SS ’07), pp. 12:1-12:16, Aug. 2007.

[8] G. Gu, J. Zhang, and W. Lee, “BotSniffer: Detecting Botnet Command and Control Channels in Network Traffic,” Proc. 15th Ann. Network and Distributed Sytem Security Symp. (NDSS ’08), Feb. 2008.

[9] O. Sheyner, J. Haines, S. Jha, R. Lippmann, and J.M. Wing, “Automated Generation and Analysis of Attack Graphs,” Proc. IEEE Symp. Security and Privacy, pp. 273-284, 2002,

[10] “NuSMV: A New Symbolic Model Checker,” http://afrodite.itc. it:1024/nusmv. Aug. 2012.

[11] P. Ammann, D. Wijesekera, and S. Kaushik, “Scalable, graphbased network vulnerability analysis,” Proc. 9th ACM Conf. Computer and Comm. Security (CCS ’02), pp. 217-224, 2002.

[12] X. Ou, S. Govindavajhala, and A.W. Appel, "MulVAL: A Logic- Based Network Security Analyzer," Proc. 14th USENIX Security Symp., pp. 113-128, 2005.

[13] R. Sadoddin and A. Ghorbani, "Alert Correlation Survey: Framework and Techniques," Proc. ACM Int'l Conf. Privacy, Security and Trust: Bridge the Gap between PST Technologies and Business Services (PST '06), pp. 37:1-37:10, 2006.
[14] L. Wang, A. Liu, and S. Jajodia, "Using Attack Graphs for Correlating, Hypothesizing, and Predicting Intrusion Alerts," Computer Comm., vol. 29, no. 15, pp. 2917-2933, Sept. 2006.
[15] S. Roschke, F. Cheng, and C. Meinel, "A New Alert Correlation Algorithm Based on Attack Graph," Proc. Fourth Int'l Conf. Computational Intelligence in Security for Information Systems, pp. 58-67, 2011.

[16] A. Roy, D.S. Kim, and K. Trivedi, "Scalable Optimal Countermeasure Selection Using Implicit Enumeration on Attack Countermeasure Trees," Proc. IEEE Int'l Conf. Dependable Systems Networks (DSN '12), June 2012.

[17] N. Poolsappasit, R. Dewri, and I. Ray, "Dynamic Security Risk Management Using Bayesian Attack Graphs," IEEE Trans. Dependable and Secure Computing, vol. 9, no. 1, pp. 61-74, Feb. 2012.

[18] Open Networking Fundation, "Software-Defined Networking: The New Norm for Networks," ONF White Paper, Apr. 2012.

[19] "Openflow," http://www.openflow.org/wp/learnmore/, 2012. [20] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L.Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling Innovation in Campus Networks," SIGCOMM Computer Comm. Rev., vol. 38, no. 2, pp. 69-74, Mar. 2008.

[21] E. Keller, J. Szefer, J. Rexford, and R.B. Lee, "NoHype: Virtualized Cloud Infrastructure without the Virtualization," Proc. 37th ACM Ann. Int'l Symp. Computer Architecture (ISCA '10), pp. 350-361, June 2010.

[22] X. Ou, W.F. Boyer, and M.A. McQueen, " A Scalable Approach to Attack Graph Generation," Proc. 13th ACM Conf. Computer and Comm. Security (CCS '06), pp. 336-345, 2006.
[23] Mitre Corporation, "Common Vulnerabilities and Exposures, CVE," http://cve.mitre.org/, 2012.

[24] P. Mell, K. Scarfone, and S. Romanosky, "Common Vulnerability Scoring System (CVSS)," http://www.first.org/cvss/cvss-guide. html, May 2010.

[25] O. Database, "Open Source Vulnerability Database (OVSDB)," http://osvdb.org/, 2012.