ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852

IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

Design and Implementation of Intrusion Detection System (Ids) Sensor Deployment

Mr.Patil Mahesh A.*, Prof: Y. M. Patil
* Research Scholar in Electronics And Telecommunication Engg. Shivaji University Kolhapur(MS), India
Professor in Electronics Eng Dept.,K.I.T.College Of Engg. Shivaji University Kolhapur(MS), India
mahesh.patil37@gmail.com

Abstract

Network intrusion detection systems provide proactive defense against security threats by detecting and blocking attack-related traffic. This task can be highly complex, and therefore, software-based network intrusion detection systems have difficulty in handling high speed links. This paper describes the design and implementation of a high-performance network intrusion detection system that combines the use of software-based network intrusion detection sensors deployment. In large network environments multiple intrusion detection sensors are needed to adequately monitor network traffic. However, deploying and managing additional sensors on a large network can be a demanding task, and organizations have to balance their desire for detecting intrusions throughout their network with financial and staffing limitations. This paper investigates how intrusion detection system (IDS) sensors should best be placed on a network when there are several competing evaluation criteria. This is a computationally difficult problem and we show how Multi-Objective Genetic Algorithms provide an excellent means of searching for optimal placement

Keywords: Network intrusion detection systems (NIDSes), multi-criteria optimization, Intrusion Detection System (IDS) sensors, probe attack , Denial of Service(DOS) attack, Attack Graphs.

Introduction

The increasing importance of network infrastructure and services along with the high cost and difficulty of designing and enforcing end-system security policies has resulted in growing interest in complementary, network-level security mechanisms, as provided by firewalls and network intrusion detection and prevention systems. High-performance firewalls are rather easy to scale up to current edge-network speeds because their operation involves relatively simple operations such as matching a set of Access Control List-type policy rules against fixed-size packet headers. Unlike firewalls, network intrusion detection systems (NIDSes) are significantly more complex and, as a result, are lagging behind routers and firewalls in the technology curve. Moreover, the function of NIDSes needs to be updated with new detection components and heuristics, due to the continuously evolving nature of network attacks. Both complexity and the need for flexibility make it hard to design high-performance NIDSes.

In large network environments, particularly those with many network segments and those with multiple Internet access points, network administrators have generally placed multiple IDS sensors along the network perimeters, typically around firewalls, or near the node to be protected, to monitor network traffic. By deploying sensors on various network segments, tune each of them to the traffic that typically on that segment, which means identify and locate suspicious activities more quickly. However, the detection of intrusions in large volumes of data, in the absence of semantic hints provided by prior knowledge of the intrusion type, is fundamentally limited by the low ratio of malicious events [2]. It is not obvious that deploying IDS sensors in larger numbers would improve detection quality – diminishing returns are likely to be evident early. Neither is it feasible to deploy more and more sensors given the costs and the manual engagement required to monitor for potential intrusions.

Determining where to place a set of sensors to create cost effective intrusion detection is a difficult task. There may be several evaluation criteria for placements, seeking to maximize various desirable properties (e.g. various attack detection rates), whilst seeking to reduce undesirable properties (such as false alarm rates as well as purchase, management, and communications costs). Subtle tradeoffs may need to be made between the properties; different placements may have complementary strengths and weaknesses, with neither placement being uniformly better than the other. However, engineering regularly deals with such

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852

difficult *multi-criteria optimization* problems and has developed a powerful suite of technical tools to facilitate the search for high performing solutions.

Relevance

An IDS Sensor placement may be optimal for the detection of one type of attack, but not for a second type of attack. To find out a placement that gives good chances of detecting each of several types of attack; this may yield a different optimal placement. To determine the "optimal" placement required a means to evaluate a particular placement. In some cases, this may be carried out with respect to statically assigned information (e.g. location of firewalls and servers). In others, it requires to simulate attacks and measure the effectiveness of the placement.
Intrusion Detection System (IDS) sensors should best is placed on a network when there are several competing evaluation criteria. This is show how Multi-Objective Genetic Algorithms provide an excellent means of searching for optimal placements. A cost-effective decision for multi-objective optimization demonstrates the validity and potential of the multi-objective approach to sensor placement trade-offs and provide incremental placement options. The work presented is a deliberate attempt to use GA and MOO techniques to assist network administrators to choose IDS sensor placement that effectively satisfies multiple criteria. A multi-objective genetic algorithm (MOGA) can be harnessed to address the sensor placement problem. The placement strategies generated, although simple, are typical places that network administrators would likely deploy IDS sensors.

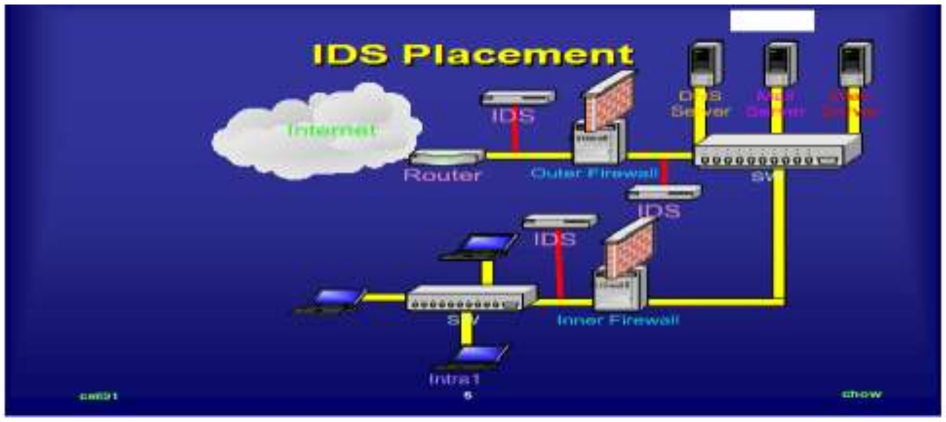


Fig 1 : IDS Sensor placement

Design and implementation

Performance: The primary metric of interest in the design of a NIDS is throughput. That is, to be able to operate at network speeds of at least 1 Gbit/s without packet losses, so as to detect any attempted attack. Therefore, the system must be capable of analyzing all the incoming traffic under the most stringent conditions. Network intrusion detection systems (NIDSes) based on commodity PCs are able to monitor at speeds much lower than 1Gbit/s2,5. This necessitates the use of a distributed design with several intrusion detection sensors operating in parallel and supported by a load balancing traffic splitter. At the same time, we want to minimize cost and use as few resources as possible. We also want to minimize the number of sensors needed. A key focus of our work is therefore on how to exploit the processing capacity on the IDS to reduce the load of the sensors. A second important performance goal is minimizing the latency induced by the NIDS. There is a direct relationship between latency introduced by a networking device and the maximum throughput of TCP flows. If the NIDS will be used at the boundary between an enterprise network and the Internet, latencies in the order of a few milliseconds may be tolerable. If the NIDS is deployed internally, and the network needs to support high-bandwidth local services (such as file sharing, etc.) the latency requirements are even more stringent. Particularly, there is a critical value for the

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852

round trip time (RTT) of a packet in each network. If the latency is below this critical value, TCP throughput is unaffected -- it is the line speed of the underlying network which becomes the bottleneck -- above this critical value, however, TCP throughput is Negatively impacted. The critical value for RTT in a network supporting Gigabit speeds is 0.5 milliseconds. Thus, if we want the throughput of TCP to be unaffected, we must ensure that the latency imposed by our NIDS is less than 0.5 milliseconds. However, Gigabit Ethernet links will rarely carry only a single TCP connection. Rather, a Gigabit Ethernet link supports hundreds, if not thousands of TCP connections, and this multiplexing mitigates the impact of latency on the overall throughput of the link.

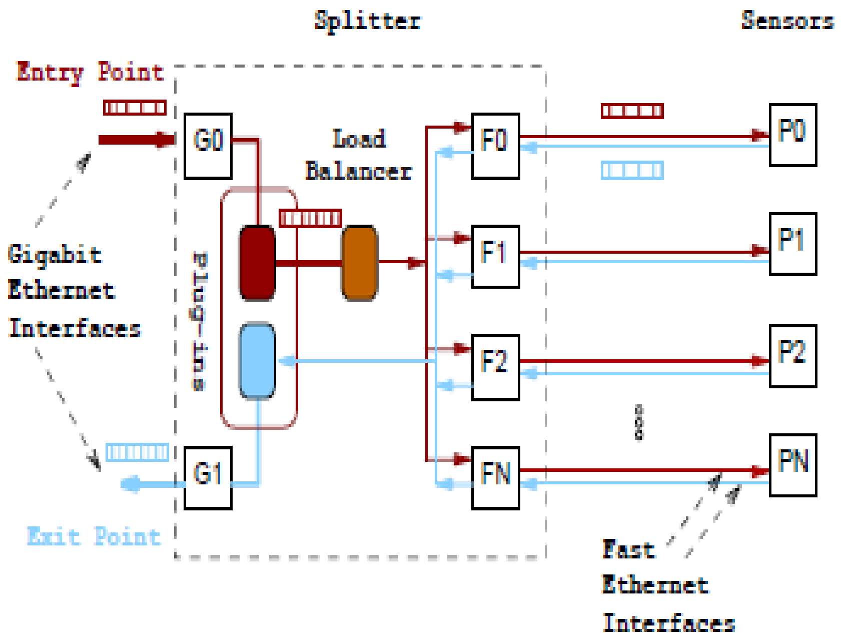


Fig : 2 Architecture of IDS Deployment

To impose latency greater than 0.5 milliseconds without affecting the throughput of a link due to the high number of TCP connections.

Flexibility and Scalability: A NIDS needs to be flexible and scalable, both for scaling up to higher link speeds and more expensive detection functions, as well as for updating the detection heuristics. If the protection of a faster link or a more fine-grained detection is required, it would be desirable to reuse as much as possible of the existing hardware. Clearly, this property does not hold for ASIC-based NIPSes. However, it is remarkable that almost all NIPS providers ignore this dimension. Furthermore, a prerequisite of flexibility is simplicity as extending a complex system may be hard and error-prone. It is therefore desirable for the hard-to-program elements of our system to be as generic as possible.

A. Architecture

Fig. 2 is composed of a customized load balancing splitter and a number of contentbased network intrusion detection sensors connected with the splitter (Figure 2). The splitter is the entry and exit point of the traffic that runs through the system. The basic task of the splitter is to evenly distribute the traffic across the sensors and to transmit the non-attack packets back to their destination. The sensors are responsible for the heavy task of inspecting the traffic for intrusion attempts. They maintain the required information for recognizing all the malicious traffic and deciding whether to forward or drop the packet. For every input packet, the splitter computes which sensor will be responsible to analyze this packet. Then, it forwards the packet to this sensor for inspection. The sensor searches for known attack patterns contained in the packet. If a pattern is found, then the packet is blocked, otherwise the packet is forwarded back to the splitter.

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852

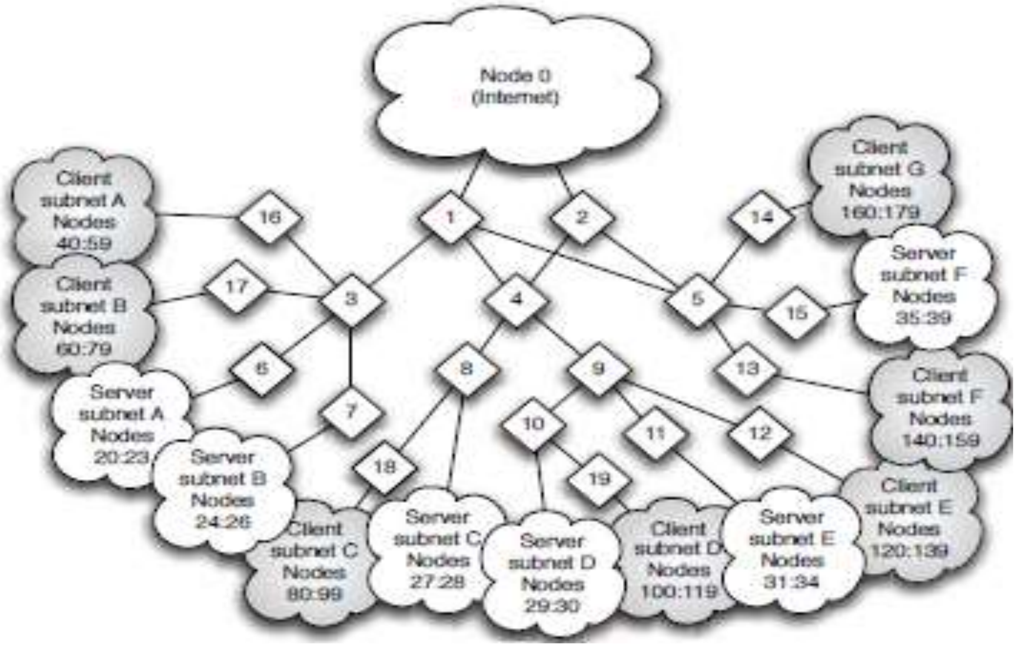
The splitter receives the analyzed packet and transmits it to its destination.
Additionally, it supports plug-ins that implement operations necessary to improve the performance of the system. A plug-in has two parts, one running on the splitter and one running on the sensors. These two parts cooperate in order to accomplish their task. In the context of this work we have designed a plug-in for attempts to minimize the cost of sending a packet from a sensor to the splitter.

Splitter: The functionality of the splitter can be divided into the basic operations and the plug-ins that provide adequate operations to boost performance. The basic part of the splitter integrates the functionality of a load balancer -- it is responsible for distributing the incoming traffic across the output interfaces (ports). However, it differs from a common load balancer in that it must be *flow-preserving*, that is, all the packets belonging to the same flow must be forwarded to the same output interface.
In case of TCP/UDP traffic, we define a flow to consist of all the traffic of a TCP or UDP connection. Otherwise, flow consists of all the traffic originating from a particular IP address and destined to a particular IP address. Regarding load balancing, there are two possible approaches that we could use: stateful load balancing that requires from the system to hold state and hash based load balancing, that experiences greater load imbalances. For the purposes of this paper, we assume that load imbalances are tolerable and use the simpler hash-based method. The input of the hash function is composed of the source and destination IP addresses of the packet.

Sensor: A sensor is a commodity PC that runs a modified popular NIDS and is connected with the splitter (through an Ethernet connection). A sensor receives traffic from the splitter and analyzes it for possible known attacks. In case that an attack is found, it notifies the splitter to block the offending packet(s), otherwise it informs the splitter that the packet(s) should be forwarded. A sensor maintains state about the traffic it analyzes in order to operate
Correctly. The maintained state includes the active TCP connections it has captured in the near past, TCP connections tagged as offending, fragmented packets and statistics about the connections per second to TCP/UDP destination ports.

B. Proposed Methodology:

The proposed scheme is outlined as follows,
Considering the whole network consists of 180 nodes, where node 0 represents the outside world, nodes 1 to 19 are the routers interconnecting various parts of the network, nodes 20 to 39 are servers offering valuable services to users and therefore critical assets that need to be protected, and nodes 40 to 180 are ordinary clients some of which may be compromised by intruders to attack critical assets.


Node 0 (Internet)
Client subnet A Nodes 40:59
Client subnet B Nodes 60:79
Server subnet A Nodes 20:23
Server subnet B Nodes 24:26
Client subnet C Nodes 80:99
Server subnet C Nodes 27:28
Server subnet D Nodes 29:30
Client subnet D Nodes 100:119
Server subnet E Nodes 31:34
Client subnet E Nodes 120:139
Client subnet F Nodes 140:159
Server subnet F Nodes 35:39
Client subnet G Nodes 160:179
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19

Probe attack

A real intrusive behavior to analyze how such behaviors could be efficiently detected by the proposed approach. The intrusive behavior is to do with probing and information gathering, the purpose of which is to assess a potential target's weaknesses and vulnerabilities [9]. An intruder may strive to detect active hosts and networks that are reachable and the services they are running that could be successfully exploited. Detecting and preventing such probes therefore is important both to inhibit exposure of information and prevent attacks that follow.

A probe attack scenario where various servers are probed from the outside node and inside from clients, hence the simulation consists of both external and internal attacks. An intruder may subvert a node in any of the client subnets to probe any of the servers. Client nodes also attempts a probe on neighbors client node. A possible number of probe attack are injected. In order to investigate how the false alarms may influence sensor placement strategy, simulation consist not only a number of attacks but also background network traffic. If the testing data set is a very representative sample of the operation environment, can use the metrics in the testing data to approximate the real world situation. In this experimental framework assume all sensors are identical and configured to exhibit a detection rate of 95% and a false positive rate of 0.1%.Expected monitoring costs for the network are dependant on the load of the traffic at a specific location in the network: the busier the location, the higher the levels of activity monitored (including false alarms), and therefore bigger the effort.

In the experiments, expected monitoring costs to reflect an operational network in the real world: routers nodes serving at the heart of the network are assigned a cost relatively much higher. Router nodes are assigned a cost with down the hierarchy, client nodes have minimum cost .

Fitness Measurement

The fitness of a sensor placement is determined by its ability to satisfy four objectives: - number of sensors, detection rate, false alarm rate and monitoring cost.

1.To minimize the number of sensors,
2.To maximize the detection rate of a sensor placement. It is relationship between
number of distinct attacks that have been detected and the number of all simulated attacks which have injected in the data set (i.e. probe attacks).
3.To minimize the false alarm rate of a sensor placement. It is relationship between the number of false alarms that are raised by the sensors and all alerts that are reported by the sensors. All alerts is a sum of the number of detected attacks (true alarms) and the number of false alarms.
4. To minimize the total monitoring cost.

3. Sensor Placement Representation

A feasible sensor placement is represented by *n* (i.e. number of network nodes) bits.

1. To investigate the relations between the number of sensors and detection quality (in terms of the pair of detection rate and false alarm rate), and search for placement given constraints on the number of sensors available to deploy.
2.Designed to determine the minimum monitoring cost needed to detect certain amount of attacks, and the criteria of amount of sensors is omitted. Nevertheless, given a reasonable budget, it is possible to effectively detect a majority of the attacks if the sensors are optimally placed.

3.Multi-optimization technique can be a very powerful tool to help to find cost-effective sensor placements.

Simulation and result

We are constructing attack graphs for sensor placement[6].Attack graphs predict the various possible ways of penetrating a network to reach critical assets. We then place IDS sensor to cover all these paths, using the fewest numbers of sensor.

We characterize expected monitoring costs for the network. We restrict the costs to a range of values 1 to 10 to express relative monitoring costs for different locations on a network. Router nodes 1 and 2 are assigned a cost of 8,router nodes 3,4,5 and 9 are assigned cost of 7,router nodes 8 and 10 are assigned cost of 6,router nodes 6,11,15 are assigned cost of 5.We assign a flat cost of 4 for all the other subnet router nodes.

We are designed different tcl script through NS2 simulator(For here showing v3.tcl,v4.tcl,v5.tcl,normal1.tcl).For attack Simulation Denial of Service(DOS) attack(eg.Normal1.tcl) is simulated for attack penetration. For probing of attack Worm attack (eg.Worm.tcl) is simulated.

1. Attack Graphs:

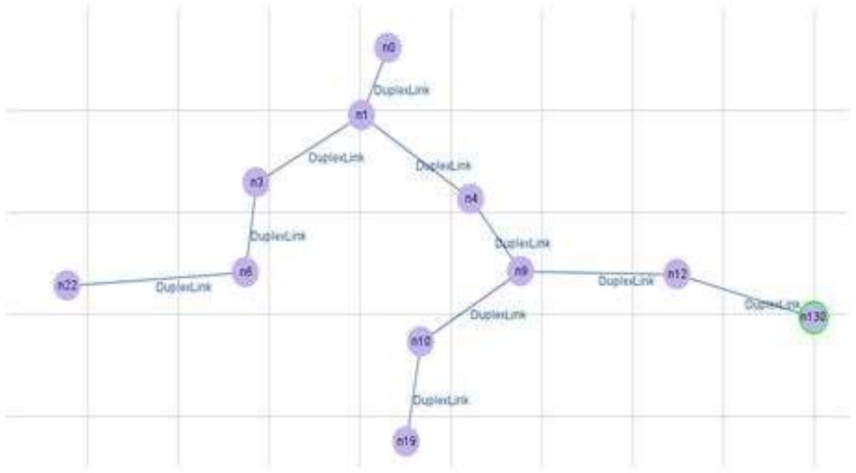


Fig:V3.tcl Attacker Node 130client On Sever Node22

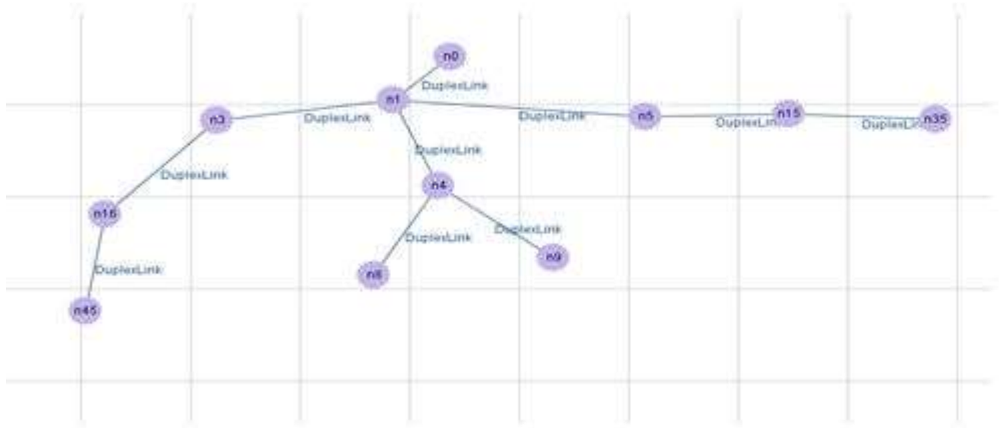


Fig:V4.tcl Attacker Node45 client On Server Node35

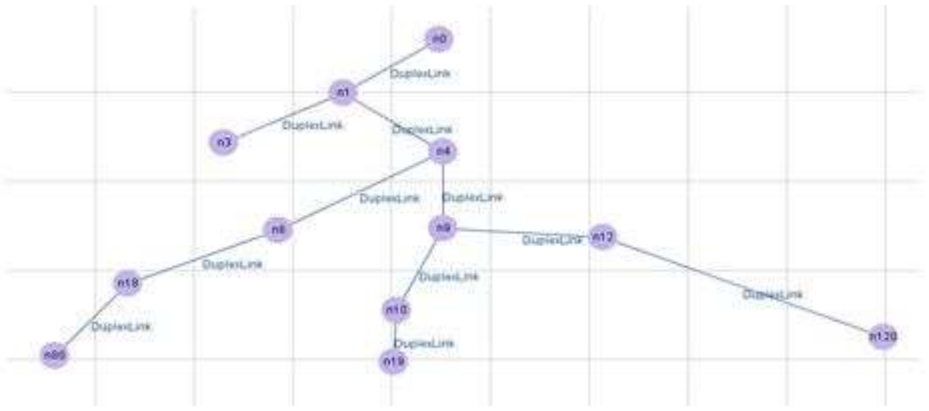


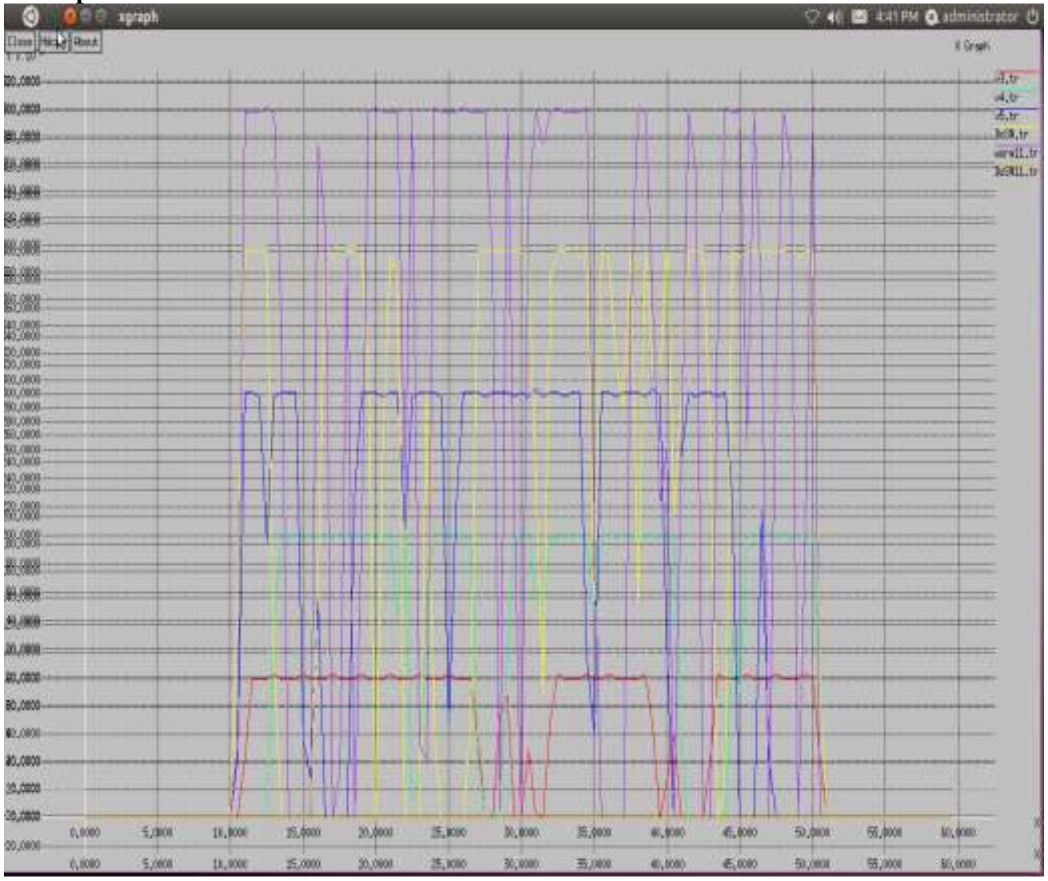
Fig:V5.tclAttacker Node80client On Server Node120

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852

2.Sensor Placement Representation:

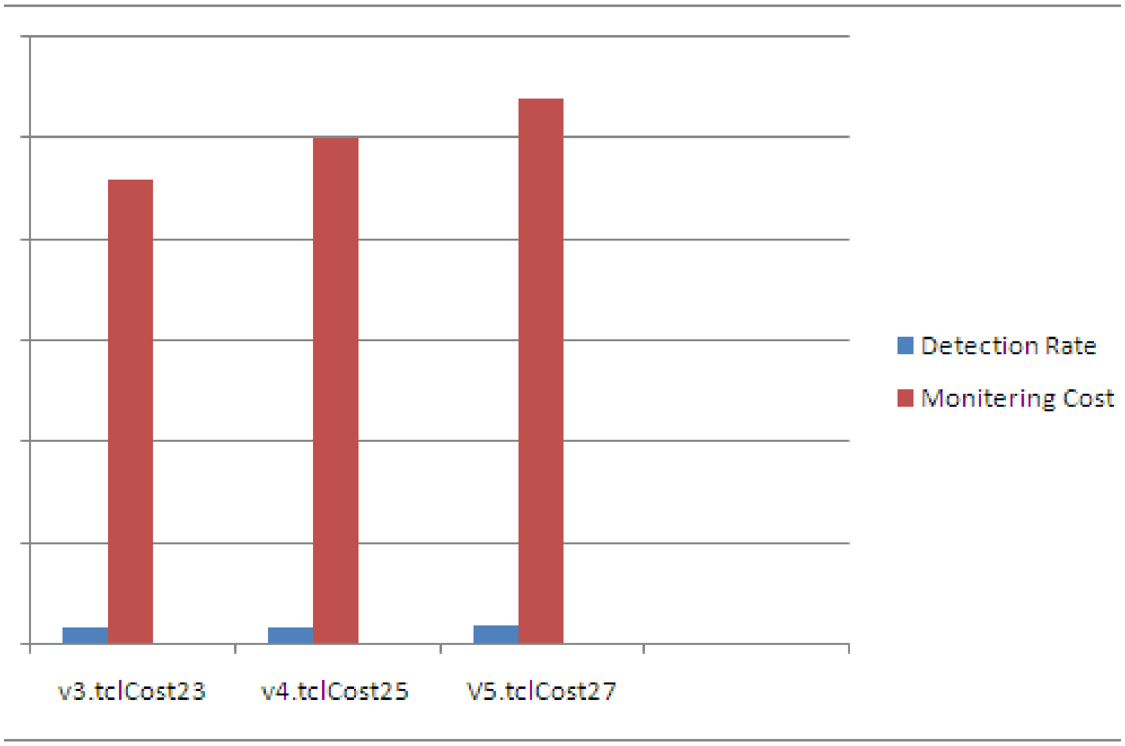
Example	No of sensor	Detection rate	Placement option	Monitoring cost	Energy consumed
V3.tcl	4	88.98%	NODES 1,3,12,19	23	300
V4.tcl	4	91%	NODES 3,8,9,15	25	400
V5.tcl	5	94%	NODES 1,3,12,18,19	27	520
Worm11.tcl	-	76%	-	-	510
Normal1.tcl	-	72%	-	-	510

3.Xgraph for Optimum Solution



Xgraph for v3.tcl,v4.tcl,v5.tcl,normal1.tcl,worm.tcl

4.Fttness Measurement



Through multi-objective optimization analysis we find out three placement option of IDS sensor placement.

Summary and concluding remarks

We have presented the design of a high-performance Network Intrusion Prevention System (NIDS). The number of sensors implemented on commodity PCs. We have focused on one method for boosting system performance by optimizing the coordination between the load balancer and the sensors. The result is a 45% improvement in performance, allowing the system to reach speeds of at least 1 Gbit/s. There are several directions that we are currently pursuing. First, we are re-examining the structure of the sensor software. We try to move part of the protocol processing functionality. Second, we are looking at ways for building a 10 Gbit/s NIDS .

Scope of the work

The placements satisfying realistic security requirements merits further investigation of the technique. Experimentation and general knowledge of intrusion detection systems have allowed identifying numerous possible improvements to the approach and tool support. A straightforward extension of this work would be to incorporate an increased number of security requirements. Sensor placement is critical to providing effective defense. Optimal placement for this purpose would seek to minimize damage caused by intrusions. Placements that seek to maximize the number of victims detected could be useful in identifying locations best for detecting attacks likely to have more adverse impact. Such placements could be particularly important to detect and mitigate worm propagation and network probes . One future work are planning is to assign quantitative information (e.g. level of risk) to individual nodes and provide a model (e.g. the sensor deployment model by Shaikh [13]) to assess the information and incorporate it into the multiobjective optimization framework.

References

1. S. A. Shaikh, H. Chivers, P. Nobles, J. A. Clark, and H. Chen, proposed “Optimizing IDS Sensor placement” 2010 IEEE International Conference on Availability, Reliability and Security IEEE computer society pages 315-320
2. P. Helman and G. Liepins, “Statistical foundations of audit trail analysis for the detection of computer misuse,” IEEE

Transactions on Software Engineering, vol. 19, no. 9, pp. 886–901, 1993.
3. D. E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning. Boston, MA, USA: Addison-Wesley Longman PublishingCo.,Inc.1989.Available:http:/portal.acm.org/citation.cfm?id=534133
4. C. A. C. Coello and L. Nacional, "An updated survey of gabased multiobjective optimization techniques," ACM Computing Surveys, vol. 32, pp. 109–143, 1998.
5. W. Lu and I. Traore, "Detecting new forms of network intrusion using genetic programming," in Proceedings of the 2003 Congress on Evolutionary Computation, 2003.
6. S. Noel and S. Jajodia, "Attack graphs for sensor placement, alert prioritization, and attack response," in Cyberspace Research Workshop, 2007.
7. M. Rolando, M. Rossi, N. Sanarico, and D. Mandrioli, "A formal approach to sensor placement and con.guration in a network intrusion detection system," in SESS '06: Proceedings of the 2006 international workshop on Software engineering for secure systems. ACM, 2006, pp. 65–71.
8. T. Issariyakul and E. Hossain, An Introduction to Network Simulator NS2.Springer,2008.[Online].Available: http://www.springer.com/engineering/signals/book/978- 0-387-71759-3
9. S. A. Shaikh, H. Chivers, P. Nobles, J. A. Clark, and H. Chen, "Network reconnaissance," Network Security, vol.11,pp,12-16 ,2008.elsevier.
10. G. Gu, P. Fogla, D. Dagon, W. Lee, and B. Skoric, "Measuring intrusion detection capability: an information-theoretic approach," in ASIACCS '06: Proceedings of the 2006 ACM Symposium on Information,computer&communications security. ACM, March2006,pp.90-101
11. S. Luke, "A java-based evolutionary computation research system," 2008, available as http://cs.gmu.edu/eclab/projects/ecj/.
12. E. Zitzler, M. Laumanns, and L. Thiele, "Spea2: Improving the strength pareto evolutionary algorithm," Swiss Federal Institute of Technology, Tech. Rep. 103, 2001.
13. S. A. Shaikh, H. Chivers, P. Nobles, J. A. Clark, and H. Chen, "A deployment value model for intrusion detection sensors," in 3rd International Conference on Information Security and Assurance, ser. Notes On Computer Science, vol. 5576, 2009,
14. Design and implementation of a high performance network intrusion prevention system Konstantinos Xinidis1, Kostas G. Anagnostakis2, Evangelos P. Markatos1 1Institute of Computer Science, Foundation for Research and Technology Hellas, P.O Box 1385 Heraklio, GR-711-10 Greece {xinidis, markatos}@ics.forth.gr ; 2Distributed Systems Laboratory, CIS Department, Univ. of Pennsylvania, 200 S. 33rd Street, Philadelphia, PA 19104 anagnost@dsl.cis.upenn.edu
15. Aaron Turner and Matt Bing. tcpreplay Tool. http://tcpreplay.sourceforge.net.
16. S. Antonatos, K. G. Anagnostakis, and E. P. Markatos. Generating realistic workloads for intrusion detection systems. In Proceedings of the 4th ACM SIGSOFT/SIGMETRICS Workshop on Software and Performance (WOSP 2004), January 2004.
17. Z. Cao, Z.Wang, and E.W. Zegura. Performance of hashing based schemes for internet load balancing. In Proceedings of IEEE Infocom, pp. 323-341, 2000.
18. Y. Charitakis, K. G. Anagnostakis, and E. Markatos. An active splitter architecture for intrusion detection (short paper). In Proceedings of the Tenth IEEE/ACM Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS 2003), October 2003.
19. Y. Charitakis, D. Pnevmatikatos, E. P. Markatos, and K. G. Anagnostakis. Code generation for packet header intrusion analysis on the IXP1200 network processor. In Proceedings of the 7th International Workshop on Software and Compilers for Embedded Systems (SCOPES 2003), September 2003.
20. Intel Corporation. Intel PRO/1000 MT Dual Port Server Adapter. http://www.intel.com.