Parameterized Analysis of Intrusion Detection and Prevention Systems and their Implications on Attack Alerts and Event Co-relation

Shalvi Dave
Asst. Prof, Indus University
Ahmedabad

Bhushan Trivedi, PhD.
Director, GLSICT
Ahmedabad

Jimit Mahadevia
Asst.V.P, Elitecore
Ahmedabad

ABSTRACT
Intrusion Detection and/or Prevention Systems (IDPS) represent an important line of defence against a variety of attacks that can compromise the security and proper functioning of an enterprise information system. Along with the widespread evolution of new emerging services, the quantity and impact of attacks have continuously increased, attackers continuously find vulnerabilities at various levels, from the network itself to operating system and applications, exploit them to crack system and services. Network defence and network monitoring has become an essential component of computer security to predict and prevent attacks. Unlike traditional Intrusion Detection System (IDS), Intrusion Detection and Prevention System (IDPS) have additional features to secure computer networks.

In this paper, we present a detailed study of how architecture of an IDPS plays a key role in its performance and the ability to co-relate known as well as unknown attacks. We categorize IDPS based on architecture as local or distributed. A detailed comparison is shown in this paper and finally we justify our proposed solution, which deploys agents at host-level locally to give better performance in terms of better attack co-relation and accurate detection and prevention.

General Terms
IDPS architecture, Network Security

Keywords
Intrusion Prevention, IDPS sensors/agents, attack and event co-relation, architecture, information source, relevance of attacks.

1. INTRODUCTION
In order to apply admission and access control for a network, various Intrusion Detection and Prevention systems (IDPS) are available in the market. Intrusion detection system is used to manage traffic in real-time for increasing the accuracy detection and decreasing false alarm rate. In some instances, IPS adopts techniques from intrusion detection, such as detection approach, monitoring sensor, and alert mechanism. An IDPS is also used for gateway appliance, perimeter defence appliance, all-in-all capability, and network packet inspection/prevention. It is designed to identify and recognize potential security violations in stream network. However, the primary intrusion prevention systems use signature mechanism to identify activity in network traffic and host perform detect on inbound – outbound packets and would block that activity before they access and damage network resources.

Figure.1 and Figure. 2 shows the basic scenario of an Intrusion Detection System (IDS) and an Intrusion Prevention System (IPS).

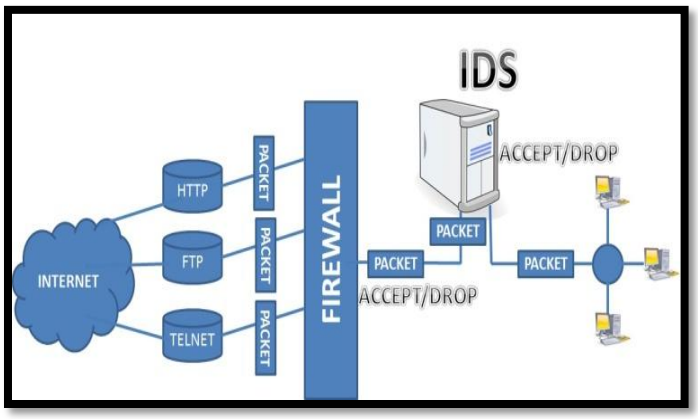

Fig.1 Intrusion Detection System

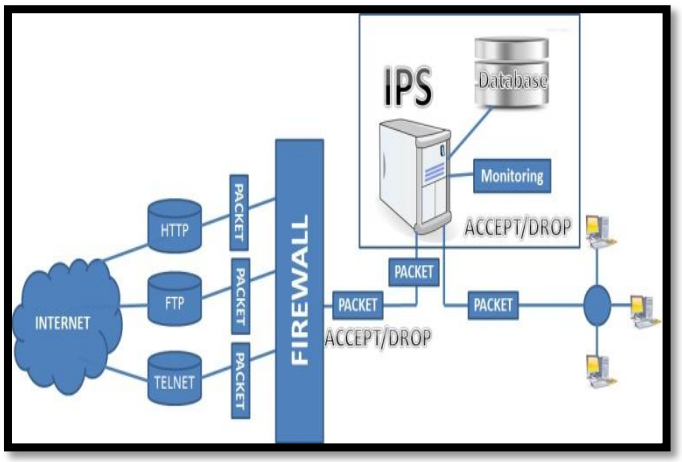

Fig.2 Intrusion Prevention System

An IDPS is an inline approach to monitor network activity. The detection technique used by the IDPS classifies it into two categories: signature based if it detects an attack by comparing it against a stored set of pre-defined signatures. It is anomaly-based if any abnormal behavior or intrusive activity occurs in the computer system, which deviates from system normal behavior. System normal behavior such as kernel information, system logs event, network packet information, software-running information; operating system information etc is stored into the database. [1]

In this paper, we present a case study on working of existing IDPS, including problem areas faced in today's environment and enhancements possible to address each of these problem areas. We also present a roadmap of hybrid IDPS approach. We have taken into consideration the following four parameters, in order to justify a Hybrid IDPS system:

a) Deployment

b) Architecture

c) Source of Information

d) Relevance of attacks.

The architecture of an IDPS can be centralized or distributed. In addition, when deployed around the boundary of a network, it is known as perimeter-based IDPS. In distributed architecture of IDPS, certain tasks are handled at the host-level and remaining at the network-level. Figure. 3 shows general architecture of an IDPS.

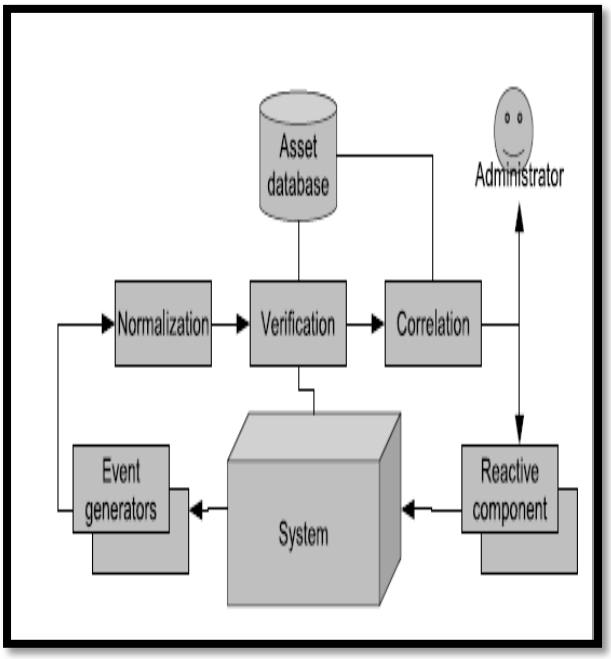

Fig.3 General architecture of IDPS

An NIDPS works on packet stream and not on host information. For e.g., operating environment of host. The information source for recognizing attacks are network packets, which are monitored by the IPS sensor. However, these network packets only contain limited amount of information, which includes source and destination IP and port addresses. Therefore, they can detect network vulnerability but would block the host itself instead of infected application. In addition, nowadays, IP and port addresses alone are not sufficient, since the attacks launched by intruders are immune to most firewall and IDS.

In case of HIDPS, it works more on host information like operating environment and logs and does not work on network packets to detect the vulnerabilities. Therefore, it lacks central analysis and chances are it cannot detect network vulnerabilities and attacks being performed from outside of the network to that particular host.

A system is considered robust if it does not produce false positives and does not completely fail to detect intrusions. One of an approach after an intrusion takes place is getting intruder's IP address and then to track all the activities done by the intruder system to generate its activity log and to do cross attack on the intruder system [2]. But, this cannot be applied in all situations. Therefore, it is essential for any IDS to know whether the attack is relevant or not. Also, relevance of an attack depends on the kind of operational environment where IDPS is deployed. For this, attack classification is very essential. The classification can be done using various techniques such as Data mining, artificial intelligence on neural networks, etc.

In this paper, we present a case study on the above mentioned techniques of architecture of existing IDPS, including problem areas faced in today's environment and enhancements possible to address each of these problem areas. We also analyze various existing systems architecture, information source and its implications on attack and event co-relation.

2. ARCHITECTURE AND INFORMATION SOURCE OF IDPS

The architecture of an IDPS is either centralized or distributed. The architecture also determines the information analyzed by an IDPS to generate attack or event alerts. An IDPS, whether network-based or host-based, performs two basic functionalities: Monitoring and Analysis. In distributed environment, intrusion detection data is both collected and analyzed in a distributed fashion. One of its earliest exponents is DIDS [3], which uses some NSM components and has the ability to do both local and global analysis of the data. At the local level it uses both statistical and rule-based detection, and at the global level it uses a rule-based expert system. In this sense, DIDS can be described as a number of host-based and network-based intrusion detection systems that can communicate and share results with one another. This is the form of almost all intrusion detection systems that call themselves distributed. The same techniques used in host-based and network-based intrusion detection systems are used, but the results are shared and can be analyzed at different levels.

Traditional centralized intrusion detection and prevention systems rely on a limited number of data sensor and only one event analyzer to obtain, process and analyze all the data in the network, there will be varying degrees of missed and false negative phenomena on the attack lack of resilience of the global assault. In addition, the system scalability is limited and difficult to configure and expand [3]. Figure. 4 describes a typical centralized intrusion detection and prevention system.

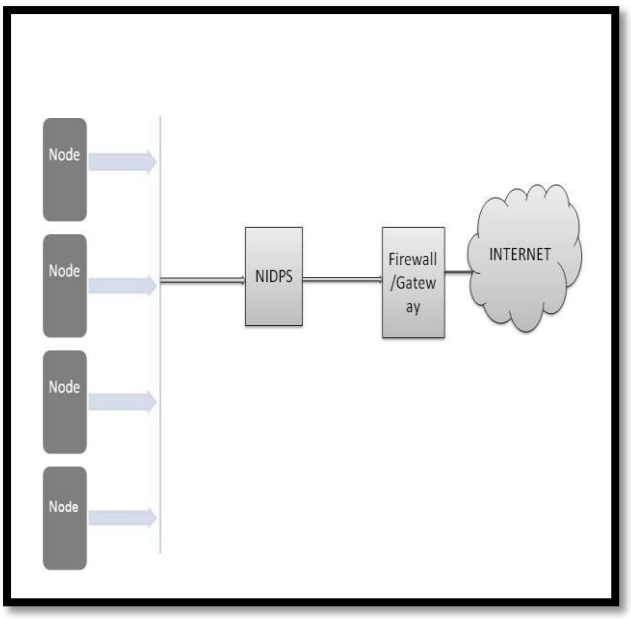

Fig.4 A Centralized IDPS architecture

Earlier examples of distributed intrusion detection systems are EMERALD [4] and APHIDS [5]. EMERALD is network-based IDS with distributed architecture while APHIDS is host-based IDS also with distributed architecture. These systems use different sources for data and different mechanisms for analyzing it. However, they share a similar general structure: a hierarchical arrangement where host-local components perform some part of the work and relay their results to components higher in the hierarchy. This continues until the partial results reach the top-level components, which have a network-wide view of the systems. Because all these systems ultimately depend on a centralized component, we could argue that they are not truly distributed. Figure. 5 describes a typical distributed intrusion detection and prevention system.

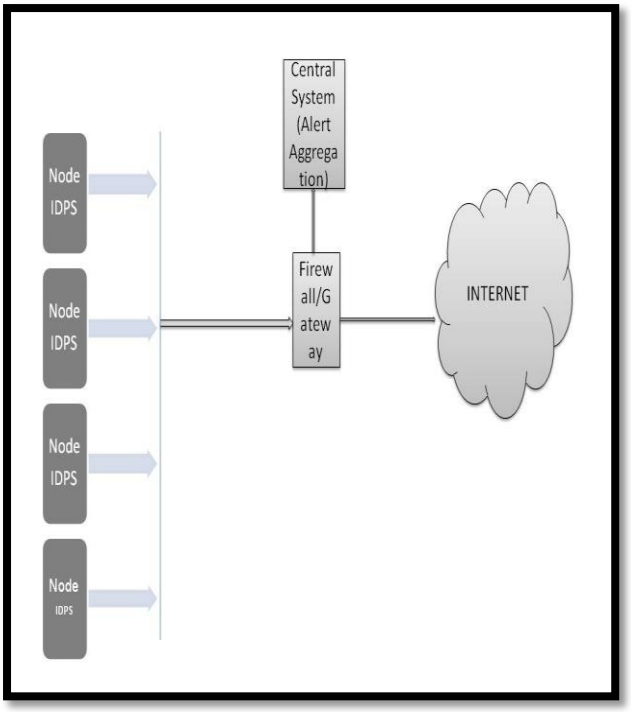

Fig.5 Distributed IDPS architecture

As mentioned earlier the architecture of an IDPS determines the source of information it analyzes to generate an attack or event alert. APHIDS is realized as a distributed layer, which operates on top of a set of distributed agent engines. These agents analyze trigger event notifications and generate an alarm. This kind of architecture provides reduction in delay of the analysis. However, these agents do not store all the generated network events at a common place. Therefore, it is not possible to detect vulnerabilities, which require analysis of all stored events. I.e. DDOS attacks cannot be detected using this methodology, as these types of attacks need to be analyzed after being monitored over a larger time-span.

Emerald monitors variety of sources like audit data, network datagram, application logs and intrusion detection events. These events are forwarded for further processing after they are parsed, filtered and formatted. Emerald's profiler engine detects vulnerabilities by implementing different analysis methods on this stream. As it is a host-based analysis, and no further analysis has been done at a common level, it fails to detect vulnerabilities, which can be detected only by analyzing overall network streams.

Thus, we need detailed information for event co-relation and analysis. For example, if we want to implement quota-based application access policy, which is very common and is an important requirement in any corporate office, only information contained in network packets is not sufficient.

One needs to capture information at the operating system level also. For example, socket information (source IP, source Port, destination IP, Destination port, protocol) and details of application (application name, version, upload and download data etc.). We achieve this by intercepting socket calls using hooks in socket. Administrator uses this connection and event log to implement quota-based application security policy. This restricts network access by applications during corporate working hours. For example, if one wants to restrict the use of yahoo messenger during working hours, then the connection log helps the administrator to implement such quota-based security policy. The Source and Destination IP provides to the administrator the network level information of the attacker and the victim. Name and severity of the attack gives information on criticality of the attack.

3. IMPLICATIONS OF ARCHITECTURE AND INFORMATION SOURCE ON ATTACK AND EVENT CO-RELATION

Monitoring of attacks and event co-relation should be done using distributed architecture, which is a feature of HIDS. However, after monitoring and event co-relation is done, analysis of attack log should be done locally. Administrator on admin server should perform this analysis.

The CIDP architecture [3] showed in the figure.6 talks about multiple IDP sensors at edge router, subnet or host. Every sensor will generate the alerts as per their configuration and rules deployed. Correlation happens on central location. However, when same kinds of alerts are being generated repeatedly then it is hard to filter them out and correlate as some of the basic correlation attributes were missing.

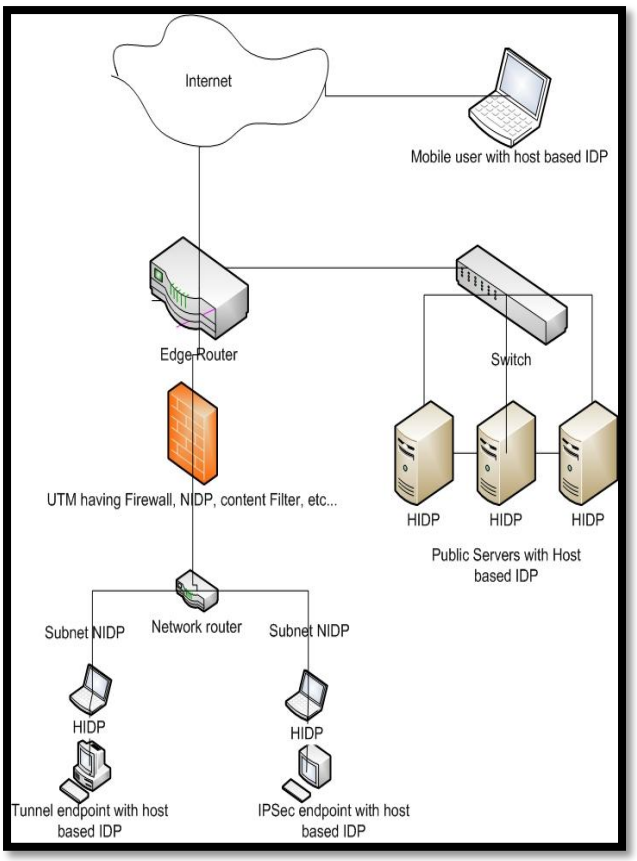

Fig.6 CIDP Architecture and Components

One of the example would be if certain security events which are generated when someone visits particular site. This site is vulnerable and can launch a possible attack on X version of IE browser. However, other browsers such as Mozilla or Chrome are not vulnerable to this attack. Now if system tries to correlate events without prior information of application then such correlation can mislead to appropriate security response. So event correlation module should be aware of all necessary information required to correlate security events to implement proper security response. In this case, a response should be to update the IE browser or patch the system with fix so it cannot be compromised. It will minimize the response focus and hence security can be implemented at its best.

In order to implement better response action and security we believe that event correlation module should be aware of application and application version. As we have explained that combination of vulnerable site and vulnerable IE version is a security threat. None of them can be considered as a security threat alone. So instead of generating alert on any web request to that server or that page, alert should be generated when request from particular IE version to visit that page or site.

Citing another example [6], it advocates implementation of verification layer at central location before applying correlation technique and algorithm. They suggest four kind of verification when sensor sends an alert. Sensor reliability check, attack reach ability check, vulnerability verification and target health check. Objective of this implementation is to reduce False Positives and False Negatives.

This kind of implementation might lead to two possible problems:

- As verification a check has been implemented at central level so there might be a case that central server gets lots of alerts and run out of capacity. Therefore, certain check has to be implemented at sensor level itself. E.g., attacks reach ability check. If targeted application is not going to compromise because of possible attack then there is no need to raise an alert for the same. Implementation of such checks can reduce false positives at first place itself.
- Attack Reach ability check verifies the possibility of attack on targeted system. It requires information about whole network model along with asset database. Implementation of such verification after every alert is relatively impossible to implement. It may lead to false positive or false negative if two protocol servers are running on same host. Alert has to be verified against the application servers or applications running on host instead of network layer IP address and protocol only.

As we have described, target reach ability check is one of the necessary focus area in study of reducing false positives. However, it should be implemented at a sensor level itself instead of central analysis server. Correlating non-reachable attacks alerts (false positives) will harm the outcome of central module and it can run out of resources due to heavy load of false positives. Implementation of the same looks relatively impossible from central location in absence of application or server running on that machine at that time. Inventory information might give idea about total applications installed on that system but it may not give a snapshot of running application at the time of alert has been generated.

Statistical analysis shown in this paper [2] states that in live scenario 92.85% of false alerts are false positives and 7.15% are false negatives. So controlling false positive is crucial for any IDPS. They also state that out of these FPs 91% of FPs occur only because policy configuration and not due to any security issue. It is also observed that all such FPs majorly occur due to traffic similarities between protocols.

Examples of such events are as follows.

- The “SQL Injection comment attempt” alert results from Bit Torrent clients who happen to bind port 80, and the traffic happens to be similar to an injection attempt.
- The “VERITAS Backup Agent DoS attempt” alert results from Bit Torrent clients who bind port 10000 (the port monitored by the rule), and the traffic happens to be similar to a DoS attempt.

In both the example applications were just using the standard protocol ports but they were not sending any malicious traffic. But IDP sensor will see them as malicious traffic reason being normal IDPS sensor works on host and protocol port. IDPS sensor should also consider application itself for the same. If sensor can correlate application and signature then the rate of such false positives can drastically reduce.

Comparison of this paper is more towards improvement of the approach they have suggested. It supports our approach in a way where false positives are higher because of wide range of application and application protocol running on same port or on random ports. Therefore, to reduce false positives one should relate attack with application itself.

Another solution [7] talks about finding a vulnerable attack when any application is running with root privileges. Argument they are giving is when application runs with root privileges, they can harm more. It doesn't look right if we talk about application vulnerability or malware attacks. Someone can hack username and password of a person when running browser with user mode privileges. Someone can launch NTLM vulnerability attack and can cause more harm than it could have made it on the process running with root privileges.

We believe that implementation of the security should be at every level. User privilege might give a firsthand idea about possible harm but it cannot be a parameter to deploy security. IDPS sensors should be deployed in such a way that it can track all the malicious traffic and can correlate the traffic with application and application behaviour if possible. So far, we have not studied in the area of application behaviour but any attack should be related to application and application property instead of application privileges.

[8] Deals with two types of attacks, Partial Completion Detection and Scan attacks. Partial completion attacks are more like a DoS attacks. Solution works in three phases. First phase is to find out the operating range of PCF (Partial Completion Filter, proprietary data structure to hold value of counters), second stage is about finding out the flows that are outside of this threshold and third stage analyzing false positives and false negatives.

There might be some problem in calculating initial operating range for counters in today's world. Counters are implemented in this algorithm is majorly based upon difference between SYN and FIN along with source IP and source port. If we consider Web protocol as an example then we can see two different behaviours for the same. If source is using a proxy server then there are chances that one connection is kept alive for many http requests and responses for same domain. At the same time, some of the advanced browser uses connect ahead functionality to setup a connections for future requests. So operating time has to be sufficiently long and should consider both the scenarios. It looks tricky and troublesome. It also does not talk about deployment strategy clearly but seems like it advocates deployment at outgoing edge of network. Maintaining flows at router level would make this scenario more complex to implement.

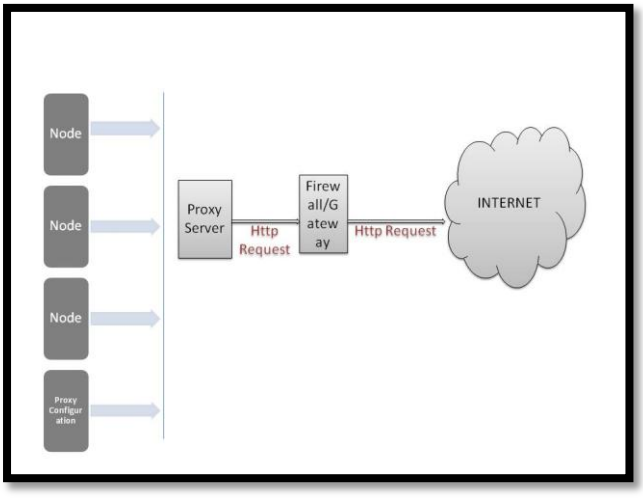

Fig.7 Typical Deployment of Proxy Server in Local Network

If someone wants to implement above-mentioned approach then it can better implemented using our work. As we have explained that implementing, it at edge router level might lead chances to run into a situation of false positives due to wrong operative frequency calculation. If the same can be implemented at host level then one can find out operating frequency of services easily using our Application Aware Logger System. Using our system one can get the hold on application data also to determine if the request is coming from proxy server as whenever any proxy server sends a request on behalf of any host we can find out actual host using "X-ForwardedFor" tag in request header. Using this parameter, we can implement operating frequency of concurrent connection in a better way.

In [9], implementation of the approach this paper suggested is Locality Buffering. It is a technique for adapting the packet stream in a way that accelerates sensor processing by improving the locality of its memory accesses and thus reducing its cache misses. It holds packets and arranges them as per the locality buffer allocation methods. So far, they have suggested methods based on source port and destination port, destination port only or known destination port. However, going further they also advocate that positive effects of these methods can get diluted in today's world where most of the applications use unknown and different ports. To encounter this problem, one must implement the locality buffer allocation on the bases of application or application protocol itself. If that can be done then it can give the best use of locality buffering.

As described, locality buffering has been done by grouping signatures by combination of source port and destination port. We suggest that it should be based on application also. Nowadays more applications are using either a same port or random ports so it might kill the main objective of signature locality caching.

Another case [10] talks about snort rules generalization. Technique they are using is to compare internet packet with snort rules and if any of the condition matches of the rule then lower severity alert should be generated as it may have some variation of a known attack. It is majorly to deal with the fact that application might be working on different ports and can choose random ports to attack. But rule generalization and alert merging at central location is tough and time consuming process. Instead, rule generalization should be bound with application protocol or application itself if possible.

Rule generalization also talks about content generalization which has been referred here [10] but not elaborated or discussed. IDS takes 10 times more time with generalized rules, which paper claim, is in operating limit. However, with gigabit Ethernet it is definitely not. So generalized content rules cannot be a viable method for large throughput oriented networks.

Signature generalization because of condition matching leads to unnecessary load on sensor. E.g. If we ignore one matching condition then all packets will be matched against that rule. At the end, it is being done to deal with the situation where application is working on the different port as HTTP server is running on port 8080 instead of port 80 or web traffic is being passed on port other than 80. Our approach suggests achieving the generalization because of applications. It will make signatures port independent without compromising performance of the overall IDPS system.

[11] Covers two different concepts: anomaly detection and signature generation. With context to our study, we largely focus on model proposed by them to generate weighted signature and use of the same. Anomaly detection system proposed by them has ability to find novel attacks. By mining anomalous traffic episodes from Internet connections, it detects anomalies. A weighted signature generation scheme is proposed to integrate ADS with SNORT by extracting signatures from anomalies detected. IDS extract signatures from the output of ADS and add them into the SNORT signature database for fast and accurate intrusion detection. Fig. 8 describes the same. The weighted signature extracted is as follows:

alert icmp$EXTERNAL NET any <> $HOME NET any (msg :00 possible pod attack00; itype : 8; dsize : 1; 480 <> 1; 490; threshold : type both; track by dst; count 10 seconds 1; sid : 900; 001; rev : 0;)

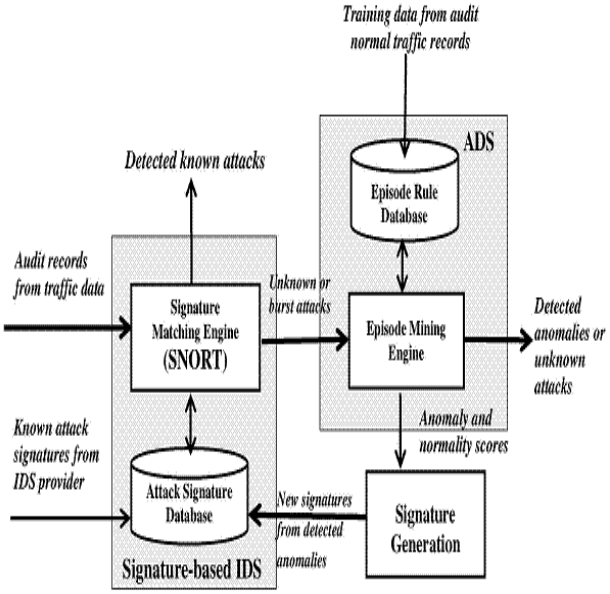

Fig.8 Anomalous Data Detection Technique

The systems mentioned above tracks the destination for data size between 1480 and 1490 for request type ICMP Echo. If such 10 occurrence happen then it will generate the alert. Signature generation has been done using anomaly analysis done by the anomaly detection system. Now when such alert occurs then possible action response can be to block the particular host (source or destination or both). Major bottleneck in this approach is it blocks the host and not the malicious application generating attack. If signature alteration or alert generation tackles down this situation and reveal the information about application, which is causing this attack, then one can deploy maximum-security measures.

4. PROPOSED SOLUTION & CONCLUSION

In our proposed system, we have developed a logging agent, which is installed on each individual host. This logging agent sends event information to Event Collector, which uses UDP protocol and stores event log in a database. This database is implemented on admin server. The administrator then performs analysis of event log stored in the database and takes policy decisions to allow, deny or drop packets. In this way, monitoring and event co-relation is distributed and analysis is done locally.

A system is considered robust if it does not produce false positives and does not completely fail to detect intrusions. One of an approach after an intrusion takes place is getting intruder's IP address and then to track all the activities done by the intruder system to generate its activity log and to do cross attack on the intruder system [12]. However, this cannot be applied in all situations. Therefore, it is essential for any IDS to know whether the attack is relevant or not. In addition, relevance of an attack depends on the kind of operational environment where we deploy IDPS. For this, attack classification is very essential. The classification can be done using various techniques such as Data mining, artificial intelligence on neural networks, etc. We propose classification of attacks by implementing concepts of data mining in the following manner:

First, we are using rule-set of Suricata, which is an IDPS used widely nowadays. From the existing rule-set of Suricata, we have taken two sets: Web-client and Web-server rules. Since our proposed system is designed taking into consideration corporate environment, we have classified the rule-set into further four categories:

1) Server-side Inbound.
2) Client-side Inbound.
3) Server-side Outbound.
4) Client-side Outbound.

This categorization is because an attack can be launched from within the network or from outside the network. In a typical network, there are two types of applications running: Client application and Server Application. Whenever a client application in the network requests for any service outside the network, it may become vulnerable to attacks from servers running outside the network. In addition, when any service provided by a Server application within the network is requested by an outside application, it may also launch an attack on server application. Apart from this, any vulnerable or infected application, client or server can possibly make attacks, to an application outside the network.

5. REFERENCES

[1] Usman Asghar Sandhu, Sajjad Haider, Salman Naseer, and Obaid Ullah Ateeb, A Study of the Novel Approaches Used in Intrusion, International Journal of Information and Education Technology, Vol. 1, No. 5, December 2011 Detection and Prevention Systems

[2] Cheng-Yuan Ho; Yuan-Cheng Lai; I-Wei Chen; Fu-Yu Wang; Wei-Hsuan Tai; , "Statistical analysis of false positives and false negatives from real traffic with intrusion detection/prevention systems," *Communications Magazine, IEEE* , vol.50, no.3, pp.146-154, March 2012

[3] Sourour, M.; Adel, B.; Tarek, A.;, "Security Implications of Network Address Translation on Intrusion Detection and Prevention Systems," IEEE *International Conference on Network and Service Security, 2009. N2S '09.* , vol., no., pp.1-5, 24-26 June 2009

[4] P.G. Neumann and P.A. Porras. EMERALD: Event monitoring enabling responses to anomalous live disturbances. In NCSC '97: Proc. 20th NIST National Information Systems Security Conference, pages 353–365, 1997.

[5] Ken Deeter, Kapil Singh, Steve Wilson, Luca Filipozzi and Son Vuong, "APHIDS: A Mobile Agent-Based Programmable Hybrid Intrusion Detection System", Mobility Aware Technologies and Applications, Lecture Notes in Computer Science, Springer, 2004, Volume 3284/2004, 244-253, DOI: 10.1007/978-3-540-30178-3_23

[6] Thomas Heyman, Bart De Win, Christophe Huygens, and Wouter Joosen, "Improving Intrusion Detection through Alert Verification", IEEE Transaction on Dependable and Secure Computing, 2004.

[7] Koller, R.; Rangaswami, R.; Marrero, J.; Hernandez, I.; Smith, G.; Barsilai, M.; Necula, S.; Sadjadi, S.M.; Tao Li; Merrill, K.; , "Anatomy of a Real-Time Intrusion Prevention System," *International Conference on Autonomic Computing, 2008. ICAC '08.* , vol., no., pp.151-160, 2-6 June 2008

[8] Ramana Rao Kompella, Sumeet Singh, and George Varghese, *On Scalable Attack Detection in the Network*, IEEE/ACM transactions on networking, vol. 15, no. 1, february 2007

[9] Konstantinos Xinidis, Ioannis Charitakis, Spiros Antonatos, Kostas G. Anagnostakis, and Evangelos P. Markatos, An Active Splitter Architecture for Intrusion Detection and Prevention, IEEE transactions on dependable and secure computing, vol. 3, no. 1, january-march 2006

[10] Uwe Aickelin, Jamie Twycross and Thomas Hesketh-Roberts, Rule Generalisation in Intrusion Detection Systems using SNORT, International Journal of Electronic Security and Digital Forensics (IJESDF), (1), pp 101-116, 2007

[11] Kai Hwang, Min Cai, Ying Chen, Min Qin, Hybrid Intrusion Detection with Weighted Signature Generation over Anomalous Internet Episodes, IEEE transactions on dependable and secure computing, vol. 4, no. 1, January-March 2007

[12] Sumit A. Khandelwal, Shoba. A. Ade, Amol A. Bhosle and Radha S. Shirbhate, A Simplified Approach to Identify Intrusion in Network with Anti Attacking Using .net Tool., International Journal of Computer and Electrical Engineering, Vol. 3, No. 3, June 2011

[13] Khalid Alsubhi, Nizar Bouabdallah, Raouf Boutaba, Performance analysis in Intrusion Detection and Prevention Systems, Proceedings of the 12th IFIP/IEEE International Symposium on Integrated Network Management, IM 2011, Dublin, Ireland, May 2011,pages 369-376

[14] Ke Yun; Zhu Jian Mei; , "Research of Hybrid Intrusion Detection and Prevention System for IPv6 Network," *2011 International Conference on Internet Technology and Applications (iTAP)*, , vol., no., pp.1-3, 16-18 Aug. 2011