ISSN (Online): 2230-7893
www.IJCEM.org

A REVIEW ON INFORMATION FLOW IN INTRUSION DETECTION SYSTEM

Yogesh Kumar[1], Swati Dhawan[2]
[1]Astt. Prof. in BPR College of Engg., Gohana, Sonipat, Haryana
yogs_crsce@yahoo.com

[2]Mtech Scholor, BPS Mahila Vishwavidyalaya
swatidhawan_04@yahoo.co.in

Abstract

An Intrusion Detection System (abbreviated as IDS) is a defense system, which detects hostile activities in a network. The key is then to detect and possibly prevent activities that may compromise system security, or a hacking attempt in progress including reconnaissance/data collection phases that involve for example, port scans. One key feature of intrusion detection systems is their ability to provide a view of unusual activity and issue alerts notifying administrators and/or block a suspected connection. This paper explains IDS classification and different phases of Intrusion analysis. This paper also describes the working of each component involved in multi-tiered architecture.

Keywords: *IDS, IPS, HIDS, NIDS, Protocols, DoS.*

1. INTRODUCTION

An intrusion is an active sequence of related events that deliberately try to cause harm, such as rendering a system unusable, accessing unauthorized information or manipulating such information. To record the information about both successful and unsuccessful attempts, the security professionals place the devices that examine the network traffic, called sensors. These sensors are kept in both front of the firewall (the unprotected area) and behind the firewall (the protected area) and values through comparing the information recorded by the two.

An Intrusion Detection Systems(IDS) can be defined as the tool, methods and resources to help identity, access and report unauthorized activity. Intrusion Detection is typically one part of an overall protection system that is installed around a system or device. IDS work at the network layer of the OSI model and sensors are placed at the choke points on the network. They analyze packets to find specific patterns in the network traffic- if they find such a pattern in the traffic, an alert is logged and a response can be based on data recorded

2. CLASSIFICATION OF INTRUSION DETECTION SYSTEMS

Intrusion detection systems fall into one of three categories:, , and hybrids of the two.

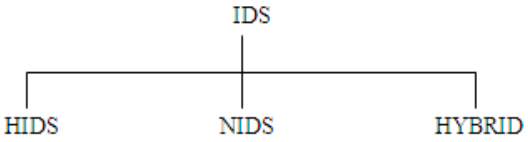

Fig. 1

2.1 Host Based Intrusion Detection Systems (HIDS):

It will require some software that resides on the system and can scan all host resources for activity; some just scan syslog and event logs for activity. It will log any activities it discovers to a secure database and check to see whether the events match any malicious event record listed in the knowledge base.

2.2 Network Based Intrusion Detection Systems (NIDS):

It is usually inline on the network, and it analyzes network packets looking for attacks. NIDS receives all packets on a particular network segment, including switched networks. It carefully reconstructs the streams of traffic to analyze them for pattern of malicious behavior. Most NIDS are equipped with facilities to log their activities and report or alarm on questionable events.

2.3 Hybrid Intrusion Detection System (HIDS) :

It monitors events occurring on the host system, with a NIDS which monitors network traffic. The basic process of an IDS is that a NIDS or HIDS passively collects data and preprocesses and classifies them. Statistical analysis can be done to determine whether the information falls outside normal activity, and if so, it is then matched against a knowledge base. If a match found, an alert is sent. Goal of the Intrusion Detection Systems is to improve an information system's security.

ISSN (Online): 2230-7893
www.IJCEM.org

3. PHASES OF INTRUSION ANALYSIS.

Intrusion analysis process can be broken down into four phases and they are as follows:

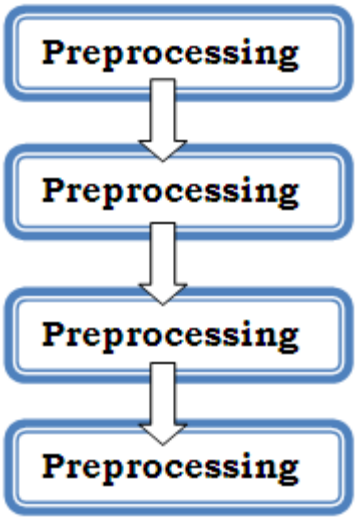

Fig. 1

3.1 Preprocessing:

Is a key function once collected from an IDS or IPS sensors. In this step data, are organized in some fashion for classification. This stage would help in determine the format the data are put into, which would be a canonical format or a structured database. Once the data are formatted they are further classified, this classifications depends upon the analysis schemas being used.

3.2 Analysis:

Once the preprocessing is completed, the analysis stage begins. The data record is compared with the Knowledge base. The data record will either be logged as an intrusion event or it will be dropped and next data record is analyzed.

3.3 Response:

In the intrusion detection systems we get the information passively after the fact, so we would get an alert after the fact. The response can be set to be automatically performed, or can be done manually after someone manually analyzed the situation.

3.4 Refinement:

This is the stage where fine tunings is done, based on the previous usage and detected intrusions. This helps in reducing false positive levels and to have more security tool. These are tool like CTR (Cisco Threat Response) that helps with the refining stage by actually making sure that an alert is valid by checking whether you are vulnerable to the attack or not. Rule based detection, even known as signature detection, pattern matching and misuse detection.

4. INTRUSION DETECTION ARCHIETECTURE

The roles performed by and relationships among machines, device, applications and processes, including the conventions used for communicating between them, define architecture. The intrusion detection architecture is a designed structure on which every element fits. An effective architecture is one in which each machine, device, component and process perform its role in an effective and coordinated manner resulting in information processing and output.

Different types of tired architectures are as follows:

- Single tired architecture
- Multi tired architecture

4.1 Single tired Architecture:

A single tired architecture, the most basic of the architecture discussed here is one in which components in an IDS collect data and process data themselves, rather than passing the output they collect to another set of components. Example HIDS tool that takes the output system logs and compares it to known patterns of attack.

4.2 Multi tired Architecture:

A multi-tired architecture involves multiple components that pass information to each other. IDS mainly consists of three parts and they are as under:

- Sensors
- Analyzers or agents
- Manager

4.2.1. Sensors:

Sensors are critical in intrusion detection architectures. Sensors are critical in intrusion-detection architectures; they are the beginning point of intrusion detection and prevention because they supply the initial data about potentially malicious activity. A deeper look at sensor functionality, deployment, and security will provide insight into exactly what sensors are and how they work.

Sensor Functions:

Considering all the possible intrusion-detection components within a particular architecture, sensors are usually (but not always) the lowest end components. In other words, sensors typically do not have very sophisticated functionality. They are usually designed only to obtain certain data and pass them on.

There are two basic types of sensors:

- network-based
- and host-based sensors.

(i) Network Based Sensor:

Network-based sensors, the more frequently deployed of the two types, are programs or network devices (such as physical devices) that capture data in packets traversing a local Ethernet or token ring or a network switching point. One of the greatest advantages of network-based sensors is the sheer number of hosts for which they can provide data. The cost-effectiveness of this approach is huge. Additionally, if configured properly, sensors do not burden the network with much additional traffic, especially if two network interfaces—one for monitoring and the other for management—are used. A monitoring interface has no TCP/IP stack whatsoever, nor does it have any linkage to any IP address, both of which make it an almost entirely transparent entity on the network.

The programs that intrusion-detection tools most frequently use as sensors are **tcpdump** and **libpcap**. To reiterate, tcpdump captures data from packets and prints packet headers of packets that match a particular filter (or Boolean) expression. Packet parameters that are particularly useful in intrusion detection and prevention are time, source and destination addresses, source and destination ports, TCP flags, initial sequence number from the source IP for the initial connection, ending sequence number, number of bytes, and window size. tcpdump is an application, but libpcap is a library called by an application. libpcap is designed to gather packet data from the kernel of the operating system and then move it to one or more applications—in this particular case, to intrusion-detection applications. For example, an Ethernet card may obtain packet data from a network. The underlying operating system over which libpcap runs will process each packet in many ways, starting with determining what kind of packet it is by removing the Ethernet header to get to the next layer up the stack.

The next layer will be the IP layer; if so, the IP header must be removed to determine the protocol at the next layer of the stack (although it is important to note that in the case of the IP protocol, hexadecimal values of 1, 6, or 11 starting at byte position 40 within the packet header indicate that the transport protocol is ICMP, TCP, or UDP (User Datagram Protocol), respectively). If the packet is a TCP packet, the TCP header is also removed and the contents of the packet are then passed on to the next layer up, the application layer. libpcap provides intrusion-detection applications with this data (payload) so that these applications can analyze the content to look for attack signatures, names of hacking tools, and so forth. libpcap is advantageous not only in that it provides a standard interface to these applications, but also because, like tcpdump, it is public domain software.

(ii) Host-Based Sensors:

Host-based sensors, like network-based sensors, could possibly also receive packet data captured by network interfaces and then send the data somewhere. Instead of being set to promiscuous mode, the network interface on each host would have to be set to capture only data sent to that particular host. However, doing so would not make much sense, given the amount of processing of data that would have to occur on each host. Instead, most host-based sensors are programs that produce log data, such as Unix daemons or the Event Logger in Windows NT, 2000, XP, and Windows Server 2003. The output of these programs is sent (often through a utility such as scp, secure copy, which runs as a cron job, or through the Windows Task Scheduler) to an analysis program that either runs on the same host or on a central host. The program might look for events indicating that someone has obtained root privileges on a Unix system without entering the su (substitute user) command and the root password—a possible indication that an attacker has exploited a vulnerability to gain root privileges.

4.2.2 Agents or Analyzers:

Agents are relatively new in intrusion detection, having been developed in the mid-1990s. Their primary function is to analyze input provided by sensors. Although many definitions exist, we'll define an *agent* as a group of processes that run independently and that are programmed to analyze system behavior or network events or both to detect anomalous events and violations of an organization's security policy. Some agents may, for example, examine network traffic and host-based events rather generically, such as checking whether normal TCP connections have occurred, their start and stop times, and the amount of data transmitted or whether certain services have crashed. Having agents that examine UDP and ICMP traffic is also desirable, but the UDP and ICMP protocols are stateless and connectionless. Other agents might look at specific aspects of application layer protocols such as FTP, TFTP, HTTP, and SMTP as well as authentication sessions to determine whether data in packets or system behavior is consistent with known attack patterns.

Our definition of agent states that agents run independently. This means that if one agent crashes or is impaired in some manner, the others will continue to run normally (although they may not be provided with as much data as before). It also means that agents can be added to or deleted from the IDS or IPS as needed.

Although each agent runs independently on the particular host on which it resides, agents often cooperate with each other by using a particular communication protocol over the network. When an agent detects an anomaly or policy violation (such as a brute

force attempt to *su* to root, or a massive flood of packets over the network), in most cases, the agent will immediately notify the other agents of what it has found. This new information, combined with the information another agent already has, may cause that agent to report that an attack on another host has also occurred.

Agents sometimes generate false alarms, too, thereby misleading other agents, at least to some degree. However, a good IDS or IPS will allow the data that agents generate to be inspected on a management console, allowing humans to spot false alarms and to intervene by weeding them out.

Agent Deployment Considerations:

Decisions about deployment of agents are generally easier to make than decisions concerning where to deploy sensors. Each agent can and should be configured to the operating environment in which it runs. In host-based intrusion detection, each agent generally monitors one host, although, as mentioned before, sometimes sensors on multiple hosts send data to one or more central agents. Choosing the particular hosts to monitor is thus the major dilemma in deciding on the placement of host-based agents. Most organizations that use host-based intrusion detection select "crown jewel" hosts, such as servers that are part of billing and financial transaction systems, more than any other. A few organizations also choose a few widely dispersed hosts throughout the network to supplement network-based intrusion detection.

In network-based intrusion detection, agents are generally placed in two locations:

Where they are most *efficient*, Efficiency is related to the particular part of a network where connections to sensors and other components are placed. The more locally co-resident the sensors and agents are, the better the efficiency.

Where they will be sufficiently *secure*, Placing agents in secure zones within networks, or at least behind one or more firewalls, is essential.

4.2.3 Manager Component:

The final component in a multi-tiered architecture is the *manager* (sometimes also known as the *server*) component. The fundamental purpose of this component is to provide an executive or master control capability for an IDS or IPS.

Manager Functions:

We've seen that sensors are normally fairly low-level components and that agents are usually more sophisticated components that, at a minimum, analyze the data they receive from sensors and possibly from each other. Although sensors and agents are capable of functioning without a master control component, having such a component is extremely advantageous in helping all components work in a coordinated manner. Additionally, the manager component can perform other valuable functions.

- **Data Management:**

IDSs can gather massive amounts of data. One way to deal with this amount of data is to compress (to help conserve disk space), archive it, and then periodically purge it. Having sufficient disk space for management purposes is, of course, a major consideration. One good solution is RAID (Redundant Array of Inexpensive Disks), which writes data to multiple disks and provides redundancy in case of any disk failing. Another option is optical media, such as worm drives (although performance is an issue).

Ideally, the manager component of an IDS or IPS will also organize the stored data. A relational database, such as an Oracle or Sybase database, is well suited for this purpose. Once a database is designed and implemented, new data can be added on the fly, and queries against database entries can be made.

- **Alerting**:

Another important function that the manager component can perform is generating alerts whenever events that constitute high levels of threat occur. Agents are designed to provide detection capability, but agents are normally not involved in alerting because it is more efficient to do so from a central host. Agents instead usually send information to a central server that sends alerts whenever predefined criteria are met. This requires that the server not only contain the addresses of operators who need to be notified, but also have an alerting mechanism.

- **Event Correlation:**

Another extremely important function of the manager component is correlating events that have occurred to determine whether they have a common source, whether they were part of a series of related attacks, and so forth. The manager component may, for example, track the progression of each attack from stage to stage, starting with the preparatory (doorknob rattling) stage.

- **Monitoring Other Components**:

It is important for monitoring component to check the health of sensors and agents. The manager is the ideal component in which to place this function because (once again) this function is most efficient if it is centralized.

In host-based intrusion detection, the manager can monitor each host to ensure that logging or auditing is functioning correctly. The manager component can also track utilization of system and network resources, generating an alert if any system or any part of the network is overwhelmed.

- **Policy Generation and Distribution:**

ISSN (Online): 2230-7893
www.IJCEM.org

Another function that is often embedded in the manager component is policy generation and distribution. In the context of the manager component, *policy* refers to settings that affect how the various components of an intrusion-detection or intrusion-prevention system function. A policy could be set, for example, to activate all agents or to move an agent from one machine to another.

- **Security Management and Enforcement :**

Security management and enforcement is one of the most critical functions that can be built into the manager component.

- **Management Console :**

Providing an interface for users through a management console is yet another function of the manager component. The management console should display critical information—alerts, the status of each component, data in individual packets, audit log data, and so forth—and should also allow operators to control every part of an IDS.

Manager Deployment Considerations:

One of the most important deployment considerations for the manager component is ensuring that it runs on extremely high-end hardware (with a large amount of physical memory and a fast processor) and on a proven and reliable operating system platform (such as Solaris or Red Hat Linux). Continuous availability of the manager component is essential—any downtime generally renders an IDS totally worthless.

Using RAID and deploying redundant servers in case one fails are additional measures that can be used to help assure continuous availability.

5. INFORMATION FLOW IN IDS

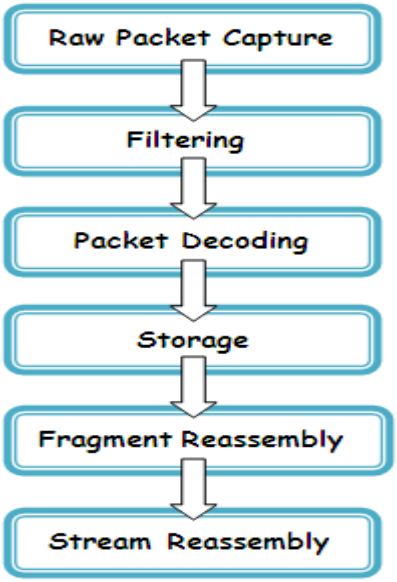

Fig. 2

(i) Raw Packet Capture :

IDS internal information flow starts with raw packet capture. This involves not only capturing packets, but also passing the data to the next component of the system. P*romiscuous* mode means a NIC picks up every packet at the point at which it interfaces with network media. To be in *non-promiscuous* mode means a NIC picks up only packets bound for its particular MAC address, ignoring the others. Non-promiscuous mode is appropriate for host-based intrusion detection and prevention, but not for network-based intrusion detection and prevention. A network-based intrusion detection system normally has two NICs-one for raw packet capture and a second to allow the host on which the system runs to have network connectivity for remote administration.

The IDS must save the raw packets that are captured, so they can be processed and analyzed at some later point. IDSs typically experience all kinds of problems, but one of the most-common problems is packet loss. A frequent variation of this problem is that the NIC used to capture packets receives packets much faster than the CPU of the host on which the IDS runs is capable of despooling them. A good solution is simply to deploy higher-ended hardware.

Another problem is this: the IDS itself cannot keep up with the throughput rates. Throughput rate is a much bigger problem than most IDS vendors publicly acknowledge-some of the best-selling products have rather dismal input processing rates. One solution is to filter out some of the input that would normally be sent to the IDS.

(ii) Filtering :

Filtering means limiting the packets that are captured according to a certain logic based on characteristics, such as type of packet, IP source address range, and others. An organization might be interested in only certain types of incoming traffic, (as often occurs) only TCP traffic because, historically, more security attacks have been TCP-based than anything else.

Filtering raw packet data can be done in several ways. The NIC itself may be able to filter incoming packets. Although early versions of NICs (such as the 3COM 3C501 card) did not have filtering capabilities, modern and more sophisticated NICs do.

Another method of filtering raw packet data is using packet filters to choose and record only certain packets, depending on the way the filters are configured. *libpcap*, for example, offers packet filtering via the bpf interpreter. You can configure a filter that limits the particular types of packets that will be processed further. The bpf interpreter receives all the packets, but it decides which of them to send on to applications.

(iii) Packet Decoding :

Packets are subsequently sent to a series of decoder routines that define the packet structure for the layer two (datalink) data (Ethernet, Token Ring, or IEEE 802.11) that are collected through promiscuous monitoring. The packets are then further decoded to determine whether the packet is an IPv4 packet or IPv6, as well as the source and destination IP addresses, the TCP and UDP source and destination ports, and so forth.

Some IDSs such as Snort go even further in packet decoding in that they allow checksum tests to determine whether the packet header contents coincide with the checksum value in the header itself. Checksum verification can be done for one, or any combination of, or all of the IP, TCP, UDP, and ICMP protocols.

(iv) Storage :

Once each packet is decoded, it is often stored either by saving its data to a file or by assimilating it into a data structure while, at the same time, the data are cleared from memory. But writing intrusion detection data to a file also has some significant disadvantages. For one thing, it is cumbersome to sort through the great amount of data within one or more file(s) that are likely to be accumulated to find particular strings of interest or perform data correlation. Additionally, the amount of data that are likely to be written to a hard drive or other storage device presents a disk space management challenge. An alternative is to set up data structures, one for each protocol analyzed, and overlay these structures on the packet data by creating and linking pointers to them.

(v) Fragment Reassembly :

Decoding "makes sense" out of packets, but this, in and of itself, does not solve all the problems that need to be solved for an IDS to process the packets properly. Packet fragmentation poses yet another problem for IDSs. A reasonable percentage of network traffic consists of packet fragments with which firewalls, routers, switches, and IDSs must deal. One packet fragment can overlap another in a manner that the fragments will be reassembled so subsequent fragments overwrite parts of the first one instead of being reassembled in their "natural" sequential order. Overlapping fragments are often indications of attempted denial-of-service attacks (DoS) or IDS or firewall evasion attempts.

(vi) Stream Reassembly :

Stream reassembly means taking the data from each TCP stream and, if necessary, reordering it. This requires determining when each stream starts and stops, something that is not difficult given that TCP communications between any two hosts begin with a SYN packet and end with either a RST (reset) or FIN/ACK packet.

6. IDS PROS AND CONS

Pros of IDS are as follows:

- Detects external hackers and network based attacks.
- Offers centralized management for correlation of distributed attacks.
- Provides the system administrator the ability to quantify attacks.
- Provides an additional layer of protection.
- Provides defense in depth.

Cons of IDS are as follows:

- Generates false positives and negatives.
- Require full time monitoring.
- It is expensive
- Require highly skilled staff's.

7. CONCLUSION

The intrusion detection and intrusion prevention arenas are extremely dynamic, with new findings, functions, and models being created all the time. A considerable amount of research on data visualization methods for intrusion detection data is also currently being conducted.

8. REFFERENCES

1. Peng Ning(2005), "*Intrusion Detection Systems Basics*", published in "Hand Book of Computer", Volume 3, edited by Hossien Bidgoli, Published by John Wiley & Sons, Inc (PP 685 to 700).
2. Simon Edwards, (September 2002), "*Network Intrusion Detection Systems: Important IDS Network Security Vulnerabilities*", white paper Top Layer Networks, Inc.
   Webpage:
   Http://www.toplayer.com/pdf/WhitePapers/wp_network_intrusion_system.pdf/
3. Hassen Sallay, Khalid A. AlShalfan, Ouissem Ben Fred, (2009)**,** "*A scalable distributed IDS Architecture for High speed Networks"*, IJCSNS International Journal of Computer Science and Network Security, VOL.9 No.8,August 2009.
4. S. Northcutt, *Network Intrusion Detection*, New Riders, Indianapolis, 1999.
5. NIST, SP 800-31, *Intrusion Detection Systems*. Website:http://csrc.nist.gov/publications/nistpubs/
6. Nazario, Jose, *Defense and Detection Strategies Against Internet Worms*, Artech House Publishers, 2003.
7. Rash, Michael et al, *Intrusion Prevention and Active Response: Deployment Network and Host IPS*, Syngress, 2005.