International Journal of Multidisciplinary and Current Research ISSN: 2321-3124
Research Article Available at: http://ijmcr.com

Honey Pot Enabled Intrusion Detection System

Sameer

Shah Satnam Ji P.G'Boys College, Sirsa, India

Accepted 03 Feb 2017, Available online 26 Feb 2017, **Vol.5 (Jan/Feb 2017 issue)**

Abstract

Intrusion detection system (IDS) is the process of monitoring computers or networks for unauthorized entrance, activity, or file modification. It is hardly difficult to provide secure information systems and to maintain them in such a secure state in their lifetime and utilization as everyday intruders have been increasing whether it was inside or outside. This paper first presents a thorough survey on the field of network intrusion detection and a classification of the systems according to the survey. The most common shortcomings in the existing intrusion detection systems are unknown attacks, false positives and false negatives. We present the design of HPEIDS (Honey Pot Enabled Intrusion Detection System) which solves the problems previously mentioned. Using honey pot with IDS also increases the flexibility and security of IDS. We also present the definition of the honey pot along with the Multi-level log mechanism.

Keywords: *IDS etc.*

Introduction

Over the past few years demands for "more secured" systems have increased heavily. So there is currently a need for an up-to-date, thorough taxonomy and survey of the field of intrusion detection. This paper presents such taxonomy, together with a survey of the important research intrusion detection systems up-to-date and a classification of these systems both quantitatively and qualitatively. When a user of an information system takes an action that user was not legally allowed to take, it is called intrusion. The intruder may come from outside, or the intruder may be an insider, who exceeds his limited authority to take action.

Whether or not the action is harmful, it is of concern because it might be harmful to the health of the system, or to the service provided by the system. Most intrusion detection systems attempt to detect suspected intrusion, and then they alert a system administrator. Original intrusion detection systems assumed a single, stand-alone processor system, and detection consisted of post-facto processing of audit records. Today's systems consist of multiple nodes executing multiple operating systems that are linked together to form a single distributed system. Intrusions can involve multiple intruders.

The presence of multiple entities only changes the complexity, but not the fundamental problems. However, that increase in complexity is substantial. This survey states the basic assumptions and the alternative technical approaches used to detect intrusions. In this survey we attempt to determine the fundamental approaches and based on the shortcomings of existing IDSs like unknown attack, false positives we propose architecture called Honey Pot Enabled Intrusion Detection System (HPEIDS) which solves the shortcomings of earlier IDSs. Section 2 gives a brief overview on early research in intrusion detection systems Section 3 describes the summary of early findings. Section 4 classifies on intrusion detection systems. Section 5 elaborates on the proposed architecture of Honey Pot Enabled Intrusion Detection System (HPEIDS).

2. Early research in intrusion detection systems

The field of Intrusion detection is currently some twenty five years old. The seminal paper that is most often cited is James P. Anderson's technical report [1], where he divides the possible attackers of a computer system into the four groups:

External penetrator: The external penetrator has gained access to a computer that he is not a legitimate user of. Anderson uses this definition to include users that are, e.g. employees of some organization, where they have physical access to the building that houses the computing resource, even though they are not authorized to use it.

Masquerader: The masquerader is a user who, having gained access to the system. The masquerader can be both an external penetrator, and another authorized user of the system attempts to use the authentication information of another user, in effect becoming him, as far as the compute system is concerned. This is an interesting case, since there is no direct way of

differentiating between the legitimate user and the masquerader.

Misfeasor: The legitimate user can operate as a misfeasor, that is, although he [2] has legitimate access to privileged information, he abuses this privilege to violate the security policy of the installation.

Clandestine user: The clandestine user operates at a level below the normal auditing mechanisms, perhaps by accessing the machine with supervisory privileges. Since there is little, if any, evidence of this type of intrusive activity, this class of perpetrator can be difficult to detect.

While this problematisation in itself does not open the field of intrusion detection, Anderson goes on to state in reference to the masquerader class that: Masquerade is interesting in that it is by definition extra use of the system by the unauthorized user. As such it should be possible to detect instances of such use by analysis of audit trail records to determine:

a. Use outside of normal time
b. Abnormal frequency of use
c. Abnormal volume of data reference
d. Abnormal patterns of reference to programs or data.

Dorothy Denning [3] presented the idea that intrusions in computer systems could be detected by assuming that users of a computer system would behave in a manner that would lend itself to automatic profiling, i.e. that some model of the behavior of a particular user could be constructed by the intrusion detection system, and that subsequent behavior of a presumed user could be verified against that user's model, with the intention that behavior that deviated sufficiently from the norm would be flagged as anomalous, and hence indicative of a possible intrusion. Denning mentioned several such models, based on the use of statistics, Markov chains, time-series, etc. Denning stressed that the work presented gives the basis for performing these functions in real-time, or near real-time. This paper has its base in the earliest prototype of IDSs. Many different approaches to building detection models have been proposed. A survey of detection techniques is given in [3]. Stephanie Forrest presents an approach for modeling normal sequences using look ahead pairs [4] and contiguous sequences [5]. Helman and Bhangoo [6] present a statistical method to determine sequences which occur more frequently in intrusion data as opposed to normal data. Lee et al. [7] uses a prediction model trained by a decision tree applied over the normal data. Ghosh and Schwartzbard [8] use neural networks to model normal data. Lane and Brodley [9] examine unlabeled data for anomaly detection by looking at user profiles and comparing the activity during an intrusion to the activity under normal use. But all these approaches use a database of known signatures or algorithms to determine what production traffic is and what malicious activity is. However, information overload, unknown activity, false positives and false negatives can make analyzing and determining activity extremely difficult.

3. Summary of early findings

The early research uncovered several features of the two major approaches, anomaly based and signature based intrusion detection. The problems and advantages of the approaches can be summarized as:

Anomaly detection

Advantages: The operator need not configure the system, it automatically learns the behavior of a large number of subjects, and can be left to run unattended. Since it contains no knowledge, some would say prejudice, about how an intrusion would manifest itself, it has the possibility of catching novel intrusions, as well as variations of known intrusions.

Disadvantages: By definition it only flags unusual behavior, not necessarily illicit behavior per use. This can be a problem when the two types of behavior do not overlap. A system that learns to accept dangerous behavior as \normal" for a particular user, that slowly changes his behavior over time, will not find anything out of the ordinary when that user normally mounts his attack. The updating of the subject's profiles and the correlation of current behavior with those profiles is typically a computationally intensive task that can tax the available computing resources hard.

Signature detection

Advantages: The system knows for a fact, either suspect behavior, or how normal behavior should manifest itself. This leads to simple and efficient processing of the audit data. The rate of false positives being activity classed as an intrusion can also be kept low.

Disadvantages: Specifying the detection signatures is a highly qualified, and time consuming task. It is not something that \ordinary" operators of the system would do. Depending on how these signatures are specified, subtle variations of the intrusion scenarios can lead to them going undetected. Of course, the method has limited predictive powers. It cannot detect intrusions that are novel to it, especially not those of a fundamentally new class of intrusions. As previously stated it was hoped that by combining these approaches into a hybrid approach, the best of both worlds could be attained.

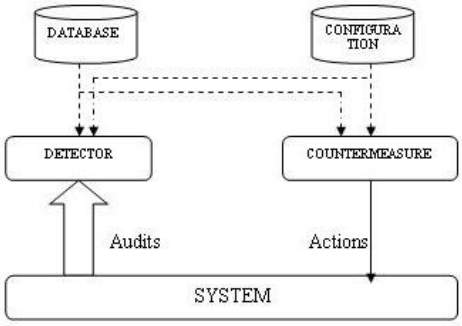

Fig. 1 A Simple Intrusion Detection System

4. A simple classification of IDS

Time of detection: Two main groups can be identified: those that attempt to detect intrusions in real time or near real-time, and those that process audit data with some delay, postponing detection i.e. non-real time, which in turn delays the time of detection.

Granularity of data-processing: This is a category that contrasts systems that process data continuously with those that process data in collection at a regular interval.
Source of audit Data: The two major sources of audit data in the surveyed systems are network data typically data read directly off a multicast network such as Ethernet) and host based security logs. The host based logs can include operating system kernel logs, application program logs, etc.

Response to detected intrusions: Passive versus active. Passive systems respond by notifying the proper authority, and they do not themselves try to mitigate the damage done, or actively seek to harm or hamper the attacker. Active systems exercise control over the attacked system, i.e. they modify the state of the attacked system to thwart or mitigate the effects of the attack. Such control can take the form of terminating network connections, increasing the security logging, killing errant processes, etc.

Locus of data-processing: The audit data can either be processed in a central location; irrespective of whether the data originates from one possibly the same site or is collected and collated from many different sources in a distributed fashion.

Locus of data-collection: Audit data for the processor/detector can be collected from many different sources in a distributed fashion, or from a single point using the centralized approach.

Degree of inter-operability: The degree to which the system can operate in conjunction with other intrusion detection systems, accept audit data from different sources, etc. This is not the same as the number of different platforms on which the intrusion detection system itself runs.

Here all these approaches use a database of known signatures or algorithms to determine what production traffic is and what malicious activity is. However, information overload, unknown activity, false positives and false negatives can make analyzing and determining activity extremely difficult.

Name of System	Detection Time	Granularity	Audit Source	Response Type	Location of Data	Data Collection	Inter-operability	KB IDS	BB IDS
Haystack	non-real	batch	host	passive	centralized	centralized	low		X
IDES	real	continuous	host	passive	centralized	distributed	low	X	
W & S	real	continuous	host	passive	centralized	centralized	low		X
Comp-watch	non-real	batch	host	passive	centralized	centralized	low		X
NSM	real	continuous	n/w	passive	centralized	centralized	low	X	X
DIDS	real	continuous	host	passive	centralized	distributed	low	X	
NIDES	real	continuous	host	passive	centralized	centralized	high	X	X
GrIDS	non-real	batch	both	passive	distributed	distributed	low		X
CSM	real	continuous	host	active	distributed	distributed	low	X	
Janus	real	continuous	host	active	centralized	centralized	low		X
EMERALD	real	continuous	host	passive	distributed	distributed	low	X	X
Bro	real	continuous	n/w	passive	centralized	centralized	low	X	

Abbreviations used: KB-Knowledge based, BB-Behavior based
Table 1: Classification of the surveyed Systems

5. Honeypot enabled intrusion detection system

A. Definition of Honeypot

Honeypots are closely monitored network decoys serving several purposes: they can distract adversaries from more valuable machines on a network, they can provide early warning about new attack and exploitation trends and they allow in-depth examination of adversaries during and after exploitation of a honeypot. Honeypots are a highly flexible security tool with different applications for security. They don't fix a single problem. Instead they have multiple uses, such as prevention, detection, or information gathering. Honeypots all share the same concept: a security resource that should not have any production or authorized activity. In other words, deployment of honeypots in a network should not affect critical network services and applications. A honeypot is a security resource whose value lies in being probed, attacked and compromised. In a word, honey pot provides an environment where intruders can he trapped or vulnerabilities accessed before an attack is made on real assets.

B. How the use of honey pot can improve the characteristics of an IDS

We propose that an IDS with honey pot as its component solves all the problems mentioned.

- A honey pot is designed to he compromised, not to he used for production traffic. Any traffic entering or leaving the network is suspicious by definition. This concept of no production traffic greatly simplifies the data capture and analysis.
- False positives are a constant challenge for most organizations. But a honey pot is a host that has no real purpose, other than to capture unauthorized activity. So honey pot reduces this problem by not having any true production traffic.
- False negatives are another challenge. Because there is little or no production activity within a honey pot, the honey pot reduces false negatives by capturing absolutely everything that enters and leaves itself. This means all the activity that is captured is most likely suspect.

As to unknown activity, even if IDS misses it, we have captured the activity. We can review all of the captured activity and identify the attack

C. Multi-level log mechanism (MLLM)

The purpose of the MLLM is to log all of the attacker's activity. This is the whole purpose of the honey pot, to collect information. Without it, the honey pot has no value. The key to MLLM is collecting information at as many layers as possible. Single layer is not secure and no single layer tells us everything. The HPEIDS has identified two critical layers of MLLM.

The honey pot captures the attacker's activity. There is detailed information of attacks such as the processes started, compiles, file adds, deletes, changes, and even key strokes etc in the system logs. This information is critical, as it's our first indication of what an attacker is doing. Obviously the system logs cannot be kept on the honey pot exposed to the hacker. Thereby we transmit them via UDP to a remote machine named "Remote Log Server". Attackers cannot see, nor sniff these packets.

But more advanced attackers will compromise the "Remote Log Server'' in an attempt to cover their tracks. So the second element is capturing every packet and its full payload as it enters or leaves the honey pot. The "Sniffer Server" can do it and writes down all the packets in the binary log files. In this way, even if backers have broken into the "Remote Log Server" and destroyed all the logs in this host there are still intruder's behaviors in those binary log files.

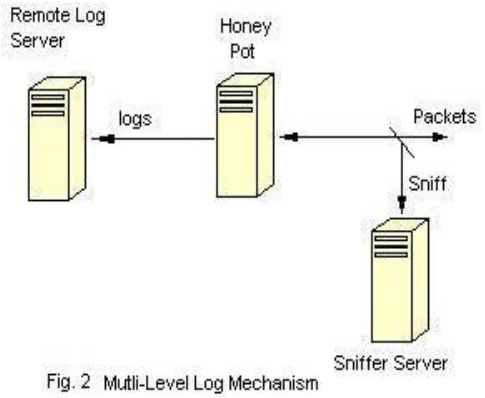

Fig. 2 Mutli-Level Log Mechanism

D. Architecture of HPEIDS

The Architecture of the HPEIDS is shown in Figure 3.This figure shows eight essential components of the architecture: "Remote Log Server", "Sniffer Server", "Honey Pot", "IDS", "WWW Server", Switch, Router and Fire Wall. "IDS" is the host for intrusion detection and "WWW Server''is the secured host in the network. Switch is used for the Data Control and Router for the Route Control. There is another function to set up the Router here. It creates a network environment that more realistically mirrors a production network. So the trap of the honey pot is not easy to be found. In this paper, we work hard at the integration of the honey pot with IDS and Fire Wall. We want to buildup a cooperative system to detect intrusion.

- Honey pot is by no means the only method to collect data; however it has the advantage of reducing false negatives. Even if IDS misses some attacks, we can identify the attack according to MLLM. IDS can protect against these threats the next time.
- Traditional IDS is purely defensive. But in HPEIDS, there is enough information about threats that' exist. New tools and attack patterns can be discovered. Hence, future compromise can be predicted.
- We use the information captured by the honey pot to correlate with the IDS'S logs. IDS can carry on frequency analysis, source analysis and statistical analysis of given theme and so on. New methods and ways of intrusion can be learned too. Further more, IDS maybe know who invade into the system. By these means, the capability of defense will be improved.
- The honey pot system can cooperate with Fire Wall. The system will refuse the visit of the intruder whose IP address is set in the Fire Wall as blacklist by the honey pot.

By combining data from multiple systems, these data can be used for such things as early warning and prediction, statistical analysis, or identification of new tools or trends. The main characteristics that we would like to achieve in the HPEIDS are flexibility and security.

Flexibility: Honey pot creates a network environment that more realistically mirrors a production network.

Security: Intruders can be trapped in the honey pot before an attack is made on real assets. It is obvious that HPEIDS solves the unknown attacks, false positives and false negatives. At the same time, it also increases flexibility and security of IDS.

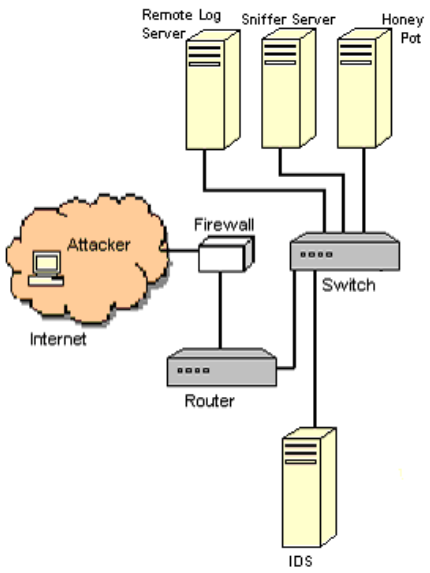

Fig. 3 Architecture of Honey Pot Enabled Intrusion Detection System (HPEIDS)

Conclusion

A novel architecture called Honey Pot Enabled Intrusion Detection System (HPEIDS) has been proposed which solves the problem of flexibility and security. In this paper I have taken only co-operative approach. Our future aim to take distributed and co-operative approach for efficient intrusion detection .User interface is also a big issue for future work. Most of the work that has been done in Intrusion Detection over the last few years focuses on how to perform the detections, but very little has been done in the way of presenting the information

to the user, as well as how to allow the user to specify policies such that the IDS can understand and therefore enforce them.

References

[1] James P. Anderson. Computer security threat monitoring and surveillance. Technical Report Contract 79F26400, James P. Anderson Co., Box 42, Fort Washington, PA, 19034, USA, February 26.

[2] Mark Crosbie, Bryn Dole, Todd Ellis, Ivan. The COAST Project, Dept. of Computer Science, Purdue University, West Lafayette, IN, USA, September 4 1996. Technical Report TR-96-050

[3] D. E. Denning and P. G. Neumann. Requirements and model for IDES|A real-time intrusion detection system. Technical report, Computer Science Laboratory, SRI International, Menlo Park, CA, USA,2003.

[4]Stefan Axelsson, Ulf Lindqvist, Ulf Gustafson, and Erland Jonsson. An approach to UNIX security logging. In Proceedings of the 21st National Information Systems Security Conference, pages 62{75, Crystal City, Ar- lington, VA, USA, October 5{8 1998. NIST, National Institute of Standards and Technology/National Computer Security Center.

[5] Jai Balasubramaniyan, Jose Omar Garcia-Fernandez, David Isaco, E. H.Spaord, and Diego Zamboni. An architecture for intrusion detection us- ing autonomous agents. Technical Report Coast TR 98-05, The COAST Project, Dept. of Comp. Sciences, Purdue Univ.,West Lafayette, IN, 47907{ 1398, USA, 1998.

[6] Mark Crosbie, Bryn Dole, Todd Ellis, Ivan Krsul, and Eugene Spaord. IDIOT|Users Guide. The COAST Project, Dept. of C omputer Science, Purdue University, West Lafayette, IN, USA, September 4 2002. Technical Report TR-96-050.

[7] Herve Debar, Monique Becker, and Didier Siboni. A neural network component for an intrusion detection system. In Proceedings of the 1992 IEEE Computer Sociecty Symposium on Research in Security and Privacy, pages 240{250, Oakland, CA, USA, May 1992. IEEE, IEEE Computer Society Press, Los Alamitos, CA, USA.

[8] Herve Debar, Marc Dacier, and Andreas Wespi. Towards a taxonomy of intrusion detection systems. Computer Networks, 31(8):805{822, April 2002

[10] A Revised Taxonomy for Intrusion-Detection Systems. Herve Debar, Marc Dacier and Andreas Wespi. IBM Research. Zurich Research Laboratory. 8803 Ruschlikon.

[11] B. Mukherjee, L. T. Heberlein, K. N. Levitt, “Network intrusion detection”, IEEE Network, Vol. 8, Issue: 3, Pages: 26-41, 1999