Review on Data Security using Honeypot System

Prof. Ms. Rutuja A. Gulhane
Assistant Professor
Prof. Ram Meghe Institute of Technology & Research,
Badnera, Amravati, Maharashtra, India
Email: gulhanerutuja@gmail.com

Prof. Ms. Meghna A. Deshmukh
Assistant Professor
Prof. Ram Meghe Institute of Technology & Research,
Badnera, Amravati, Maharashtra, India
Email: meghnadeshmukh9@gmail.com

Prof. Ms. Pranita P. Deshmukh
Assistant Professor
Prof. Ram Meghe Institute of Technology & Research,
Badnera, Amravati, Maharashtra, India
Email: pranita.33deshmukh@gmail.com

Prof. Ms. Rani Lande
Assistant Professor
Prof. Ram Meghe College of Engineering & Management,
Badnera, Amravati, Maharashtra, India
Email: lande.rani@gmail.com

Abstract- In the area of computer and internet security a Honeypot is used. It is a resource used to trap attacks, records intrusion information about events of the hacking process, and avoids attacks outbound the compromised computer system. It can also be deployed to attract and divert an attacker from their real targets. The paper describes the classification and types of Honeypots and the possible solution use in a research as well as productive environment. Honeypot is an active defense system for network security. It traps attacks, records intrusion information about tools and activities of the hacking process. Located either in or outside the firewall, the Honeypot is used to learn about the technique of intruder as well as determine vulnerabilities in the real system.

Index Terms- Data Security, Honeypot, Honeynet.

I. Introduction

In computer terminology, a Honeypot is a trap set to detect, deflect, or counteract attempts at unauthorized use of information systems. A Honeypot consists of a computer, data, or a network site which is part of a network, but is actually isolated and monitored, and seems to contain information or a resource of value to attackers [2]. This is similar to the police baiting a criminal and then conducting undercover surveillance.

Honeypots are categorized by their level of interaction [3]. So-called low interaction Honeypots are defined as simulated services, anything from an open port to a fully-simulated network service. The low interaction honeypots use simple script-based languages to describe the honeypots reactions to attacker inputs. Low interaction Honeypots are secure because of the limited capabilities and are easy to set up. The drawbacks are that they are easy to detect for attackers, because the service's reactions are not implemented completely. Its use is limited to the logging of automated attacks and intrusion detection. So-called high interaction honeypots, it is emphasizes that do not make a distinction between medium and high interaction Honeypots are real service.

A. Motivation:

The main objective is to develop a secure communication system which will scan every message and mail transfer between users for malware and spam. Honeypot system is used to check every mail or message for unwanted spam and malware which are stored in the database as spam words and malware signatures [1]. The Honeypot system will check every mail or message and if any spam or malware detected it will alert the administrator about the activity and the message or mail which will store in spam table.

II. Related Work

Honeypot is a non-production system, used for exploiting the attacker and notice the attacking techniques and actions. The objective of Honeypots is not only to notice but to tackle the risk and abate it. There are various definitions of Honeypots are available as few people take it as a system to lure the attackers and inspect their activities where as other take it as a technology for detecting attacks or real systems formed for getting attacked.

L. Spitzner defines the term Honeypot as follows: A Honeypot is a resource whose value is being in attacked or compromised. This means, that a Honeypot is expected to get probed, attacked and potentially exploited. Honeypots do not fix anything. They provide us with additional, valuable Information [5].

In network security, Honeypots are used to detect the attackers and learn from their attacks and then modify and develop the system accordingly for security. The loop holes of the network security can be covered with the help of information provided by Honeypots.

Honeypot can be figured as a computer system connected with a network for inspecting the vulnerabilities of a computer or a complete network. The loopholes can be examined collectively or individually of any system, as it is an exclusive tool to study about the attackers and their strategies on the network [6]. Honeypots are normally virtual machines which acts like a real system.

A. Honeypot based on Categories

- Research Honeypots:

These are the Honeypots which are manipulated by researchers and are used to acquire information and knowledge of the hacker society. The knowledge gained by the researchers are used for the early warnings, judgment of attacks, enhance the intrusion detection systems and designing better tools for security. These are run by a volunteer, non-profit research organization or an educational institution to gather information about the motives and tactics of the Blackhat community targeting different networks. These Honeypots do not add direct value to a specific organization. Instead they are used to research the threats organizations face, and to learn how to better protect against those threats. This information is then used to protect against those threats. Research Honeypots are complex to deploy and maintain, capture extensive information, and are used primarily by research, military, or government organizations.

- Production Honeypots:

These are the Honeypots derived by the industries as a part of network security backbone. These Honeypots work as early warning systems. The objectives of these Honeypots are to abate the threats in industries. It provides the information to the administrator about the attacks before the actual attack [7]. This is easy to use, capture only limited information, and are used primarily by companies or corporations; Production Honeypots are placed inside the production network with other production servers by organization to improve their overall state of security. Normally, production Honeypots are low interaction Honeypots, which are easier to deploy. They give less information about the attacks or attackers than research Honeypots do. The purpose of a production Honeypot is to help mitigate risk in an organization. The Honeypot adds value to the security measures of an organization.

Honeypots that provide only some fake services, these acts as an emulator of the operating system and services. These Honeypots are simple to design but also simply detectable. Attacker can just use a simple command to identify it that a low involvement Honeypot does not support. An example of this type of Honeypot is Honeyd. High level interaction Honeypots provides the real like operating systems and some real services with some real uncertainties. These allow the capturing of information of attacker and record their activities and actions. These are the real machine with one system, with one network interface on network. An example of this type of Honeypot is Honeynet [8].

III. Classification of Honeypot

Honeypots can be classified into two categories: low-interaction honeypots are used for production purposes and high interaction honeypots are used for research purposes [4].

- Low-interaction Honeypots

A typical low-interaction Honeypot is also known as a Gen1 Honeypot. This system is very effective against automated attacks or beginner level attacks. Honeyd is one such Gen1 Honeypot which emulates services and their responses for typical network functions from a single machine, while at the same time making the intruder believe that there are different operating systems. It allows the simulation of virtual network topologies using a routing mechanism that constitutes various network parameters such as delay, latency and ICMP error messages. The architecture consists of a routing mechanism, a personality engine, a packet dispatcher and the service simulators. The most important of these is the personality engine which gives services a different avatar for every operating system that they emulate.

Low-interaction honeypots simulate services frequently requested by attackers. Multiple virtual machines can easily be hosted on one physical system, since they require relatively few resources. The virtual systems have a short response time, and for reducing the complexity of the virtual system's security, less code is required. Example: Honeyd.

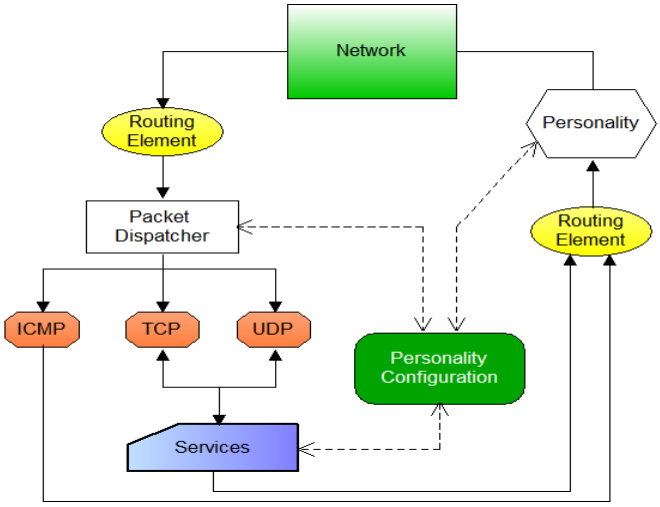

Fig. 1: Low Interaction Honeypot Architecture

Low-interaction Honeypots work by emulating certain services and operating systems. The attacker's activities are limited to the level of emulation provided by the Honeypot. Low-interaction Honeypots have advantages that they are simple and easy to deploy and maintain. The limited emulation available, allowed on low interaction Honeypots reduces the potential risks brought about using them in the field. However, only limited information can be obtained with low-interaction Honeypots, and it is possible that experienced attackers will easily recognize a Honeypot when they come across one.

- High-interaction Honeypots

A typical high-interaction Honeypot has following elements: resource of interest, data control, data capture and external logs. It also known as Gen2 Honeypots and started development in 2002. It provides better data capture and control mechanisms. It increases complexity level to deploy and maintain in comparison to Low-Interaction Honeypots.

High-interaction honeypot makes a copy of the activities of the production systems that host a variety of services and, therefore, an attacker may be allowed to waste his time in lot of services. Multiple Honeypots can be hosted on a single physical machine by employing virtual machines. Therefore, it can be restored more quickly, if the Honeypot is compromised. In general, high-interaction Honeypots provide more security because of difficult to detect, but they are expensive to maintain. One physical computer must be maintained for each Honeypot, if virtual machines are not available which can be exorbitantly expensive.

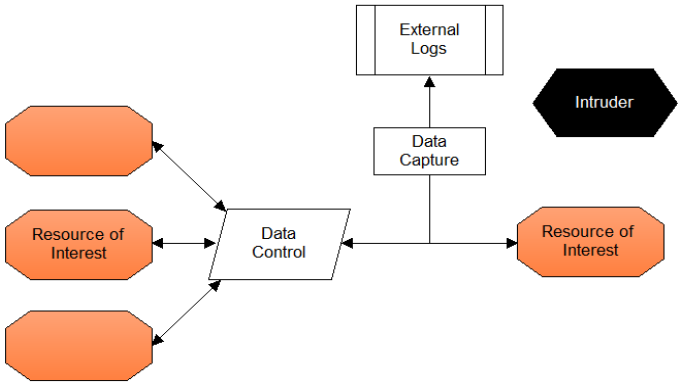

Fig. 2: High Interaction Honeypot Architecture

High-interaction Honeypots involve real operating systems and applications, therefore are more complex. For example, if the aim is to collect information about attacks on a particular FTP server or service, a real FTP server will be built. By giving attackers real systems to interact with, no restrictions are imposed on attack behavior and this allows administrators to capture extensive details about the full extent of an attacker's methods. However, it is not impossible that attackers might take over a high-interaction honeypot system and use it as a stepping-stone to attack other systems within the organization. Therefore, protection measures need to be implemented accordingly. The network connection to the honeypot may need to be disconnected in the worst case to prevent attackers from further penetrating the network and machines beyond the Honeypot system itself. Example: Honeynet.

A. Types of Honeypot

There are five Honeypots that are discussed in the following section.

1] ManTrap

2] Back officer friendly

3] Specter

4] Honeyd

5] Honeynet

1] ManTrap:

ManTrap is a high-interaction commercial honeypot created, maintained, and sold by Recourse Technologies. An attacker can interact with a highly controlled operating environment created by ManTrap. It creates a fully functional operating system containing virtual cages that are logically controlled environments from which the attacker is unable to exit and attack the host system. However, instead of creating an empty cage and filling it with certain functionality, ManTrap creates cages that are mirror copies of the master operating system. Each cage is a fully functional operating system which has the same capabilities as a production installation [8]. The approach creates a very powerful and flexible solution. Each cage has own virtual world with few limitations. An administrator can customize each cage as he would a physically separate system. He can create users, install applications, run processes, and even compile his own binaries. When an intruder attacks and gains access to a cage, to the attacker it looks as if the cage is a truly separate physical system. He is not aware that he is in a caged environment where every action and keystroke is recorded.

2] BackOfficer Friendly (BOF):

BackOfficer Friendly, or BOF is a simple, free Honeypot solution developed by Marcus Ranum. It is extremely simple to install, easy to configure, and low maintenance. However, this simplicity comes at a cost. Its capabilities are severely limited. It has a small set of services that simply listen on ports, with notably limited emulation capabilities. It works by creating port listeners, or open sockets, that bind to a port and detect any connections made to these ports. When a connection is made to the port, the port listeners establish a full TCP connection (if the service is TCP), log the attempt, generate an alert, and then close the connection, depending on how the service is configured. Everything BOF does happen in user space. It does not build or customize any packets when responding to connections. Because of this simple model, BOF can run on any Windows platform, including Windows 95 and Windows 98 [9].

3] Specter:

Specter is a commercially supported Honeypot developed and sold by the folks at NetSec. Like BOF, Specter is a low-interaction Honeypot. However, Specter has far greater functionality and capabilities than BOF. Not only can Specter emulate more services, it can emulate different operating systems and vulnerabilities. It also has extensive alerting and logging capabilities. Because Specter only emulates services with limited interaction, it is easy to deploy, simple to maintain, and is low risk. However, compared to medium- and high-interaction Honeypots, it is limited in the amount of information it can gather. Specter is primarily a

production Honeypot. Specter shares the same limitations as BOF. Specifically, it cannot listen on or monitor a port that is already owned by another application. If any service listening on the FTP port (port 21), then Specter is unable to monitor on that port. Specter can only monitor ports that are not owned by any other applications. It also has the capability of emulating different operating systems. This is done by changing the behavior of the services to mimic the selected operating system [10].

4] Honeyd:

Honeyd is developed and maintained by Niels Provos of the University of Michigan and was first released in April 2002. It is designed as a low-interaction solution; there is no operating system intended for an attacker to gain access to, only emulated services. Honeyd is designed primarily as a production Honeypot, used to detect attacks or unauthorized activity [11]. Honeyd works on the principle that when it receives a probe or a connection for a system that does not exist, it assumes that the connection attempt is hostile, most likely a probe, scan, or attack. When Honeyd receives such traffic, it assumes the IP address of the intended destination (making it the victim). It then starts an emulated service for the port that the connection is attempting. Once the emulated service is started, it interacts with the attacker and captures all of his activity. When the attacker is done, the emulated service exits and is no longer running. Honeyd then continues to wait for any more traffic or connection attempts to systems that do not exist. As Honeyd receives more attacks, it repeats the process of assuming the IP address of the intended victim, starting the respective emulated service under attack, interacting with the attacker, and capturing the attack. It emulates multiple IP addresses and interacts with different attackers all at the same time.

5] Honeynets:

Honeynet represents the extreme of high-interaction Honeypots. Not only does it provide the attacker with a complete operating system to attack and interact with, it may also provide multiple Honeypots. Honeynets are nothing more than a variety of systems deployed within a highly controlled network. Since their value is in being probed, attacked, or compromised, these systems become honeypots. The controlled network captures all the activity that happens within the honeynet and decreases the risk by having the attacker's activity. Honeynets are a simple mechanism that works on the same principle as a Honeypot. Anything sent to the Honeynet is suspect, potentially a probe, scan, or even an attack. Anything sent from a honeynet implies that it is compromised of an attacker or tool is launching activity. However, honeynet takes the concept of Honeypots one step further: a honeynet is a physical network of multiple systems, instead of a single system. Honeynet is an architecture which builds a highly controlled network, within which you can place any system or application [12].

IV. Solution of Honeypot

The goal of the approach is the design and realization of a generic high interaction honeypot framework which allow to identify application layer based attacks (e.g. buffer overflows, format string attacks, etc.) automatically. Figure 3 depicts an example scenario that includes two attackers, a firewall / intrusion prevention system (IPS) and Honeypot framework. The purpose of the IPS / firewall is to filter the incoming traffic for known attacks. The Honeypot framework consists of a proxy and a Honeypot host. The proxy host is responsible for the session-individual logging of the network traffic that was sent to the Honeypot.

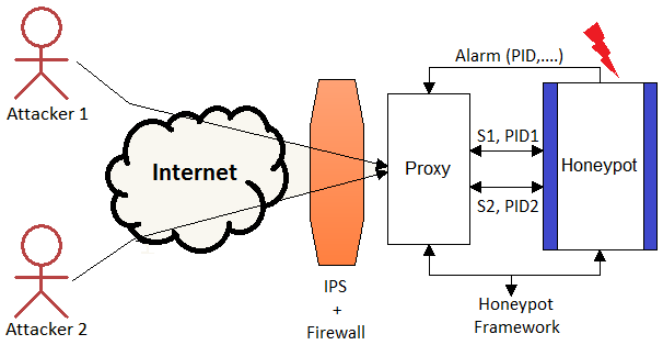

Fig. 3: Architecture of Honeypot

Furthermore, in case of a detected attack the proxy provides a mechanism to replay a specific previously logged session. An advantage of the bipartite approach is that in case of a complete system takeover of the honeypot host by an attacker the log files remain save, for this purpose the honeypot and the log files are kept on separate hosts.

The Honeypot host consists of a Honeypot service, namely a real service, and a host intrusion detection system (HIDS). The running service is bait that attracts worms respectively hackers whereas HIDS supervises honeypot service. On the basis of system-call signatures, realized detection mechanism allows detection of attacks. The reasoning behind this approach is that the main part of current attacks exploits a vulnerability that is specific for software. In case of a successful attack, hacker will sooner or later exploit its newly gained authorizations which results in an observable system change. An example for this would be an attacker that tries to open a new network socket in order to download further hacking utilities. The interface between Honeypot service and HIDS is generic such that is possible to exchange or add a Honeypot service in an easy and flexible manner.

V. Conclusion

A Honeypot can be anything from Windows to UNIX. Compared to other intrusion detection systems, Honeypots do not generate incorrect alerts or log files like other intrusion detection systems because no productive components are running on system. There is no need to manage data base of intrusions signature or definition, as honeypot system logs every byte that flows through network. This data helps researcher to draw picture of an attacker. Honeypots have their advantages and disadvantages. They are clearly useful tool for trapping attackers, capturing information and generating alerts when someone is interacting with them. The activities of attackers provide valuable information for analyzing their attacking techniques and methods. Because Honeypots only capture and archive data and requests coming in to them, they do not add burden to existing network bandwidth.

REFERENCES

[1] Spitzner L., "Honeypot: Definitions and Values", May, 2002. http://www.spitzner.net

[2] Bao, J., Gao, M. "Research on network security of defense based on Honeypot", International Conference on Computer Applications and System Modelling, 2010.

[3] Phrack magazine, http://www.phrack.org

[4] Levine, J., Grizzard, J. "Using honeynets to protect large enterprise networks," Security & Privacy Magazine, IEEE, vol. 2, pp. 73-75, 2004.

[5] Spitzner, L.: Tracking Hackers. Addison Wesley, September 2002.

[6] Zanoramy, W., Zakaria, A., et. al, "Deploying Virtual Honeypots on Virtual Machine Monitor".

[7] Qassrawi, M., Hongli, Z. "Deception methodology in virtual Honeypots", Second International Conference on Network Security, Wireless Communication and Trusted Computing, 2010.

[8] Kuwatly, I., Sraj, M. A, "Dynamic Honeypot Design for IntrusionDetection". http://webfealb.fea.aub.edu.lb/proceedings/2004/SRC-ECE-04.pdf.

[9] Levin, J., Labella, R. Henry, "The Use of Honeynets to Detect Exploited Systems Across Large Enterprise Networks", IEEE Proceedings, June 2003.

[10] Lanoy, A., and Romney, G.W., "A Virtual Honey Net as a Teaching Resource", Information Technology Based Higher Education and Training, 7th International Conference, pp. 666-669, 2006.

[11] Lance Spitzner, "Tracking hackers". http://www.tracking-hackers.com

[12] Lance Spitzner, "Honeytokens". http://www.securityfocus.com/infocus/1713

[13] Niels Provos, "Honeyd". http://www.honeyd.org