Intelligent Firewall System using ANN and Decoy-data
Aditya Samant1 Sudip Kumar Shrestha2 Ashish Agarwal3 Rahul Samant4
1, 2, 3, 4 Department of Information Technology
1, 2, 3, 4 NBNSSOE, Sinhgad Technical Education Society

Abstract— A honey-pot is a non-production system, designed to interact with cyber-attackers to retrieve intelligence on attack techniques and behaviors. With the increase in the dependency on the Internet both at the personal and commercial level, intrusion detection becomes a challenging process. The challenge here is not only to be able to actively monitor large numbers of systems, but also to be able to react quickly to different situations. Implementations however falling into 3 major categories: high-interaction, medium-interaction, low-interaction. Honey-d, created by Niels Provos, is an open source project implemented as a small software daemon that creates virtual hosts on a network (low-interaction). High-interaction honey-pot systems are typically hardware replicas of existing operational components that include the appropriate software. These systems do not mimic services, but are deployed with working instances. Our type of system provides a high-interacting solution that is less prone to discovery of its deceptive purpose by network intruders. Therefore these issues can be addressed by our efficiently as the systems are at higher risk of facing an attack from an intruder. The system we speak of will perform interactions in Real time, data and records will be saved on a database primarily will be developed an NIDES(Next Generation Intrusion detection system). The system will provide the modules of port/socket scanning, packet scanning, SQL injection modules in the working manner. As such a system having or running these 3 modules together has not been taught of in the practical manner.

Key words: Honey-pot, FF (Feed Forward), BPNN (Back Propagation neural network), ANN (Artificial neural network)

I. INTRODUCTION

In computer terminology, a honey-pot is a trap set to detect, avoid, or in some manner, subdue attempts at unauthorized use of information system. Generally, a honey-pot consists of a computer, data, or a network site that appears to be part of a network but is actually isolated and monitored, and which seems to contain information or a resource of value to attackers. This is similar to the police trapping a criminal and then conducting undercover surveillance.

We can explain this or more properly illustrate this in the form of a small diagram.

Essentially honey-pots can be classified according to their level of involvement. Based on their deployment they are classified as:-

1) Production honey-pots
2) Research honey-pots

Fig. 1: Honey-pot System diagram.

Production honey-pots are easy to use, capture only limited information, and are used primarily by companies or corporations; Production honey-pots are placed inside the production network with other production servers by an organization to improve their overall state of security. Normally, production honey-pots are low-interaction honey-pots, which are easier to deploy. They give less idea about the attacks or attackers than research honey-pots do.

Research honey-pots are run to gather information about the motives and tactics of the Black-hat community targeting different networks. These honey-pots do not add direct value to a specific organization instead, they are used to research the threats that organizations face and to learn how to better protect against those threats. Research honey-pots are difficult to deploy and maintain, capture extensive information, and are used primarily by research, military, or government organizations.

Based on their design criteria they can be further classified as:-

1) Pure honey-pots
2) High Interaction honey-pots
3) Low-Interaction honey-pots

Pure honey-pots are full-fledged production systems. The activities of the attacker are monitored by using a casual tap that has been installed on the honey-pot’s link to the network. Even though a pure honey-pot is useful,
stealth of the defense mechanisms can be ensured by a more controlled mechanism.

High-interaction honey-pots imitate the activities of the production systems that host a variety of services and, therefore, an attacker may be allowed a lot of services to waste his time. By employing virtual machines, multiple honey-pots can be hosted on a single physical machine. Therefore, even if the honey-pot is compromised, it can be restored more quickly. In general, high-interaction honey-pots provide more security by being difficult to detect, but they are highly expensive to maintain. Example: Honey net.

Low-interaction honey-pots simulate only the services frequently requested by attackers. Since they consume relatively few resources, multiple virtual machines can easily be hosted on one physical system, the virtual systems have a short response time, and less code is required, reducing the complexity of the virtual system’s security. Example: Honey -d

II. LITERATURE SURVEY

A honey-pot is a program, machine, or system put on a network as bait for attackers. The idea is to fool the attacker by making the honey-pot seem like a legitimate system. Honey-pots are typically virtual machines that imitate real machines by acting on running services and open ports, services which one might find on a typical machine on a network. Currently the honey-pot systems which are in use as well as those which have been used are given:

1) Lance Spitznzer introduced the concept of a dynamic honey-pot (DHP) in 2004. The idea was based on automatically configuring a honey-pot by gathering information obtained from network traffic.

2) Deception Toolkit: DTK was the first Open Source honey-pot released in 1997. It is a collection of Perl scripts and C source code that copies a variety of listening services. Its primary purpose is to fool human attackers.

3) La-Brea: This is designed to slow down or stop attacks by acting as a sticky honey-pot to detect and trap worms and other malicious codes. It can run on Windows or UNIX.

4) Honey-wall CDROM: The Honey-wall CDROM is a bootable CD with a collection of open source software. It can capture, manipulate and monitor all incoming and outgoing honey-net activity.

5) Honey-d: This is a powerful, low-interaction Open Source honey-pot, and can be run on both UNIX and Windows platforms. It can monitor unused IPs, simulate operating systems at the TCP/IP stack level, simulate thousands of virtual hosts at the same time, and monitor all UDP and TCP based ports.

6) Honey-trap: This is a low-interactive honey-pot developed to observe attacks against network services. It helps administrators to collect information regarding known or unknown network-based attacks.

7) Honey C: This is an example of a client honey-pot that initiates connections to a server, aiming to find malicious servers on a network. It aims to identify malicious web servers by using known clients that are able to solicit the type of response from a server that is necessary for analysis of malicious content.

8) Honey-Mole: This is a tool for the deployment of honey-pot farms, or distributed honey-pots, and transport network traffic to a centralized system for detection and analysis.

In the corporate environment, the commercial products are available:

1) Symantec Decoy Server: This is a "honey-pot" intrusion detection system (IDS) that detects, contains and monitors unauthorized access and system misuse in real time.

2) Specter: This is a smart honey-pot-based intrusion detection system. It can emulate 14 different operating systems, monitor up to 14 different network services and traps, and has a variety of configuration and notification features.

III. PROCEDURE AND SYSTEM DETAILS

This section of the paper describes the software tool evaluation and implementation logic of the system we have developed. The diagram shows the basic data flow of the system in development, namely the components being used and the manner in which they are used.

The Administrator is the main component of the system along with the functions it serves, it is uses snort rule for the basic adding, viewing, deleting operations to be performed on the system.

Our system employs the use of ANN (Artificial Neural Networks) for the training module of the system. ANN basically uses 2 algorithms for training the system, namely:

- FF (FEED FORWARD)
- BPNN (BACK PROPAGATION NEURAL NETWORK)

The back propagation algorithm (Rumelhart and McClelland, 1986) is used layered feed-forward ANNs. The back propagation...

![Fig. 2: Data flow diagram](image)

All rights reserved by www.tjsrd.com 135
algorithm uses supervised learning, which means that we provide the algorithm with examples of the inputs and outputs we want the network to compute, and then the error (difference between actual and expected results) is calculated. The idea of the back propagation algorithm is to reduce this error, until the ANN learns the training data. The training begins with random weights, and the goal is to adjust them so that the error will be minimal. The Feed Forward method is used as a part of the BPNN also the weighted index corrections are made once the calculations from BNN are established.

The algorithm works in the following manner:

1) First apply the inputs to the network and work out the output – remember this initial output could be anything, as the initial weights were random numbers.

2) Next work out the error for neuron B. The error is what you want – What you actually get, in other words:

   \[ \text{Error}_B = \text{Output}_B \times (1 - \text{Output}_B) \times (\text{Target}_B - \text{Output}_B) \]

   The “Output (1 - Output)” term is necessary in the equation because of the Sigmoid Function – if we were only using a threshold neuron it would just be (Target – Output).

3) Change the weight. Let \( W_{+AB} \) be the new (trained) weight and \( W_{AB} \) be the initial weight.

   \[ W_{+AB} = W_{AB} + \text{Error}_B \times \text{Output}_A \]

   Notice that it is the output of the connecting neuron (neuron A) we use (not B). We update all the weights in the output layer in this way.

4) Calculate the Errors for the hidden layer neurons. Unlike the output layer, we can’t calculate these directly (because we don’t have a Target), so we Back propagate them from the output layer (hence the name of the algorithm). This is done by taking the Errors from the output neurons and running them back through the weights to get the hidden layer errors. For example if neuron A is connected as shown to B and C then we take the errors from B and C to generate an error for A.

   \[ \text{Error}_A = \text{Output}_A \times (1 - \text{Output}_A) \times (\text{Error}_B \times W_{AB} + \text{Error}_C \times W_{AC}) \]

   Again, the factor “Output (1 - Output)” is present because of the sigmoid squashing function.

5) Having obtained the Error for the hidden layer neurons now proceed as in stage 3 to change the hidden layer weights. By repeating this method we can train anetwork of any number of layers. The Exemplary diagram of the system shows the IDS being deployed with ANN’s being assigned to do it. This system diagram also shows the modules developed as a part of the system.

IV. SOFTWARE DETAILS AND SCOPE

A. Project Scope:
- Honey-pots have tremendous usage in industry where spying or phishing is utilized in order to minimize competition or plagiarize ideas.
- Honey-pots therefore are used as a means of trapping for such intruders.

B. User Classes and Characteristics:
- Main users will be the administrator, intruder and the users (end-users).
- Basically it will act as trapping tool by showing intruder a false data known as decoy data in order for him to be trapped and his ref. id blocked from further usage.
- It will be an IDS system which will not only detect intruders but also block them by access.
C. Assumptions and Dependencies:

- In the very basic assumption we shall be providing a diagram which is self-explanatory.
- SMTP (Simple mail transfer protocol) used for sending and POP3 (Post Office Protocol) used for receiving.

V. CONCLUSION AND FUTURE WORK:

In this work, we explored the concept of honey-pots in depth and saw how it might be useful to the field of network security. The concept of honey-pots is an important addition to the security field. Honey-pots offer an attacking approach to intrusion detection and prevention. Most importantly, they serve as a learning tool for system administrators and also involved studying issues concerning intrusion detection systems the challenges that these systems faced. The Internet has become indispensable both at the organizational and personal level and so it will be the case with security systems. The use of honey-pots and related technologies is on the rise. As awareness and interest in honey-pots increases so will its use in an organization as a security tool.

There is scope for development of honey-pot tools which facilitate the different aspects of honey-pots like logging, tracing back to the source etc. System modules for sophisticated keystroke logging, better filtering tools and utilities to catch encrypted traffic are a few things that could be worked on. One can even consider an external honey-pot distribution with a modified kernel to make it easy for system administrators to deploy honey-pots.

REFERENCES


