Journal of Electrical and Computer Engineering Innovations
(JECEI)
Journal homepage: http://www.jecei.sru.ac.ir

Research paper

An Advanced Hybrid Honeypot for Providing Effective Resistance in Automatic Network Generation

M. Amiri, A. Barati*

Department of Computer Engineering, Dezful Branch, Islamic Azad University, Dezful, Iran.

Article Info

Article History:
Received 06 August 2018
Reviewed 12 October 2018
Revised 19 January 2019
Accepted 19 April 2019

Keywords:
Honeypot
Social engineering
Vulnerable scanner
Dynamic honeypot

*Corresponding Author's Email Address:
abarati@iaud.ac.ir

Abstract

Background and Objectives: Increasing usage of Internet and computer networks by individuals and organizations and also attackers' usage of new methods and tools in an attempt to endanger network security, have led to the emergence of a wide range of threats to networks.
Methods: A honeypot is one of the basic techniques employed for network security improvement. It is basically designed to be attacked so as to get the attackers' information and trap them. By using a vulnerable scanner in this paper, we obtained the required network vulnerabilities and normalized them via the proposed method. Then, a dynamic hybrid honeypot has proposed by high and low interaction honeypots. Also, in the proposed method, by footprinting and scanning of an integrated network, a detailed picture of the production network and a honeypot configuration file are generated.
Results: As a result, more devices could be detected via automated production by the proposed method.
Conclusion: This method could accelerate honeypot production and reduce the users' mistakes during their manual production. Monitoring network traffic, collecting the information of network machines, determining network operating systems, and storing data in a database are the specific features of this system that could be performed by using the selected network scanning tools and modules.

Introduction

Today's world has been radically influenced by the Internet. Nevertheless, security mechanisms that use the Internet are required to be applied to almost all personal communications, business interactions, and academic researches. At increased use of computers, millions of dollars have been dedicated to the identification of attackers attempting to perform targeted and automated security attacks and penetrate into computer systems all over the world. Some of mechanisms such as authentication [1], [2], firewall, and honeypot can be used to security in the network. In 1990, honeypot systems were introduced to provide researchers and organizations with tools to identify vulnerabilities in their systems. These tools would allow administrators to register attackers through a deceptive scenario called honeypot. According to Lance Spitzner [3], the author of "honeypot", honeypot is a security resource that its worth relies on exploring, attacking and compromising. Honeypots can be used in a variety of scenarios, such as intrusion detection, defense mechanism or response. In addition, honeypot can be used to invade an attacker's resources or neglect valuable goals and spend his time on honeypot instead of the attacked production system [4].

Honeypots can throw off attackers' attention from

the real network so that the main sources of information are not compromised. In addition, they can monitor new viruses and worms in the future. Their other functions are as follows: attacker profile creation, identification of new vulnerabilities and undetected hazards on various operating systems, programs, and environments at present and higher-priority attack methods in the same way law-enforcement agencies use criminal profiles to identify a criminal [4]. Based on their methods of detecting attackers on a network, honeypots can be either active or passive. Suspicious web servers or malicious malware executed by attackers can be actively visited via client-based honeypots [5].

Nevertheless, the time-consuming process of honeypot configuration is a quite big challenge to be dealt with most security administrators. Since being accessible to a wide number of audiences, such systems will need to be constantly modified and configured by the relevant administrators if they are to truly represent the present network services and environments or emulate the desired vulnerabilities. To ensure an uncompromised performance for the host and reduce the probability of honeypot detection by applying the latest patches to both honeypots and hosts, system administrators have to properly monitor and maintain the systems besides trying to hide the presence of honeypots within the network.

The contributions of this paper can be summarized as follows:

- Implements a dynamic honeypot and provides the framework of an advanced hybrid honeypot.
- Automatically generates a network to collect more information from attackers at the right time.
- Scans the network as soon as a module is added to the network by honeypot.
- Stores the data from scan to give a general scheme of the network.
- Adapts to the changing environment.
- Constantly supervises individual processes and reuses them if necessary.
- For each tool, a module is simultaneously designed with the activity of that tool to process the collected data and store the relevant information in the database.
- Employs a management system including net scan tools, footprinting, and scanning to reduce the responsibility of users to monitor scan, modules, and generate configuration files as soon as a major change in system is detected.

The rest of the paper is organized as follows: Section 2 represents an overview of the related work. In Section 3, the proposed honeypot method is presented. Then we provide the simulation of the proposed method and the analysis of its performance in Section 4. Finally, Section 5 concludes this paper.

Related Works

NESSUS as a famous and powerful software for detecting holes and vulnerabilities was developed by Renaud Deraison in 1998. This computer program was designed to access the weaknesses of applications, networks, and computer systems. Today, there are a variety of vulnerability scanners for specific purposes.

Honeypot scanner program receives primary configuration information, stored in the database, and propagates it to all corresponding modules after they are equipped. Footprinting and scanning modules are used as threads. The threads are monitored through process identification (PID) to confirm the operation. In case of modules failure, they can be reused through constant monitoring. Also, honeypot scanner program examines the results from all scanned modules and on finding the setpoint or base, determined by users, it generates the configuration files of the honeypot. These scanners have huge databases of malicious codes in their files to be used for a very quick detection on the network in a way that a malicious code related to a security vulnerability factor can work successfully and show the results if the devices within a network have a security hole.

NESSUS is a tool for detecting vulnerabilities during the network scanning by finding the running devices and testing them against various vulnerabilities. Another feature of this tool is reporting the possibility of the network damages and fixing them as quickly as possible. Scanning vulnerabilities takes place in the entire network layers, including operating systems, databases, files, and applications. By using this scanner in this paper, we examined system vulnerabilities via social engineering to obtain the real points on the network. The main reason for applying this method was to persuade the attackers to interact with honeypot as the real agent of the network by fixing any simple security holes in the system and hence removing their doubts. The non-simplicity of the network vulnerability was considered to avoid the attackers' suspicions of being trapped and subsequent detections of the honeypot. Honeypots can select one or more machines or subnet targets for scanning based on IP addresses. They can also support various methods in their scanning portals. Some other features of these scanners include detection of services on non-standard ports, identification of the same services that are active on separate ports and scanning all the services, for instance, by separately repeating the scans if two FTP servers are active on two separate ports, and examination of a specific vulnerability executed if only the service and the related operating system are located in the target machine to prevent the system resources from being wasted (for example, not running the tests

related to Apache server vulnerabilities on a machine when the service is not active on it).

In [6] a solution for detecting adversaries attacking the communication channel of a BAN called a wearable honeypot system is proposed. The proposed method by communicating fake user information between the base station and decoy nodes in the BAN is worked. The decoy nodes and the base station are continuously having pre-decided fake communication between them.

This communication simulates the exchange of sensed data from a sensor device to the base station. Any modification of this fake data en-route either in the form of data tampering or delay in arrival at the base station is considered to be an indication of the presence of adversaries and which an alarm is generated.

In [7], SIPHON architecture for Internet of thing (IoT) devices is proposed. This method leverages IoT devices that are physically at one location and are connected to the Internet through so-called wormholes distributed around the world.

SIPHON allows portraying physical IoT devices in a single lab as being geographically distributed by establishing tunnels between the public IP addresses and the physical devices. It also allows for collecting traffic for further processing and analysis.

In [8], a novel HIHP framework is proposed. The framework is able to monitor host and network activities. In addition, it is possible to assign activities to sessions. This method is a honeypot system capable of a high level of interaction, different embedded and common architectures as well as a monitoring mechanism for all relevant interaction.

In [9], the first honeypot that is specifically designed for the protection of unmanned aerial vehicles (honeydrone) is proposed. HoneyDrone emulates a number of UAV-specific and UAV-tailored protocols, making it possible to lure adversaries into attacking it. HoneyDrone provides a medium-interaction interface for many UAV-specific and UAV-tailored protocols.

This allows for the emulation, recording and analysis of malicious activity in UAVs.

In [10], a centralized honeypot-based approach with a software-defined switching is proposed. This method provides security and to detect unauthorized and malicious access to the VLAN. The architecture of the developed system consists of four main modules; these include: honeypot system module, intrusion detection and prevention module, anomaly and misuse detection module, and software switch.

The growing honeypot field is divided into high and low interaction domains. The criteria for distinguishing the high and low interactions include the amount of obtainable information from a honeypot and the risks and costs of deploying and operating the resources [4].

The different honeypot types are summarized in Table 1.

Table 1: Honeypot Types [11]

Type	Category
Low interaction	Virtualization of the transmission layer
Medium interaction	Virtualization of the application layer
High interaction	Real vulnerable systems

A. Low-Interaction Honeypot

There is no operating system for attackers to low interaction with a honeypot, which works by simulating the systems and services. In this mode, the attackers' activities are merely the same as those allowed by the simulated services. This would significantly reduce the risk of hijacking this kind of honeypot. A particular LIH associated with the relevant researches is honeyd as an open-source framework allowing thousands of IP addresses to be simulated by network services. Honeyd, an open source solution, is free and allows users to access its source code. This honeypot, designed for UNIX systems, can be used for windows systems as well. Honeyd is a honeypot with limited interaction, installed on a computer to allow the simulation of a lot of operating different systems and services. Editing setting file allows determining the IP addresses that should be controlled by honeyd as well as the operating systems and services that should be simulated. For instance, it is possible to simulate the kernel of Linux 2, 4, and 10 through an FTP server, listening to port 21. If attackers visit that honeypot, they think that they are interacting with a Linux operating system; when they link to that FTP service, they think that the service is a true FTP service. A simplified view of the honeyd architecture is shown in Figure 1.

This honeypot simulates the operating systems both on the IP stack level and through the changing behaviors of services. When honeypot identifies an attempt to connect to one of the unused IP addresses, it disconnects the call, while dynamically sacrificing himself and interacting with the attacker. This feature greatly increases the chance of an interaction with attackers [12]. To obtain attackers' information, Levine [13], Levine and et al. [14], Azadegan et al. [15] utilized a honeypot of high interaction.

B. Medium-Interaction Honeypot

Low-interaction honeypots provide less-advanced performances than medium-interaction honeypots, which have more security holes for the hackers to access the systems. Thus, further information from the hackers and more complicated attacks can be achieved in this

mode though no operating systems exist in this case either. Some medium-interaction honeypots include honeytrap, mwcollect, and nepenthes.

Table 2: A Comparison of Honeypot Types [12]

Honeypot type	Low interaction	High interaction
Inclusion degree	Low	High
Real operating system	No	Yes
Danger	Low	High
Data collection	Low	High
Being at danger	No	Yes
The degree of knowledge for running	Low	High
The degree of knowledge for development	Low	Medium-high
Hold time	Low	Very high

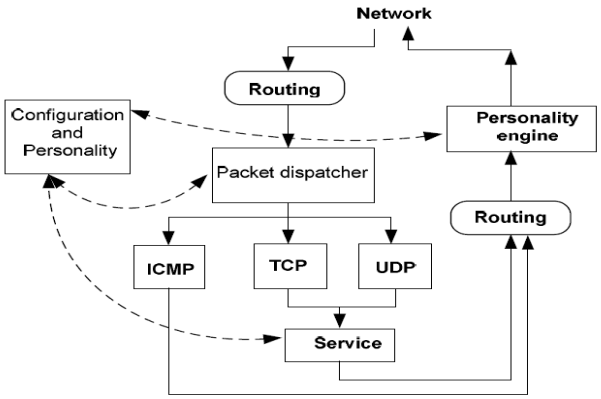

Fig. 1: A simplified view of the honeyd architecture.

C. High-Interaction Honeypot

In a high-interaction honeypot, the attacker interacts with real operating systems, services, and applications. It can be used for observing attackers' behaviors, tools, and motivations. Tools like Sebek can help honeypots of high interactions to measure the logs or system calls. Using virtualization software like Xen [16], Qemu [17], and VMware [18], a honeypot of high performance can be installed inside a virtual machine [19]. According to the different criteria, we can compare a high-interaction honeypot with a low-interaction honeypot. This comparison was carried out by Pouget and et al. [12], the results of which are presented in Table 2. Honeypots can provide security professionals with many benefits if they learn sufficient knowledge and experience about the attackers' powers through some risk margins. By having enough knowledge about the risks of setting up a honeypot, it will be possible to reduce the associated risks and disadvantages [20]. The following benefits are unique and special for this advanced technology [21]:

Requiring minimal resources for preventing harmful activities: Systems with minimum specifications are enough to run a honeypot.

Discovering new tools and tactics: honeypots consider everything they interact with.

Working with encryption or IPv6: honeypots can act in an encrypted environment/system or IPv6.

Being simple: honeypots operate in a very simple and flexible way thus; no sophisticated algorithms are needed for their correction functions.

Like any other security solutions, trapping technology has certain disadvantages as follows:

Takeover risk: if an attacker takes control of a honeypot, he/she can use it for attacking the internal and external systems.

Limited monitoring: only part of the network traffic directly coming to a honeypot can be monitored by it.

D. Advanced Hybrid Honeypots

Honeypots with high and low interactions can be applied and developed simultaneously and hybrid in a coherent unit. The combination of honeypots with high and low interactions can provide the potential for deploying thousands of low-interaction honeypots onto a host, while a limited number of high-interaction honeypots are required for gathering data. Identification and obtainment of information from Internet threats can be accomplished via the different interaction levels involved in this approach.

E. Dynamic Honeypots

Development of the concept of a "dynamic honeypot" has been originally attributed to Spitzner [3]. He mentioned that the biggest problems with most security technologies, including honeypots, are their challenges and time-consumption for generating configuration files. He then presented a list of demands for idealizing honeypots, e.g., determination of computer and service types to be performed in a honeypot through the active or passive reviews of the network. A honeynet is a set of honeypots, designed to be accepted as network servers and services. For instance, a honeypot may seem a DC, an Internet web server, mail server, or other types of server. Honeynets are consisted of strong honeypots, weak honeypots, or a combination of both. They are usually implemented behind a system, called honeywall. Honeywalls create a bridged route and are able to monitor and capture packets, and IDS/IPS. As a result, a honeynet representing a real network or subset can be created by network learning with the help of the derived information. In 2004, a dynamic honeynet design was introduced to present more details on Spitzner's concept [22]. In this project, a dynamic honeypot server (DHS) could collect some information about the

environment through active and passive reviews based on the network architecture. During system testing, any problems for identifying the open ports on the devices were cited by the author. Nevertheless, most the above-mentioned designs had the disadvantage of not immediately adjusting a newly introduced system to the network and updating the databases after a predefined time by rebooting the honeyd or a new configuration file.

The Proposed Method

The proposed solution for the problems caused by dynamic systems is using a vulnerability scanner to scan the network and obtain vulnerabilities through special footprinting and scanning methods. Scanning can be used to detect the target operating system, its services, and possible vulnerabilities. In fact, scanning is the supplementary phase of footprinting and provides more detailed information about the target. The main idea of scanning is to detect unsafe channels and listening ports. In general, scanning process includes 3 steps:

1. Port scanning: detecting open ports and services of the target system
2. Network scanning: detecting the range of IP addresses of the target server
3. Vulnerability scanning: detecting the vulnerabilities of the operating system

The proposed approach was based on active or passive network scanning to obtain the necessary information. As the network environment becomes more realistic, intruders are more likely to believe it, and even professional intruders are easily deceived, so the network should be made as attractive as possible, albeit doubtful. Do not be professional intruders. In this system, a vulnerable sca nner was employed to obtain the system vulnerabilities. Then, the vulnerable points could be realized by using social engineering. The network is scanned actively or passively by using a modular method after a real network environment was achieved. The obtained data is stored in a database to generate an overview of the network and produce a honeyd configuration file based on the low and high interaction honeypots. It also generates an XML file to equip the high-interaction honeypots. Implementation of virtual honeypots in all parts of the system is facilitated by honeyd. The two objectives pursued in this process are briefly summarized as follows:

- Implementation of a dynamic honeypot and presentation of an advanced hybrid honeypot framework.
- Assessment of the effectiveness of the automatic production of the network organization to collect more information from attackers at the best time. Figure 2 provides an overview of the proposed scheme:

For each tool, a module is simultaneously designed with the activity of that tool to process the collected data and store the relevant information in the database. The advantage of the modular method is its ability to provide an independent operation for each scanning module. The commands with various symptoms allowed the user to start implementing any modules. To reduce the user's role in continuously monitoring the processes or performing individual processes, the two management programs of "Net Scan Tools" and "scanning" are created to monitor the performances of the modules and control the productions of the honeypot configuration files. Net scan tools is a detection tool of the network. Net Scan Tools Pro allows detecting, monitoring, and restoring the tools of the network. The tool enables us to obtain some information about internal LAN, Internet IP addresses, ports, etc. In addition, it allows detecting certain vulnerabilities and dangerous ports in the network systems. In fact, Net Scan Tools Pro is a combination of several network software and tools, which have been aggregated. One advantage of this tool is its ability to gather information about systems and networks simply and quickly.

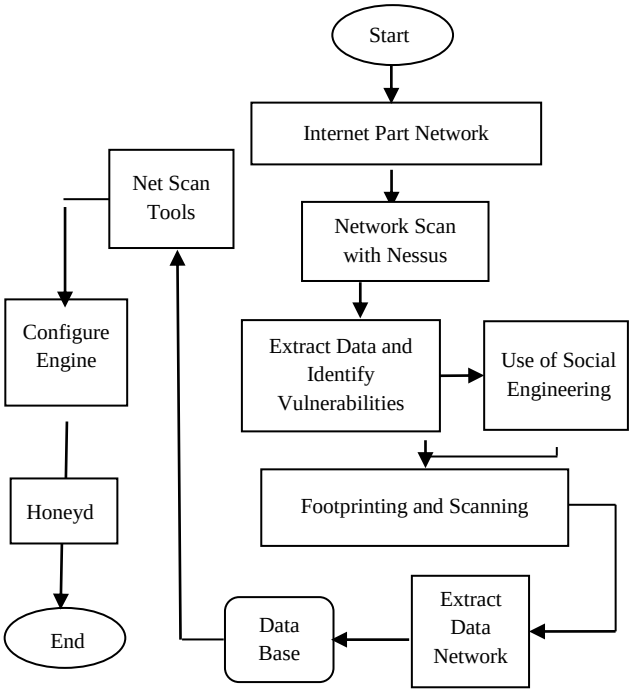

Fig. 2: An outline of the proposed method.

F. Net Scan Tools and Scanning Programs

The purposes of creating a management system were reducing the user's responsibility for monitoring the scanning processes and modules and creating honeypot configuration files as soon as identifying the major changes occurred to the system so as to reach a set of predetermined conditions.

G. Net Scan Tools and Scanning Programs

The purposes of creating a management system were reducing the user's responsibility for monitoring the scanning processes and modules and creating honeypot configuration files as soon as identifying the major changes occurred to the system so as to reach a set of predetermined conditions.

B.1. The Proposed Net Scan Tools

The net scan tools program is the main program in this investigation to monitor both the footrinting and scanning programs. After equipping the stored information of the initial configuration in the database, the net scan tools disseminated the collected information to all the corresponding modules. As represented in figure 3, the footprinting and scanning modules were utilized as threads, which were monitored through process identification (PID) number to validate the operations. Continuous monitoring made it possible to reuse these modules in case of their accidental failure. The net scan tools program also reviewed the results collected from all the scanning modules. As soon as it reached the reference point or set point determined by the user, it created its configuration files. The first set point depended on the number of devices to be identified prior to the initial uses of the honeypot configuration files. Therefore, the set point was determined based on the percentages of the altered devices or identified services. This system was designed in such a way that it could create a honeyd LIH configuration file with properties similar to those of an HIH configuration file with an XML format. Thus, the capabilities of this system were extended by creating honeyfarms, which were able to change the addresses and direct the attackers from LIH to HIH, thus facilitating the attacker direction task.

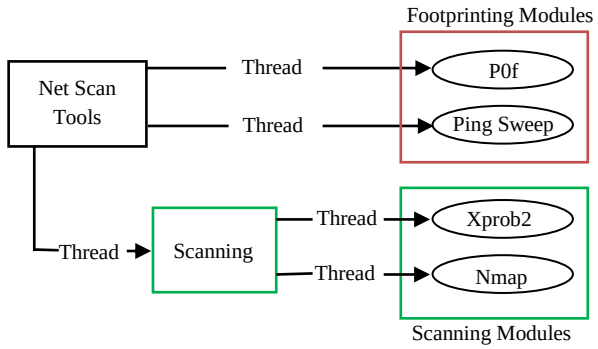

Fig. 3: The honeypot scanner management program.

B.2. Footprinting Program

A footprinting program is utilized to monitor the footprinting modules. Footprinting is defined as a process, during which the map of the network and systems of an organization is provided. The process begins with detecting the target systems, applications, or physical situation of the target. For example, the web page of an organization may contain a file on its personnel's biography. The information becomes valuable when hackers try to obtain it through social engineering. The information obtained in footprinting phase includes the network architecture, server, network services and applications, and the location of storing useful data.

H. Footprinting Modules

Two inactive scanner modules are used to collect data from devices on the network. Since these modules collect various data about the devices, it was necessary to use two separate sub-programs to access operating system information and determine active ports.

Any information about the network environment can be provided by p0f. In this research, fingerprinting of the IP addresses and open ports provided by ping sweep in the operating systems was done with the help of P0f, the results of which were then stored in the dynamic honeypot database [23]. A ping sweep (also called an ICMP sweep) is a technique to scan the network. The technique is used to scan a range of IP addresses and detect live systems in the network. While a ping command sends an ECHO request to one system, a ping sweep can send the packet to several computers of the network. If a host or system of the network is on, it sends back an ECHO reply. This technique can be almost used for all platforms. The table 3 shows as follows:

Table 3: The Initial Implementation Commands for the Footprinting Modules

Scan mode	Sub-programs	Scanning commands
Footprinting	p0f+Xprobe2	P0f –i {interface} -p -l Xprobe2-r –m 2 –o {file} –X {ip}

I. Scanning Modules

Similar to the footprinting method, Nmap and Xprobe2 modules are used for active scanning [24]. Various protocols are employed by each sub-program for determining the operating system types of the detected devices. More precise and accurate identification of operating systems can be achieved by each subprogram. The velocities and precisions of properly detecting devices and the traffic generated by the network for collecting data provided us with some leading factors for applying various active scanning modules.

J. Scan Network

To operate the active and passive scanning programs, the modules of P0f and ping sweep were created for collecting any device information from the network traffic and Nmap and Xprobe2 modules were generated

for actively scanning the detected devices on the network.

As afore-mentioned, footprinting modules constantly collect any information about the devices on the network and activate their corresponding scanning modules, respectively.

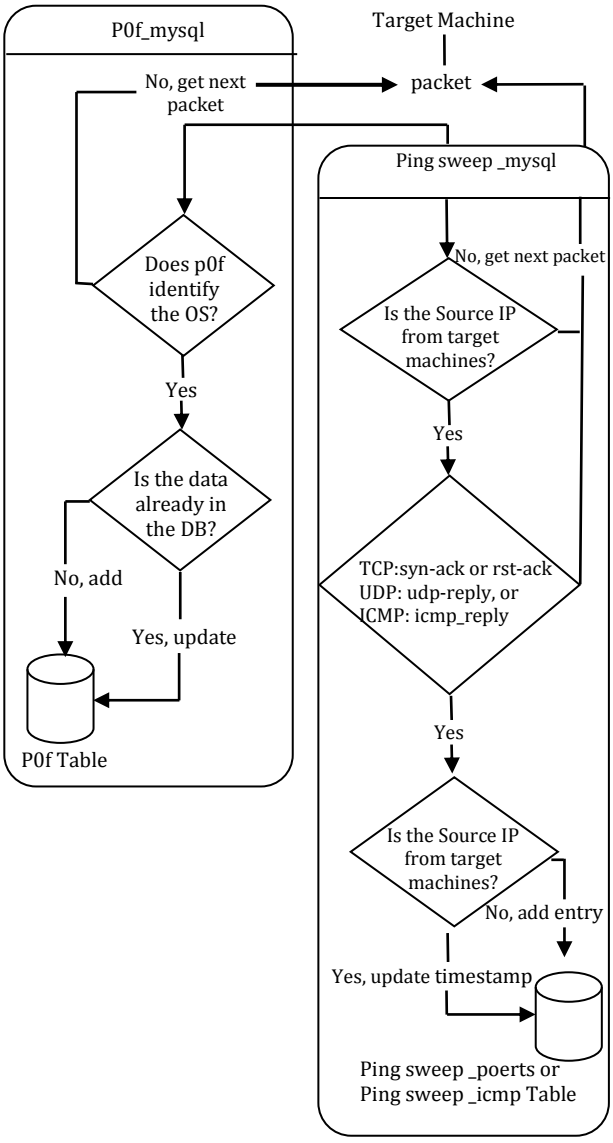

Fig. 4: The footprinting modules.

Table 4: The Initial Implementation Commands for the Scanning Modules

Scan mode	Sub-programs	Scanning commands
Scanning	Xprobe2+ nmap	Xprobe2-r –m 2 –o {file} –X –T 1-1024,3306 –U 1-1024 {ip} Nmap –sT –sU –T3 –sV –O –oX {file} –host-timeout 180 {ip}

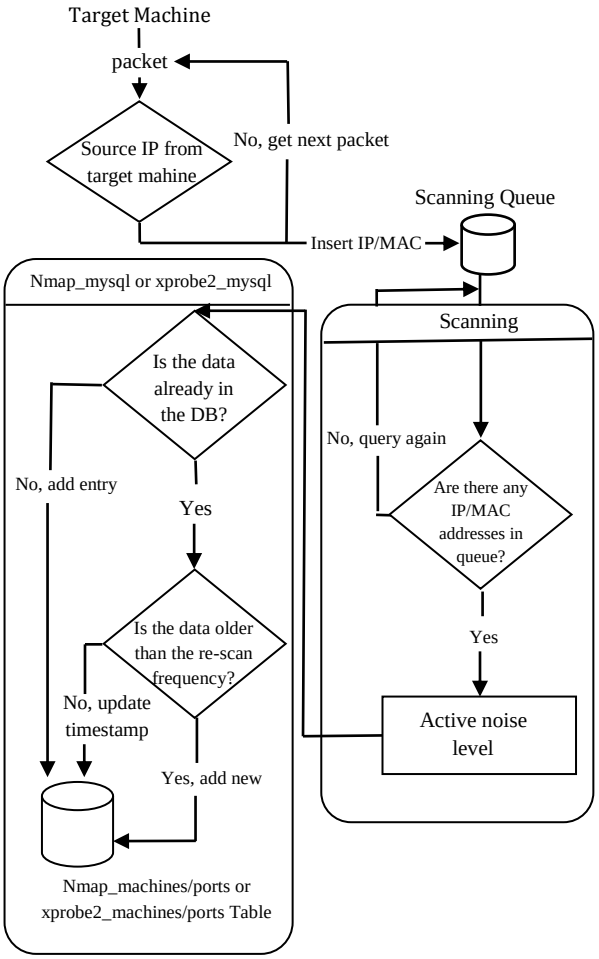

Fig. 5: Scanning modules.

K. Database

During the scan operation, query scan results are inserted and updated to remove additional data. This ensures that relevant information is used in generating honeypot configuration files. The collected data is stored in a database. One advantage of the database is that the user can browse the information collected by each module.

L. Low/high-Interaction Honeypot Configuration Engine

Since honeyd was more prevalent and could provide better support and interaction with attackers, it was selected for the low-interaction honeypot program, which aimed at combining the results of the scanning modules and creating the most accurate honeyd file for the production network. Once reinstalled, the net scan tools system generates a configuration file whose data is collected by the scanner modules. Any device or port that has been detected or updated in the interval between scans is added to the configuration file. The configuration is updated with all known data about any device detected by the scanner modules.

Upon generating the configuration file in some parts of the database, the coupled low- and high-interaction

honeypots provided an opportunity for collaboration. This matter facilitated the traffic direction from the low to high interaction honeypots.

The XML file was generated by the net scan tools program so that the user could create a high-interaction honeypot with any virtual machine programs or utilize that file as a formula for generating an independent high-interaction honeypot.

M. Management System Design and Application

The abilities to constantly monitor separate processes and reuse them if necessary were the essential features of this system.

Table 5 shows the applications of the modules and management programs.

Simulation Results

The testing process was originally started with the NESSUS vulnerability scanner program to identify vulnerabilities.

Upon accessing this point, changes were made to remove and fix them using social engineering. Afterward, the testing process was conducted with two different noise levels and two types of scans. Network formation was identical in each testing process, the sub-program capabilities of which were measured in terms of identification of the beginning and ending configuration files.

Different versions of Linux and Windows were used to evaluate the capability of identifying unique changes and determine which sub-program had the best function in various operating systems. The operating systems as virtual machines were installed on VMWare's ESXi 5.1 server.

The honeypot scanning program is allowed to monitor the network traffic via a virtual switch set in a promiscuous mode. The program was stopped between each test and the web browsers in each machine were closed to collect the data of the previous test from the database. The next test was initiated after clearing all the information of the previous test from the database tables.

For analyzing and simulating the network topology in the testing phase, different modes were assessed. The tests were conducted in 4 steps. After scanning the network with NESSUS vulnerability scanner and taking the necessary measures in each case, the results were obtained as follows:

Assuming the vulnerability of the above program against attacks to be definite, the program was updated with the latest Microsoft tools, the security documentation was provided in connection with a proper Interactive Information System (IIS) management, and a security hole was found in one of the Windows computers in the IIS service.

Table 5: Applications of the Modules and Management Programs

Modules and programs	Application
Net scan tools	1. Monitoring the statuses of the mentioned programs 2. Creating honeypot files after a certain change in percentage 3. Monitoring the percent changes in the network configuration and thread
Scanning	1. Scanning the ports and network 2. Allowing the user to determine the noise set level and scan Nmap or Xprobe2 IP addresses 3. Starting the module
P0f	1. Monitoring the network traffic 2. Collecting information about the present machines on the network 3. Identifying machines for the operating systems 4. Storing information in the database
Ping sweep	1. Collecting information by scanning the present machines on the network 2. Identifying the machines for the operating systems 3. Identifying UDP or TCP ports 4. Storing information in the database
Xprobe2	1. Collecting information by scanning the present machines on the network 2. Identifying machines for the operating systems 3. Identifying UDP or TCP ports 4. Storing information in the database
Nmapl	1. Collecting information by scanning the present machines on the network 2. Identifying the machines for the operating systems 3. Identifying UDP or TCP ports 4. Storing information in the database

Microsoft SQL Server, which has several vulnerabilities, was installed on Windows 7 so as to allow critical data to be accessed, contents of the database to be changed, and the lives of SQL Servers to be endangered by attackers (1433 and 1434 ports were the default ports of the server, which were checked permanently and sophistically by the attackers).

Upon fixing these points, system configuration was properly conducted by checking and protecting each of the existing systems so as to ensure that they had a password associated with the corresponding account. Filtering was conducted on port 1434. Internet Explorer (IE) browser is extensively available in Microsoft

software and generally exists in most Windows systems. The patches required for vulnerable points were identified to be used in the IE browser.

Table 6: Input Parameters

Parameter	Measure
Ethernet interface	Eth1
MAC address	30:5A:3A:7A:32:5F
DHCP server	192.168 1.1
Honeypot initial deployment	5
Percent change (redeployment)	30
Noise level	For the active scan
Active scan (sec)/rescan	43200
OS	Linux, Windows
Modules	P0F, ping sweep, Nmap, Xprobe2, NESSUS

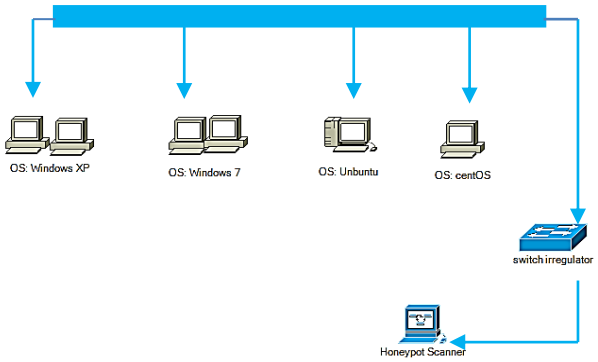

Fig. 6: Network topology in the testing phase.

Another defined point for vulnerable accounts was a weak password or no password while assuming that the accounts were defined by the network user and the nature of the honeypot was based on seeking the attacker. Also, due to using social engineering, no changes were made to resolve this vulnerability so that the attackers would look for the accounts and their passwords for hours, during which the honeypot could get their rootkits. The automatic execution feature of WSH is the main source of supplying and releasing ILOVEYOU worm. Due to this vulnerability in some Windows systems and because WSH technology is an optional part of them, it was removed or disabled on the computers securely and without any particular concerns in several cases. When Office pack is installed in Windows operating systems, Outlook Express will be installed too, which is associated with several security issues. There was no need for fixing this vulnerable point when using social engineering. NESSUS functions on Linux operating systems will be explained below.

Considering the widespread use of BIND software for implementing DNS and its vital position on a computer network, the attackers could choose it as an appropriate target for conducting a variety of attacks against it. Nearly all Linux operating systems have a version of BIND software, the vulnerability of which was fixed by updating it with the latest patches provided by the manufacturer. The existences of several RPC security vulnerabilities have allowed attackers to exploit them for various attacks. In most cases, RPC services run with more permissions than usual. To let the honeypot track down the attacker and interact with it through social engineering, only the permissions that were out of the ordinary were resolved, while no operations were conducted to completely remove this vulnerability. Vulnerabilities of the accounts with weak or unencrypted passwords for Linux operating systems were monitored by NESSUS program. In fact, social engineering has prevented this problem to be resolved. An SNMP version was enabled on most Linux systems by default since this protocol was assumed to have much vulnerability; nevertheless, we filtered SNMP in the network input ports (TCP/UDP port 161 and TCP/UDP port 162). SSH is a general service to secure logins, execute commands, and send files on the network. Although most Linux-based systems on our network used it, NESSUS identified several security vulnerabilities related to it. Most of the vulnerabilities detected were merely minor bugs and only a few of them were important. The newest version of SSH was installed on the operating system after downloading the latest patch from the related site. After achieving the vulnerabilities and fixing them with the help of social engineering, we scanned the network to produce the best configuration file. The net scan tools program generated additional traffic during the footprinting test. The p0f program could correctly identify 8 out of 11 Windows computers while recognizing only two of the Linux computers. Ping sweep was able to identify only 5 open ports.

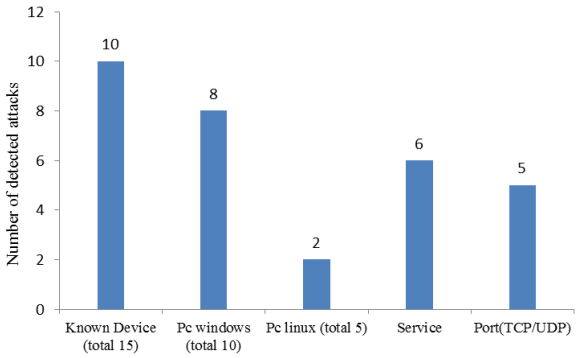

Fig. 7: The footprinting at the testing phase.

For testing noise, Nmap and Xprobe2 applications were applied to actively scan the networks. The test was performed using the active scan method and the Nmap module. TCPSYN scan was run on each machine. In this test, all the devices were identified as the "up" state, but none of the Linux devices were identified.

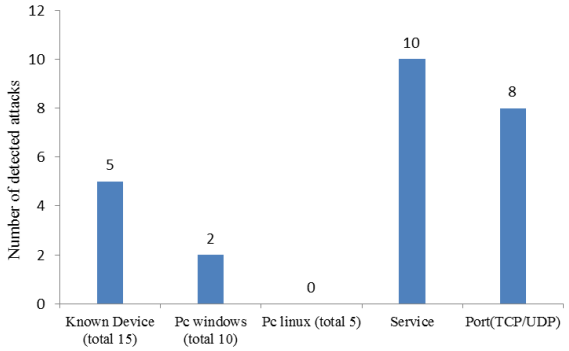

Fig. 8: The scanning at the testing phase.

Nmap application was able to identify only two Windows operating systems. Also, few open ports were identified. The combination of Nmap and Xprobe2 applications provided a more complete image of the network through our goals were not still met. Only one kind of the Linux operating systems could be identified by Xprobe2 application. Two of the Windows operating systems besides most open ports on the devices could be identified by Nmap application.

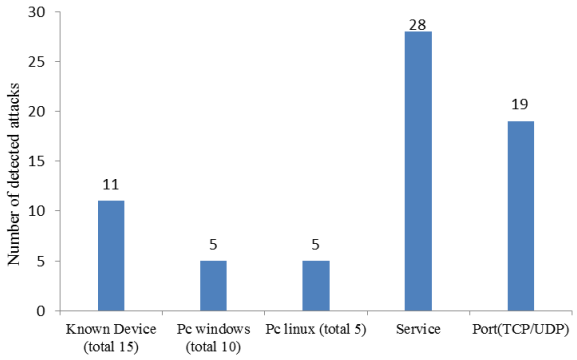

Fig. 9: A chart of the hybrid scanning mode at the testing phase.

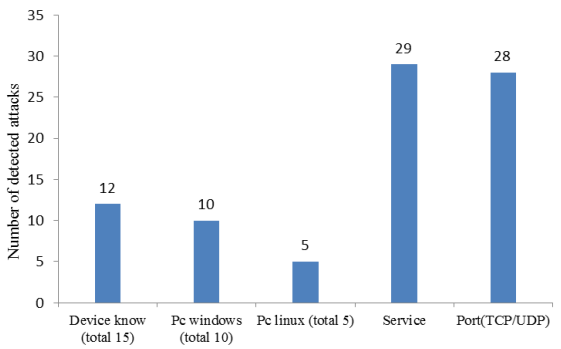

Fig. 10: A chart of the footprinting and scanning mode results.

The results revealed that footprinting and scanning methods can be used simultaneously to get a detailed picture of the network environment. Extensive information can be collected by creating a network of honeypot (honeynet) that represents the production environment. Using a vulnerable scanner to identify security vulnerabilities and fixing them with the help of social engineering, which provided the effectiveness of network automatic generation led to the reduction of the user's role. Moreover, a more accurate image of the network topology was obtained with a much higher speed of detection. In an attempt to improve the identification process, the best-fitted image of the network was introduced to produce a honeypot.

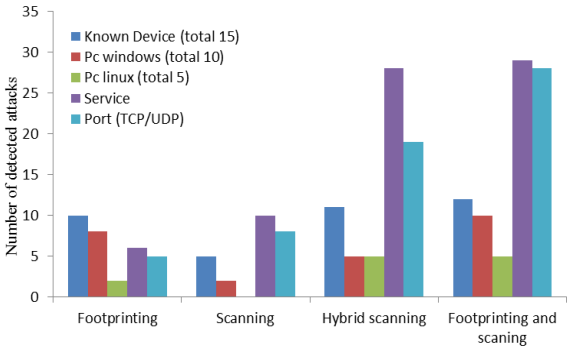

Fig. 11: A comparison of the different scenarios.

The windows operating systems could be identified by the p0f program. This feature was reduced when the scanning method was utilized. The Xprobe2 program succeeded in identifying Linux operating systems. Nmap and ping sweep programs were able to identify most open ports that were exchanging information.

Results and Discussion

A combination of both scanner applications made it possible to get a more complete picture of the network. The automated network production led to the decreased user's role in preventing human errors, detailed image of the network, and better effectiveness of the identification speed.

Conclusion

The most important part of a complicated honeypot is the way it gets information for developing a network. This information may include operating system types and methods of application in the existing environment. Honeypots can obtain the network parameters to provide a comprehensive plan and quick response to the present system environment. One of the simplest ways to do this process is actively exploring and specifying the types of systems and services utilized. In our systems, honeypots were established and developed as independent devices capable of physically communicating with the network of distributed computer systems.

Table 7: A Comparison of the Relevant Studies on Honeypot

Author(s)	Specification	Honeypot level	Flexibility	Form of honeypot	Nature of honeypot	Anti-Modification	Dynamic Information			Application Environment
							Deployment	Configuration	Object	
Leonard et al. [4]	Detect a variety of communication attacks	Low interaction	NA	Virtual	Passive	✓	✓		Honeypots	Wearable networks
Guarnizo et al. [5]	Capture and analyze traffic and attacks	High interaction	NA	Physical	Passive		✓		Honeypots	Internet of things
Fraunholz et al. [6]	Monitor host and network activities	High interaction	NA	Virtual	Active		✓	✓	Honeypots	Industrial and embedded applications
Daubert et al. [7]	detecting active attackers	Medium interaction	NA	Virtual	Active	✓		✓	Honeypots	Unmanned aerial vehicle
Baykara et al. [8]	Honeypot IDPS	Low interaction and Configurable	NA	Virtual	Active		✓		Honeypots	Institutional network (Adaptable to other network systems)
Proposed method	Advanced hybrid Honeypot/ actively exploring and specifying	Low and high interaction	High	Physical	Active and Passive	✓	✓	✓	Honeypots and real services	Automatic network generation

The tracking and training phase began after connecting to the network devices. Topology learning by the honeypots occurred at this phase. The varied durations of the learning phases depend on the system topologies. Improvement in the diagnostic process was the most important part of this study. A more comprehensive picture of the network could be available only when the optimal results for the identification of penetration were achieved within a short time. Proposed method is compared with the relevant studies on honeypot.

The table 7 shows the numbers and types of the applied operating systems and services could be determined by the proposed method after performing modular network scanning. Furthermore, our honeypots had the ability to determine who represented the actual systems or communication services and how they occurred. Once the honeypots collected important information, they could begin to use and develop themselves.

Information obtainment through our integrated analysis of the active and passive scanning methods had no end and could be continuously in progress. The entire network was monitored by the systems, which increased its flexibility. The proposed complicated honeypots significantly reduced the amount of work required for configuration and management in a changing environment.

The proposed honeypots have another advantage of minimizing the risk of any human mistakes during manual configuration. The risk of an attacker's suspicion is also minimized by the surrounding integration environment.

Author Contributions

M. Amiri designed the experiments and interpreted the results. A. Barati collected the data. M. Amiri and A. Barati wrote the manuscript.

Acknowledgment

We would like to thank the Islamic Azad University, Dezful branch for providing support in conducting this study.

Conflict of Interest

The author declares that there is no conflict of interests regarding the publication of this manuscript. In addition, the ethical issues, including plagiarism, informed consent, misconduct, data fabrication and/or falsification, double publication and/or submission, and redundancy have been completely observed by the authors.

Abbreviations

DHS	Dynamic honeypot server
PID	Process identification
IIS	Interactive information system
IoT	Internet of thing

References

[1] M. E. Namin, M. Hosseinzadeh, N. Bagheri, A. Khademzadeh, "RSPAE: RFID search protocol based on authenticated encryption," Journal of Electrical and Computer Engineering Innovations, 6(2): 179-192, 2018.

[2] M. Safkhani, "Cryptanalysis of R2AP an ultra lightweight authentication protocol for RFID," Journal of Electrical and Computer Engineering Innovations, 6(1): 107-114, 2018.

[3] L. Spitzner, Honeypots: tracking hackers, Addison Wesley Professional, 1: 2002.

[4] P. Diebold, A. Hess, G. Schäfer, "A honeypot architecture for detecting and analyzing unknown network attacks," in Proc. 14th Kommunikation in Verteilten Systemen (KiVS05): 245-255, 2005.

[5] J. P. John, F. Yu, Y. Xie, A. Krishnamurthy, M. Abadi, "Heat-seeking honeypots: design and experience," in Proc. The 20th International Conference on World Wide Web, ACM: 207-216, 2011.

[6] A. M. Leonard, H. Cai, K. K. Venkatasubramanian, M. Ali, and T. Eisenbarth, "A honeypot system for wearable networks," in Proc. IEEE 37th Sarnoff Symposium: 199-201, 2016.

[7] J. D. Guarnizo, A. Tambe, S. S. Bhunia, M. Ochoa, N. O. Tippenhauer, A. Shabtai, Y. Elovici, "Siphon: Towards scalable high-interaction physical honeypots," in Proc. The 3rd ACM Workshop on Cyber-Physical System Security: 57-68, 2017.

[8] D. Fraunholz, D. Krohmer, H. D. Schotten, C. Nogueira, "Introducing FALCOM: A multifunctional high-interaction honeypot framework for industrial and embedded applications," in Proc. International Conference on Cyber Security and Protection of Digital Services (Cyber Security)): 1-8, 2018.

[9] J. Daubert, D. Boopalan, M. Mühlhäuser, E. Vasilomanolakis, "HoneyDrone: A medium-interaction unmanned aerial vehicle honeypot," in Proc. NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium: 1-6, 2018.

[10] M. Baykara, R. DAŞ, "SoftSwitch: A centralized honeypot-based security approach using software-defined switching for secure management of VLAN networks," Turkish Journal of Electrical Engineering & Computer Sciences, 27(5): 3309-3325, 2019.

[11] R. Danford, 2nd Generation Honeyclients, SANS Internet Storm Center, 2006.

[12] F. Pouget, M. Dacier, V. H. Pham, "Leurreé.com: On the advantages of deploying a large scale distributed honeypot platform," in Proc. The E-Crime and Computer Evidence Conference, 2005.

[13] J. Levine, R. LaBella, H. Owen, D. Contis, and B. Culver, "The use of honeynets to detect exploited systems across large enterprise networks," in Proc. Information Assurance Workshop, IEEE Systems, Man and Cybernetics Society: 92-99, 2003.

[14] J. G. Levine, J. B. Grizzard, and H. L. Owen, "Using honeynets to protect large enterprise networks," IEEE Security & Privacy, 2(6): 73-75, 2004.

[15] S. Azadegan and V. McKenna, "Use of honeynets in computer security education," in Proc. IEEE Fourth Annual ACIS International Conference on Computer and Information Science: 320-325, 2005.

[16] 'The Xen Hypervisor,'December 2019.

[17] F. Bellard, "QEMU-open source processor emulator," 14 November 2019.

[18] 'VMware", 01 October 2018.

[19] N. Provos, T. Holz, Virtual honeypots: from botnet tracking to intrusion detection, Pearson Education, 2007.

[20] R. Baumann, C. Plattner, Honeypots, Swiss Federal Institute of Technology, 2002.

[21] P. Fanfara, M. Dufala, J. Radušovský, "Autonomous hybrid honeypot as the future of distributed computer systems security," Acta Polytechnica Hungarica, 10(6): 25-42, 2013.

[22] I. Kuwatly, M. Sraj, Z. Al Masri, and H. Artail, "A dynamic honeypot design for intrusion detection," in Proc. IEEE/ACS International Conference on Pervasive Services: 95-104, 2004.

[23] C. Hecker, B. Hay, "Securing E-government assets through automating deployment of honeynets for IDS support," in Proc. 43rd Hawaii International Conference in System Sciences (HICSS): 1-10, 2010.

[24] C. Hecker, B. Hay, "Automated honeynet deployment for dynamic network environment," in Proc. 46th Hawaii International Conference In System Sciences (HICSS): 4880-4889, 2013.

Biographies

Mehdi Amiri was born in 1983 in the Aleshtar city of Lorestan province. He received his B.Sc. degree in Computer Hardware Engineering, M.Sc. degree in Computer Systems Architecture Engineering in 2007 and 2019, respectively. His major research interests include Computer networks, network security, and new information and communication technologies.

Ali Barati is an Assistant Professor in the Department of Computer Engineering at Dezful Branch, Islamic Azad University, Dezful, Iran. He received his B.Sc. degree in Computer Hardware Engineering and M.Sc. degree in Computer Software Engineering and a Ph.D. degree in Computer Software Engineering in 2001, 2004 and 2014, respectively. Currently, he is a faculty member of Dezful Branch, Islamic Azad University, Dezful, Iran. Between 2007 and 2011, he was appointed as the Head of Department of Computer Science at the University. His major research interests include wireless sensor networks, VANET, high-speed networks, fault-tolerant systems, and network security.

How to cite this paper:

M. Amiri, A. Barati, "An advanced hybrid honeypot for providing effective resistance in automatic network generation," Journal of Electrical and Computer Engineering Innovations, 7(2): 133-144, 2019.

DOI: 10.22061/JECEI.2020.5621.241

URL: http:// jecei.sru.ac.ir/article_1185.html