

# Implementation of Honeypot to Trap and Track Cyber Attacks

Praful Nair<sup>1</sup>, Vishak Nair<sup>2</sup>, Karan Nair<sup>3</sup>, Prof. K.S Charumathi<sup>4</sup>

<sup>1-3</sup>BE student, Department of Computer Engineering, Pillai College of Engineering, Navi Mumbai, India-410206

<sup>4</sup>Assistant Professor, Department of Computer Engineering, Pillai College of Engineering, Navi Mumbai-410206

\*\*\*

**Abstract** - The domain of Cyber Security and System Security is an ever-expanding domain and this is due to increasing rate of cyber-attacks in the world. Even though there are various security measures continuously being developed and deployed in order to detect and mitigate cyber-attacks, the effectiveness of these measures are still at question, especially against newer attacks like new ransom wares and zero day exploits. One way to build a better preventive measure is to understand how the attack works, how hackers behave, what they try to do once they penetrate inside a system. Tracking hacker's movement and understanding the steps taken by them to compromise a machine is important to build a particular counter measure. This is where Honeypot system proves to be an important element in Cyber Security. Honeypot is an age-old technique of luring an attacker into a weak, vulnerable, decoy system, (which is meant to be illicitly used by hackers) which is carefully rigged by monitoring software and data collecting software. The honeypot system captures and track intruder's movement inside the system, and transfer these data to the system administrator/ security personnel that deployed and configured the honeypot. The collected data can prove to be crucial for setting up counter measures against various types of attack, especially newer types of attacks. Use of honeyspots provides effective solution to increase the security and reliability of the network or a system and helps researchers, scientists and engineers establish a defense strategy against cyber-attacks.

**Key Words:** Honeyspots, Research Honeyspots, Monitoring Network, Decoy system, Monitoring and Data Collection.

## 1. INTRODUCTION

Concept of Honeypot is widely used all over the world in variety of ways as a security measure. We can categorize honeypot as Research Honeypot or Production honeypot. Honeyspots comes in different shapes, size and nature. The beauty of this concept is the versatility in implementing honeypot. Research based honeypot is deployed in the open to analyze the ongoing attacks, whereas Production honeyspots are deployed for an organization inside its perimeter. Such a mechanism indicates security breaches. In this paper, we are proposing a simple high interaction honeypot system that is capable of monitoring the activities of hacker, once successfully inside the system and capture data and tracks left by him/her and further analyze these collected data. The proposed honeypot is suggested to be installed in a

virtual environment, that is, it will be hosted by main system(host) virtually over the network. But it could be plainly installed on a computer system as it is. The types of vulnerabilities that will be introduced in the system must be carefully planned and placed; as for a given vulnerability, there are numerous exploits available and numerous ways to bypass it.

## 2. RELATED WORKS

The overall objective of any honeypot system is to identify intrusion and understand in depth about the intrusion, and it may even further involve mitigation of attacks. One way to achieve few of these is via applying the concept of honeypot into IDS and IPS, as described in [2]. The researchers in [2] has made honeypot configuration on IDS to identify unauthorized users in a given network, who take advantage of unused IP address, username etc. They have implemented eight different methodologies to trap the attacker. Another interesting take on implementation of Honeypot is mentioned in [4]. The authors, Chao-Hsi Yeh and Chung-Huang Yang described how easy it is to deploy virtual honeyspots with the use of Honeyd. Honeyd is a complete low-interaction honeypot system that is capable of deploying Honeypot over a network with different personalities and settings. Honeyd is outdated now. Maintenance of Honeypot is an equally important factor. One way to get an Adaptive Honeypot configuration, which is automatically deployed and configured is explained in [3], which involves Machine learning as well.

## 3. PROPOSED ARCHITECTURE

### 3.1 Concept and Framework

The concept of this particular honeypot has two key striking features. First, the honeypot can also be a virtual machine. The operating system could be a Windows operating system, (preferably older versions like Windows Server 2003) or a Debian based OS, like Linux (In case of installing the honeypot as it is on a system, any OS, preferably Windows would work). These machine(s) would be running in a host computer, with the help of software like VMware or Oracle's Virtual Box. These Software, helps to run number of virtual machines from a host system, which can be connected to a common network. This will

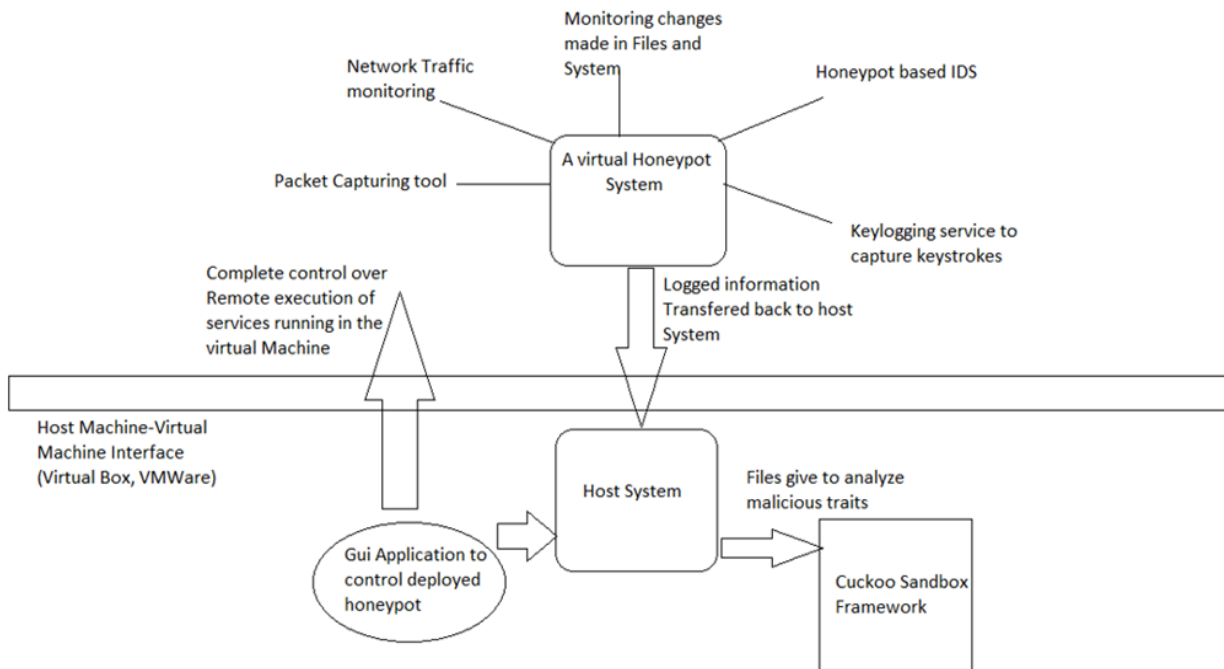


Fig 1- Proposed Framework

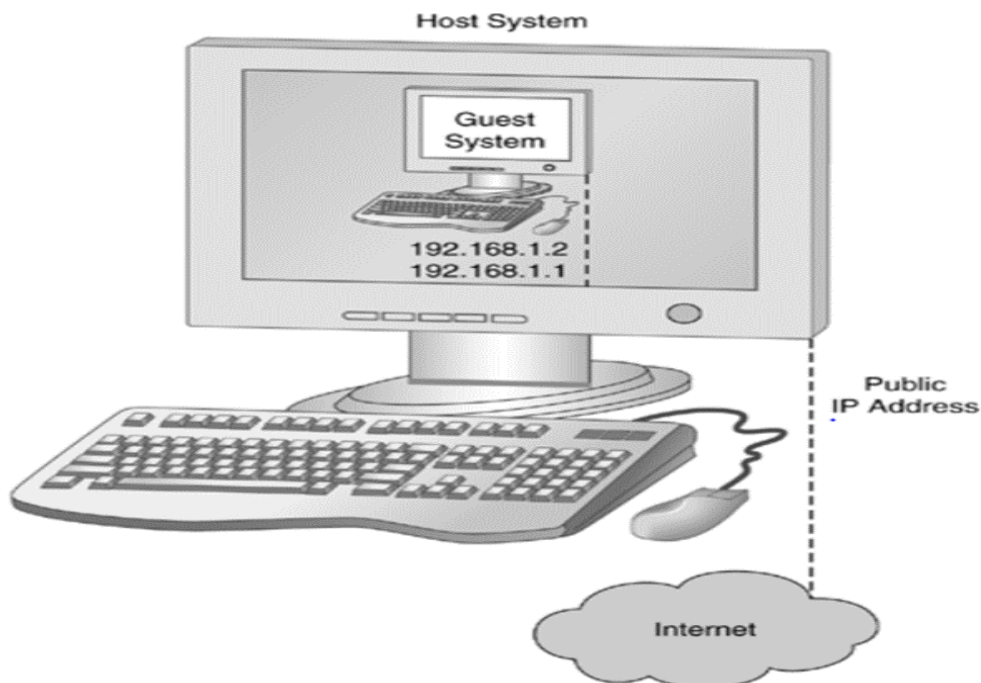


Fig 2 – Depiction of Honeypot Setup

be elaborated further in the next subsection. The deployed Honeypot, that is, these machines will have number of loop holes, which is to be illicitly used by the attacker. The proposed system, hence could be installed on a system or installed on virtual machines hosted by a main host OS. Easy way in into the system is supposed to lure the attacker. Concept of honeypot immersion is crucial here.

### 3.2 Modules in the Framework

The three key functioning involves Monitoring the deployed honeypots, Collecting the information, and further analyzing the collected information. Therefore, appropriate tools and program has to be placed and executed in the virtual machine. Python and C will be preferred for scripting manual programs, that will be running continuously in the machine. All these scripts will be running in the background as daemon services. A Monitoring program will be running, to inspect over network traffic. Pcap is a module for packet capture using libpcap. LibPCAP is the packet capture library written for TCPDUMP and also used in Wireshark. IP packet capturing can also be implemented via Socket programming. Moreover, the captured packet can be further analyzed for malicious trait, with the help of frameworks like cuckoo sandbox. The captured information will be stored as log, which will be transferred back to the host system. Apart from monitoring all the network aspect, programs will also be deployed to look over all the changes made in the system by the attacker. This involves changes being made into a file, movement of files, transfer of new files, deletion of files etc. File-system events monitoring with use of Python, watchdog library and other OS related library like Blob, shutile etc. If we specifically talk about a windows system, changes made in the registry is also important to be tracked and monitored. UNIX system has '/etc' folder which are hot target for attackers.

## 4. IMPLEMENTATION

### 4.1 A Main Driver Program

In case of installing the honeypot program on virtual environment, the host system handling the virtual machines deployed as honeypots will control them with the help of a GUI drive program. The program will offer a default configuration for less experienced users, which they can use for testing the working of the programs and services included. Researchers and Analysts can manually configure the entire honeypot setup. They have the liberty to evoke as many services as they want, or just even a single service. For an example, in case of insider attack, to monitor and analyze the situation, initiating a SSH module is not necessary, only modules like keylogger and IP packet sniffer is enough to capture the attack being performed. This is how the entire design provides a 'flexibility'. Figure 1 explains how the system will look in case of honeypot being deployed on a virtual machine.

### 4.2 Network Monitoring and Packet Capturing

The running services like Network traffic monitoring program will continuously look for abnormal spike or variation in usage of network resources. The data will be logged and sent back to base System. For this, Scapy in python can be used to manipulate or forge and decode network packets. It is important to keep these data in a standardized format so that it can be passed to Cuckoo sandbox for further analysis. Other libraries like dpkt can also be used to make more features available in this program. Our IP packet Capturing script is based on socket programming, written in python, is capable of capturing IP packets in the Honeypots. The IP packets are captured, presented in Raw data format so that its useful for Analyst to parse. It's parsed again and all the values like Protocols, Flags, Version etc. are presented separately.

### 4.3 Monitoring Port Activities.

Ports are important part of system, hosting a number of services for the users. It is important to keep track of activities in this space. Most of the attack begin with understanding what kinds of ports are open and what kind of services they are running, and this is usually done with the help of network mappers (N- map). A simple program can keep a look at ports for possible intrusive scans from attackers or unauthorized users scanning these honeypots. The program also captures the IP address of attacker within the network which attempts a scan. For this program, Nmap library for python will be used, socket programming to make a listener. This will constantly look for particular traffic and listen to a given port.

### 4.4 History of Commands in Terminal

It is not completely necessary to rely on Key-logging script to understand what is being done on the computer. Certain attack can open terminal access remotely for the attacker, and in such case, key-logging service may not be able to extract keystrokes events. In such cases, we can easily analyze the HISTFILE or .bash-history, which is mostly hidden. This stores all the terminal commands entered by the user, and such data would prove to be really useful to understand what kind of changes are being made into the system.

### 4.5 Changes Observed in File System

A complete dedicated program has been developed which will be used to monitor all the changes happening in the file system of the honeypot. Considering a Windows OS in mind, two programs have been developed. One is a File System Monitor, and another one, which looks out for activities in file directory. File system Monitor when executed in the honeypot, automatically checks if its running for the very first time and in that case, traverse through the entire system, count and collects the Number of Files. Further it calculates the Md5 hash value of each files and stores it in a CSV file in the system. Upon subsequent execution of this program, it just recalculates the hash values of Files present in the system and detect in case of

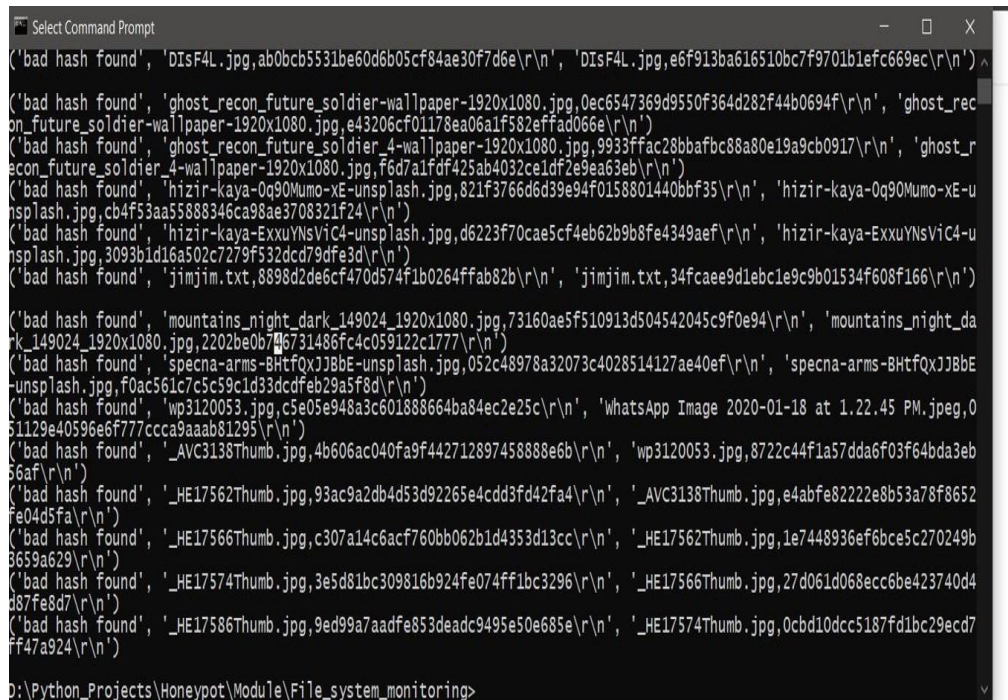


abnormality in hash values. This can happen if the attacker is targeting at the integrity of data present in the system. Moreover, this program lets the Administrator/Security analyst know if data has been hampered at the bit level. Figure 2 shows how bad hash is found which means those files may have been compromised. The second program that we developed, which come under this category, logically hovers over the honeypot system to see if hacker is creating or sending new files (possibly malicious), or deleting files or modifying the files. The program just detects the activities on the directory and saves the

information in a log. For this Watchdog module is used in Python.

#### 4.6 Key-Logger

A very basic keylogger. A keylogger is considered to be a fundamental part of most of the honeypot that exist out there. It is important for capturing keystrokes being entered in the system. Analyzing the captured information gives an in-depth insight on what exactly the hacker is doing, once inside the system.



```
Select Command Prompt
('bad hash found', 'DIsF4L.jpg,ab0bcb5531be60d6b05cf84ae30f7d6e\r\n', 'DIsF4L.jpg,e6f913ba616510bc7f9701b1efc669ec\r\n')
('bad hash found', 'ghost_recon_future_soldier-wallpaper-1920x1080.jpg,0ec6547369d9550f364d282f44b0694f\r\n', 'ghost_recon_future_soldier-wallpaper-1920x1080.jpg,e43206cf01178ea06a1f582effad066e\r\n')
('bad hash found', 'ghost_recon_future_soldier_4-wallpaper-1920x1080.jpg,9933ffac28bbafbc88a80e19a9cb0917\r\n', 'ghost_recon_future_soldier_4-wallpaper-1920x1080.jpg,f6d7a1fdf425ab4032ce1df2e9ea63eb\r\n')
('bad hash found', 'hizir-kaya-0q90Mumo-xE-unsplash.jpg,821f3766d6d39e94f0158801440bbf35\r\n', 'hizir-kaya-0q90Mumo-xE-unsplash.jpg,cb4f53aa55888346ca98ae3708321f24\r\n')
('bad hash found', 'hizir-kaya-ExxuYNSViC4-unsplash.jpg,d6223f70cae5cf4eb62b9b8fe4349aef\r\n', 'hizir-kaya-ExxuYNSViC4-unsplash.jpg,3093b1d16a502c7279f532dcd79dfe3d\r\n')
('bad hash found', 'jimjim.txt,8898d2de6cf470d574f1b0264ffab82b\r\n', 'jimjim.txt,34fcaee9d1ebc1e9c9b01534f608f166\r\n')
('bad hash found', 'mountains_night_dark_149024_1920x1080.jpg,73160ae5f510913d504542045c9f0e94\r\n', 'mountains_night_dark_149024_1920x1080.jpg,2202be0b746731486fc4c059122c1777\r\n')
('bad hash found', 'specna-arms-BHtfQxJJBbE-unsplash.jpg,052c48978a32073c4028514127ae40ef\r\n', 'specna-arms-BHtfQxJJBbE-unsplash.jpg,f0ac561c7c5c59c1d33dcdfcb29a5f8d\r\n')
('bad hash found', 'wp3120053.jpg,c5e05e948a3c601888664ba84ec2e25c\r\n', 'WhatsApp Image 2020-01-18 at 1.22.45 PM.jpeg,051129e40596e6f777ccca9aaab81295\r\n')
('bad hash found', '_AVC3138Thumb.jpg,4b606ac040fa9f442712897458888e6b\r\n', 'wp3120053.jpg,8722c44f1a57dda6f03f64bda3eb56af\r\n')
('bad hash found', '_HE17562Thumb.jpg,93ac9a2db4d53d92265e4cdd3fd42fa4\r\n', '_AVC3138Thumb.jpg,e4abfe82222e8b53a78f8652fe04d5fa\r\n')
('bad hash found', '_HE17566Thumb.jpg,c307a14c6acf760bb062b1d4353d13cc\r\n', '_HE17562Thumb.jpg,1e7448936ef6bce5c270249b8659a629\r\n')
('bad hash found', '_HE17574Thumb.jpg,3e5d81bc309816b924fe074ff1bc3296\r\n', '_HE17566Thumb.jpg,27d061d068ecc6be423740d4387fe8d7\r\n')
('bad hash found', '_HE17586Thumb.jpg,9ed99a7aadfe853deadc9495e50e85e\r\n', '_HE17574Thumb.jpg,0cbd10dcc5187fd1bc29ecd7ff47a924\r\n')
D:\Python_Projects\HoneyPot\Module\File_system_monitoring>
```

Fig -3 The program finding bad hash values in files attacked

#### 4.7 SSH Honeypot

A dedicated SSH honeypot has been developed using Python's Twisted Matrix Lab's Twisted Conch. Conch is an SSHv2 implementation written in Python. SSH is a protocol designed to allow remote access to shells and commands. It's generic enough to allow everything from TCP forwarding to generic File system access. So, what we essentially did here with this library is to open a SSH service at port 22. This is not fully emulating all SSH server, but only mimicking partially what SSH do. When the attacker tries to scan this port for information, the data is logged with Ip address of the attacker. Further, while accessing, it captures password attempts, all the dictionary used to brute force it. Even upon successfully cracking it, the shell access only Echo out the entered command and gives zero features and power of SSH. But by the time attacker realizes it's a ssh honeypot, enough data would be captured. Figure 3 shows a brute force attempt on our SSH module.

```
ib/site-packages/twisted/python/log.py:214>. Remove one of the calls to beginLoggingTo.
2020-02-21 15:38:25+0530 [-] Log opened.
2020-02-21 15:38:25+0530 [-] SSHFactory starting on 2022
2020-02-21 15:38:25+0530 [-] Starting factory <twisted.conch.ssh.factory.SSHFactory instance at 0x0000000004407c88>
2020-02-21 15:38:38+0530 [twisted.conch.ssh.factory.SSHFactory] disabling non-fixed-group key exchange algorithms because
2020-02-21 15:38:38+0530 [SSHServerTransport,0,127.0.0.1] kex alg, key alg: 'ecdh-sha2-nistp256' 'ssh-rsa'
2020-02-21 15:38:38+0530 [SSHServerTransport,0,127.0.0.1] outgoing: 'aes128-ctr' 'hmac-sha2-256' 'none'
2020-02-21 15:38:38+0530 [SSHServerTransport,0,127.0.0.1] incoming: 'aes128-ctr' 'hmac-sha2-256' 'none'
2020-02-21 15:38:38+0530 [SSHServerTransport,0,127.0.0.1] NEW KEYS
2020-02-21 15:38:38+0530 [SSHServerTransport,0,127.0.0.1] starting service 'ssh-userauth'
2020-02-21 15:38:38+0530 [SSHService 'ssh-userauth' on SSHServerTransport,0,127.0.0.1] 'admin' trying auth 'none'
2020-02-21 15:38:41+0530 [SSHService 'ssh-userauth' on SSHServerTransport,0,127.0.0.1] 'admin' trying auth 'password'
2020-02-21 15:38:42+0530 [-] 'admin' failed auth 'password'
2020-02-21 15:38:42+0530 [-] unauthorized login:
2020-02-21 15:38:51+0530 [SSHService 'ssh-userauth' on SSHServerTransport,0,127.0.0.1] 'admin' trying auth 'password'
2020-02-21 15:38:52+0530 [-] 'admin' failed auth 'password'
2020-02-21 15:38:52+0530 [-] unauthorized login:
2020-02-21 15:38:53+0530 [SSHService 'ssh-userauth' on SSHServerTransport,0,127.0.0.1] 'admin' trying auth 'password'
2020-02-21 15:38:54+0530 [-] 'admin' failed auth 'password'
2020-02-21 15:38:54+0530 [-] unauthorized login:
2020-02-21 15:38:54+0530 [SSHServerTransport,0,127.0.0.1] connection lost
```

**Fig 3 - SSH honeypot capturing attempts made by attacker**

## 5. CONCLUSIONS

The proposed Honeypot Modules will help us understand the behavior of attackers while they try to exploit given vulnerabilities. This opens door to many pro-active security measures that can be applied to actual system with important data. Since, this system doesn't rely much on already present open source tools for deploying honeypot, it is easier to deceive the attacker. The logged data are systematically stored and transferred to Base machine (in-case of honeypot deployed on virtual machine), and then cleared from the honeypot. Since the Admin has no need to directly interact with the honeypot, there will be less traces left behind for the attacker to get suspicious about. Further analysis of data can reveal more information about particular files and activities seen in the honeypots

## REFERENCES

[1] The HoneyNet Project, "The HoneyNet Project", <http://www.honeynet.org/>.

[2] K.R.Sekar, V.Gayathri, Gollapudi Anisha, K.S.Ravichandran, "Dynamic Honeypot Configuration for Intrusion Detection", IEEE Conference Record: 42666, May 2018.

[3] Daniel Fraunholz, Marc Zimmermann, Hans D. Schotten, "An Adaptive Honeypot Configuration, Deployment and Maintenance Strategy", 2017 19th International Conference on Advanced Communication Technology (ICACT), February 2017.

[4] Chao-Hsi Yeh and Chung-Huang Yang, "Design and Implementation of Honeypot Systems Based on Open-Source Software", 2008 IEEE International Conference on Intelligence and Security Informatics, June 2008.

[5] Jin-hak Park, Jang-won Choi, and Jung-suk Song, "How to Design Practical Client Honeypots Based on Virtual

Environment", 2016 11<sup>th</sup> Asia Joint Conference on Information Security, August 2016.

[6] Ionut, Cernica and Nirvana Popescu, "Wordpress honeypot module", 2018 IEEE 16th International Conference on Embedded and Ubiquitous Computing (EUC), October 2018

[7] Matthew L. Bringer, Christopher A. Chelmecki, and Hiroshi Fujinoki I. J., "A Survey: Recent Advances and Future Trends in Honeypot Research", "Computer Network and Information Security", 2012,

[8] Chuvakin, A. "Honeynets: High Value Security Data": Analysis of real attacks launched at a honeypot. Network Security, 2003,(8), 11-15.