ISSN:0975-9646

Analysis of Client Honeypots

Jhilam Biswas, Ashutosh

[1&2] *Student (8th semester), Department of Electronics and Communication*
Manipal Institute of Technology, Manipal
Karnataka, India

***Abstract*— With the growing popularity of Internet, security has become one of the most important concerns. Honeypot is a security resource whose value lies in being probed or attacked. It can be used to wave off the security issues arising nowadays. Also one can obtain a considerable amount of information about the attacker and his attacking methodologies. This paper includes brief discussion about different types of honeypot technology based on the site of the attack namely client honeypots and server honeypots. The server honeypots enable us to understand the server side attacks whereas client honeypots enable us understand the client side attacks. During the research on honeypot technologies, main focus was on the analysis of Client honeypots as they considerably are more vulnerable to the attacks. We will be discussing different types of attacks on client honeypots and different approach to detect and tackle them.**

***Keywords*—honeypot, client honeypot, 0day attacks, detection, obfuscation, effectiveness, integration**

I. Introduction

A large number of malware such as virus, Trojan horse are invariably present in the Internet. These malwares decode personal credentials, user names, passwords etc. Hence Internet security and privacy are a matter of great concern. Broadly, there are two kinds of attacks which persist: server side attacks and client-side attacks. Server-side attacks, aim at the servers that provide services to client machines. Client-side attacks target client applications, such as web browsers, email client and office software. These client applications interact with a server or file. Malicious client side attacks aim at attacking client application software. Antivirus systems can help to detect them, however, antivirus software is mostly based on virus signatures, so it is useful to detect known malware but it cannot effectively detect metamorphic or unknown malware. The appearance of honeypot can detect and obtain metamorphic and unknown malware.

Broadly, there are two kinds of honeypot, *server-side honeypot* and *client-side honeypot*. Server-side honeypot is the passive or traditional honeypot which provides with deep insight of server side attacks. In contrast to server honeypots, client honeypots provide the thorough knowledge of client side attacks; therefore they are also called as active Honeypots or Honeyclient. A further elaborate classification of honeypots is based on the intensity of interaction of the honeypots with attackers. They are classified as follows: *Low- Interaction Honeypots, Medium-Interaction Honeypots* and *High- Interaction Honeypots.*

- *Low-Interaction Honeypots*: Installation, configuration, maintenance and implementation are the easiest to perform in these honeypots. They limit the hacker to interact with pre-configured basic services like FTP and Telnet.
- *Medium-Interaction Honeypots:* In terms of interaction with attackers, this is a little more advanced than low-interaction honeypots, but a little less advanced than high-interaction honeypots. Medium-Interaction honeypots do not possess a real operating system, but the fake services provided are more technically sophisticated.
- *High-Interaction Honeypots:* These kinds of honeypots are time consuming to design, manage and maintain. Installing and maintaining this honeypot is a tedious task, but the valuable information and evidence gathered for analysis are enormous. The goal of a high interaction honeypot is to give the attacker an access to a real operating system where nothing is emulated or restricted. In another words, the sole purpose to build this honeypot is to let the attacker gain root or super user access to the machine.

This paper gives a deep insight into client honeypot characteristics. Starting with the various attacks in client honeypots, the paper goes on to discuss the objective, invisibility, detection issues and effectiveness of client honeypots. Each section of the paper analyzes different aspects of client honeypots.

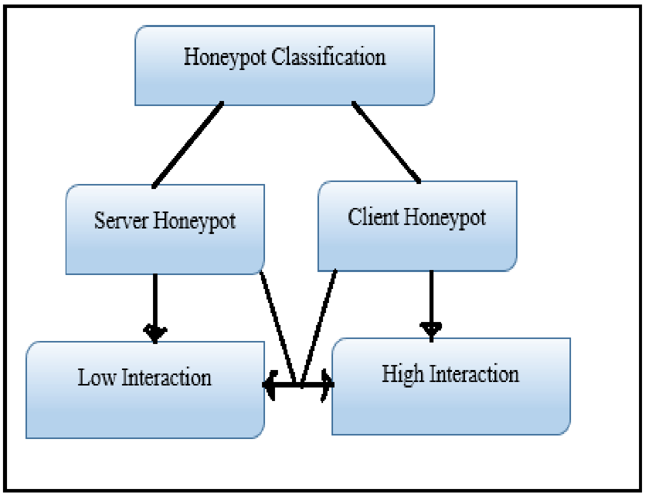

Fig 1: Honeypot classification

II. Attacks In Client Honeypots

One of the major type of attack that have been faced recently are client-side attacks. Client-side attacks are those which are launched against client user. In this type of attack, an attacker makes use of client application's vulnerabilities to take control of the client system by the malicious server. However, client side attacks are not limited only to the web browser vulnerabilities, but can occur on any client/server pairs, for example e-mail, adobe, instant messaging, multimedia streaming, etc. In this section we will discuss some issues relating to client-side threats: drive-by download, code obfuscation, phishing, and exploit servers.

A. Drive-by Download

A drive-by download is an attack where malicious servers can change the state of client machine without user's consent, which usually refers to the ability to download and install a program to client system without user's consent.

B. Code Beclouding

Attackers usually want to hide the exploit vector by using various encoding options to make the code vague and hard to understand. This technique aims for evading static detection tools such as IDSs, anti-virus tools, and firewall filters. Attacker can use beclouding to make the exploit code of JavaScript or VBscript unreadable during transportation from web server to client web browser. Attacker can use multiple layers to encode the code which make the code harder to be decoded.

C. Phishing

Phishing is an attack that combines between social engineering techniques and sophisticated attack vectors to acquire sensitive information or data from end users. Phisher typically try to lure her victim into clicking a URL pointing to a rogue page which does not have real exploit code; it just tries to deceive the client by entering credential to log to a fake site. Phisher typically uses redirection method to redirect the user to benign website.

III. Objective Of Client Honeypots

The prime motive of client honeypots is to identify and detect malicious activities across the Internet. The main functions of client honeypots are listed as follows:

- Client honeypot should enable real-time detection of attacks.
- Client honeypots should be able to detect all known and unknown threats against any client/server user application. Client honeypot should be able to check various URLs (images, executable files, html, scripts)
- Researches of client-side honeypot most aim at finding malicious website. Their data source comes from search engine or blacklist.
- Dynamic modification of the detection and security policy rules are the other features client honeypots should possess.

The general approach of client honeypots has the following two phases:

Crawling: In this phase, honeypot trace malicious websites. This phase is common for all client honeypots. Client-side honeypot needs data source. Crawlers are used to get URLs, which are later inspected in the honeypot. However, in general, the speed of crawler and the speed of the honeypot don't match. Normally, the crawler is faster than the behavior of opening processes in the honeypot. The behaviors in the honeypot are restricted by network bandwidth and performance of operating system. The process of open URL or file needs a certain amount of time. It is a bottleneck that affects system efficiency. As a result, two factors: the efficiency and coverage need consideration while designing a crawler.

Detecting: In this phase, honeypot identifies whether the queued sites are malicious or benign. Two approaches are used by client honeypots to detect malicious website:

1) *Pattern-Matching*: It is used by low interaction client honeypots. Low interaction client honeypots do not use fully functional operating system or web browser, instead they use simulated client. Low interaction client honeypots are often emulated web browsers, or web crawlers, which do have or only have limited abilities for attackers to interact with. Low interaction client honeypots send HTTP requests to the web server and detect malicious servers by applying signature based or heuristic methods on the server response for a fast analysis. They can directly detect the security violation by applying static signature or heuristics based method on web server's response. Thus, honeypot which use this method are quick in detecting attacks. However obfuscated attacks and other unimplemented attack types are likely to be missed by this detection method.

2) *Inegrity Check (State Changes Check):* It is used by high interaction client honeypots. High interaction client honeypot gives an attacker the oppurtunity to interact with real system rather than simulation. State changes check mechanism is a process that enables high interaction honeypots to detect security violations. Various client honeypots use approaches such as *HoneyClient, HoneyMonkey, and Capture* through which users access the suspicious sites of the presorted sites. Simultaneously, honeypot is monitored closely to detect occurance of any changes happening on the client system. Any change should give first insight that the system has been affected. Monitoring the followings can provide an indication whether the system has been exploited:

- File system activities.
- Registry entries.
- Processes.
- Network connections.
- Memory. This is the ultimate state change check.

However, such investigation is a tedious task. In order to achieve easier and faster implementation, current high interaction client honeypots (*HoneyClient, HoneyMonkey, Capture,*) are limited to monitor file system, registry entries, and processes. While using the integrity check method, more attention has to be given to avoid false positives. As an example, website may create cookies on the system to save some information like IP addresses, number of user visits to the website, etc. Thus, there should be some kind of exclude lists to prohibit false positives. Client honeypots operating in different networks can report the collected information findings to central sites that can correlate the data. The analysis synthesized enables the operator to keep track of collected information. Client honeypot can be run in virtual machines as *Vmware*. This is helpful to easily reset the machine to the clean state after a compromise on the system takes place.

IV. INVISIBILITY OF CLIENT HONEYPOTS

Client honeypots also show "invisibility" feature similar to server honeypots. Invisibility of client honeypot means preventing malicious websites from detecting the HTTP request that is sent by client honeypot. There are various issues relating to the invisibility in client honeypots. They are anti-crawling techniques, virtual environment detection, geo-location attacks, and IP blacklisting. All these issues will be discussed briefly.

A. Anti-Crawling Techniques

Automated crawlers allow malicious servers to fingerprint client honeypots. They normally send requests to resources which are invisible to human user and hence malicious web sites are able to detect crawlers. These websites then cease triggering. This problem is difficult to solve, thus the crawler should be refined to behave as identical to a browser as possible. Also, client honeypots would send many http requests to crawl websites. Anti-crawlers can be used to limit the amount of http request per IP. To mitigate this problem, intelligent crawling is used instead of crawling the whole web site, by looking for suspicious files as scripts and images.

B. Virtual Environment Detection

Use of *VMware* which is a virtual machine is a good choice for resetting the client honeypot after the system has been compromised. However, presence of virtual machines can be detected by attackers using several methods; a detection code can be incorporated in the exploit page to detect the virtual environment, and hence the malicious site can stop triggering the exploit, behave differently, block honeypot IP or do something else to keep hidden from detection.

C. Geo-location Attacks

Some attacks target users at specific geographical places. Attackers can find out the location of visitors, and then attack visitors in certain country or location. This issue can be handled by two approaches. First being implementation by allowing running honeypot across many different networks. Second approach can be using TOR service to run client honeypots behind various proxies.

D. IP Blacklisting

As malicious websites can detect presence of honeypots by various means, they can even block honeypot IP. It is not possible to hide client honeypot behavior completely, nevertheless it becomes hard to tackle this counter technique unless we operate honeypots using various ISPs. This will force the malicious site to block various ISPs, which deprives the attacker from a large percentage of his victims.

V. DETECTION ISSUES

Detection accuracy can be expressed by the rate at which false negatives and false positives occur. In the light of detection approaches discussed in the previous section, various detection problems have been discussed.

A. Human Behavior Simulation

The ultimate aim of client honeypots is to achieve the same behavior as humans which might not be possible due to absence of full features. This problem is more visible when dialog boxes pop up. A user is left with typically two options; either to accept the request or to deny it. The website might react differently depending on the user selection. It can even introduce dialog boxes and ask the user to fill out; the user then has to click the OK button to prove he/she is human and not a spam, and the web site drops a cookie to suppress the dialog box for future visits. In this case, user input is necessary to determine the server's response. Malicious website even can use CAPTCHA, which is a type of challenge-response test, to counter client honeypot. Using such response tests allow the malicious website to hide its malicious activity from client honeypot. At the same time, the end users will be deceived into believing that such website is trying to protect itself against a spam abuse.

B. Delayed Exploit

A delayed exploit is also an important issue that needs to be considered when implementing high interaction client honeypots. Sometimes, there may be a delay between initial infection and complete compromise. Low interaction honeypots will not be evaded by this delay, as they apply directly pattern-matching algorithm on the server's response.

This delay might be due to any or all of these three possible reasons:

- Downloading more malware: Generally, a web page first successfully exploits vulnerability in client application; then downloads a process to install more malware on the system. In such case the download process consumes some time. In the meantime, client honeypot has already accepted another web page.
- Logic Bombs: Logic bombs are exploits contained on a malicious web page in which the exploit triggers only after a given period of time and hence they also delay the compromise.
- User-Triggered Exploits. This scenario arises when the exploit needs a user action to trigger, such as mouse clicking. Correct pages are needed to be flagged so that user can get to know which page actually triggered and started the compromise.

C. Real-Time State Change Check

State changes checks can find out whether the web page has modified or changed something on client system. It can be performed periodically but there will be some delay. Such checks are unreliable, since installed malware may also install rootkit which may further hide subsequent malware instances, and thus make it hard to detect any changes. Therefore, the integrity check should be performed in real-time.

D. Attacks Against Internal Security Policies

In the present scenario high interaction client honeypots cannot detect exploits that do not make any persistent-state changes .Thus, present high interaction client honeypots might neglect attacks that are targeted at violating the internal security policies of the browser.

E. 0day and Beclouding Attacks

0day and beclouding attacks may not be detected by low interaction client honeypots. The reason being that the detection algorithm used by low interaction client honeypots depend on implementing signatures for known attacks. On the other hand, a high interaction client honeypots may detect these attacks if they try to make any change on the system state.

VI. EFFECTIVNESS OF CLIENT HONEYPOTS

Client honeypot is sometimes also referred as computer-human interaction tool. The effectiveness of client honeypots can be measured by the accuracy, reliability and completeness of the tasks a client honeypot performs. Broadly four factors are used to measure the effectiveness of a client honeypot. They are speed, detection, accuracy and invisibility. All the four factors are discussed briefly.

A. Speed

Speed of client honeypots can be expressed by number of sites that can be connected and inspected in a given time period. It has significance in describing the ability of the client honeypot to identify malicious servers quickly and to safeguard client user against them. The speed of client honeypots depends on various factors such as hardware, network connection, etc. It also depends on the client honeypot implementation which means more complex the implementation, slower the honeypots are. Detection algorithms play an important role in speeding up the detection process.

B. Detection Accuracy

Client honeypot should have high accuracy rate while detecting malicious servers. Detection accuracy can be measured by rate at which false positives (FPs) and false negatives (FNs) occur. With high interaction client honeypots, FP rate can be neglected; hence FN rate drives the accuracy of detection of malicious web pages. With low interaction client honeypots both FP and FN can be expected to exist. Hence both FP and FN has to be taken care of while detection. The ability of client honeypots to detect malicious contents in website is influenced by both the type of honeypot and also the operating environment characteristics.

C. Invisibility

The value of honeypots depend on the amount of data that it has gathered about the attacker after being probed. Unlike server honeypots, client honeypots do not use deception to lure malicious server to initiate attack. However, client honeypots should be kept undetectable by malicious websites which can cease exploits trigger. Thus keeping the client honeypots hidden allows it to gather more and more information and eventually identifying more attackers.

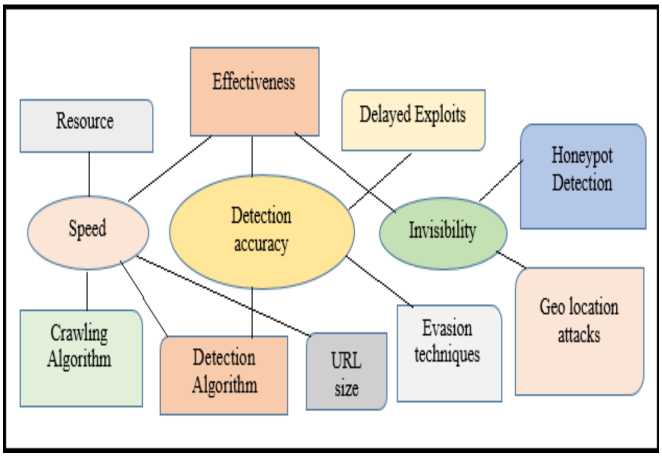

Fig 2: Effectiveness factors in client honeypots

VII. CLIENT HONEYPOT INTEGRATION

Till date, there is no compact client honeypot development that integrates various detection mechanism and capabilities of both low and high interaction honeypots that are available in public. More so, no open source client honeypots are integrated with commercial tools like web browsers to provide real-time security for the end-user. It is a herculean task to deploy client honeypot that would allow bulk processing of URLs acquired from different sources with different confidence and priority levels. Client honeypots are a budding new technology used to secure client-side system sturdily. Thus, they have to deal with large web space, various web technologies, evasion techniques, various browser behavior and strong integration with operating system. However, client honeypots still being a developing, immature technology, various tools are not available and open for public research. Client honeypots need to operate as a service, rather than just a research tool to inspect some contents. Thus, there is a need for elastic frameworks which allow easier integration with the latest client honeypots, and enable to analyze the detection of large space of attacks trends.

VIII. DISCUSSION

In this current generation of the Internet, a large number of malware exploit vulnerabilities in client applications. This is the main motivation of developing client honeypots. Lately, client honeypots have been greatly used in various areas in network security. They can be used a useful tool to evaluate websites by examining websites contents which helps to identify malicious sites, applications, files, etc. They can be effectively used to evaluate and test client user applications. Web browser being the most preferable target for attackers, researchers use client honeypots to test web browser security. To add to the list, client honeypots are able to identify and detect various client-side attacks. Integrity checks can be effective in discovering new threats; this can be of great help to change the configuration and security policy to prevent such attacks. Lastly, client honeypots can help in mapping malicious neighborhoods because malicious websites typically redirect to another malicious web sites [20].

Implementation of client honeypot should depend on the goals of honeypots and circumstances of operating.

Choosing honeypots specification are required to meet these goals as they expose the full functional spectrum of a computer system for the attacker to interact with and therefore allow for collection of the desired data [7].

Attacker uses various hiding and evading so that they are not detected. Thus, developing efficient protection mechanisms against malicious websites attacks requires effective analysis tools which allow studying current attacks and foreseen future attacks. For instance, many malicious sites attack a client side only once in a given timeframe, then all subsequent requests are redirected to harmless sites such as search engines. This aims to hold down the analysis and keep hidden from further tracking. Therefore, an analysis tool should have the ability to record and reply all requests and responses involved in a detected attack. Till now, open source client honeypot systems have not leveraged the benefits of using both low and high interaction solutions together. With combining low interaction honeypot and high interaction honeypot, a scalable architecture can be achieved at constant levels of false negatives. Low interaction components can be used to search quickly for potential malicious sites, tag them as suspicious and only then hand them over to the high interaction component for detailed analysis. To check if the low interaction component is missing some attacks, a small percentage of URLs can be passed over to the high interaction component and the results from both components are compared.[7], [11].Theoretically, high interaction components can be used to extract signatures of threats, which can then be used in low interaction components.

IX. Conclusions

Client honeypot is a new technology that aims to overcome the weakness of server honeypots and other security tools in dealing with client side attacks. Client honeypots use two approaches to detect client-side attacks: pattern matching and integrity check. Each approach has benefits and shortcomings.

There are several detection issues with the case of client honeypots which are needed to be addressed. Nevertheless, some implementations need to be thought of with regard to client honeypots so as to increase its effectiveness be it in terms speed or detection accuracy or both. Invisibility is also a big issue regarding client honeypots which needs to be dealt with. Current client honeypots are still in the developing phase. They have various shortcomings relating to their inability to detect and evade various attacks by malicious attackers. In this paper, we introduced factors to measure the effectiveness of client honeypots: speed, detection accuracy and invisibility.

This is a review paper that gives an overview on client honeypots. It talks in depth about various aspects of client honeypots like attacks, objective, invisibility, detection and effectiveness of client honeypots. The concept of honeypots being a new technology, they come in to help in three ways that is prevention, detection and how users react to an attack. Not only do client honeypots become cost-effective to deploy and maintain, but they also have a better integration into the organization network. This paper can be of immense help to the novice as well as the experienced in the field of network security with the help of client honeypots.

Acknowledgement

Sincere thanks to the professors of the Department of Electronics and Communication and the department of Information and Communication Technology for providing excellent laboratory facilities to carry out the research study and gather valuable knowledge on honeypots.

References

[1] Ren Liu. China virus status & Internet Security Report in 2006.20070201.http://www.donews.com/Content/200702/eda7daf7970448608b2881d97c9a1868.shtm.

[2] VMware Server.2007.http://www.vmware.com/download/server

[3] An efficient approach to collect malware. In Proceedings of 9th Symposium on Recent Advances in Intrusion Detection (RAID'06), 2006.

[4] Paul Baecher, Markus Koetter, Thorsten Holz, Maximillian Dornseif, and Felix C. Freiling. The nepenthes platform: An efficient approach to collect malware. In Proceedings of 9th Symposium on Recent Advances in Intrusion Detection (RAID'06), 2006.

[5] Jan Goebel, Thorsten Holz, and Carsten Willems. Measurement and Analysis of Autonomous Spreading Malware in a University Environment. In Proceeding of 4th Conference on Detection of Intrusions & Malware, and Vulnerability Assessment (DIMVA'07), 2007.

[6] Moheeb Abu Rajab, Jay Zarfoss, Fabian Monrose, and Andreas Terzis. A multifaceted approach to understanding the botnet phenomenon. In Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement, ACM Press, New York, NY, USA, 2006, pp. 41–52

[7] J. Zhuge, T. Holz, X. Han, C. Song, and W. Zou. Collecting autonomous spreading malware using high-interaction honeypots. In Proceedings of ICICS'07, 2007.

[8] Roger A. Grimes.Tracking malware with honeyclients. 2006-04-14.http://www.Infoword.com/article/06/04/14/77378_16OPsecadvise_1.html.

[9] Kathy Wang.Using Honeyclient to Detect New Attacks. In Proceedings of RECON 2005, Crowne Plaza Montreal, Canada, 2005.

[10] Y.M. Wang, D. Beck, X. Jiang, R. Roussev, C. Verbowski,S. Chen, and S. T. King. Automated web patrol with strider honeymonkeys: Finding web sites that exploit browser vulnerabilities. In Proceedings of 13th Network and Distributed System Security Symposium (NDSS'06), 2006.

[11] Krisztion Piller,Sebastian Wolfgarten.Honeymonkeys-Chasing hackers with a bunch of monkeys. 2005-12-30.http://events.ccc.de/congress/2005/fahrplan/attachments/686 slides_honeymonkeys.pdf

[12] Websense,Inc.Overview of Our Investigative Process.200507http://www.websense.com/docs/WhitePapers/WSLabsOverview.pdf.

[13] The Honeynet Project. Know Your Enemy: Malicious Web Servers, August 2007. http://www.honeynet.org/papers/

[14] C. Clementson," Client-Side Threats and a Honeyclient-Based Defense Mechanism, Honeyscout", Master's Thesis, Linköping University Electronic Press, 2009.

[15] C. Seifert, "Know Your Enemy: Behind the Scenes of Malicious WebServers", The Honeynet Project, 2008 http://www.honeynet.org/papers/wek

[16] C. Seifert, Improving Detection Speed and Accuracy with Hybrid Client Honeypots, Victoria University of Wellington, PhD Thesis, 2008.

[17] C. Seifert, R. Steenson, T. Holz, Y. Bing, and M. A. Davis, "Know your enemy: Malicious web servers." The HoneynetProject2007. http://www.honeynet.org/papers/mws/

[18] M. Pennock, S. Lawrence, and L. C. Giles, "Methods for Sampling Pages Uniformly from the World Wide Web",In AAAI FallSymposium on Using Uncertainty within Computation (NorthFalmouth 2001), pp 121–128.