RESEARCH ARTICLE OPEN ACCESS

Client Honeypot Based Malware Program Detection Embedded Into Web Pages

Supinder Kaur*, Harpreet Kaur**
*(Computer Science and Engineering, Sant Baba Bhag Singh Insitute of Engineering and Technology – Punjab Technology University)
** (Computer Science and Engineering,Sant Baba Bhag Singh Insitute of Engineering and Technology –Punjab Technology University)

ABSTRACT
In today's world where internet is hosting major resources of this world, the malware programs embedded into web pages have become a severe threat in today's internet which launches the client side attacks by exploiting the browser based vulnerabilities. With the improvement of software security, vulnerabilities based attacks declined whereas the attacks based on the client side application is being increasing which becoming the major threat to the current resources hosted on the internet. The spread of malware using these software vulnerabilities has become a severe threat to today's Internet. In allusion to this kind of threat, detection of such client side attacks is desperately required to make the internet more secure. Here in this research paper, we show and describe the detection of malicious programs linked with web pages with the help of client honeypots. Client Honeypots are security devices that detect these malicious web pages on a network. We are ensuring that most of software tools used in our implementation is open source.
Keywords **-** Honeypots, Client Honeypots, Network Intrusion Detection System, Network Security.

I. INTRODUCTION

The volumes of applications hosted on the internet are growing exponentially and internet is becoming the most popular medium of communication and global information reservoir. In the recent years, the new trends which targeting the client side vulnerabilities are increasing and thereby there is a need of new tools and techniques to defend these new kinds of attacks on the cyber space. To supplement these new kinds of attacks, many security technologies such as Intrusion detection system (IDS) and firewall exist as well honeypot is also one kind of new technology to defend against these attacks. In the area of information security, the term of honeypot refers to a closely monitored computing resource that we want to be probed, attacked or compromised. Lance Spitzner defines a honeypot to be "a resource whose value is in being probed, attacked or compromised." [1].

Basically there are two kinds of attacks: server side attacks and client side attacks Server-side attacks, such as attacks using Windows RPC service vulnerabilities, aim at the servers that provide services clients can interact with. Client-side attacks are those attacks that target vulnerabilities of client applications, such as web browsers, email client and office software [3].

There are two kinds of honeypot, server-side honeypot and client-side honeypot. Server-side honeypot is the traditional honeypot. This kind of honeypot must have some vulnerable service, and attacker can detect them, so they are passive honeypots. The concept of client-side honeypot [2] was brought forward by Lance Spitzner. Client-side honeypot aims at vulnerabilities of client applications. It needs a data source, and visits the data source actively, and detects all activities to judge if it is safe. Client-side honeypot actively "requests" to accept attack. This kind of honeypot actively acquires malware spreading through client application software which traditional honeypot can't get [3]

The traditional server based honeypots are not able to detect the client side attacks which exploit the client side applications such web browsers. The server side honeypots are exposing the vulnerabilities to be exploited so that those vulnerabilities is to be exploited by the attacker and the intent of the attackers can be caught. Whereas to detect the client side attacks, there is a need to actively interact with the malicious website using client side applications and collect the attack data. A new type of honeypot is therefore needed: the client honeypot. Client honeypots crawl the network, interact with servers, and classify servers with respect to their malicious nature.

The main differences between a client-side honeypot and traditional honeypot are [16]:

- Client-side: it simulates/drives client-side software and does not expose server based services to be attacked.
- Active: it cannot lure attacks to itself, but rather it must actively interact with remote servers to be attacked.

- identify: whereas all accesses to the traditional honeypot are malicious, the client-side honeypot must discern which server is malicious and which is benign

The format of the remaining paper is: section 2, defines and explains the technology that has been employed and discusses the client honeypots in brief and other detection approaches. Section 3 deliberates the framework design and discusses our detailed design of the implemented system. Section 4 discusses conclusion and future work of the research problem.

II. BACKGROUND AND MOTIVATION

A Honeypots

As compare to traditional security mechanisms, honeypots are very different than them, as such they are not providing the security to the network but they collect the attacks and threats present in the network. They are security resources that have no production value; no person or resource should be communicating with them. Any interactions with the honeypot placed in the network are suspected by default. Any traffic sent to the honeypot is most likely a probe, scan, or attack. Any traffic initiated by the honeypot means the system has most likely been compromised and the attacker is making outbound connections [4].

Honeypots are the types of resources whose values are being attacked or probed by the attacker. Generally honeypots are used for gathering the intelligent information about the attacker or black hat community targeting the intrenet cyber space. In terms of network security, it is well accepted that honeypots play a biggest role including other security devices such firewall, IDS etc. To make the network full proof against attacks, honeypots play the major role to protect the network from unknown and unclassified kind of attacks. Honeypots also play an important role in protecting servers and hosts against attacks targeted at resources available on a production network by directing attacks to decoy systems. When placed with other technologies such as intrusion detection system, intrusion prevention system, firewalls, honeypots become highly effective tools against attacks performed by black hat community.

Most of the network security devices such as firewall, intrusion detection system, are based on pre-defined signatures embedded into them to detect the attacks and prevent the network from these kind of attacks but what will happen in case of zero day attacks when there are no signatures exists in their database, honeypots plays biggest roles here to tighten the networks from these kind of nknown attacks. Honeypots are effective tools to detect internal attacks and propagation of worms within an internal network which other tools such as firewalls fail to achieve

Usually Honeypots are very different than other network security devices because they are not directly providing any kind of security to the organizational network but they give us the useful information to study the behavior of attackers so that we can take the remedial actions further.

Honeypots can be classified as per the attack classes and targeted attacks such as server side attacks and client side attacks. Classification of Honeypots can be as server honeypots and client honeypots. Server Honeypots which provide us the deep knowledge of server side attacks, which are a kind of passive honeypots. In contrast to server honeypots, client honeypots provide us the deep knowledge of client side attacks; therefore they are also called as active Honeypots or Honeyclient. Both of the Honeypot Technology has emerged as a widely research areas in the field of cyber security.

Client Honeypot

In recent times, black hat community has mainly targeting client side applications such internet browser, pdf, media player etc. The client honeypot is new concept [21] and is quite different than server honeypots. In case of active honeypot, client honeypot acts as client and actively visit the website to see whether the attack has happened or not. Following diagram depicts high interaction and low interaction client honeypots.

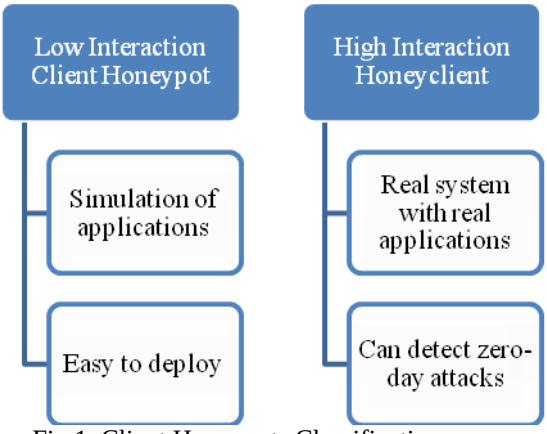


Fig 1. Client Honeypots Classifications

III. MALWARE PROGRAM DETECTION BASED ON CLIENT HONEYPOTS

A honeypot is a security technology that provides organisations with a way to catch viruses, malware or attackers, as well as acting as an alarm system that can discover attempts to attack a network. Honeypot technology is defined as a 'security resource whose value lies in being probed, attacked or compromised' [1]. There are two main types of honeypot: passive and active. The passive honeypot is a technology that passively waits for attacks in order to detect them, while the active honeypot, also called a client honeypot, interacts with a target web page to identify and determine its potential effect on the browser or operating system.[Honeyware]

The advantages of using honeypots are:

- They collect and log data with small amounts of false positives and negatives as it logged data only from the target web page.

- The data has a high value.

The disadvantages of using honeypots are:

- The risk after a system is compromised, which could be an attack on another system on the same network or outside.
- Highly skilled people and their time are needed to operate and analyse the data from honeypots.

The Architecture of client honeypots:

The architecture of client honeypot is shown in figure 2, divided mainly into three parts know as queuer, the client and the analysis engine. A client honeypot is composed of three components. The first component, a queuer, is responsible for creating a list of servers for the client to visit. This list can be created, for example, through crawling. The second component is the client itself, which is able to make a request to servers identified by the queuer. After the interaction with the server has taken place, the third component, an analysis engine, is responsible for determining whether an attack has taken place on the client honeypot [honeyware].In addition to these components, client honeypots are usually equipped with some sort of containment strategy to prevent successful attacks from spreading beyond the client honeypot. This is usually achieved through the use of firewalls and virtual machine sandboxes. Analogous to traditional server honeypots, client honeypots are mainly classified by their interaction level: high or low; which denotes the level of functional interaction the server can utilize on the client honeypot. In addition to this there are also newly hybrid approaches which denotes the usage of both high and low interaction detection techniques.

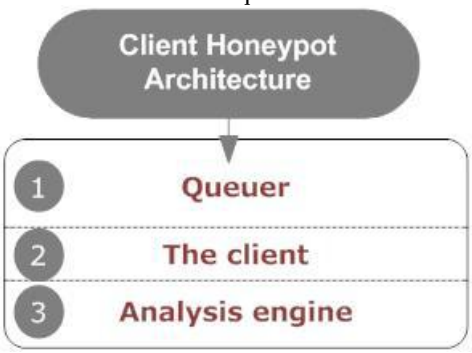


Figure 2: Client Honeypot Architecture

Study of existing proof-of-concept Client Honeypots solutions:

- High Interaction Client Honeypots

The High interaction client honeypots are real systems with real applications installed on them; the websites are being browsed using real browsers and plug-ins. As such, no functional limitations (besides the containment strategy) exist on high interaction client honeypots. Attacks on high interaction client honeypots are detected via inspection of the state of the system after a server has been interacted with. The detection of changes to the client honeypot may indicate the occurrence of an attack against that has exploited a vulnerability of the client. An example of such a change is the presence of a new or altered file.

1. Capture-HPC

Capture is a high interaction client honeypot developed by researchers at Victoria University of Wellington, NZ [5]. Capture differs from existing client honeypots in various ways. First, it is designed to be fast. State changes are being detected using an event based model allowing reacting to state changes as they occur. Second, Capture is designed to be scalable. A central Capture server is able to control numerous clients across a network. Third, Capture is supposed to be a framework that allows utilizing different clients. The initial version of Capture supports Internet Explorer, but the current version supports all major browsers (Internet Explorer, Firefox, Opera, Safari) as well as other HTTP aware client applications, such as office applications and media players.

2. HoneyClient

HoneyClient[6] is a web browser based (IE/FireFox) high interaction client honeypot designed by Kathy Wang in 2004 and subsequently developed at MITRE. It was the first open source client honeypot and is a mix of Perl, C++, and Ruby. HoneyClient is state-based and detects attacks on Windows clients by monitoring files, process events, and registry entries. It has integrated the Capture-HPC real-time integrity checker to perform this detection. HoneyClient also contains a crawler, so it can be seeded with a list of initial URLs from which to start and can then continue to traverse web sites in search of client-side malware.

3. HoneyMonkey

HoneyMonkey[7] is a web browser based (IE) high interaction client honeypot implemented by Microsoft in 2005. It is not available for download. HoneyMonkey is state based and detects attacks on clients by monitoring files, registry, and processes. A unique characteristic of HoneyMonkey is its layered approach to interacting with servers in order to identify zero-day exploits. HoneyMonkey initially crawls the web with a vulnerable configuration. Once an attack has been identified, the server is reexamined with a fully patched configuration. If the attack is still detected, one can conclude that the attack utilizes an exploit for which no patch has been publicly released yet and therefore is quite dangerous.

4. SHELIA

Shelia[8] is a high interaction client honeypot developed by Joan Robert Rocaspana at Vrije Universiteit Amsterdam. It integrates with an email reader and processes each email it receives (URLs & attachments). Depending on the type of URL or attachment received, it opens a different client application (e.g. browser, office application, etc.) It monitors whether executable instructions are executed

in data area of memory (which would indicate a buffer overflow exploit has been triggered). With such an approach, SHELIA is not only able to detect exploits, but is able to actually ward off exploits from triggering.

5. UW Spycrawler

The Spycrawler[9] developed at the University of Washington is yet another browser based (Mozilla) high interaction client honeypot developed by Moshchuk et al. in 2005. This client honeypot is not available for download. The Spycrawler is state based and detects attacks on clients by monitoring files, processes, registry, and browser crashes. Spycrawlers detection mechanism is event based. Further, it increases the passage of time of the virtual machine the Spycrawler is operating in to overcome (or rather reduce the impact of) time bombs.

6. Web Exploit Finder

WEF [10] is an implementation of an automatic drive-by-download – detection in a virtualized environment, developed by Thomas Müller, Benjamin Mack and Mehmet Arziman, three students from the Hochschule der Medien (HdM), Stuttgart during the summer term in 2006. WEF can be used as an active HoneyNet with complete virtualization architecture underneath for rollbacks of compromised virtualized machines.

- ***Low Interaction Honeyclient***

Low interaction client honeypots differ from high interaction client honeypots in that they do not utilize an entire real system, but rather use lightweight or simulated clients to interact with the server. Responses from servers are examined directly to assess whether an attack has taken place. This could be done, for example, by examining the response for the presence of malicious strings.

1. HoneyC

HoneyC [11] is a low interaction client honeypot developed at Victoria University of Wellington by Christian Seifert in 2006. HoneyC is a platform independent open source framework written in Ruby. It currently concentrates driving a web browser simulator to interact with servers. Malicious servers are detected by statically examining the web server's response for malicious strings through the usage of Snort signatures.

2. Monkey-Spider

Monkey-Spider [12] is a low interaction Honeyclient that utilizes many existing freely available software systems. The modular framework of Monkey-Spider consists of following components:

- **Queue/Seed Generation**

Sources of initial set of seed URLs

  - **Web Search Seeding**

Web Search APIs of three search engines, namely Google, Yahoo and MSN are considered with commonly used topics or keywords.

  - **SpamTrap Seeding**

URLs extracted from spam mails (collected from e-mail account targeted for spammers) are also enlisted.

  - **BlackList Seeding**

A tool is implemented that automatically downloads blacklist from major blacklist providers and use them as seed for the crawler.

- **Web Crawling**

Heritrix Crawler is integrated into the Monkey-spider architecture with two predefined parameters

  - Maximum Link hops (Count of links to be included in Crawl)
  - Maximum transitive hops (Count of URLs extracted from seed URLs)

- **Content/Malware Analysis**

  - **Static Analysis**

The downloaded contents (in ARC format) of the URL are scanned by Clamav antivirus and alerts generated based on pattern matching. The nomenclature is provided for the downloaded binary. A separate repository of malware is created.

  - **Dynamic Analysis**

The analysis of the suspicious binaries with automatic, behavior-based malware analysis tool like CWSandbox is performed.

3. PhoneyC

PhoneyC [13] is a low-interaction Honeyclient, which is designed to mimic the behavior of a user-driven network client application, such as web browser, and can be exploited by an attacker's content. PhoneyC is a virtual Honeyclient, meaning that it is not real application but rather an emulated client. By using dynamic analysis, PhoneyC performs dynamic analysis of Javascript and Visual Basic Script to remove the obfuscation from malicious pages. Furthermore, PhoneyC emulates specific vulnerabilities to pinpoint the attack vector

PhoneyC supports the various client emulations, dynamic languages such as Javascript, and mimics ActiveX add-ons. To analyze the malicious content, obfuscated or encrypted Javascript is decoded and reanalyzed, mimicking what a real browser would do with such content.

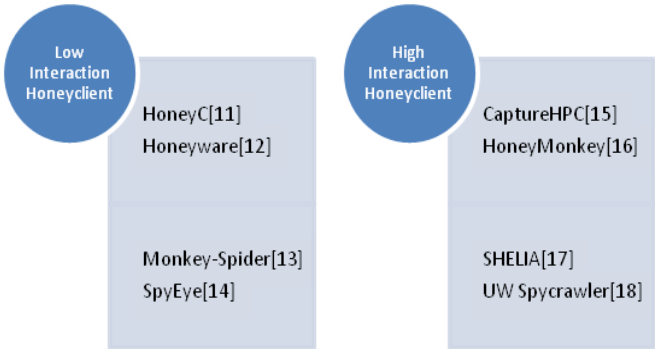
4. SpyBye

SpyBye [14] is a low interaction client honeypot developed by Niels Provos. SpyBye allows a web master to determine whether a web site is malicious by a set of heuristics and scanning of content against the ClamAV.

5. HoneyWare

Honeyware [15] is a low-interaction tool that interacts with a given URL by simulating a specific web-browser through its user-agent field and

downloading the response of the target web site. The response is then examined using the scan engine provided by Honeyware which can detect known malicious signatures.



IV. Problem Statement

The problem addressed in this paper is detection amd extraction of malicious web programs embedded into malicious websites based on the client honeypot technologies. Our main objective to address the problem of identifications of malicious website based on the maliciousness into that websites. This solve the problem of determination of the trust weather we should visit the website or not. Client honeypot is technologies with the help of which we can be able to determine the client side attacks by actually browsing the website in real environment in the case of high interaction client honeypots as well as by emulating the environment in the case of low interaction client honeypots technologies.

During the implementation, following are the steps of the generic algorithms used in URL extraction and feed them into virtual machine for actively visiting of them:

1. List of URL to be visited say N number of URLs
2. Save the list of URLs into database
3. Fetch the URL one by one from Mysql database

Figure 3: Process of Malicious Program detection.

Figure 3 depicts the process of the system in which list of websites have been submitted for visitation using actual real browsers on a client honeypots. Using clean machine of client honeypots, a website is to be visited in that clean environment and if after the visitations of the website, we are taking the snapshot of that machine and save the logs generated during the website visitations.

The system logs generated on the client machine are further analyzed either manually or by using 3rd party analysis tools such as popular anti-viruses to determine the infections on those collected logs.

The main aim of client honeypots is detect malicious websites by actively visiting them. In this research, the infected websites have been visited by using real system & real applications. With the help of real client applications, the websites will be actively visited and all the interactions of client machines with the malicious server will be logged. If there is any malicious activity performed during the visitation of websites which can be an indication of maliciousness of websites.

V. Conclusion

In this paper, we only present the literature study of the existing solution of client honeypots for detection of malicious websites and found that most of the solution is either not available for public users and closely bound, thereby we propose the system which is able to detect the malware programs with the help of client honeypot as well as by applying the intelligent forensic investigation of the collected network PCAP data.

VI. Acknowledgements

We would like to sincerely thank Ms. Harpreet Kaur for her contribution and help in writing this paper.

References

[1] Honeynet Project, Know Your Enemy: Defining Virtual Honey/nets. 2003. URL: http://old.honeynet.org/papers/virtual/
[2] R. Danford, ―2nd Generation Honeyclients‖ , SANS Internet Storm Center,2006 http://handlers.dshield.org/rdanford/pub/Honeyclients_Danford_SANS 06.pdf
[3] Xiaoyan Sun, Yang Wang, Jie Ren, Yuefei Zhu and Shengli Liu, "Collecting Internet Malware Based on Client-side Honeypot", 9th IEEE International Conference for Young Computer Scientists (ICVCS 2008), pp. 1493 – 1498, 2008.
[4] L. Spitzner, Honeypots: Tracking Hackers. Addison Wesley, 2002.
[5] https://projects.honeynet.org/capture-hpc
[6] http://www.honeyclient.org/trac
[7] http://en.wikipedia.org/wiki/HoneyMonkey
[8] http://www.cs.vu.nl/~herbertb/misc/shelia/
[9] en.wikipedia.org/wiki/Client_honeypot
[10] http://www.xnos.org/security/overview.html
[11] https://projects.honeynet.org/honeyc
[12] http://monkeyspider.sourceforge.net/

[13] http://code.google.com/p/phoneyc/
[14] http://www.spybye.org/
[15] Yaser Alosefer , Omer Rana, Honeyware: a web-based low interaction client honeypot, Software Testing, Verification, and Validation Workshops (ICSTW), 2010 Third International Conference on.
[16] http://www.honeynet.org/node/158
[17] Ramaswamy, C. and R. Sandhu. (1998), Role-based access control features in commercial database management systems: *Citeseer*.
[18] Skoudis, E., and Zeltser, L., *"Malware: Fighting Malicious Code"*, Prentice Hall,2003, Page 3, ISBN = 978-0131014053.
[19] [Provos, N., McNamee, D., Mavrommatis, D. W., K and Modadugu, N., *the Ghost In The Browser Analysis of Web-based Malware*. 2007. [Online]. Available at:http://www.usenix.org/events/hotbots07/tech/full_papers/provos/provos.pdf [Accessed 11 Feb 2009]
[20] Secure Browsing | Malware Protection | Trustwave https://www.trustwave.com/securebrowsing/
[21] Google Safe Browsing www.google.com/tools/firefox/safebrowsing/
[22] Detection and Analysis of Drive-by-Download Attacks and Malicious JavaScript Code www.cs.ucsb.edu/~vigna/.../2010_cova_kruegel_vigna_Wepawet.pdf
[23] Prophiler: A Fast Filter for the Large-Scale Detection of Malicious Web Pages www.cs.ucsb.edu/.../2011_canali_cova_kruegel_vigna_Prophiler.pdf
[24] Secunia. (2010), "Factsheets by Windows Operating System - 2010". 20 - March - 2011]; Availablefrom: http://secunia.com/resources/factsheets/2010_win_os/.
[25] PcPitstop. (2010), "The State of PC Security". 20 - December 2010]; Available from:http://techtalk.pcpitstop.com/2010/05/13/the-state-of-pc-security/.
[26] Secunia. (2010), "Secunia Yearly Report - 2010". Available from:secunia.com/gfx/pdf/Secunia_Yearly_Report_2010.pdf.
[27] Secunia. (2010), "Research Reports, Factsheet by Browser - 2010". [Cited 2011 5 - January]; Available from: http://secunia.com/resources/factsheets/2010_browsers/.
[28] VirtualBox. (2004). Sun VirtualBox® User Manual. Available:http://www.virtualbox.org/manual/UserManual.html Last accessed 20 July 2008.
[29] Sanjeev Kumar, at al, Hybrid Honeypot Framework for Malware Collection and Analysis, ICIIS-2012, IIT Chennai
[30] www.honeyclient.org
[31] en.wikipedia.org/wiki/Client_honeypot
[32] www.honeynet.org

Journal Papers:

[33] Masood Mansoori and Ray Hunt, *International Journal of Network Security & Its Applications (IJNSA), Vol.3, No.5, Sep 2011, AN ISP BASED NOTIFICATION AND DETECTION SYSTEM TO MAXIMIZE EFFICIENCY OF CLIENT HONEYPOTS IN PROTECTION OF END users*

Thesis:

[34] Yaser Alosefer, Analysing Web-based Malware Behaviour through Client Honeypots Cardiff University School of Computer Science & Informatics, Feb-2012