ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology
Volume 1, Issue 4, June 2012

Design and Implementation of Virtual Client Honeypot

Himani Gupta, Gurpal Singh Chhabra
School of Mathematics and Computer Applications, Thapar University, Patiala
himanigupta4@gmail.com, gurpalsingh123@gmail.com

Abstract— Computers security has become a major issue in many organization. There are different solutions to response to this needs but they remain insufficient to truly secure network. Honeypot is used in the area of computer and Internet Security. It is resource which is intended to be attacked and comprised to gain more information about the attacker and their attack techniques. Compared to an intrusion detection system, Honeypots have the big advantage that they do not generate false alerts as all traffic is suspicious, because no productive components are running on the system. Client Honeypot is a honeypot actively searches for malicious sites on the web. In this paper, we design and implement virtual Client Honeypot to collect the internet malwares.

Index Terms—**Intrusion detection system; Honeypots; Honeyclients; client-side attacks; malware; crawler;**

I. INTRODUCTION

Malwares have become a major threat to the internet as their occurrence in the internet had significantly increased in past few years. In response to this increasing malware attacks, honeypots has emerged as one of the popular practical defence technique. The Honeypots are the information system resources capable to attract, capture and collect malware attacks.

While the fight is ongoing on the Internet between blackhats and whitehats, attackers have started to transfer the battlefield to the client user; as they believe the client applications are more likely to have security breaches and vulnerabilities. Client user has become the weakest link in the network security chain, and since the security chain is only robust as its weakest link, we need to detect attacks against client side to protect the whole security system [1].

Traditional honeypots are servers (or devices that expose server services) that wait passively to be attacked. **Client Honeypots** are active security devices in search of malicious servers that attack clients. The client honeypot poses as a client and interacts with the server to examine whether an attack has occurred. Often the focus of client honeypots is on web browsers, but any client that interacts with servers can be part of a client honeypot (for example ftp, ssh, email, etc.). There are several terms that are used to describe client honeypots. Besides client honeypot, which is the generic classification, honeyclient is the other term that is generally used and accepted. The concept of client honeypots was firstly articulated by Lance Spitzner (2004). Later several client honeypots were developed: Honeyclient; HoneyMonkey [2]; HoneyC [3]; and Capture [4]. *HoneyClient* was the first open source client honeypot, which was developed in 2004 by K. Wang [5], and subsequently developed at MITRE. However, in spite of the continuous progress with client honeypots technology, they are still immature technology. In this paper, we will study threat against client user, Goals of Client Honeypot, Architecture of Client Honeypot, Functional Diagram of Virtual Client Honeypot and Comparison of Honeyclient with IDS.

II. THREATS AGAINST CLIENT USERS

One of the new major attack types that we are faced recently are client-side attacks. Client-side attacks refer to the attacks launched in opposition to client user. In this type of attacks, an attacker uses client application vulnerability to take control of client system by malicious server. A typical target is web browser. However, these attacks can occur on any client/server pairs such as email, instant messaging, FTP, multimedia streaming, etc[6] In this section we will discuss some issues relating to client-side threats: drive-by download, code obfuscation, phishing and Typo-squatting.

A. Drive-by download

A very effective way to infect a victim's machine is to exploit vulnerabilities and execute malware without the user noticing such actions and without any user interaction. A drive-by download usually initiates a number of downloads and installations, after the successful exploitation of a vulnerability in the browser or one of its plug-ins. The executables are malware used for different purposes that cause changes to the system state and affect the user's machine depending on their type. The main changes are observed in the registry, the system's processes and network's activity. [7] Once a user visits a page that launches drive-by attacks, a common first step in the attack is to perform fingerprinting of the visitor's browser. To this end, a script collects information about the browser version and language, operating system version, or enumerates the installed plug-ins.

B. *Code Obfuscation*

Obfuscation means using encoding to make the code ambiguous, and more difficult to interpret. Hiding the exploit vector is an effective way of evading signature-based detection systems such as virus scanners and filtering firewalls. Criminals use code obfuscation to make the malicious JavaScript or VBscript unreadable during transportation from the web server to the browser. These scripts are decoded and interpreted by the browser. [8]

C. *Phishing*

Phishing is an attack combines between social engineering techniques and sophisticated attack vectors to harvest financial information or sensitive data from end users. Phisher typically tries to lure her victim into clicking a URL pointing to a rogue page In phishing, users could be easily tricked into submitting their username and password into fraudulent web sites whose appearance look similar to the genuine one. [9]

D. *Typo-squatting*

Typo-squatting refers to the practice of registering domain names that are typo variations of popular websites, which usually host websites with significant traffic. The individuals or organizations who register ***typo-squatting domains*** (or ***typo domains***) are referred to as ***typosquatters***. Some major typo-squatters are known to have registered thousands or more of typo domains.

III. GOALS OF CLIENT HONEYPOT

The ultimate goal of client honeypots is to detect and identify any malicious activity coming from the Internet. This ideal case of client honeypot can be summarized as follows:

1. Client honeypot should detect any known and unknown threats against any client user application. Application can be any server/client based application. Client honeypot should be able to check various URLs (images, executable files, html, scripts, etc). Ideal client honeypot has rate zero false positive.
2. Client honeypot should detect the attacks in real-time.
3. Client honeypot should be able to dynamically modify the detection and security policy rules to fit the current situation. [10]

IV. CLIENT HONEYPOT

Client honeypots are client-side, they simulates drives client-side software and do not expose services to be attacked. Client honeypots typically are active, they actively initiate interact with remote servers to be attacked. The client-side honeypot must recognize which server is malicious and which is benign. Honeyclient is an active honeypot that mimics, either manually or automatically, the normal series of steps a regular user would make when visiting various websites. [11] The intended goal of honeyclients is to identify malicious websites which target the client application vulnerabilities.

V. ARCHITECTURE OF CLIENT HONEYPOT

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 requests 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.

The Active honeypot architecture is divided into following three modules:

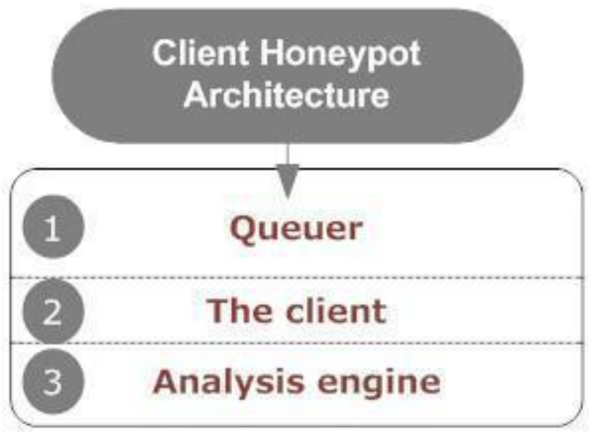

Fig 1. Architecture of client honeypot

The components are explained as:

1. **Queuer**: the queuer is responsible for creating the list of the URLs that has to be browsed by the Active Honeypot. There are several techniques used to create URL lists, including search engines, Blacklists, Phishing and spam messages, and instant messaging.
2. **Client Module**: the client is the component that makes requests and interacts with the web servers. It emulates the browser level vulnerabilities.
3. **Analysis engine**: the analysis engine is responsible for determining and checking the state of the client honeypot to see if an attack has occurred or not.

VI. VIRTUAL HONEYCLIENT

With the improvement of software security, attacks based on RPC vulnerabilities declined, however, attacks based on client application software vulnerabilities have increased. Such client application software includes web browsers, Email client and Office. The spread of malware using these software vulnerabilities has become a severe threat to today's Internet. In allusion to this kind of threat, we have tried to develop a prototype system to collect the internet malwares by actively visiting the malicious websites using client honeypots. This system can not only collect malware but also detect malicious website. Here when we are visiting the websites in a virtual machine, we monitor the activities such as file system, network monitor etc. The end results of the system are collected malware executable binaries, PCAP network data.

VII. DESIGN AND IMPLEMENTED VIRTUAL CLIENT HONEYPOT

High-interaction honeyclients give an attacker the capability to interact with real system rather than simulation. They detect the security violations via state changes check; which means the need to monitor filesystem, registry entries, processes, network connection and physical resources such as memory and CPU, etc. State change checks should give first insight into whether a system has been compromised. There are various honeyclients developed based on this approach such as *Capture-HPC*, *HoneyClient* and *HoneyMonkey*. installed on the machine starts monitors the file system for suspicious activity caused by malware

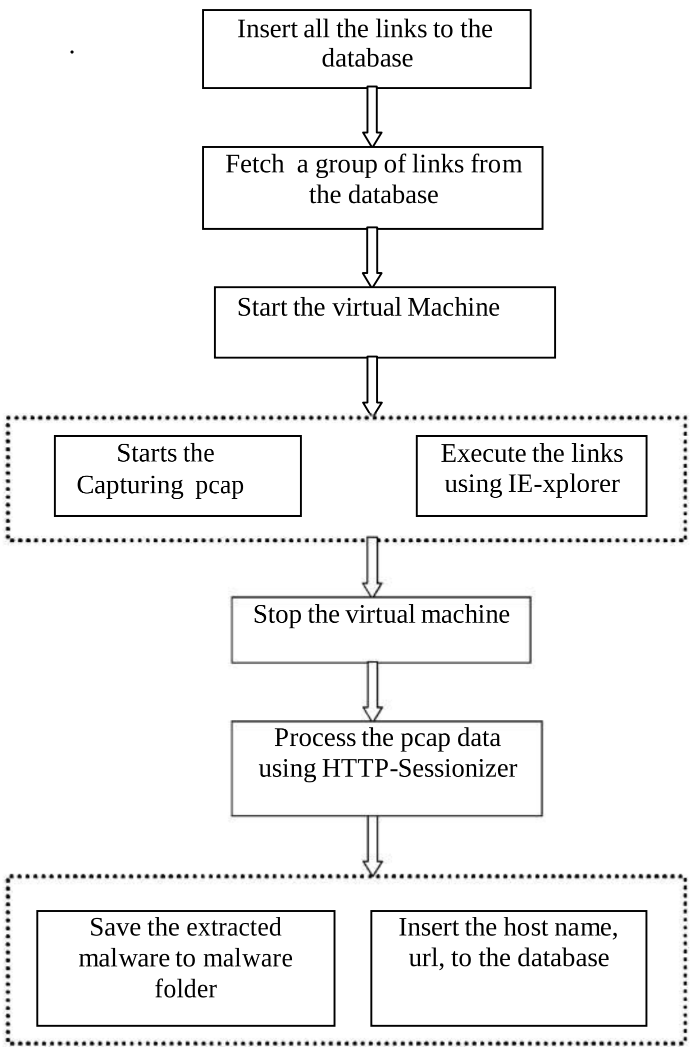
Insert all the links to the database
Fetch a group of links from the database
Start the virtual Machine
Starts the Capturing pcap
Execute the links using IE-xplorer
Stop the virtual machine
Process the pcap data using HTTP-Sessionizer
Save the extracted malware to malware folder
Insert the host name, url, to the database

Fig 2. functional diagram of virtual honeyclient

In the implementation of virtual client honeypot, we have used linux red hat as base machine and Virtual Box based honeypot for browsing of URLs and monitoring file system, network activities. Firstly, we manually feed the URL's in the log file which we want to check for malwares or we can a crawler to collect web page URLs, and store them in a database. After that when we fetch the links from the database and start the virtual machine. The machine starts to open these fetched links one by one and MwWatcher tool infections. We have set the execution of each site for 90 sec. Also we use the DCHSniffer for capturing PCAP data. After all the processing has been done virtual machine stops and all the executable and binary files be shown on the base machine with the URL from where they came.Then analysis and reporting, we are inserting the mailicious URLs into database.We have also used bridge-util is used for creation of bridge, gcc compiler is GNU C compiler used in linux platform, HTTP: sessionizer is for re-session of http communication and Fuse util is being used for virtual file system

VIII. EXPERIMENTAL RESULTS

URL	STEM	HOSTNAME	md5
http://admarcontabil.sites.uol.com.br///live.txt	///live.txt	admarcontabil.sites.uol.com.br	cc4c77ee54de37e9089c7aae2e24d9a2
http://ew.correa.sites.uol.com.br///RITINHA.jpg	///RITINHA.jpg	ew.correa.sites.uol.com.br	5912d4f1845de44a4e5c9e9db891c65f
http://pixwall.net///summer/XvidSetup.exe	///summer/XvidSetup.exe	pixwall.net	ae8621d33a5d184534bab844a0716d1b
http://strandednaked.com///media/XvidSetup.exe	///media/XvidSetup.exe	strandednaked.com	ae8621d33a5d184534bab844a0716d1b
http://depaulamdp.sites.uol.com.br///aut.jpg	///aut.jpg	depaulamdp.sites.uol.com.br	337877a8689824558ba8c17a03763776
http://gucosilva.sites.uol.com.br///downloada.jpg	///downloada.jpg	gucosilva.sites.uol.com.br	5d1cdf7ff4c57503c2352f1d6bf3a149
http://loys.com.br///oportunidade/images/01.jpg	///oportunidade/images/01.jpg	loys.com.br	3f7d7f857f13174261540d6db7c48e2d

Table 1. Experiment results collecting malwares

In the above table the term "URL" means the website which we opened, "stem" means from where the malware found and "md5" means the unique number for malware just like a numeric value.

IX. COMPARISON OF HONEYCLIENT WITH IDS

Client Honeypot is an active honeypot,which uses client application and collects the malwares. As we know, Client Honeypot and IDS are both network security terms but Client honeypot is better than IDS because IDS only generates the alerts when the signature of attacker matches with the database but client honeypot detects the malware of unknown signatures also. Also Intrusion detection systems in large networks suffer from the high amount of traffic while client honeypot in contrary just have to handle traffic directed to themselves. Client Honeypot does not need high configurations.

X. CONCLUSION AND FUTURE WORK

Computer networks have brought the world together by bridging the information gap among people. Network technology has undergone a revolution with better and faster ways of sending information between computers. Unfortunately security systems and policies to govern these networks have not progressed as the same speed. Today's network is very complex and the whole world is focusing on ease of use and functionality. This is diversity to our concern for the security towards the ease of use and increase of functionality. Cyber crime is also no longer the prerogative of lone hackers or random attackers. So there is a huge need of detecting and preventing the threats and intrusion. In this work, we presented the Internet malware system using client-side honeypot. We use the active ability of client-side honeypot to collect malware that traditional honeypot cannot get in the Internet. We introduced the category of Internet malware, the client side attack techniques and overall framework of the system in detail. We mainly gave the design and implementation of client honeypots based malware collection. During the work done so far, client honeypot based solution is very useful to collect the internet malwares and to detect the malicious websites.

Our developed Virtual Box powered Honeyclient is very useful for collection of internet malwares but it is having a limited capabilities or we can say that it is just a prototype. There is a requirement of integration of crawler as data acquirement, at present there is no such component in our developed module. Further there is also a possibility of addition of various client side applications such as firefox, pdf etc because currently we only using Internet Explorer for actively visiting the websites. And there is also a possibility of addition of automatically analysis of collected malwares. We can confirm that we cannot cover all the challenges such human user simulation, logic bomb, time triggered websites but we have developed a prototype solution to get better understanding of client honeypots**.**

REFERENCES

[1] R. Danford, "2nd Generation Honeyclients", SANS Internet Storm Center,2006
http://handlers.dshield.org/rdanford/pub/Honeyclients_Danford_SANS 06.pdf
[2]Zero Day Initiative, "Adobe Flash Player JPEG Parsing Heap Overflow Vulnerability", 9 December 2009.
http://www.zerodayinitiative.com/advisories/ZDI-09-092/
[3]C. Seifert. HoneyC - The Low-Interaction Client Honeypot. 2006. CiteSeerX
http://citeseer.ist.psu.edu/seifert06honeyc.html.
[4]R. A. Grimes, "Tracking Malware with Honeyclients", InfoWorld, 2006 http://www.infoworld.com/d/security-central/tracking-malwarehoneyclients- 852
[5] K. Wang. Honeyclient Development Project.
http://www.honeyclient.org/
[6] Offensive-Security, Client Side Attacks, 2009
http://www.offensive-security.com/metasploit-unleashed/Client-Side-Attacks
[7] C. Seifert, R. Steenson, T. Holz, Y. Bing, and M. A. Davis, "Know your enemy: Malicious web servers." The Honeynet Project, 2007.
http://www.honeynet.org/papers/mws/
[8] HoneySpider Network Project, "The Honeyspider Network – Fighting Client-Side Threats",2009 http://www.honeyspider.net/wpcontent/uploads/2009/06/hsn-first2008-article-v02.pdf
[9] S. Garera, N. Provos ,M. Chew , and A. D. Rubin, "A Framework For Detection And Measurement Of Phishing Attacks", Proceedings of the 2007 ACM workshop on Recurring Malcode, 2007
[10] C. Clementson," Client-Side Threats and a Honeyclient-Based Defense Mechanism, Honeyscout", Master's Thesis, Linköping University Electronic Press, 2009.
[11] R. A. Grimes, "Tracking Malware with Honeyclients", InfoWorld, 2006 http://www.infoworld.com/d/securitycentral/tracking-malware-honeyclients-852 .