A Study on the Automatic Malware Collecting System Based on the Searching Keyword

Byung-Ik Kim, Jongil Jeong, and Hyuncheol Jeong*

**Korea Internet & Security Agency, Seoul, Korea*
{kbi1983, jijeong, hcjung}@kisa.or.kr

Abstract

With the development of the Internet, many people are able share information freely, and check for information from various parts of the world in real time. Internet brought about the development of diverse industries, and its utilization is likely to increase through combination with many types of media in the future. However, the Internet also causes some problems. A computer can be infected with malicious codes only from a user visiting website, and personal information can be leaked from an infected PC. In particular, malicious codes are spreading, using the search word containing the social issue. Because search work rankings are provided with various categories, such as real time search words and "sharply rising search words", malicious code spreading using these search words seem to increase. Therefore, this paper proposes a system that automatically collects malicious codes, which are disseminated using the search function.

Keywords: *Malware, Searching Keyword, Hybrid Interaction Client Honeypot, Hidden Iframe, Obfuscation Code*

1. Introduction

The Internet brought convenience of life to many people. We now can see the news from the other side of the world practically in real time. Offline personal connections are now moving online using social network services (SNS). We can obtain the news about our acquaintances by only having to use the Internet. As the commercial transaction on the Internet becomes more popular, many people are using e-commerce, Internet banking, and Internet payment services, and are also making online transactions. As described above, the Internet makes our life more comfortable than before.

However, the Internet also entails several problems such as the leaking of personal information, malicious code dissemination, enterprise hacking, and the collection of personal information, even though it makes our lives more convenient. A more serious social problem is that an article slandering a particular person is posted on an Internet website or on a celebrity's home page. These problems didn't exist before the Internet, but they have become a serious social issue, as the Internet becomes more popular and the number of users increases.

These days, a user's PC can be infected by a malicious code only by accessing an Internet website, and the infected PC can be exploited to obtain a user's personal information and credit card information, or can be used to make a random attack against a particular organization or target. Accessing a particular web page, or downloading the illegal file from a P2P site, or from visiting a hacker's attack site can infect a user's PC. From out of these malicious code distribution methods, this paper focuses on malicious code distribution that

occurs from using a search word. Recently, a malicious code was disseminated, using the misleading title, "A picture of Osama bin Laden dead". Likewise, users can be infected by malicious codes without meaning to, when they search on the Internet using a recommended search word.

This paper proposes a system that collects the search word that is connected to a social issue, and also collects the malicious code that is distributed using the search word in question.

This paper is composed of as follows: Chapter 2 explains the method of collecting malicious codes that are disseminated by a website. Chapter 3 describes the automatic malicious code system based on the search word, and Chapter 4 explains the performance measurement results of the proposed system. Lastly, Chapter 5 describes the conclusion and future studies.

2. The Trend for Collecting Malicious Codes from a Website

There are active and passive methods for collecting malicious codes from a website and for website analysis. The passive method is that a malicious attacker injects a malicious code into the user's PC. The server side honey pot system is the example of the passive method.[1] The active method is that the malicious code collection system attempts to connect to a particular web site, and performs malicious actions on the web site in question. This is called the "client honey pot."[1] The client honey pot method is divided into the low interaction client honey pot and high interaction client honey pot, depending on the operation scope of the malicious code collection system.[3,4,5,8] Chapter 2 will review the honey pot system and its shortcomings and strengths, and Chapter 3 proposes a system that supplements those shortcomings.

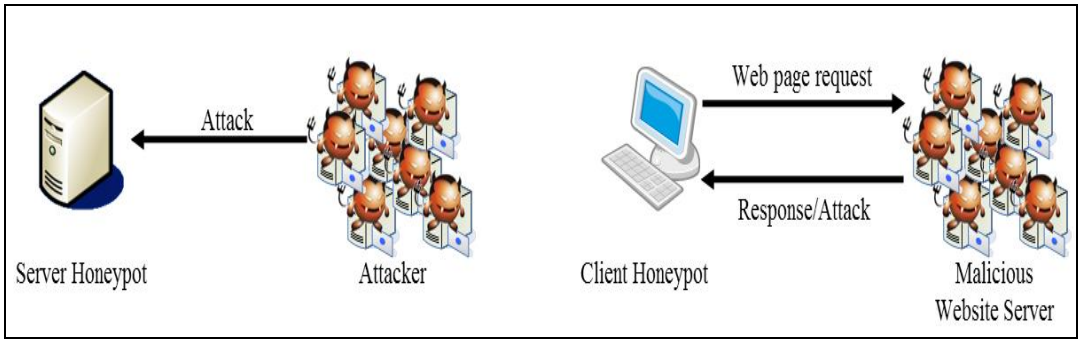

Figure 1. Comparison of the Server Honeypot and the Client Honeypot [8]

2.1. Low Interaction Client Honeypot

A low interaction client honey pot includes systems like HoneyC[8], Monkey-Spider[3], PhoneyC, and SpyBye[7]. These low interaction client honey pots determine what is a malicious website by checking the response between the website to check and the honey pot.[3] The malicious behavior pattern should be reviewed to check the website response. When checking the website, the actual site is not visited but the source code of the target website is crawled, using a virtual browser, such as GNU Wget or HERITRIX. The collected website source is compared with the malicious action pattern of the system, and the website will be selected as a malicious website, if any malicious behavior is found.

2.2. High Interaction Client Honeypot

A high interaction client honey pot[4,5,6] includes Capture-HPC, HoneyClient, and UW Spycrawler. These high interaction client honey pots actually visit the website to check by using the web browser, unlike the low interaction honey pot. The high interaction client honey pot analyzes the malicious website by monitoring malicious behaviors without analyzing the website source. The high interaction client honey pot monitors the file (file creation, deletion, and hiding), process creation, and registry modification.

Therefore, the high interaction client honey pot can check the malicious behavior pattern that is dynamically created, as well as a new attack that cannot be detected by the low interaction client honey pot, and an attack that uses the obfuscated script. In addition, it has the benefit of collecting the file by visiting the actual target website, and identifying the malicious site and the path to the malicious site. However, the client can be infected by the malicious code, as the actual website needs to be visited. Therefore, the system should be initialized after visiting the website that needs to be checked. There is another shortcoming in that the malicious behavior cannot be found if the client program required by the website that needs to be checked, the browser type and version, and OS don't match with one another.

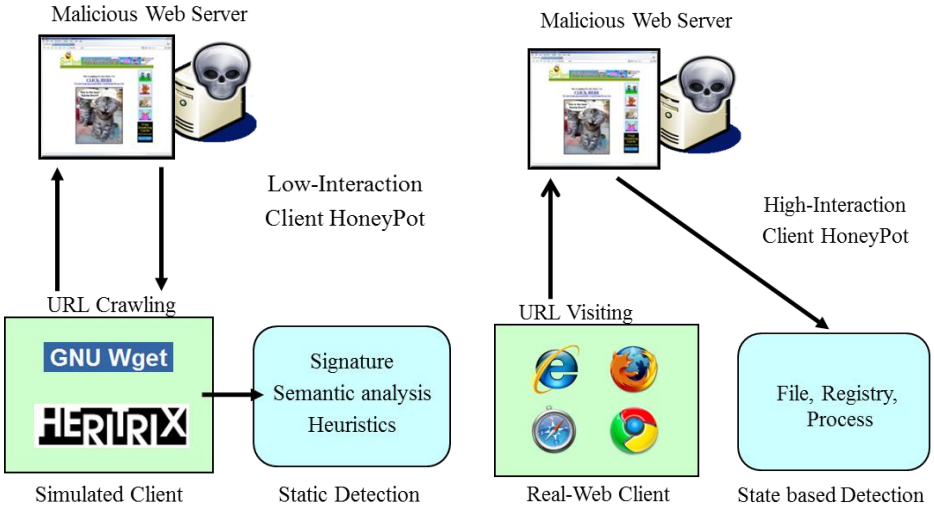

Figure 2. Concept of the Low Interaction Client Honeypot and the High Interaction Honeypot

The hybrid client honey pot was proposed to supplement the strengths and shortcomings of the low interaction client honey pot and the high interaction client honey pot. The hybrid client honey pot checks the website source. It then selects the suspicious website, and actually visits the website.

The system proposed by this paper also adopts the hybrid client honey pot. Chapter 3 briefly explains its composition and operation method.

3. The Automatic Malicious Code Collection System Based on a Searching Keyword

As described in Chapter 2, the low interaction client honey pot can quickly analyze the website by checking the website source, whereas the high interaction client honey pot can

detect the unknown attack and collect the malicious code. This paper proposes a system that automatically collects the malicious code using the hybrid client honey pot that combines those strengths. The check target will be selected, using the real time search word and daily popular search word. Using the low interaction method then checks the availability of the malicious behavior. Afterwards, the malicious code that was generated when visiting the targeted suspicious website will be collected. All of these procedures will be automated to minimize interference from an administrator.

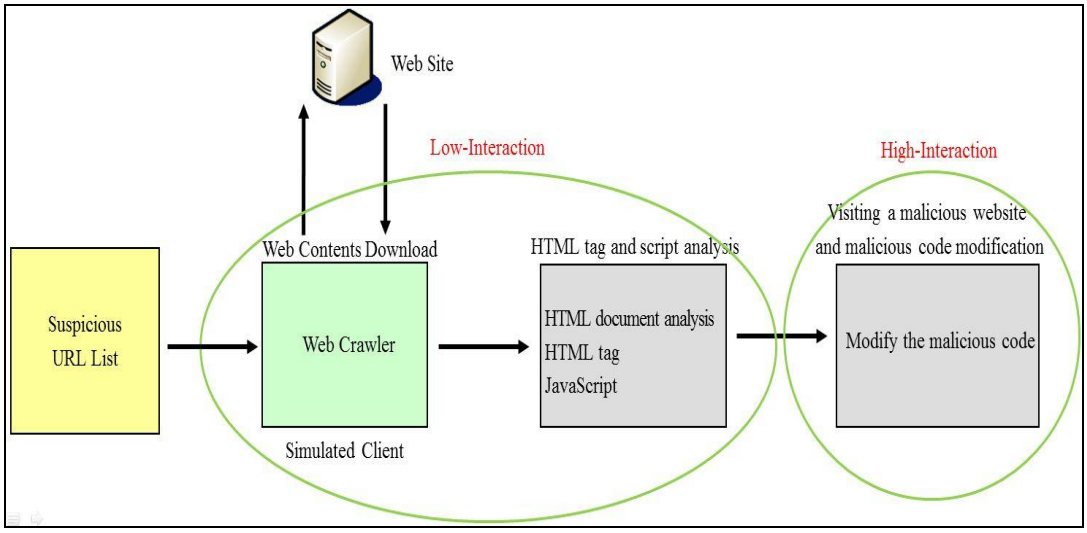

Figure 3. Simplified Configuration Diagram of the Proposed System

3.1. Utilization of the Search Keyword that Reflects the Social Issue

Most malicious website detection or malicious code detection is started with a report from the security administrator or Internet user. Malicious code detection based on the report is started when the malicious code is already disseminated and abnormal behavior is detected. In this case, there is a possibility that the malicious code is already widely distributed. The proposed system adopts the method that proactively selects the target, instead by a report from the security administrator or Internet users..

One of the malicious code dissemination trends is to use the search engine optimization method, or the social issue. For instance, the malicious code related to the death of Osama bin Laden was distributed as soon as the news was broadcast. The pre-created malicious code was disseminated when the matching social issue was raised. Therefore, the search words that are connected to social issues should be collected in advance, and the search result should be also collected. Many portals provide a list of search words with various categories, such as a real time search word, a sharply rising search word meaning is unclear, and the popular daily search words. The proposing system sends a query to search engines such as Naver, Google, Yahoo, and Bing, using the search words that were collected in this way. Then, the URLs of the search results are collected to detect malicious codes.

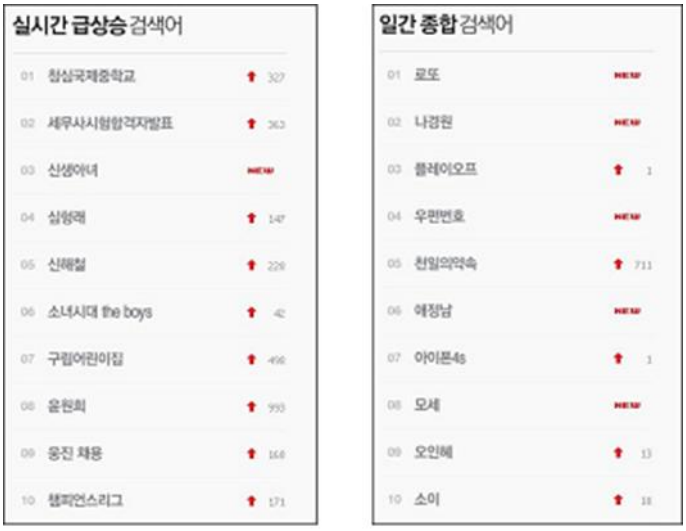

Figure 4. Example of Search Keyword Ranking by Type that in Provided by Naver.com in Korea

3.2. Checking for Malicious Behavior by Reviewing the URL Components

Actual malicious code dissemination sites are detected by using the URLs that have been collected by search engines. A URL component check takes the form of the low interaction client honey pot that was described before. Crawling will be started by using a single URL that needs to be checked. The URL that needs to be checked can be identified by using the simple HTTP Request and Response, when the crawling method is used.

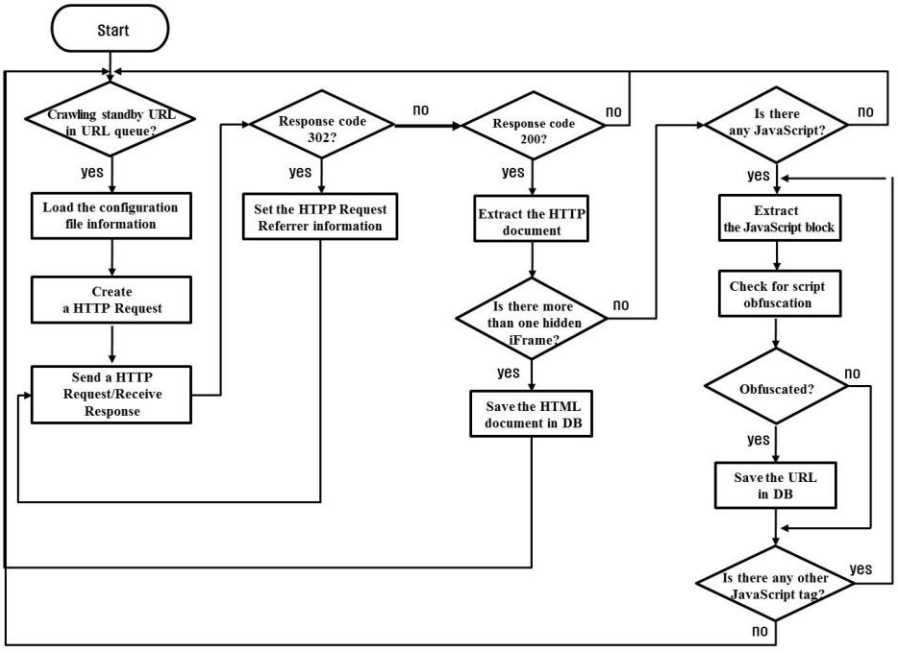

Figure 5. URL Component Check Flowchart

The proposing system checks the HTTP packet responses and the URL's web page source. It is checked whether there is any hidden iFrame with a zero size, and whether any web page script is obfuscated or not. It is also checked whether the web page to check actually has the PE (Portable Executable) format. The hidden iFrame is one of the components that is most widely used to disseminate malicious codes. If both the horizontal or vertical height are "0," or if the multiplication of two height values is "0," iFrame will be hidden from the web browser but connected to the actual path.[2] Therefore, the visitor to this website is connected to the particular website without realizing the connection, and the malicious code is disseminated without the user knowing.

Exploit Kit, one of the malicious code distribution tools, obfuscates the script (e.g., JavaScript) to proliferate a malicious code, so that normal users cannot interpret the script. Obfuscation can be checked using an obfuscation check algorithm, and a malicious website can be determined to be such, depending on the intensity of obfuscation.[9]

For check the obfuscated JavaScript, proposal system check several point of target URL component.

First, the system will check density of target web page. The density elements are the longest unique character stream size and used special character set number. If the longest character stream is over 200 characters, the system can scored 12 point. And give 2 point to target web page URL, if used special character set number is over threshold value.

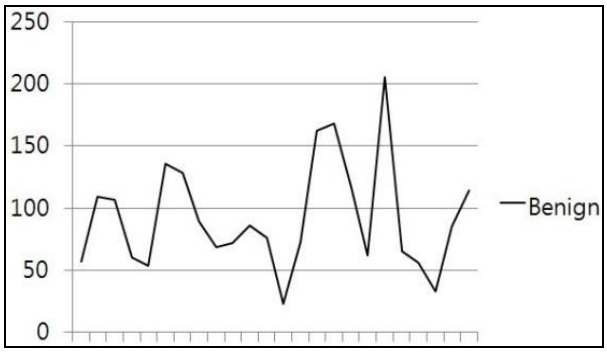

Figure 6. Maximum Length of a Unique Character Stream in a Normal Web Page

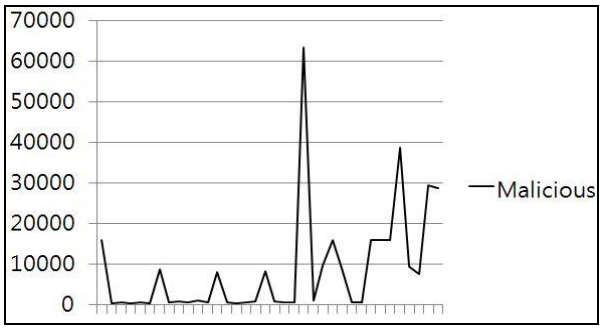

Figure 7. Maximum Length of a Unique Character Stream in a Malicious Web Page

Second, the system will count some feature of target web page for checking frequency score. The system check a particular function's frequency and encoding mark and % symbol occurrence.

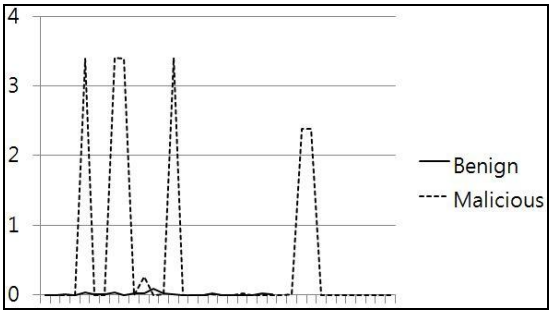

Figure 8. Frequency of % Symbol Percentage Besides the http Link in Web Page Code

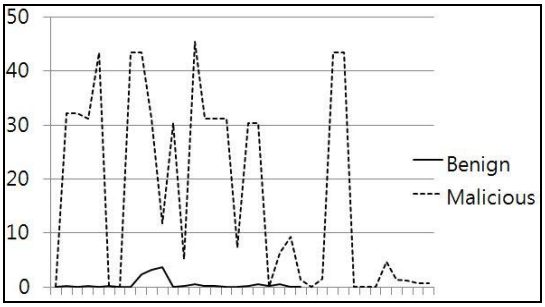

Figure 9. Frequency of a Particular Function in Web Page Code

Finally, web page entropy will be checked for verified obfuscated JavaScript. Whole web code's entropy, total JavaScript entropy, each JavaScript block entropy, variable's name entropy and function's name entropy are checked. Also, the proposal system check other component of web code of target URL, but above descriptions are most important check point in the system.

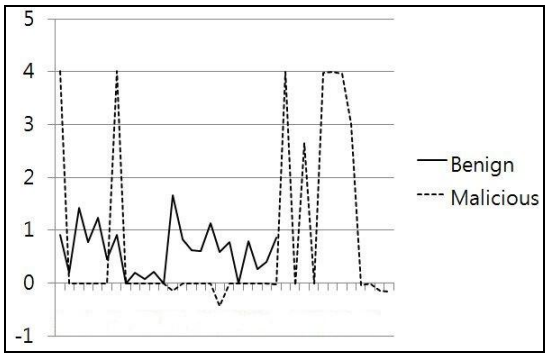

Figure 10. Difference between total JavaScript Entropy and Average Java Script Block Entropy

The proposed system checks the website with a hidden iFrame, an obfuscated website, and PE style website. If the website that needs to be checked belongs to any of these types of websites, it will be determined to be a malicious website. If it is determined to be a malicious website, it is sent to an automatic website module in the form of a high interaction client honey pot.

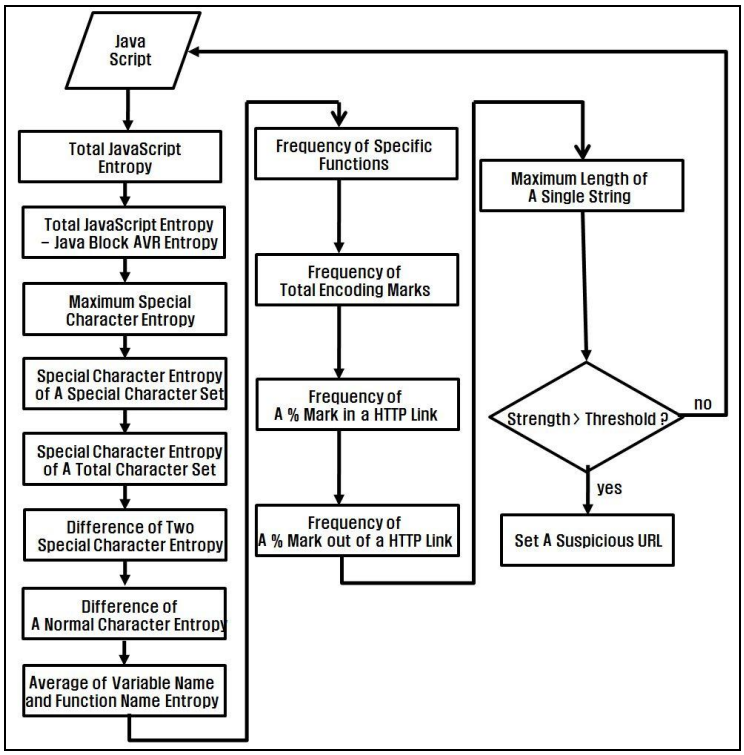

Figure 11. Obfuscated JavaScript Checking Diagram

3.3. Automatically Visiting a Website and the Collection of Malicious Codes

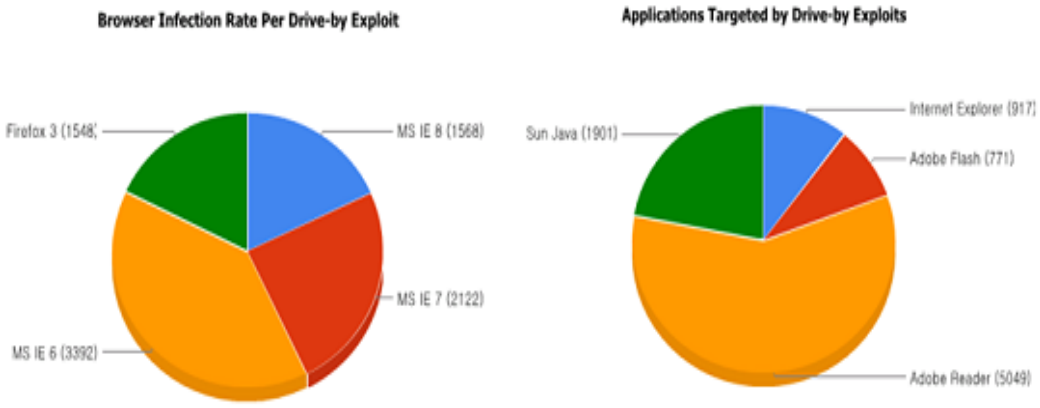

Figure 12. Example of the Environment Configuration used by Driven-by Exploit, Blade, 2010

A website that has selected as a suspicious website by the URL component check module will be reviewed in the form of the high interaction client honey pot. The actual attack is checked using the strength of the high interaction client, and the malicious file is collected.

The proposed system visits suspicious websites using the actual browser on the virtual PC, after several virtual PCs have been installed on a single server, using the virtualization

solution.[10] Then, the suspicious files are collected by visiting the suspicious websites. To collect these files, the environment of the accessing PC is important, because the response can be different, depending on the environment of the PC that accesses the website in question. Therefore, the environment of the virtual PC is created, based on the environment that proliferated the malicious code most frequently up until now.[11]

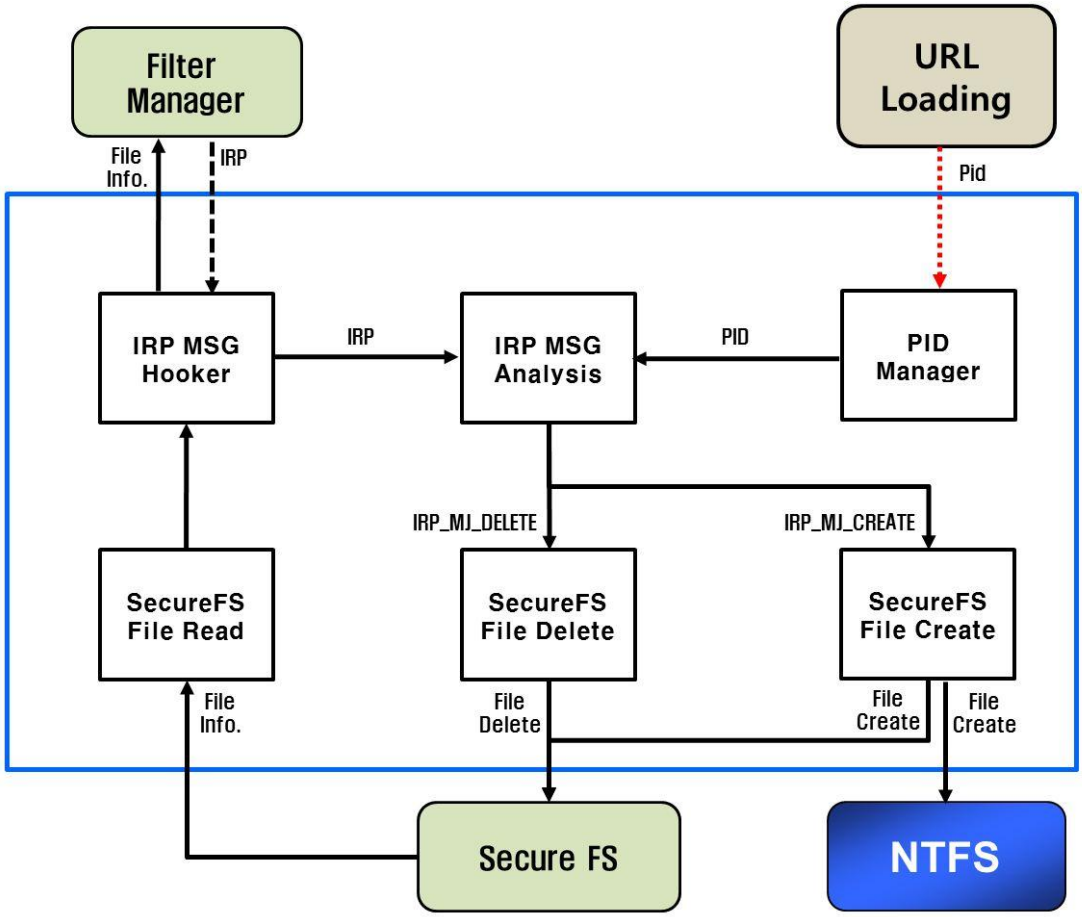

Figure 13. The Module that Separates the Generated File using IRP Message Hooking

The proposed virtual machine minimizes malicious code infection from having visited a website, by isolating the created file using IRP message hooking, and execution prevention. In addition, the machine provides the function that decodes the obfuscated redirection code. Obfuscated malicious codes can be distributed in the following two ways: redirection to the website, and using the code that actually execute malicious behaviors. Therefore, hooking the IPR messages can collect the file created by actual malicious behavior. In addition, the obfuscated redirection script can be normalized, using the function like document.write, document.writeln, and eval. The normal return value of these functions can be collected to normalize the obfuscated redirection code.

The malicious code collected by the proposed system, and its actual malicious behavior is not verified. Therefore, these suspicious files should be sent to the analysis system to have their actual behavior checked.

4. Performance Verification of the Proposed System

The structure of the automatic malicious code collection system that is based on the search word, which is implemented by the hybrid client honey pot, was described in Chapter 3. The website to visit will be selected using the low interaction client honey pot, and will be visited using the high interaction client honey pot, and the generated file will be collected. The following procedure was carried out, in order to evaluate the performance of the proposing system. The anti-virus program checked the files collected by the proposing system, and the normal files were entered into the PE file behavior analysis system again, in order to check the malicious behavior. Performance was evaluated regarding the files collected from November 22, 2010 to January 11, 2011.

4.1. Comparison of System Functions that Automatically Collect Malicious Codes that are Proliferated by a Website

Systems that automatically collect malicious codes that are proliferated by a website, were compared to check the quantitative performance of the proposed system. The high interaction client honey pot is the comparison target, such as HoneyMonkey or HoneyClient, and MonkeySpider was selected as the low interaction client honey pot. As the proposed system is a hybrid client honey pot, direct comparison cannot be made but quantitative performance can be checked, because the implementation type is similar. The check items included the check target selection method, the URL pre-processing function (filter), an automatic description of the obfuscated script, the malicious code collection function, and the false positive rate.

Table 1. Comparison between the Proposed System and Existing System

	Proposed System	HoneyMonkey	HoneyClient	MonkeySpider
Type of HoneyPot	Hybrid	High interaction	High interaction	Low interaction
Review target selection method	Search engine, blacklist, spam mail	Report acceptance, internal selection	Report acceptance, internal selection	Link with the search engine
URL pre-processing function (filter)	Hidden Iframe check Script obfuscation check KillBit use check	None	None	None
Restoring obfuscated script	Available	Not available	Not available	Partial
Malicious code collection function	Available	Available	Available	Not available
False positive rate of malicious code collection	Low	High	High	·
System recovery time	10 seconds	120 seconds	180 seconds	·

The URL pre-processing function is available in the hybrid form only, and is not used by the high/low interaction client honey pot. The single client honey pot does not use the pre-

processing function, because all requested URLs will be analyzed. MonkeySpider provides the function that can check obfuscation at the limited part, when restoring the obfuscated script. It is not easy to check for the obfuscation of the website source, and to restore the obfuscated code by using the program. However, the proposed system has obfuscated source detection[9], and restores the obfuscated source based on the actual content obtained by visiting the website with the previously mentioned algorithm.

4.2. Measuring the Malicious Code Collection Rate

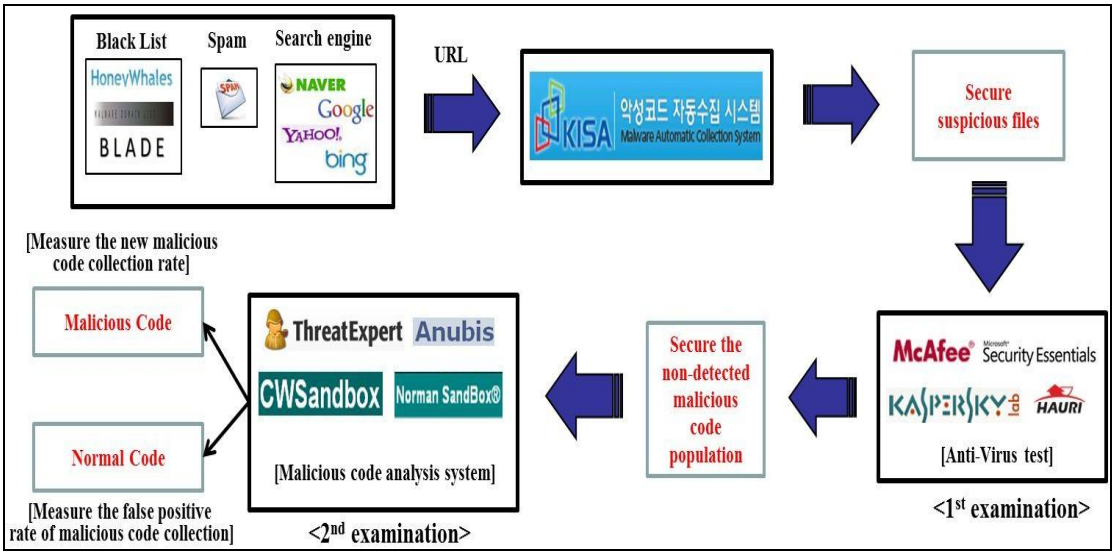

Figure 14. Process of Measure the Malicious Code Rate and the False Positive Rate in the Collected

2,256 suspicious codes were collected for one and half months between November 22, 2010 and January 11, 2011. The search engine and the URL in the body of the spam e-mail collected the collected suspicious files. The duplicated files were removed, and the malicious code was checked for 1,287 files, using 4 vaccine programs. M vaccine program diagnosed 846 malicious code infected files. When four programs were used, 867 files (67.37%) were determined to be infected with malicious codes. Behavior analysis was performed on 420 files that were determined to be normal by 4 vaccine programs by using behavior analysis systems, such as Anubis, ThreatExpert, CWSandbox, and Norman SandBox. If the malicious behavior was detected by more than three behavior analysis results, the code was deemed malicious. As a result, 119 new malicious codes were collected. The remaining 301 files were composed of 94 normal files, and 207 non-executable files or incomplete files. A total of 986 malicious codes (76.61%) were collected from 1,287 suspicious files. Based on these results, we can see that about 23.4% of normal files were collected. This false positive rate of malicious code collection is under the average of the domestic and international false positive rate of malicious code collection. Therefore, it can be said that performance improvement was made. In addition, a total 119 new malicious codes (12%) were collected, which is better than the existing malicious code collection system.

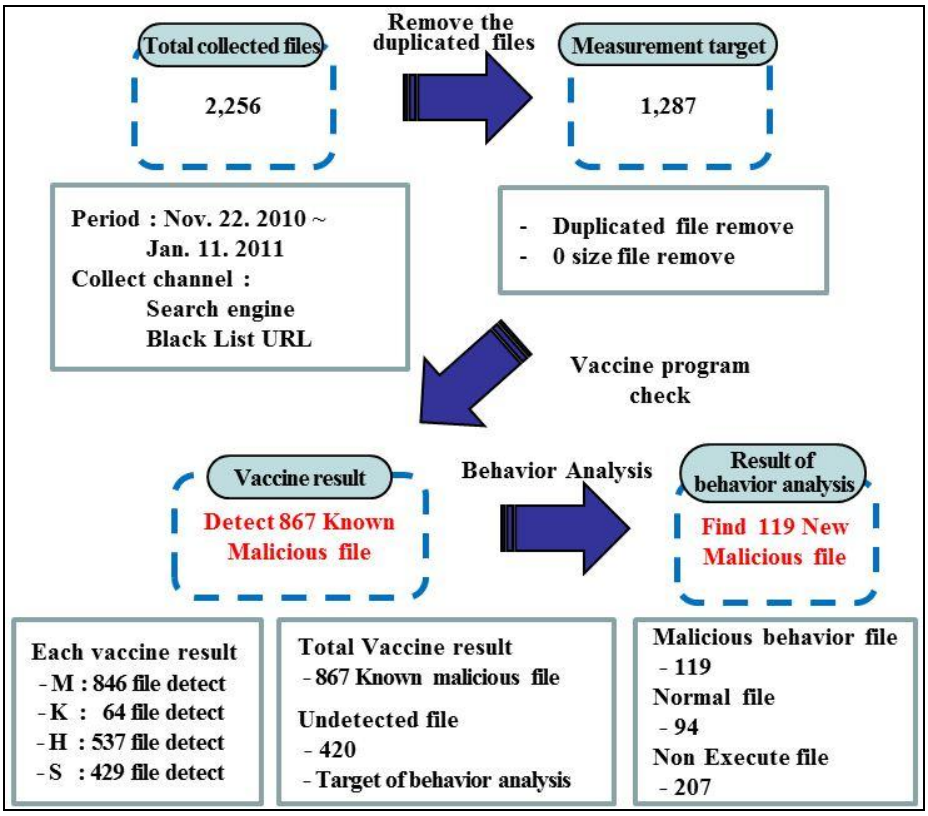

Figure 15. Results from Checking the Collected Suspicious Files

In addition, about 62.04% of the known malicious codes could be collected when the malicious URLs provided by Google were checked. It was found that 14,446 URLs out of 29,051 malicious URLs (about 50%) are still malicious. When 14,446 URLs, which are the result of a URL component check, were visited, a total of 108 suspicious files were collected from 92 URLs. Using 4 vaccine programs, 67 known malicious codes were found out of 108 suspicious files. Unlike the previous performance check results, the behavior analysis for 41 non-detected files was not carried out. Instead, their URL component was checked. A hidden iFrame was found in 8,392 URLs out of 29,051 URLs (about 29%). Out of 8,392 hidden iFrame internal URLs, http://h.nexprice.com/css/x.htm was detected 6,306 times (about a 75% concentration rate), whereas http://www.ro521.com/test.htm was detected 1,357 times (about a 16% concentration rate). These two URLs were prohibited from being accessed, and the malicious behavior of the URL could not be detected. However, it was found that the malicious codes proliferated by this method can be collected. Therefore, these URLs seem to require periodic management.

5. Conclusion and Future Studies

The proposed system has been configured with a hybrid client honey pot, and it collects the suspicious files by visiting the suspicious website, after filtering the URL through a URL component check. This system includes the benefit of both the low interaction client honey pot and the high interaction client honey pot. The validity of the search task was confirmed through comparison with existing systems. Also, the proposing system could collect the new malicious codes that cannot be detected by an anti-virus program. However, the number of

URLs to check is determined by the number of search words collected from the target. Therefore, more studies on effective search word collection are needed. In addition, more studies are required to filter out the URLs more accurately, by adding the malicious code proliferation pattern, which can be detected by checking the URL components. Also, more studies on malicious code collection environments need to be formed, so that malicious codes can be collected from more diverse environments.

Acknowledgement

"This research was supported by the KCC(Korea Communications Commission), Korea, u nder the R&D program supervised by the KCA(Korea Communications Agency)"(KCA-2011-(10914-06001)).

References

[1] Jamie, "Server Honeypot vs. Client Honeypot", The Honeynet project, http://www.honeynet.org/node/158, **(August 2008)**

[2] Dmitriy Kashitsyn "Is Your Website Already Infected? Analyzing and Detecting Malicious Content" NTOBJECTives, Inc.

[3] Ali Ikinci, Thorsten Holz and Felix Freiling, "Monkey-Spider: Detecting Malicious Websites with Low-Interaction Honeyclients", In Proceedings of Sicherheit, Schutz und Zuverl, **(April 2008)**

[4] Christian Seifert, Ian Welch and Peter Komisarczuk, "Application of divide-and-conquer algorithm paradigm to improve the detection speed of high interaction client honeypot", SAC'08, **(March 2008)**, pp. 1426-1432

[5] Y.Wang, D.Beck, X.Jiang, R.Roussev, C.Verbowski, S.Chen and S.King, "Automated Web Patrol With Strider Honeymonkeys: Finding Web Sites That Exploit Browser Vulnerabilities", in 13th Annual Network and Distributed System Security Symposium. San Die 헤: Internet Society **(2006)**

[6] New Zealand Honeynet Project, Capture-HPC - Capture - The High Interaction Client Honeypot, http://www.nz-honeynet.org/capture.html

[7] SpyBye, http://www.spybye.org/

[8] Christian Seifert, Ian Welch and Peter Komisarczuk, "HoneyC - The Low-Interaction Client Honeypot", **(August 2006)**

[9] Byung-Ik Kim, Chae-Tae Im and Hyun-Chul Jung, “Suspicious Malicious Web Site Detection with Strength Analysis of a JavaScript Obfuscation”, IJAST, Vol. 26, **(January 2011)**, pp.19-32

[10] Kim Byung-ik, Im Chaetae and Jung Hyunchul, “A Study on Advanced High-Interaction Client Honeypot”, Korea Multimedia Society, **(2010)**, Vol. 13, 2nd edition, 11.2010, pp. 195

[11] Long Lu, Vinod Yegneswaran, Phillip Porras, Wenke Lee "BLADE: An Attack-Agnostic Approach for Preventing Drive-By Malware Infection", CCS'10, 10.2010 **(2010)**