Malware Meta Crawler for MASS

MA-INF 3309 - Malware Analysis
Lab Report
Winter Semester 2016.17
University of Bonn

Ehab Qadah

April 29, 2017

Table of Contents

Abstract. On a daily basis, new malware samples are discovered in. This makes the software vulnerabilities analysis one of the top concerns for organizations. The automatic identification of vulnerable software inside the organization is fundamental to avoid cyber-attacks. In this paper, we discuss two techniques to automatically monitor software vulnerabilities using open standards and public vulnerability information repositories, and alternative method to identify a vulnerable software using information obtained from social media platforms.

1 Introduction

In last decade, the usage of the Internet has increased and adopted all sectors of business and industry as result of the digital revolution. On the other hand, the wide usage of Internet creates a new opportunities for Cyber criminals to perform their malicious activities such as information theft and espionage. Malicious Software (malware) is any software has a harmful intention and abuse the user's computer [1]. Malware is a common tool to perform cyber attacks that can be in different forms such as worm, virus, Trojan and spyware [2]. According to Symantec, in 2015, 431 million of new malwares were discovered [3], which means over one million per day. To protect the Internet's users the malware researchers community try hardly to study these malwares, in order to build the counter measures and detect the new malware software or their malicious behavior, using different malware analysis techniques like static or dynamic analysis of malware samples [4].

In this work, we provide malware crawler that contentiously retrieve new malware samples (e.g., malware domains, URLs and binary files) from different on-line sources and repositories , and submit them to MASS server to build a comprehensive database of malicious software, to make the malware samples continuously available in one place, which helps the malware researchers in their studies.

The remainder of this report is organized as follows. In Section 2, we present the related work and fundamental background . Section 3 presents the general system overview. In Section 4 we give the implementation details. Section 5 provides the evaluation results. And finally, Section 6 gives the overall conclusion and future work.

2 Background and Related Work

- about mass - general overview of malware analysis - other people work - maltrieve -Raypicker - malware resource were used

- foundation like tool were used python + mass apiclient

In this section, we review some of the related work to our system and the required concepts that are used throughout our work.

2.1 Malware Analysis and Storage System (MASS)

The Malware Analysis and Storage System (MASS) serves as a platform for providing malware samples and analysis results [5]. All collected malware samples and analysis results (reports) are stored in a database on the MASS server. The MASS database contains malware samples submitted by malware researchers or retrieved by the malware meta crawler component. The MASS server is connected to several analysis systems which ease the process of sample reception and analysis. In addition to that MASS provide a web interface and REST APIs to access the samples and analysis results information.

The aim of MASS software is to provide the malware researchers a collaboration platform for malware analysis.

2.2 Malware Analysis

This section provides an overview of the malware analysis techniques (i.e., static and dynamic analysis) and the types of malware. Bayer, Ulrich, et al. [1] define the malware analysis as the process of identifying and understanding the capabilities and goals of a malicious software sample.

2.2.1 Static Analysis:

The static analysis is a technique to analysis the malware sample by generating the corresponding assembly code to understand the control and data flow of the sample [1]. This process does not require to execute the malware sample

2.2.2 Dynamic Analysis:

While the dynamic analysis approach is to study the behavior of malware sample by executing it in isolated environment [1]. This process requires to run the malware executable on a certain environment and find the effects of execution on the host system.

2.2.3 Categories of Malware:

This section provides an overview of the different types and classes of the malware samples. The malwares are categorized based on their similar characteristic and behavior, the following are some of the malware types were observed crossed the globe [6]:

- Bot: is a malware program utilizes the exploited system to perform malicious activities such as Spaming and involving in Denial of Service (DoS) attacks [7]. Usually the bot is remotely connected to Command & Control (C&C) server (i.e., botmaster) to receive the instructions of new malicious activities and updates.
- Spyware: is a malicious code collects sensistive information without the knowledge and permission of the system's user [6].

– Virus: is a malware program that requires to be attached to other host program in order to execute [8].
– Worm: "is a program that can run independently and can propagate a fully working version of itself to other machines"[8].

2.3 Sources of Malware Samples

This section present the online sources are used in the malware meta crawler to retrieve the malware samples, while more sources can be supported in future. The following are the supported sources that contain different malware samples information:

– Malware Domain Blocklist[1]: that provides a list of malware domains that are know to used for malicious purposes.
– Malc0de [2]: a database of URIs that contain malicious executables and binaries.
– ZeuS Tracker [3]: that provides RSS feeds of IPs and domains of malicious ZeuS hosts.

2.4 Related Systems

2.5 Python

The Python programming language [8] is a high-level, easy to use, general, purpose language, which supports object-oriented and functional programming programing paradigms. It is one of most popular languages in the scientific research. For those charactisices and others we have selected it to implement our system (i.e., the malware meta crawler for mass).

3 System Overview

- idea +problem - formal algorithmic description - process flow diagram or any sort of charts

3.1 Malware Meta Crawler Architecture

The goal of malware meta crawler is to feed the MASS server database with new malware samples retrieved from different online sources (e.g., malwaredomains.com) which provide malicious domains or URIs that are known to be connected with malware activities or deliver malicious payload. Furthermore, the malware metacrawler pre-analysis the malware samples to enrich the samples with additional related data (e.g., IP of a malware domain), also it detects the relations between the malware samples and submit it to the MASS server. Figure 1 shows the overview of the MASS project and the connection of the malware meta crawler component and analysis systems.

[1] `http://www.malwaredomains.com/`
[2] `http://malc0de.com/database/`
[3] `https://zeustracker.abuse.ch/feeds.php`

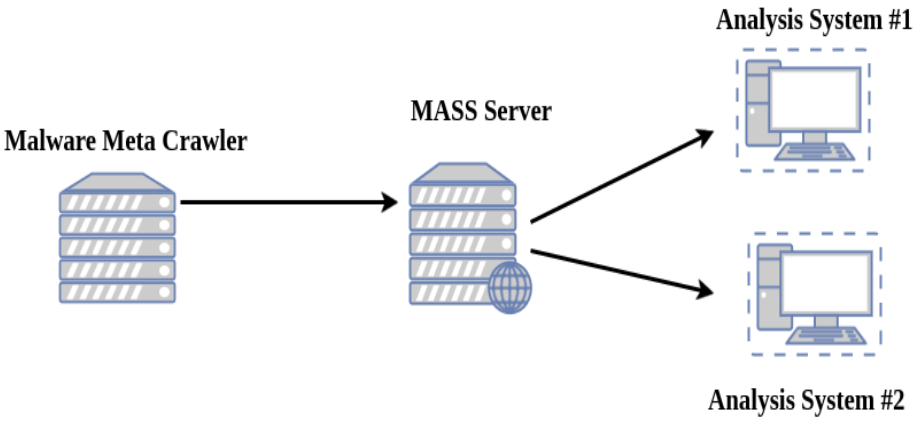

Fig. 1. Overview of the MASS components.

3.2 Process Flow of Malware Meta Crawler

This section presents the flow of the malware meta crawler component that retrieves the malware samples and submit them to the MASS server. Figure 2 illustrates the internal process flow of malware meta crawler.

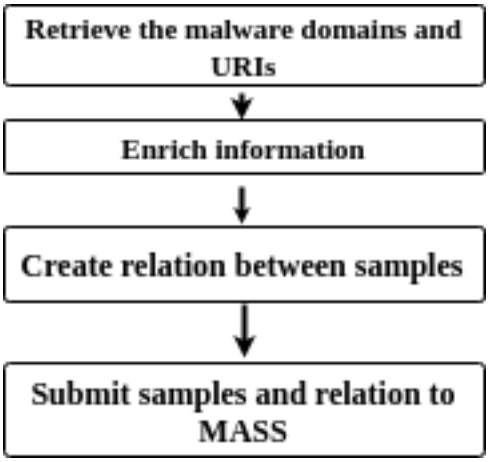

Fig. 2. Process flow of malware meta crawler.

First, the malware domains and URIs are retrieved from the mentioned sources in Section 3. Second, the retrieved list is processed to be enriched with additional information like find IP of malware domain, extract domain of malware URI and try to download the file of executable or binary URI. Third, the

malware meta crawler builds the relation between the enriched malware samples such as relation between malware domain and corresponding IP, URI with malware domain and malware sample file with corresponding URI. Finlay, the malware sample and their relation with other sample are submitted to the MASS server.

4 System Implementation

- how the idea + problem is realized + code snippet - not code docs

5 System Evaluation

Evaluation performance + number of samples - state what do you like to find and how? - state perfomance metric like time, memory usage, etc. - environment setup - present the results - conclude findings

6 Conclusion and Future Work

-briefly sumup what was include/done -state the overall achievement -state the future work - measure the difference time between submission time between samples.

References

1. Bayer, Ulrich, et al. "Dynamic analysis of malicious code." Journal in Computer Virology 2.1 (2006): 67-77.
2. Kienzle, Darrell M., and Matthew C. Elder. "Recent worms: a survey and trends." Proceedings of the 2003 ACM workshop on Rapid malcode. ACM, 2003.
3. Symantec. Internet Security Threat Report, Vol. 21 https://www.symantec.com/content/dam/symantec/docs/reports/istr-21-2016-en.pdf, 2016.
4. Egele, Manuel, et al. "A survey on automated dynamic malware-analysis techniques and tools." ACM Computing Surveys (CSUR) 44.2 (2012): 6.
5. The Malware Analysis and Storage System(MASS). URL https://github.com/mass-project/mass_server/blob/master/README.md. Accessed: 2017-04-27.
6. Manuel Egele. A survey on automated dynamic malware analysis techniques and tools. ACM Computing Surveys, to appear.
7. Li, Chao, Wei Jiang, and Xin Zou. "Botnet: Survey and case study." innovative computing, information and control (icicic), 2009 fourth international conference on. IEEE, 2009.
8. Spafford, Eugene H. "The Internet Worm Incident Technical Report CSD-TR-933." (1991).

8. Van Rossum, Guido. "Python Programming Language." USENIX Annual Technical Conference. Vol. 41. 2007.