DETERMINING MALWARE LINEAGE

TAMMY TRAN

Abstract: Current research on software lineage typically focuses on reverse engineering code, but anti-reversing techniques such as code obfuscation causes difficulty and complexity in analyzing reverse engineered code. In this project, static analysis of various metadata is used to determine malware lineage, such as: software metadata, binary imports and exports, and execution behavioral statistics. Metadata information was aggregated from the Malicia Project dataset and VirtusTotal execution reports to analyze and determine malware lineage; producing low false positives between 0.7-35%.

INTRODUCTION

Software lineage can be defined as the evolution of a software application throughout its development. Many changes can be made to software between releases, such as: new features are added, some features may get removed or modified, bugs get fixed, etc. Note the following problem:

Given a collection of binary files, how can one systematically determine the hereditary relationships between pairs of files: A is (or is not) related to B, A is an ancestor of B, B is an ancestor of A?

The methodology proposed in this particular project is to compare software metadata, binary imports and exports, and execution behavioral statistics to determine software lineage; more particularly malware lineage.

BACKGROUND

To determine malware lineage, most research inspect and analyze evolutions in code complexity. Unfortunately, code obfuscation can cause tremendous difficulty in being able to fully reverse engineer code sections properly. With malware in particular, most malware developers tend to repackage their malicious code into benign software to be able to avoid detection.

Ideally, by utilizing Microsoft specification portable executable (PE) metadata and behavioral reports from executing malware in monitored environments, performing static analysis may escape the troubles of code obfuscation and repackaging. PE files provide information such as import tables that relay information on function calls made by the application, or export tables that reveal what function calls are created. Such information can be used to determine if new features are added and/or removed between binary releases. Furthermore, behavioral statistics will also reveal similarities and differences between binaries (though these reports are less often available or thorough).

METHODOLOGY OVERVIEW

This project is based on an algorithm proposed by Jiyong Jang[1]. The methodology proposed for this project is as follows - Given a malware family:

1. *Sort binaries by file size (or number of imports); the binary with the minimum file size (or number of imports) is chosen as the root malware.*
2. *Select binaries that are most similar to a selected malware by comparing metadata:*
    a. *Packer, un/initialized data size, link date, link/image/os/file version.*
3. *Of the list of binaries that were selected as most similar, select the binary that has the minimum difference value (# of features added + # of features removed) by comparing:*
    a. *PE data – imports, exports*
    b. *Behavioral statistics – processes, runtime dlls, mutexes, registry, and files*
4. *Binary A is and ancestor of binary B if B is the most similar and has the least number of differences to A.*

DATASET

One of the main contributions of the project is utilizing a merged dataset from the Malicia Project binary metadata and the VirusTotal malware execution reports.

MALICIA

The Malicia Project dataset consists of over 11,000 malware binaries collected from drive-by download servers over 11 months. The dataset includes: the malware binary, metadata detailing when/where the malware was collected, and malware family classification. Of the binaries already classified into families, the families distributed over the longest period of time were selected for this project; that included: *harebot, cridex, zeroaccess, zbot, and winwebsec* (See Figure 1). For further details on malware families, see Table 1.

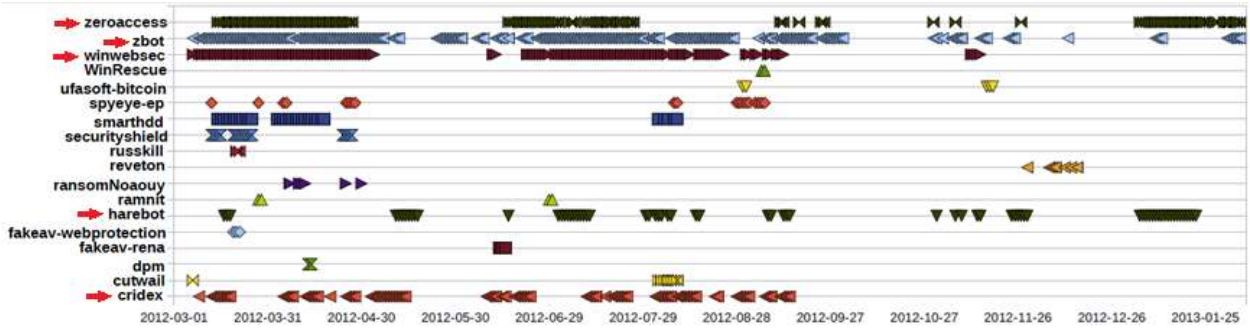

Figure 1. Malware family distribution

Family	Distribution	# Binaries	Total Collected	Repack Rate
harebot	3/17/2012–1/18/2013 (329 days)	51	1078	Every 5.92 days
cridex	3/09/2012–1/31/2013 (308 days)	72	2555	Every 4.4 days

zeroaccess	3/15/2012–2/05/2013 (328 days)	1144	3775	Every 0.30 days
zbot	3/07/2012–2/05/2013 (336 days)	2160	11619	Every 0.15 days
winwebsec	3/07/2012–11/14/2012 (253 days)	5794	16355	Every 0.04 day

Table 1. The following table details the date range over which each malware family's binaries were discovered, the number of unique binaries found, the total number of binaries collected from each family and the repacking rate (how often the malware was repackaged in new software).

VIRUSTOTAL

VirusTotal aggregates data from outputs of various antivirus engines, website scanners, file and URL analysis tools, and user contributions. For the full list of tools used by VirusTotal, see: https://www.virustotal.com/en/about/credits/. All binaries collected from the Malicia dataset were submitted to the VirusTotal engine to gather PE and behavioral reports. See Table 2 for a list of all metadata and reports collected, merged and utilized from each data source.

Malicia	**VirusTotal**
• sha1 • File size • Malware Family • Collected timestamp • Packer*	• Imports • Exports* • Code size • Un/Initialized Data Size • Timestamps (*Compilation, link, PE*) • Unpacker* • Versions (*Linker, Image, OS, File*)* • Processes (*Injected, Created*)* • Runtime Dlls* • Mutex (*Opened, Created*)* • Registry (*Deleted, Set*)* • File (*Opened, Read, Moved, Downloaded, Written, Replaced, Deleted, Copied*)*

Table 2. The above table lists all metadata collected, merged and utilized for statistical analysis. * indicates data fields that were spotty, or inconsistently available.

EXPERIMENT

For each of the selected malware families from the Malicia dataset, the methodology discussed above was implemented in two cases: 1) binaries were sorted by file size and root the binary with the minimum file size was selected as the root; 2) binaries were sorted by number of imports and the binary with smallest number of imports was selected as the root. Next, every binary was compared to all other binaries that were greater than or equal to the file size/number of imports. (According to *Lehman's Laws of Evolution* – as software evolves, so does its size and complexity). Those with the highest similarities in binary metadata, such as: packer, uninitialized or initialized data size, link date, link version, image version, OS version, and file version, were selected as possible descendents. For each of the selected binaries, a difference value was computed by

summing up the number of features added and/or removed between each binary, such as: imports, exports, processes created or injected, runtime dlls, mutexes opened or created, registry set or deleted, and files opened, read, moved, downloaded, written, replaced, deleted, or copied during execution. The binary with the least number of differences in PE data and behavioral statistics (and most similarities in metadata) is then selected as the most likely descendent. Any binaries that have a difference value of zero (i.e. binaries have no difference in their imports, exports, behavior statistics, etc) are marked as equivalent binaries; not descendents.

GROUND TRUTH

To determine accuracy, the PE timestamps (from VirusTotal report) and the timestamp from when the malware was collected/discovered (from Malicia data) were used as ground truth. There were three different cases used as ground truth:

1. Metadata PE timestamps (i.e. link date, compilation timestamp, PE header timestamp). Though these timestamps can be changed or may be inaccurate, it was found that binaries closely similar also had closely similar timestamps; though the timestamps may not reveal the exact date/time when the file was created.
2. "Discovery" timestamp from when malware was collected (from Malicia metadata).
3. Given that the time when a malware is discovered is not necessarily equivalent to its lineage, it was assumed that malware close in lineage/release might also have close discovery timestamps. For example, if malware B is a close descendent of malware A, and both were released to the public, it is possible that malware B gets discovered first, but it is assumed malware A will be detected soon afterwards since their binaries are so similar. Furthermore, according to [2], the median lifespan of the Malicia malwares collected was about 5.5 days. Therefore, given a little more leniency, ground truth was any time within a week of each discovery time, i.e. if malware B (discovered 3/1/2012) was predicted to be a descendent of malware A (discovered 3/3/2012), this would then still be considered valid because B was discovered within a week of A's discovery.

RESULTS

FILE SIZE VS. NUMBER OF IMPORTS

It was found that sorting binaries by number of imports and selecting the binary with the minimum number of imports as the root produced a slightly lower false positive, see Table 3. Imports are not as drastically affected by obfuscation or repackaging of code as would file size.

GROUND TRUTH	**PE TIMESTAMP**		**DISCOVERY DATE**		**LENIENT DISC. DATE**	
Family (# binaries)	File Size	# Imports	File Size	# Imports	File Size	# Imports
Harebot (51)	37%	35 %	35 %	35 %	**19.6 %**	27 %
Cridex (72)	29 %	36 %	36 %	28 %	30.5 %	**19 %**
Zeroaccess (1144)	4.5 %	10 %	6 %	5.5 %	**3.67 %**	5.2 %
Zbot (2160)	9.8 %	**5.7 %**	13 %	7.6 %	10.5 %	**5.83 %**
Winwebsec (5794)	.88 %	**.72 %**	7.2 %	7 %	2.7 %	2.88 %

Table 3. The above table reveals the percentage of malware lineage falsely determined (false positive).

GROUND TRUTH

It was found that using a more lenient discovery date was ground truth produced generally produced the lowest false positive, see Table 3.

CONCLUSION

By performing static analysis on binary metadata and execution behavior statistics, malware lineage can still be determined despite anti-reversing techniques such as code obfuscation or repackaging. The Malicia Project dataset and VirusTotal reports were merged and analyzed to provide more detailed information on each binary i.e. malware binaries, malware family classification and discovery timestamp from Malicia; and behavioral statistics from VirusTotal execution reports of each binary.

Binaries are first matched by similarity in metadata to cluster those binaries similarly repackaged/obfuscated. Computing the minimum difference value (the sum of the number of features added and/or removed) between all binaries determines the binary most closely related.

As a result, malware lineage was determined within 0.7-35% false positive. Further analysis revealed the majority of false positives were due to the ordering of malware by file size or number of imports. Using a more lenient discovery date as ground truth and sorting malware by number of imports generally resulted in a lower false positive.

FUTURE WORK

One of the biggest conflicts presented in this project was the selected collection of malware. The majority of binaries provided by the Malicia Project did not have a very long lifespan. Of the families selected with the longest distribution, the majority of binaries had a very high repacking rate, creating many very short lineage trees. Furthermore, a malware collection with more positive ground truth should be analyzed.

In reference to the methodology implemented, further research needs to be done to improve the selection of the root and the ordering of malware for lineage selection. The majority of false positives were due to file size/imports ordering assumptions.

REFERENCES

[1] Jiyong Jang, Maverick Woo, and David Brumley.'Towards Automatic Software Lineage Inference', USENIX Security 2013.

[2] Antonia Nappa, M. Zubair Rafique, and Juan Caballero. 'Driving in the Cloud: An Analysis of Drive-by Download Operations and Abuse Reporting', DIMVA 2013.

[3] VirusTotal. https://www.virustotal.com, 2013.