The Dropper Effect: Insights into Malware Distribution with Downloader Graph Analytics

Mingfei Gao* Doowon Kim† Tongyang Li‡ Virinchi Srinivas§

February 22, 2016

Introduction

Malicious software (malware) destroys and steals access to users' private computer systems, which can lead to breach of sensitive personal information. Malware can be categorized as virus, Trojan horses, Rootkits, Backdoors, Evasion, etc. It has been rapidly growing, spreading and infecting computer systems; it continues to be an active threat. Currently, more than 200 million unique variants of malware exist. Anti-virus is a software tool that is used to protect against attacks from malwares. Lot of work has been dedicated to detecting malwares which broadly focused on better understanding the properties of malwares such as malwares' network behaviors. However, this approach is not always able to help detecting new malwares because the malwares continuously keep evolving by re-packing and obfuscating themselves. Therefore, a new approach, called content-agnostic techniques, has come into the limelight. Content agnostic techniques do not solely rely on the content of various files (benign/malware) and stresses for the need to focus on the relationship between these different files (downloaders). The need for using content agnostic techniques can be motivated with an example that follows.

Executable files can download a variety of auxiliary malwares, called payloads, which are the main sources of malware distribution. For example, assume that a user downloads Chrome (web browser). Although the web browser is benign, it can download malicious programs. Moreover, the malicious programs downloads supplementary malwares from the Internet. The detection of this kind of the attack is not trivial because downloading software from the Internet is not a clue of malicious intents. However, the *downloader graphs* that can be generated by detecting the relationships among downloaded executable files on a host can provide a better mechanism to understand and capture relationship between different files and droppers during the software download process.

*mingfeigaobupt@gmail.com
†doowon@outlook.com
‡tongyang@cs.umd.edu
§virinchimnm@gmail.com

An example of a downloader graph is shown in Figure 1. From this figure, we can see the basic idea of how the malicious nodes might distribute in the downloader graph. The Internet searching starts at node A which represents a benign Web browser. The node A further downloaded two files, i.e. nodes B and D. We don't know if B or D is malicious because we do not have the labels (benign or malicious) of these Nodes. Then Node B downloaded nodes C and F which are labeled as malwares. Since node B is connected directly to known malwares, we can suggest that node B and all the nodes reachable from node B in the graph are probabily involved in malware distribution. For conciseness, these kinds of malware distributions are called influence graphs as shown in the figure. If we could identify the properties of benign and malicious influence graphs in advance, we would have be able to detect malware and abort it from dropping malicious softwares in the machine.

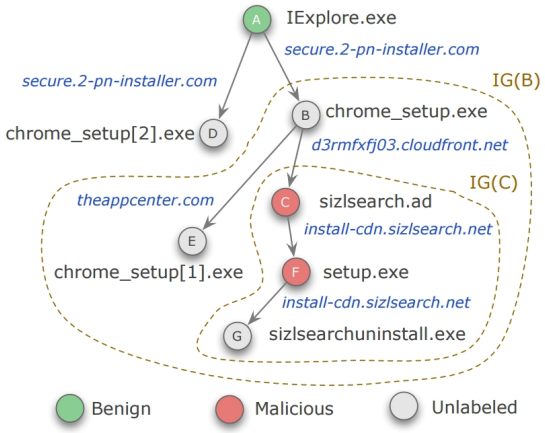

Figure 1: Example of a real downloader graph and the influence graphs of two selected downloaders [1]

Methodology and Results

This work detects malware downloads by graph analytics. To be more specific, *influence graph* is considered, a directed graph defined for each individual downloader on a given host machine, where a node represents a downloaded file and an edge represents a download on the corresponding host machine. Interestingly, malicious influence graphs have many distinct properties compared to those benign ones as shown in Figure 2: their diameters are larger, their growth rates are slower, they tend to downed fewer files per domain, and their URL access are quite different from benign ones. Based on these features, a malware detection system is built by employing supervised machine learning techniques to separate malicious and benign influence graphs. In other words, this work uses graph abstraction to analyze the relationship between different downloaders and other executables they download using downloader graphs, and the method captures relation between different malicious families and increases system performance in terms of detecting malware.

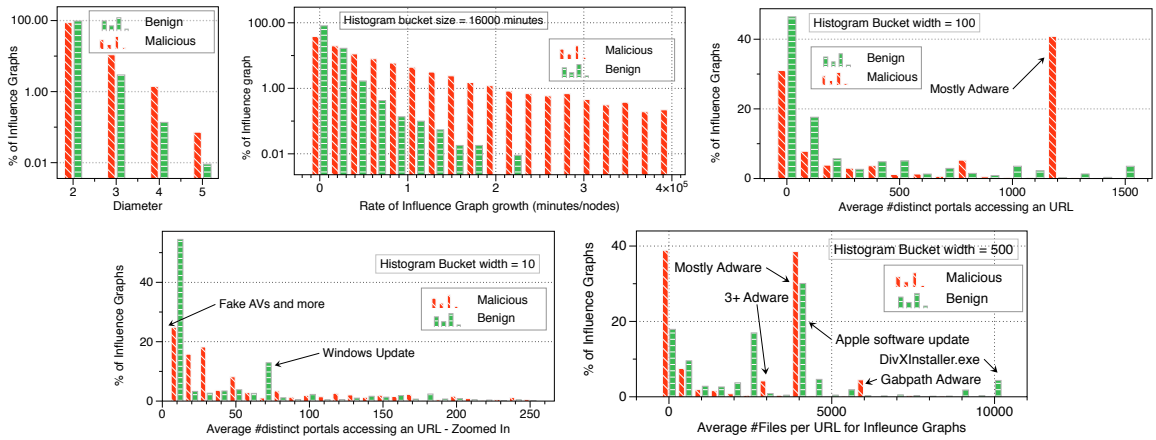

Figure 2: Malicious influence graph vs benign influence graph [1]

Experimentally, download activities were inferred on 5 million end-hosts using data on a platform for data intensive experiments in cyber security and a large number of the data were known as malicious or benign files according to records from two libraries. Experiments are done on 19 million downloader graphs from these 5 million real hosts and the result shows that the proposed classifier achieves a 96.0% true positive rate, with a 1.0% false positive rate.

Future Work

Given this being the first work in this direction, one could extend this system to identify malicious campaigns in the wild. We could use similar graph abstractions to understand the relation between downloaders, executables and domains they access and how they actively coexist in a given time interval. The system proposed in this work, considers static data setting and an obvious extension would be to build a malware detection system when data is presented in a streamed fashion. This would give rise to a complete end-to-end malware detection system.

References

[1] Kwon, Bum Jun, et al. "The Dropper Effect: Insights into Malware Distribution with Downloader Graph Analytics." *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2015.

[2] Mavrommatis, Niels Provos Panayiotis, and Moheeb Abu Rajab Fabian Monrose. "All your iframes point to us." *USENIX Security Symposium*. 2008.

[3] Zou, Cliff C., and Ryan Cunningham. "Honeypot-aware advanced botnet construction and maintenance." *Dependable Systems and Networks, 2006. DSN 2006. International Conference on*. IEEE, 2006.