Graph-based Malware Distributors Detection

Andrei Venzhega
Yandex
16, Leo Tolstoy st
Moscow, Russia
osenzen@yandex-team.ru

Polina Zhinalieva
Yandex
16, Leo Tolstoy st
Moscow, Russia
bondy@yandex-team.ru

Nikolay Suboch
Yandex
16, Leo Tolstoy st
Moscow, Russia
suboch@yandex-team.ru

ABSTRACT

Search engines are currently facing a problem of websites that distribute malware. In this paper we present a novel efficient algorithm that learns to detect such kind of spam. We have used a bipartite graph with two types of nodes, each representing a layer in the graph: web-sites and file hostings (FH), connected with edges representing the fact that a file can be downloaded from the hosting via a link on the web-site. The performance of this spam detection method was verified using two set of ground truth labels: manual assessments of antivirus analysts and automatically generated assessments obtained from antivirus companies. We demonstrate that the proposed method is able to detect new types of malware even before the best known antivirus solutions are able to detect them.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Search and Retrieval

General Terms

Algorithms, Experimentation, Security

Keywords

Graph Mining, Malware Detection, Large Data, Search Engine Security, Webspam

1. INTRODUCTION

Due to Internet propagation malware has been rapidly spreading and infecting computers around the world at an unprecedented rate [1] and malware detection became one of the top internet security topics [2, 11, 3]. Security software developers reported that the release rate of malicious code and other unwanted programs may be exceeding that of legitimate software applications [11].

Search engines (SE) have become one of the principal boosters of malware distribution. Users are looking for software with SE, but sometimes instead of sites of software developers or legal distributors, they get fake websites or malware distributors (MD).

Recently, SEs began to realize their unintentional contribution in malware distribution. To protect users from malware search results they made agreements on cooperation between SE and antivirus companies. Web services enable the identification of malware with a huge partners data about viruses collected, e.g. virustotal.com[1]. But even a huge malware database does not guarantee detection of new ones. Most of anti-malware software products, such as Kaspersky, Symantec, MacAfee typically use the signature-based method to recognize threats[2]. But malware writers successfully invent counter-measures against proposed malware analysis techniques. Today's malware samples are created at a rate of thousands per day. According to Symantec's annual report [11]: 5,5 billion malware attacks were blocked in 2011, 81% more than in 2010. More than 403 million new types of malicious software were detected in 2011, 41% more than in 2011. Symantec reports huge amount of blocked malware, but they estimate that new malware techniques are able to generate an almost unique version of their malware for each potential victim. This suggests traditional signature-based malware detection solutions will likely be outpaced by the number of innovative threats being created by malware authors. A new radically different approach to the problem is currently needed.

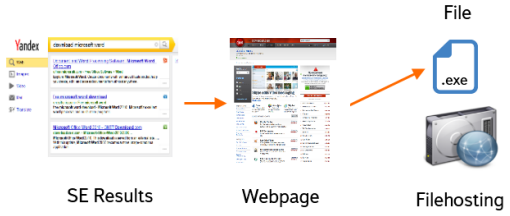

Figure 1: Malware distrubution via SE

SE companies are the first who face a threat from new malwares. That is why early detection of new malware and in particular their distributors is the principle objective of ensuring safe and high-quality web search. Some websites even if they are not MDs, but closely related to the distributors, for example, linked with hyperlinks, can also be dangerous. We can even suspect them of intentional cooperation with distributors of viral software. Therefore to find suspicious websites, we propose an approach that consists in spreading information about MD via connections between neighbours, which is similar to the idea of homophily. We used a bi-

WWW 2013 Companion, May 13–17, 2013, Rio de Janeiro, Brazil.
ACM 978-1-4503-2038-2/13/05.

[1]https://www.virustotal.com/about/credits/
[2]http://en.wikipedia.org/wiki/Antivirus_software/

partite graph with two types of nodes: website and FH. An edge represents the fact that a file hosted on FH and can be downloaded from the website.

2. RELATED WORK

Antivirus companies propose several mixed methods to detect malware files. They are based on file's data and an additional information about relations between files (e.g. usage of similar libraries, similar behavior on infected computers etc.): Polonium Technology by Symantec [2] is based on a customized belief propagation method on a huge bipartite graph with two types of nodes: user machines and files. Edges are denoting a file appearing on a user machine. They address the problem of identifying malware is to locate files with low reputation.
Valkyrie Technology by Comodo [1] is based on a semi-parametric classifier model to combine file content and file relations for malware detection.

We considered several graph mining techniques to detect web spam which could be useful to deal with MD. Most of them are based on well-known PageRank [4] and HITS [5] algorithms. One of HITS customizations is LiftHITS[6] with special edge attributes. Another trust propagation technique TrustRank [7] based on the PageRank was developed to deal with web spam and requires manually specified seed of reputable pages to initialize the method. There is another approach to detect spam - WITCH [8] method which combines two techniques: graph based and traditional webpage data analysis. Semi-Supervised PageRank [9] is more common approach than WITCH with the similar idea to use content-based features and PageRank advantages as well. MapReduce logic was used to deal with computations on a huge graph.

These graph based methods are suitable to detect traditional types of web spam, but webpage data (collected from html and links) is not enough to detect MD. Polonium Technology seems to be a good approach to solve a problem of malware files detection, but it is not our main goal. The major challenge faced by SE is not to detect files but malware distributors - webpages or websites, using all available data about file downloads on webpages.

Eventually we propose a new combined method to detect MD, based on ideas of antivirus method [2] and graph-based antispam methods [6, 7, 8] as well.

3. DATA DESCRIPTION

We used anonymized user data about downloading files by following links on webpages to detect MDs. User data was collected for the period from 1 to 7 August 2012 via a specialized browser toolbars. We obtained 26,517,355 records about downloads with the following information: the webpage where download of the file was started, file type (MIME type), the FH which hosts the file, the date of file download. We filtered out some downloads by file types: images, audio and video files, torrents. In this paper we assume that these file types can not be malware. Examining examples of the MDs, we found that if a noticeable subset of all webpages of a website distributes malware, then we can suspect other files from this site being malware. We also found out that groups of suspicious sites often use a shared set of FH to store files, so we made an assumption about the presence of relationships between some MDs. We also noticed that

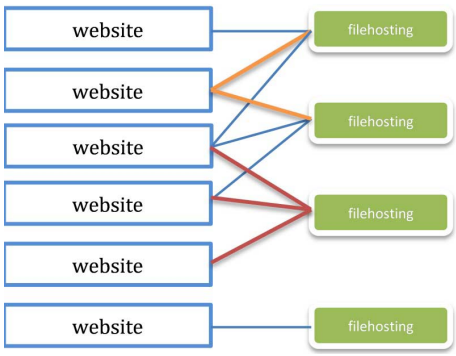

Figure 2: Bipartite graph of websites and file hostings and connections between them

MD often use FH with the names similar to well-known FH or software brands, e.g.: *media.fi1es.net, depostfiiles.com, fastfiledown10ader.com, msoffice.dld12.net* etc.

Thus we decided to transform the structure "website - webpage - file - FH" to the "website - FH". A pair "website - FH" is formed when at least one document of the website links to a file that is hosted on the FH. Each pair is weighted by the number of times a file was downloaded from the FH by following a link from the website. If the file is hosted on the same website, the website can be represented by two connected nodes in the graph: "website" as a site and "file hosting" as a FH.

$G = (V, E, W)$
V – vertices of two types: site $S \subset V$ and filehosting $F \subset V$;
$S \cap F = 0$; $S \cup F = V$;
E – the edges of the graph: $E = \{(s, f) : s \in S, f \in F\}$;
$W = \{(w_s(s, f), w_f(s, f)) : \ s \in S; f \in F\} \in [0, 1]^2$;
– edge weights set of two types:

$w_s(s, f)$ - fraction of files which were downloaded by following a link from site s and being hosted on f from all downloads from s.
$w_f(s, f)$ - fraction of files which were downloaded by following a link from site s and being hosted on f from all downloads from f.

$m(v)$ — malwareness rank of vertex $v \in V$, $m \in [0, 1]$. We exploited an idea from Belief propagation [10] to make an evaluation of the maximum likelihood state probabilities of nodes being malicious. In other words $m(s)$ is an estimation of probability of s being MD and $m(f) = 1$ means FH f hosts malware files only.

To initialize method we used an initial seed of trusted and malicious FH. Let F^0_{bad} and F^0_{good} be an initial seed of malicious and trusted FH respectively, then $m(f_i) = 1, \forall i \in F^0_{bad}$ and $F^0_{good} : m(f_j) = 0, \forall j \in F^0_{good}$.

Eventually we made a graph consisting of 293.557 websites, 305.677 FH and 622.137 edges. Each FH stores 48 files on average, about 50 files can be downloaded using links from one website. 82,9% of websites host at least one file using their own hosting. But there are websites, which host files on a single FH that is used by this website only, therefore they are not connected to the other graph components.

We call them autonomous sites, which constitute 55,2% of sites in our data.

4. ALGORITHM

During preliminary experiments with the graph, we learned that we need to use the sigma function that allows to prevent distribution of tiny ranks or "noise" on the graph, which interferes method convergence.

$$m_\sigma(v) = \begin{cases} m(v), & m(v) >= \sigma; \\ 0, & m(v) < \sigma. \end{cases}$$

At the first step we have initialized graph $G = (V, E, W)$; with prior set F^0_{bad} of bad FH vertices and prior set F^0_{good} of good FH vertices. We set $m(v) = 0$ to all other vertices, proceeding from the assumption they are a priori not bad. The idea is to find other suspicious vertices using an idea of homophily - to spread information about bad neighbours through graph vertices. We propose an iterative algorithm based on HITS technique. At each iteration t rank m is consistently calculated for all sites and all FHs with following steps:

1) computation of site malwareness using neighbours values of FH ranks ($m_{t-1}(f)$) calculated on previous iteration

$$m_t(s_i) = \sum_{j \in N_i} \frac{w_s(s_i, f_j) \cdot m_{t-1}(f_j)}{W_s(s_i)}, \forall i \in S;$$

where $N_i -$ set of all neighbours of vertexs_i;

2) computation of FH malwareness using neighbours values of websites ranks ($m_{t-1}(s)$) calculated on previous iteration

$$m_t(f_j) = \sum_{i \in N_j} \frac{w_f(s_i, f_j) \cdot m_t(s_i)}{W_f(f_j)}, \forall j \in F_u;$$

$$F_u = F \setminus (F^0_{bad} \cup F^0_{good}) - \text{FHs not from the prior set.}$$

$$m(f_i) = 1, \forall i \in F^0_{bad},\ m(f_j) = 0, \forall j \in F^0_{good};$$

$$W_s(s_k) = \sum_{m \in N_k} w_s(s_k, f_m),\ W_f(f_l) = \sum_{m \in N_l} w_f(s_m, f_l)$$

— the normalization coefficient is equal to the sum of weights of edges of all the neighbours. It allows to meet the condition: $m(v) \in [0, 1]$.

The stop condition of iterative method on t iteration:

$$\|m_t(v) - m_{t-1}(v)\| < \varepsilon;\ \forall v \in V;$$

If in a result of the method some vertex (website or FH) has $m(v) > \theta$ than we recognize it as bad or malware distributor.

The principal disadvantage of this method: the elements of the graph, which are inaccessible from the initial set could not be analyzed or detected. Autonomous sites are among them, so they can be analyzed with antivirus information only.

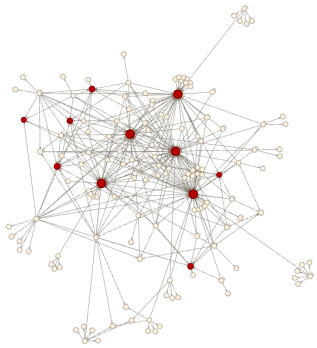

Figure 3: Bipartite graph initialization ($t = 0$). Vertices painted red are in F^0_{bad} prior set.

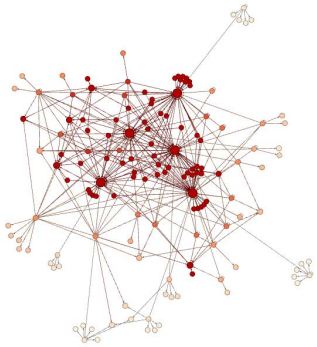

Figure 4: Bipartite graph after $t = 2$. Vertices intensity of red color represents value of badness rank. The vertices of the most saturated red color have $m_2(v) = 1$;

The expression above can be rewritten in matrix form. But matrix operations on large graphs are very resource intensive. A sparsity of relationship matrix follows from the graph topology, therefore we refused matrix computations. Operations performed on the iteration can be parallelized, thus we decided to use MapReduce approach [12] to compute malwareness rank. A similar technique of computation on a large graph is described in paper [9].

4.1 MapReduce

We used a special distributed system designed for efficient computations on the large data. During the process of computing the graph is divided into portions, every portion is processed by a separate computer node. The computing process consists of consecutive iterations. At the start of each iteration for each of the vertices there is a set of incoming messages. Then a specific function is called for each vertex. Function uses current vertex value and a set of incoming messages. After that, this function can change the vertex value and send messages to other vertices. At the next iteration these messages are available for the destination vertices. The computational process is performed in two steps:

MAP: sending messages to vertices. In our case - to send a set of neighbours ranks from the previous iteration $m_{t-1}(v_j)$, $j \in N_i$ to each vertex v_i.

REDUCE: receiving messages at the vertex and calculating the new aggregated value. In our case - computing malwareness rank $m_t(v_i)$ of vertex v_i using the information received from the MAP step.

5. IMPLEMENTATION

Our experiments started from the initialization of the graph described in Section 2. We obtained 52 FH distributing malware files, which were taken from manual assessments of antivirus analysts and automatically generated assessments obtained from antivirus companies. These FH were marked as "malicious". The values of $m(f)$ for these vertices were fixed as 1. The set of 140 most popular FH and official software distributors was marked as "trusted". Here some sites from this set: apple.com, microsoft.com, avira.com, firefox.com, kaspersky.com etc. The values of $m(f)$ for these trusted FH were set to 0. The value of $\sigma = 0.05$ from $m_\sigma(v)$ was set up empirically using some held-out test data to meet the method's stop condition.

As a result, we set malwareness threshold value of website and FH to $\theta = 0.5$. We figured out that parameters values: $\sigma = 0.05$, $\theta = 0.5$, $\varepsilon = 0.05$ allow our method to converge after 2 iterations.

When we collected the data for the subsequent periods of time, it turned out that the graph has significantly changed during these periods. Thus, the assessment became irrelevant with time. We believe that MD webmasters are probably aware of existing methods of their detection and proactively change FHs frequently, abandoning the old ones. Our assessment of the minimum lifetime of such FH is 1 week, but on average lifetime of MD websites is much longer. So, we decided to run our method every time we obtain the updated data and to use previous results to initialize the method, as well as the new data obtained from malware analysts and antivirus software developers. To minimize the false positive mistakes, random sets of MD detected by a method are double-checked periodically by a group of antivirus analysts.

We detected 209 FH distributing maleware files with 97% accuracy (with $\theta = 0.5$) after the first run and 1239 FH after 12 weekly launches with 98% accuracy. We also detected 2454 MD websites connected with these FH with 98% accuracy. After these websites had been banned by SE antispam policies, we observed 2.1 times decrease of the average number of malware distributers observed in top-10 search results. The approximate number of malware downloads on websites which were detected by this method was decreased about 3,4 times according to the toolbar logs.

Only 9% of files from malware FHs discovered by the method were detected with signature based anti-virus scanning at the moment of their detection. But almost all (96%) of these files were recognized as malware after manual verification. After two weeks we checked antiviruses markup again, eventually they confirmed more than 90% of files from our old results FHs. So we obtain an information about malware FHs (and that means files) noticeably faster than antivirus companies do. The training set of non-autonomus malware FH was marked up by the method with 86% recall. It is hard to estimate recall precisely, because the method's recall considerably depends on the initialization set. We are unable to use all antvirus data to measure recall, because in contrast to antivirus software, our method is focused on malware FHs detection, not the files. However, the high accuracy and speed of detection demonstrates the effectiveness of the method.

6. CONCLUSION

We proposed a new effective method to detect malware distributors. A large anonymized log of downloads of files linked from webpages was used to create a bipartite graph of websites and filehostings. The idea of the homophily was used as the underlying principle of the proposed graph mining technique to detect malware distributors. MapReduce programming model was used to deal with large-scale data computations.

This approach was adopted by Yandex SE antispam that caused a significant decrease of websites distributing malware in search results. We plan to explore new opportunities of application of this approach, particularly we expect to use it for other types of spam detection, as well to improve the current method, using additional information about websites.

7. REFERENCES

[1] Yanfang Ye, Tao Li, et al. Combining File Content and File Relations for Cloud Based Malware Detection. In proceeding of the 17th ACM SIGKDD, 2011.

[2] D. Chau, C. Nachenberg, et al. Polonium: Tera-scale graph mining and inference for malware detection. In Proccedings of SIAM SDM, 2011.

[3] Egele, M., et al. A Survey on Automated Dynamic Malware Analysis Techniques and Tools. In: ACM Comput. Surv., 44(2): p. 1-42, 2012.

[4] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine, Computer networks and ISDN systems, vol. 30, no. 1-7, pp. 107-117, 1998.

[5] J. Kleinberg. Authoritative sources in a hyperlinked environment, Journal of the ACM (JACM) Vol. 46 Issue 5, pp 604-632, 1999.

[6] H. Chang, D. Cohn, and A. K. McCallum. Learning to create customized authority lists. In the proceedings of the 17th ICML, pp 127-134, 2000.

[7] Z. Gyongyi, et al. Combating web spam with trustrank. VLDB Endowment, 2004, p. 587.

[8] J. Abernethy, O. Chapelle, and C. Castillo. Web spam dentification through content and hyperlinks. In the proceedings of AIRWeb'08, 2008.

[9] Bin Gao, et al. Semi-supervised ranking on very large graphs with rich metadata. In the proceedings of the 17th ACM SIGKDD, pp. 96-104, 2011.

[10] S. Pandit, D. Horng Chau, S. Wang, C. Faloutsos. Netprobe: a fast and scalable system for fraud detection in online auction networks. WWW'07, pp. 201-210. ACM, 2007.

[11] Symantec internet security threat report. 2011 *www.symantec.com/content/en/us/enterprise/other_resources/b − istr_main_report_2011_21239364.en − us.pdf*

[12] J. Dean, S. Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. Communications of the ACM, vol. 51, no. 1, pp. 107-113, 2008.