ISBN 978-981-11-3671-9
Proceedings of 2017 the 7th International Workshop on Computer Science and Engineering (WCSE 2017)
Beijing, 25-27 June, 2017, pp. 100-104
doi: 10.18178/wcse.2017.06.017

Evaluation of non-Randomness DGA Detection Method to Combat Ransomware

Marko Niinimaki [1+] and Reijer Idema [2]

[1] Department of Computer Science, Webster University, Bangkok, Thailand
[2] JOC Consulting, Liberty Tower fl. 12A, Thong Lo, Wattana, Bangkok, Thailand

Abstract. Since 2009, ransomware has plagued computer users. By analysing ransomware's communication with its controller, researchers have found out patterns used in naming the controller's domain. These names are generated by a Domain Generation Algorithm (DGA). It has been proposed that DGA generated domain names appear more random than actual domains registered for a legitimate purpose. If this was the case, we could block communication to such domains.
In this paper, we analyse the feasibility of detecting DGA generated domain names based on their randomness. We compare a large (800 000) list of actual DGA domain names with (i) a list of most popular domains in the internet and (ii) a list of actual queries to a domain name service. Unfortunately, it seems very difficult to block communication with apparently random domain names. Though some DGA generated names are apparently random, the perceived randomness of both popular domain names and actual queried names is often greater than that of DGA domains.

Keywords: DGA, Ransomware, Randomness.

1. Introduction

Ransomware is malicious software which, when run, disables the functionality of a computer in some way, typically by encrypting the user's files. The ransomware program displays a message that demands payment to restore that functionality (by decrypting the files again) [1]. This extortion is often successful. In 2016 it was estimated that ransomware now cost small and medium companies at least $75 billion in expenses and lost productivity annually [2].

Typically, a computer gets infected by ransomware when a user inadvertently downloads such malware (or its loader) from a web site. Moreover, an infected computer communicates with the ransomware's command and control (C&C) server, either by using an IP address or a host name. Thus, it may sound a feasible defense to block access to suspicious sites and C&C servers. This is often done by using blacklists in a firewall or even inside a web browser [9]. To avoid blocking, malware developers utilise a domain generation algorithm (DGA). The idea of a DGA is to use a different domain name based on time: a DGA periodically generates a number of domain names that can be utilized as C&C servers [3]. Since these domain names change frequently and are seemingly unconnected, blacklisting them will be difficult.

There are several ways to combat DGA-based malware. In this paper, we evaluate a method proposed by Baggett in [4]: the "randomness" of the domain name indicates that the domain is indeed DGA generated. We study the feasibility of this idea by checking how the randomness of DGA generated names differ from names of popular internet domains and from a set of local domain name queries.

+ Corresponding author. Tel.: +66 21066599.
E-mail address: niinimakim@webster.ac.th.

The remainder of this paper is organized as follows. Section 2 gives account of previous work. Section 3 explains our analysis of the non-randomness DGA detection method. Finally, Section 4 contains conclusions and further discussion.

2. Previous Work

According to Symantec [1], ransomware first appeared in 2009 in Russian speaking countries. The idea of using a domain generation algorithm appears at the same time, first in the context of botnets [5]. Stone-Gross et al refer to this dynamic communication with a command and control server (C&C) as "domain flux". There "each bot uses a domain generation algorithm (DGA) to compute a list of domain names. This list is computed independently by each bot and is regenerated periodically. Then, the bot attempts to contact the hosts in the domain list in order until one succeeds". The DGA presented by Stone-Gross et al (and used by malware Torpig) is not very complex so it is possible to programmatically check if a given domain name is generated by it. However, modern DGA's use complex customised features. For instance, the Matsnu DGA creates domains that are comprised of a noun, verb, noun, verb combination until the domain is 24 characters long [6].

Schiavoni et al [7] present some features of the malware-C&C communication that can be used to detect a DGA. As in [5], they note that it takes many attempts to reach the C&C server since indeed the malware tries to contact the hosts in the domain list in order until it succeeds. This leads to suspicious number of failed domain name queries. Thus, a firewall or similar software that traces such failures (NXDOMAIN responses from a Domain Name System) can block the communication, hopefully before the malware reaches the actual C&C server.

Another obvious defense is to get intentionally infected by a ransomware and observe the way it sends Domain Name System queries (or reverse-engineer the DGA). "Knowing" what domain names the DGA's are likely to generate gives us at least two ways of fighting them: (i) implement a program that for each accessed domain name checks if it could have been generated by any DGA and run this program in the user's PC or (ii) observe currently active DGA's and provide a list of domain names that they will use within the next hours. Option (i) may be simply too complex to maintain and will drain the PC's resources. For option (ii), computer security organizations like abuse.ch maintain lists of domain names that are generated by DGA's [8]. It is unlikely that a new ransomware will be able to infect all its potential victims simultaneously. Thus, an up-to-date blacklist will protect most users. However, the domain black lists are necessarily large since there are many active DGA's. Checking each Domain Name System query against such a list, and even copying such a list to an individual PC would take much of the PC's resources. If by an additional method we could eliminate blacklists or at least limit their size, we would gain both in saved CPU cycles and data transfer. An example of a blacklist is shown in Figure1. The blacklist compiled by a computer security company from several sources contains about 800 000 entries. The domain names in the list are generated by known, active DGA's.

111g6nl1klpfqt16nfybh1d56ia9.net
bzgrpartbulkyf.com
1yt0yqt1901yv61aaub2e1xsfrod.com

google.com
youtube.com
facebook.com
adexchangeprediction.com
1tv.ru

Fig. 1: Examples of entries in a DGA blacklist (above) and most popular domain list (below).

3. Non-Randomness DGA Detection Analysis

In this section, we study the feasibility of the method proposed in [4]. There Baggett proposes a solution that identifies domain names generated by typical DGA algorithms: they differ from domain names that

humans would use or register, because they are too random. Thus, we need a method to recognize non-randomness in words.

A simple way of analysing text is using bigrams (pairs of consecutive letters). Given the frequency that each possible bigram occurs in a specific language, a measure M can be calculated for the probability that some text is written in that language. In this case, the method simply sums up the probabilities that letter i is followed by letter i+1 and divides the sum by the number of letters. It is a crude measure, but it may give an indication on whether a domain name is automatically generated or not. We shall call the measure M "non-randomness". With a bigram generated from English language texts, we can observe that actual popular site names exhibit higher non-randomness than DGA-generated ones: google.com 6.63315, youtube.com 10.51936 facebook.com 6.57618 compared with 111g6nl1klpfqt16nfybh1d56ia9.net 1.62286, 1yt0yqt1901yv61aaub2e1xsfrod.com 2.36986 and bzgrpartbulkyf.com 4.93711.

We use an existing bigram frequency implementation to analyze a list of 800 000 DGA generated domains (Fig1) and a list of popular internet domains. The "1 million most popular domains" list is compiled by Alexa corporation.[1] In what follows, we shall call the 1 million domain sample "Alexa" and the DGA domain list "DGA".

We first observe differences in domain name lengths in the Alexa and DGA. Alexa has some very short domain names like "z.cn". The mean length of a domain name is 14.4 characters (including dots). In DGA the names are longer. The shortest domain name length is 8 characters and the mean length is 21.4 characters.

Table I shows the main features of non-randomness for Alexa and DGA. Given some very low values in the Alexa sample, the simple approach is not good enough to make a distinction between real popular domains and DGA generated domain names. In fact, a popular domain "m3q.jp" is ranked very random (0.01046) by our method. Likewise, the domain generation algorithm manages to produce a domain name "thezererwyatanb.com" that is ranked quite non-random (12.86393). If we used 12.86393 as a threshold ("only domains with non-randomness higher than this are valid"), we would block ca 97% of the most popular domains.

Table I. Summaries of Non-Randomness Detection on Three Samples

Sample	Min	Max	1st q	Median	Mean
DGA	0.01845	12.86393	3.22830	5.29142	5.03187
Alexa	0.01046	25.69059	5.52006	6.94199	6.90279
Alexa8+	0.02358	25.69059	5.74585	7.01283	7.01131

However, we can relatively safely assume that DGA's no not use short domain names. Either such names are already taken (like the real m3q.jp) or there simply are not enough combinations available for a DGA. We generate a sample "Alexa8+" by including domain names whose length is at least 8 characters. But this improves the situation only marginally (see Table 1). The distributions of non-randomness for the samples are shown in Figure 2.

462207 (57%) of DGA domains have lower non-randomness value than 5.74585 that is the first quartile of Alexa8+. However, even after eliminating short names, there are only a few DGA domains whose non-randomness rank is less than that of "txjsjzc.cn" (a legitimate site in Alexa8+).

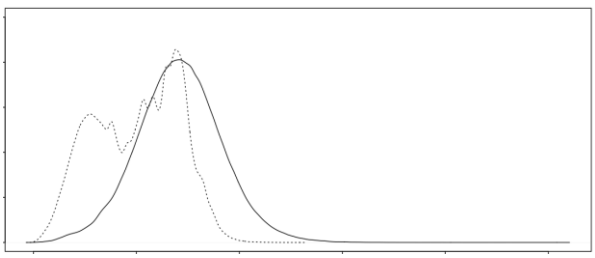

Fig. 2: Distribution of non-randomness in Alexa8 and DGA (dotted line) samples.

[1] Alexa's popularity rank is an estimate, see `http://www.alexa.com/about`.

Finally, we study if the non-randomness method would work on a sample of actual Domain Name System queries.[2] The idea of this analysis is that we do not limit ourselves to well-known domains of the world but test a more localized set. Our sample has 562 199 queries whose distribution is very zipfian: just 10 most popular queries represent 20% of all queries. 82 037 of the queries are unique (set DNSu).

About 99% of the (unique) names queried do not appear in the Alexa sample.[3] The summary for DNS is shown in Table II.

Table II. Summaries of Non-Randomness of DNS Queries

Sample	Min	Max	Median	Mean
DNSu	0.08133	19.67079	5.99333	5.96304

The sample of 562 199 non-unique queries can be used to check how many false positives we would get if we used a rule-of-thumb style measure to block seemingly random (and thus potentially DGA generated) domain names. We select value 7.01283 that we found to be the median of the Alexa8+. Blocking access to domains with names apparently more random than this would, however, block 29% of the queries.

4. Summary and Conclusions

In this paper we studied randomness of domain names in order to identify domain names that are used by ransomware. If such domain names could be identified by a quick method, communication with the domains could be blocked and a ransomware infection avoided. This would be more efficient than using DGA-domain blacklists that are large and thus costly to keep updated. Moreover, computing the randomness measure for a domain name is quite fast. The Python implementation of [4] can rank a list of 1 million domain names in about 1 minute,[4] so it would be possible to decide real-time if a domain should be blocked or not.

We compared actual DGA domain names with two samples (i) Alexa's 1 million most popular domain names and (ii) a list of domain name queries to an actual domain name server. Though both the samples had less randomness that DGA-generated domain names, a definite identification is very difficult. A DGA-generated domain name thezererwyatanb.com was ranked very non-random (12.86393) whereas some domain names in the Alexa list appeared random (bbc.co.uk 2.50081). In the Alexa list, there were only ca 30 000 domain names with a non-randomness rank higher than 12.86393. In the domain query list, there were 3000 (out of 82 000). In most cases, we will still need to resort to blacklists or other methods.

Additionally, the following improvements did not improve the situation significantly:

- Removing the top-level domain (.com, .co.uk, etc) from the domain name.
- Using a customized bigram based on Alexa, instead of standard English.
- Restricting the analysis to .com domains to rule out domains like "txjsjzc.cn" that are obviously not English names. Yet, there is a popular site wwwwwwwwwwwwvwwwwwwwwwwwwwwwwwwvwwwwwwwwwwwwwwwwwwwwwwwwwwwwvww.com.

We can improve the detection by modifying the non-randomness algorithm. The bigram frequency tool, by default, uses bigram probabilities that are conditional to the starting letter. That is, the bigram "qu" gets a very high probability (99.9%, although the tool caps it to 40% by default) because a "q" is almost always followed by a "u", while "en" gets a much lower probability (8.4%) because "ed", "es", "er", etc. are also common. However, in absolute numbers, "en" is 10 times more like to show up in an English text than "qu".

Figure 3 demonstrates non-randomness generated using a modified version of the bigram frequency tool. There we use absolute probabilities of bigram occurrence instead of the default conditional ones. We can see

---

[2] Incidentally: these queries do not contain any DGA generated domain names.

[3] This sounds unusual, but the queries are for full host names like www.google.com.bz whereas Alexa only lists google.com, google.es etc. If we consider only names like google.com, 10 250 of the unique names (12.5%) appear in Alexa's 1 million list.

[4] OSX 10.10.5, 1.4 GHz Intel Core i5.

the randomness of some DGA domains is more pronounced than in Figure 2. However, in general this will not be sufficient to determine if a domain name is DGA generated.

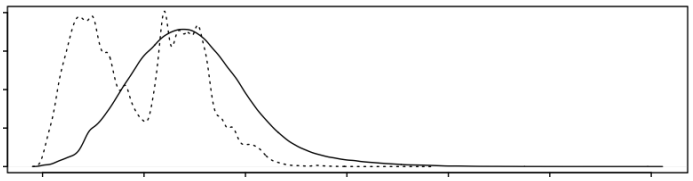

Fig. 3: Distribution of non-randomness in Alexa8 and DGA (dotted line) samples using absolute probabilities.

Using an actual set of ca 560 000 DNS queries, we tested if blocking access to sites with seemingly random names would lead to many false positives. This would be the case since 29% of these valid queries would be blocked.

5. Acknowledgment

The authors would like to thank JOC Consulting Thailand for a permission to use a DNS query sample.

6. References

[1] G. O'Gorman and G. McDonald. Ransomware: a growing menace. Symantec Corporation, 2012.

[2] A. Chandler. How ransomware became a billion-dollar nightmare for businesses. Atlantic Monthly Sep 3, 2016.

[3] K. Cabaj and W. Mazurczyk. Using Software-Defined Networking for Ransomware Mitigation: the case of Cryptowall. IEEE Network, 30(6):14–20, 2016

[4] M. Baggett. Detecting random - finding algorithmically chosen DNS names (dga). https://isc.sans.edu/forums/diary/Detecting+Random+Finding+Algorithmically+chosen+DNS+names+DGA/19893. Accessed: 2017-01-07.

[5] B. Stone-Gross, M. Cova, L. Cavallaro, B. Gilbert, M. Szydlowski, R. Kemmerer, C. Kruegel, and G. Vigna. Your botnet is my botnet: analysis of a botnet takeover. In Proceedings of the 16th ACM conference on Computer and communications security, pages 635–647. ACM, 2009.

[6] A. Raff. DGAs: A Domain Generation Evolution, Seculert Blog, http://www.seculert.com/blogs/dgas-a-domain-generation-evolution. Accessed: 2017-01-07.

[7] S. Schiavoni, F. Maggi, L. Cavallaro, and S. Zanero. Phoenix: Dga-based botnet tracking and intelligence. In International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment, pages 192–211. Springer, 2014.

[8] M. Kührer, C. Rossow, and T. Holz. Paint it black: Evaluating the effectiveness of malware blacklists. In International Workshop on Recent Advances in Intrusion Detection, pages 1–21. Springer, 2014.

[9] S. Sheng, B. Wardman, G. Warner, L. F. Cranor, J. Hong, and C. Zhang. An empirical analysis of phishing blacklists. In Proceedings of Sixth Conference on Email and Anti-Spam (CEAS), 2009.