Finding Domain-Generation Algorithms by Looking at Length Distributions

Miranda Mowbray
Security and Cloud Lab, HP Labs
HP
Bristol, UK
miranda.mowbray @ hp.com

Josiah Hagen
Tipping Point, HP Software
HP
Austin, TX, USA
josiah.hagen @ hp.com

Abstract—In order to detect malware that uses domain fluxing to circumvent blacklisting, it is useful to be able to discover new domain-generation algorithms (DGAs) that are being used to generate algorithmically-generated domains (AGDs). This paper presents a procedure for discovering DGAs from Domain Name Service (DNS) query data. It works by identifying client IP addresses with an unusual distribution of second-level string lengths in the domain names that they query. Running this fairly simple procedure on 5 days’ data from a large enterprise network uncovered 19 different DGAs, nine of which have not been identified as previously-known. Samples and statistical information about the DGA domains are given.

Keywords—domain generation algorithms; DGA; AGD; botnet; Domain Name Service; Big Data

I. INTRODUCTION

It is common for botnets to use domain fluxing in order to circumvent domain blacklisting. In domain fluxing, an infected machine periodically uses a seed and a domain-generation algorithm (DGA) to automatically generate a large batch of domain names, and does a Domain Name Service (DNS) query to discover the domains’ IP addresses [1]. The seed may be time-independent, or may depend on the current time or other time-varying public information such as top Twitter trends [2,3]. The command-and-control server contacts an infected machine by registering and using one of the generated domain names. If this domain is blacklisted, it can move to another of the generated domains. If a machine can be detected querying malware DGA domains, it may be possible to disinfect or quarantine it before the infection produces any symptoms.

The usual approach to detecting clients querying DGA domains is to use machine learning to build a classifier based on features of DGA domain names or of queries for them [4,5]. This approach only detects DGAs for which samples are available. There is therefore a need to identify previously-unknown DGAs, and obtain samples for them. This is the problem we address in this paper.

In Section II we describe a relatively simple mostly-automated procedure that can be used on DNS query data to detect sets of domains produced by previously-unknown DGAs. The procedure identifies only a small number of clients querying a DGA’s domains – often only one – but once samples from a new DGA have been found, other clients querying domains from the DGA can be identified.

The main contributions of the paper are the descriptions of the unidentified DGAs, and the description of the procedure used to find them. The identified DGAs that were found might also be of interest.

To explain the main idea behind the procedure, we need to introduce some terminology. The term second-level domain refers to the substring of a domain name consisting of all the characters after the second-last dot before the public suffix. The DGAs used for second-level domain fluxing generate domains with many different second-level domains. The substring between the last and second-last dots before the public suffix will be called the 2nd-level string, and its length the 2-length. For instance, both s1.s2.example.com and example.co.uk have 2nd-level string example and 2-length 7. The top-level domain is the substring of a domain name after the last dot, and the suffix is just the public suffix.

The main idea is to look for clients querying domains with an unusual distribution of 2-lengths. Thus, the procedure we describe can only detect DGAs that are used for second-level domain fluxing, and which generate batches of domains for which the 2-length distribution is unusual.

However, when the procedure was run on 5 days’ DNS query data from a large enterprise network, it uncovered 21 different DGAs, nine of which have not been identified and so may be previously-unknown. We describe these in Section III. Some of these may be produced by legitimate applications rather than malware, and some or even all of them may be known, but we have not found previous reports of them.

Once domains from a new malware DGA have been detected, they can be used to train a classifier so that this DGA can be detected in the future. We are currently using a classifier built using multi-feature logistic regression with L2 regularization [6], using syntactic features of second-level domain names. The features we use include counts of individual characters and character n-grams, dots, hyphens, digits, uppercase, and lowercase, and the length. The classifier performs well, with F1 scores > 95% generally, over several classes: we currently have 12 DGA classes, but aim to add new classes for the DGAs found.

A. Other Approaches

There are other approaches to detecting new DGAs. In some cases DGAs can be determined from captured
malware code, or by capturing the DNS traffic of a machine infected with malware that uses a new DGA. However this requires the prior identification of the infected machine.

One approach [7] looks for domain names that are not human-readable or human-understandable. Similarly, another more recent approach uses linguistic features from the domain names [8]. However, some DGAs are designed to produce human-readable domain names. Most bigrams in Pushdo and Expiro domains consist of a vowel and a consonant, and Dorkbot domains are derived from English words. Moreover, some benign domain names are not obviously human-understandable.

A different idea is to look for clusters of similar domains being queried by multiple clients [9,10,11]. This only catches infections in their later stages, when they have spread to several clients. The approach described in this paper potentially allows the detection of a DGA at the end of the first time slot during it is used by the first infected client in the network. However it cannot detect all DGAs. It makes sense to combine these two approaches, so that if a DGA is not detected early it may be detected later on.

B. Privacy Considerations

The reason for identifying clients based on very limited information about the domain names is that this allows for greater privacy protection. For each time slot, the automated part of the procedure identifies a few client IP addresses of interest, and the un-automated part examines the unusual domains queried by these IP addresses during the time slot. The procedure is designed so that most of it can be run on pseudonymized data, where both the IP addresses and the domains are pseudonymized. The pseudonymization of the domains preserves the public suffix and the 2-length, but may not preserve any other features. Non-pseudonymized domains need only be examined for the candidates that have been identified as having unusual 2-length distributions, and the non-pseudonymized version of a candidate IP address need not be revealed until it has been confirmed that it is infected and action is needed to remedy this.

C. The Data Set

In this paper we report results for five days’ DNS queries collected from a large enterprise network on April 9, May 12, June 11, July 18, and August 4 2014. The dates were selected at random from the days for which data was available, with the constraint that they are all from different months. The data set indicates, for each 3-hour time slot, active client IP address, and second-level domain, how many DNS queries were made by the client IP address of a domain with that second-level domain. There were 224,818 active client IP addresses, querying domains with over a million different second-level domains a day.

II. The Procedure

This section describes the procedure used to find DGAs in the data set, and the reasons for the choices made in the design of this procedure.

The first step was to convert upper-case letters in the second-level domain names to lower case. Doing this conversion is standard practice when searching for malware domains in DNS data, as the DNS service is case-invariant, and also it ensured that client IP addresses making many queries of the same domain with different capitalization as a defence against DNS spoofing [12] would not appear to query many different domains with the same 2-length.

The data entries for second-level domains with invalid public suffixes were deleted. Malware DGAs need to generate domains with valid public suffixes, so the second-level domains considered in the search for DGAs can be restricted to these.

The data entries for domains with second-level domains that were queried more than 50 times in total in the appropriate time slot were also deleted. These are unlikely to be generated by a DGA used for second-level domain fluxing, unless many machines in the network have been compromised by the same malware and are querying the same domain. 98% of all queries of valid domains in the data set are for domains with a second-level domain that was queried at least 50 times in the time slot containing the query, even though they account for less than 10% of the different second-level domains queried. This parameter could in fact have been set to a much lower value than 50, say 10, as the DGA domains found by the procedure were typically queried by only a few end-user machines (often, just one) and a few servers or aggregators.

Next, the candidates for the time slot were identified. These were the client IP addresses that during the time slot queried 50 or more of the remaining 2nd-level strings. An experiment on one day’s data reducing this parameter from 50 to 25 did not result in the detection of additional DGAs. In both cases, 7 DGA families were found.

We found it was useful to treat candidates that queried Chinese-language web sites separately, because the 2-lengths for Chinese-language web sites tend to be shorter than average, as a result of practices for representing Chinese-language domain names in Latin alphabet letters and numbers. The candidates were classified as Chinese-speakers or not using a day’s data, by determining whether what fraction of the domains with a two-letter top-level domain that a candidate queried had the top-level domain cn, tw, mo, sg or hk. If the fraction was over 50%, the candidate was classified as a Chinese-speaker. The average 2-length distributions of (non-deleted) domain names queried by candidates in the two classes were calculated, with each candidate in the class being weighted equally in the average.

For each candidate, the distance was calculated between the 2-length distribution during the time slot for domains queried by that candidate and the average distribution for either non-Chinese-speaking or Chinese-speaking candidates, candidates, depending on how the candidate was
The distance function used was the standard statistical distance function over a finite alphabet (1)

\[ D(X, Y) = \sqrt{2} \sum |X(i) - Y(i)| \quad (1) \]

where the sum is over all 2-length values \( i \), \( X(i) \) is the probability of \( i \) in distribution \( X \) and \( Y(i) \) the probability of \( i \) in distribution \( Y \).

The next step was to find candidates for which the distance from the average was anomalously large. Fig. 1 shows the distribution of the values for the distance from the average, measured to the nearest 0.1, for each day. For all five days the relatively flat part of the graph begins at a distance value of around 0.47, so values greater than 0.465 were treated as anomalous. (The blip at a distance value close to 1 turned out to be caused by candidates that queried Zbot domains.)

Next, for each of the candidates with anomalous 2-length distributions, DNS queries were made ten of the second-level domains that they queried in the time slot. If 6 or more of these queries resulted in the answer that the domain did not exist, the candidate was counted as an identified candidate. The reason for this check was that about 28% of the candidates with anomalous 2-length distributions did not appear upon manual inspection to be querying AGDs in the time slot identified. For these candidates, most of the domains they queried resolved to an IP address, whereas by design, most malware DGA domains have no DNS registration. This check will be discussed further in Section IV, which gives reasons why some candidates had anomalous length distributions without querying DGA domains.

For the reported candidates, further investigations were carried out to determine the properties of the underlying DGA, and to try to identify it. This was made easier by the fact that usually, the domains queried by the candidate during the time slot (and queried less than 50 times in total during that time slot) were almost all from the DGA. In some cases they were all from the DGA.

The procedure detected 19 different DGAs in the data set, with between 5 and 10 different DGAs detected in each day’s data. The numbers of candidates reported on the five days were 28, 34, 13, 10 and 6 respectively. (These numbers count a candidate only once if it was identified in multiple time slots). All of them were querying AGDs.

A. Identified malware DGAs

The procedure detected the DGAs used by Conficker.D, Dorkbot, Expiro, Pushdo (the variant using suffix kz [13]), Ramdo, and Zbot. For more information on these, see their entries in [14]. Note that the names refer to the DGA, and not necessarily the underlying malware. For example the Zbot domains might have been queried by GameoverZeuS or by Cryptolocker, as these use the same DGA. Dorkbot domains are hard-coded rather than being generated by infected machines, but they look as though they were originally machine-generated. As these DGAs are already known, we do not describe the domains that they generate.

B. Fixed-string DGAs

For two of the sets of AGDs, the domains are just a fixed string apart from a varying index number. These have domains of the formats firehunterN.com and testsupporturlN.com respectively, where for the firehunter domains the index \( N \) is a string of 12 decimal digits, and for the testsupporturl domains the index \( N \) is of the form

\[ hhhhhhh-hhhh-hhh-hhhhhhhhhhh \]

where each \( h \) a hex digit.. The firehunter domains appear not to be associated with malware, but rather with an Internet Service measurement tool by Agilent Technologies [14], queried by two clients. The testsupporturl domains appear to be produced by a test carried out by a single client.

C. Properties of numbered DGAs

Table I gives ten sample second-level domains from each of the eleven remaining DGAs. Larger samples are available, for research purposes, on request from the authors. Table II indicates the suffixes used by the DGAs, the minimum and maximum 2-lengths, and the set of characters used in the 2nd-level strings in the examples found in the data.

There were brief reports in 2013 of a DGA using 26 different suffixes and characters \( a-y \) in the 2nd-level string, without more details being given. DGA1 is probably that DGA, so it is not included in the total of nine unidentified DGAs, however a search for a reference was unsuccessful.

DGA6 is also not counted in the nine, because it may be the DGA reported in [9] as New-DGA-v1; however it is hard to be sure, as [9] gives no general information about New-DGA-v1, just ten sample domains. Batches of DGA6 domains often include the same 2nd-level string three times, once with each of the three suffixes.

The client that queried DGA8 domains also queried, in the same timeslot, three longer domains used in the past as command-and-control servers for the Nitol botnet [17].

---

Figure 1. Distribution of values of the distance from the average for candidates on each day.
### TABLE I. SAMPLES OF DGA DOMAINS

<table>
<thead>
<tr>
<th>DGA</th>
<th>Sample Domains</th>
</tr>
</thead>
<tbody>
<tr>
<td>DGA1</td>
<td>bvooroomedpolymray.so, eiqnrwvaqex.ms, vbmmxmhq8xpl.in, dtaacuxlucurolg.kvhj.eu, ltyhiifldackxh.cf, fxuaxaapncycyeb.eu, quucxbbsbqgj.jo, yuuumimrwdw.tj, chguuegrwqjshp.sc, hcdxaasxadwxh.bz</td>
</tr>
<tr>
<td>DGA2</td>
<td>1ppzfdsm126u4y38c.cn, u5x3ngqjikmwwmnb51.ru, kythnpfcmagqtjn2qzy, ilwqyr64szzywznmw7.net, 4y7wboxtxv7v5qyik.ru, wqddkcxevoxv471kz8s.cn, 5vdaka3xdk3wbtzg.hz, xli1uzkgz5pwsxld8.net, 16cylin3wafdp5xerus.cn, e2o7a7on7zgub7id7bf.com</td>
</tr>
<tr>
<td>DGA3</td>
<td>tetdx7tv.com, xfgusx.com, jyfjkhb.com, hsegsbt.com, iehwhgjki.com, ghtjwkgf.com, jhjdx9tk.com, sskitkxoss.com, bchdbhde.com, xhvdsvoj.com</td>
</tr>
<tr>
<td>DGA5</td>
<td>YurAlv.com, MYIECWP.com, CqCQYr.com, ORxaxgPQ.com, xnXwKxr.com, PQJAI7ZU.com, l7v9dGc.com, XIJdZlgj.com, edjibQ.com, HvDrXhY.com</td>
</tr>
<tr>
<td>DGA6</td>
<td>67add9c.net, 3142871z.com, 3142481d.net, 3142481d.net, 8293654d.net, 3293654d.com, 3293645d.com, e959e5b6.com, e959e5b6.com, bb2533l1.info</td>
</tr>
<tr>
<td>DGA7</td>
<td>Jgelv7g.net, 31lzdjpse.com, dtgjgg.com, kwwxskzb.eu, azjauu.in, etpsoprc.ru, zderyvwyw.in, lvwglvze.net, rkzlt.ru, ogypqdbk.ru</td>
</tr>
<tr>
<td>DGA8</td>
<td>uzeftk.com, dynwma.com, adgafp.com, vphfhf.com, opljeb.com, peycwgg.com, kghzaj.com, uotikso.com, aztfk.com, rkhkil.com</td>
</tr>
<tr>
<td>DGA9</td>
<td>ozlyyjehb.com, fxyonepqp.com, nmrqjgbovw.org, bovixcbgzkv.com, diylahjf.com, nqecpol6k.org, toxxs6fpyg.com, pchaledqxc.tv, jujurunxy.net, gtyzqinl.com</td>
</tr>
<tr>
<td>DGA10</td>
<td>kdrlowyj.com, dchhmkcb.com, iclclakaju.com, oysaqxcbi.com, fzaaxqadvru.com, knrj9i9i.com, heylimr0.ru, ucxqzpgav.com, nxoyntdzt.com, srxkwwkls.com</td>
</tr>
<tr>
<td>DGA11</td>
<td>marvoplqwvzxnpkfezv.net, eioyajajauqfpvdzdavv.net, odensdrfdjthiugthm.com, uaoovdstdlicuzzyrzgrshf.com, gtazlzszyfnnxsf7ftk.net, tktqtquntzqwzwwqmsg.9ov.com, hjp7zamgp7p7bpu7bqcl7p.net, nakumrwayuyewpov7dxq.net, ehfrwdborecmdbenvcz.m, vamxuxcupagehpdqgev.net</td>
</tr>
</tbody>
</table>

### TABLE II. SUFFIXES, LENGTHS AND CHARACTERS FOR DGA DOMAINS

<table>
<thead>
<tr>
<th>DGA</th>
<th>Suffixes</th>
<th>2-lengths</th>
<th>Characters in 2nd-level string</th>
</tr>
</thead>
<tbody>
<tr>
<td>DGA1</td>
<td>ac, bx, cc, cm, co, ex, de, eu, im, in, jk, ki, la, me, mn, ms, mu, nf, nu, ru, se, sh, so, sj, tv, tw</td>
<td>10-25</td>
<td>a-y</td>
</tr>
<tr>
<td>DGA2</td>
<td>biz, cn, com, net, ru</td>
<td>18</td>
<td>1-8, a-z</td>
</tr>
<tr>
<td>DGA3</td>
<td>com</td>
<td>8</td>
<td>b-k, s-z</td>
</tr>
<tr>
<td>DGA4</td>
<td>com</td>
<td>10</td>
<td>1-5, a-z</td>
</tr>
<tr>
<td>DGA5</td>
<td>com</td>
<td>7</td>
<td>A-Z, a-z</td>
</tr>
<tr>
<td>DGA6</td>
<td>com</td>
<td>8</td>
<td>0-9, a-f</td>
</tr>
<tr>
<td>DGA7</td>
<td>biz, com, eu, en, info, name, net, org, ru, se</td>
<td>6-11</td>
<td>a-z</td>
</tr>
<tr>
<td>DGA8</td>
<td>com</td>
<td>6</td>
<td>a-z</td>
</tr>
<tr>
<td>DGA9</td>
<td>com, net, org, ru, tv</td>
<td>10</td>
<td>a-z</td>
</tr>
<tr>
<td>DGA10</td>
<td>com, ru</td>
<td>9</td>
<td>a-z</td>
</tr>
<tr>
<td>DGA11</td>
<td>net</td>
<td>20</td>
<td>a-z</td>
</tr>
</tbody>
</table>

This is partly because the ordering by frequency reduces the statistical variation between different samples from the same underlying distributions. This may be the complete explanation in the case of DGA1. But in general, it is also because of the design of malware DGAs: if the underlying suffix distribution differs between batches, it seems that the DGA effectively uses the seed to determine an ordering of the suffixes, and chooses the suffix for a generated domain with a probability determined by this ordering.

Hence, it may be useful to report the suffix distributions for the domains found from the numbered DGAs, even though different batches may have different distributions. The distribution for DGA1 is shown in Figs 2 and 3. The suffix frequencies for the domains found are approximately 0.3 cn, 0.2 ru, 0.2 biz, 0.15 com and 0.15 net for DGA2, 0.5 info, 0.3 net, 0.2 com for DGA6, and 0.6 ru, 0.4 com for DGA10. For DGA7 and DGA9, the suffixes appear with approximately equal frequency.

![Figure 2. Suffix distributions of DGA1 domains queried by seven different client IP addresses.](image-url)
E. Character and Bigram Distributions

Two other features used to detect DGA domains using machine learning are character and bigram frequencies. Just as the suffix distributions may vary between different batches of the same DGA, so may the character and bigram distributions. For example, all the domains in the same Expiro batch begin with the same letter, so this letter has high frequency, but the letter can be different in different batches. However, as for the suffix distributions, the character and bigram distributions appear to be stable between batches when sorted by frequency. The batches of DGAs found were not large enough to perform n-gram analysis with confidence when n > 2.

Fig. 4 shows the character distributions for the 2nd-level strings in the DGA8 domains found. The six vowels (including y) are more frequent than the consonants, and z is the least frequent character.

Fig. 6 shows the bigram distributions, sorted in reverse order of frequency for the relevant DGA, for the 2nd-level strings in DGA3, DGA6, DGA9 and DGA10 batches. DGA9 uses only half the of the 676 possible bigrams of characters a-z in its 2nd-level strings, because two adjacent letters in a 2nd-level string for a DGA9 domain are always an odd distance away from each other in the English alphabet.

In 2nd-level strings of DGA1, DGA2, DGA4 and DGA11 domains found, all bigrams from the character set have approximately equal frequencies. As yet we have found insufficiently many sample domains from the three other numbered DGAs to report the bigram distributions of their 2nd-level strings with confidence.

II. LIMITATIONS AND FUTURE WORK

The procedure cannot detect DGAs whose distribution of 2nd-level string lengths is not sufficiently unusual. However, such DGAs may be detectable using other methods once the infection has spread to more machines. The procedure is only designed to detect second-level domain fluxing, but it might be possible to extend it to
detect fluxing at a higher level by considering looking for unusual distributions of the lengths of higher-level strings.

The candidates in the data set that had unusual 2-length distributions but were not querying DGA domains tended to have distances from the average 2-length distribution close to the cutoff point of 0.465, and typically had unusual length distributions for one of two reasons. Either they queried short Chinese-language domains, but were classified as not Chinese-speaking; or, they queried set of related long but not AGDs. For example, one candidate queried over 60 web sites belonging to a large online shop called CoolBlue [17] in the same time slot. Ten of the CoolBlue domains are below.

```
schuurmachineshop.nl, snijmachineshop.nl, broodbackmachinestore.nl, audorecordershop.nl, naaimachinestore.nl, inkcarticestore.nl, gourmetstelstore.nl, skibirlcenter.nl, videocamerashop.nl, mengpanellstore.nl
```

The check that 6 or more of the 10 sample domains were reported as non-existent prevented all these candidates from being reported. In addition, the identification of Chinese-speaking candidates could be made more accurate by considering only country-code top-level domains rather than all 2-letter top-level domains - some misclassified candidates queried many domains with top-level domain cc or vn.

The check may eliminate some candidates that are in fact querying DGA domains. In particular, it will do so if most of these domains have been sinkholed, that is, registered by others to prevent their use by the malware owner. This was the case for a candidate with an anomalous 2-length distribution in the May 12 data that was querying SillyFDC domains (see the entry in [13] for information on SillyFDC), and for several candidates querying Zbot domains after a large number of these were sinkholed by international law enforcement [18]. It is sensible for the check to count domains that resolve to a known sinkhole IP address as though they had no DNS registration. Our team is investigating how best to identify sinkhole IP addresses from DNS data. Our current method (details of which are out of scope of this paper) identified the sinkhole IP addresses involved in both of these cases. A non-automated alternative to the check is to examine the 10 sample domains by eye, and decide whether these appear to be AGDs. This typically takes only a few seconds per candidate with anomalous 2-length distribution.

It would be useful to automatically compare the features of domains queried by a reported candidate to known DGAs, to see whether the DGA is already known. Finally, this paper addresses only a small part of a large problem. In order to be used in practice, the procedure needs to be integrated with network security tools and systems. Once DGAs have been identified, there also need to be processes to develop recognizers for domains produced by the DGAs, to identify infected clients in the network, and to remediate the associated problems. As a first step toward this integration, we aim to use domains found from the newly-discovered malware DGAs to retrain our classifier, so that domains from these DGAs can be recognized quickly.

ACKNOWLEDGMENT
Thanks to all the Big Data for Security team at HP Labs.

REFERENCES


Thanks to all the Big Data for Security team at HP Labs.