MDNDS

MALICIOUS DOMAIN NAME DETECTION SYSTEM

Detecting malicious domain names using spatial co-occurrence in DNS traffic

February 7, 2016

Supervisor: Maarten Wullink (SIDN)

Auke Zwaan
University of Amsterdam
System and Network Engineering (OS3)
auke.zwaan@os3.nl

Abstract

In this research, malicious domain names in the *.nl* ccTLD are searched for by analyzing DNS traffic. Using a large dataset of DNS queries and a blacklist, spatial co-occurrence is looked for to predict which domain names are potentially malicious. It turns out the model proposed in this paper is able to find malicious domain names that were not in the initial blacklist, of which 28 percent had not been found by any online scanners before. Eventually, the results of the model can be used for further analysis and can be included as part of a broader detection system.

1 Introduction

The Domain Name System (DNS) - as defined in RFC 1034 [8] and 1035 [9] - is the standard for translating domain names into their corresponding numerical (IP) addresses. Due to the fact that DNS is so highly embedded into the workings of the Internet, cybercriminals must also make use of it to reach their malicious domains. These domains can, for instance, be used for sending spam or carry out any function as bots within a larger botnet. This is a severe threat, as botnets containing over a million nodes have been reported and efficiently fighting them is still considered difficult [12] [2]. One source even goes as far as calling it the 'largest security threat' [7].

A dataset provided by the Stichting Internet Domeinregistratie Nederland (SIDN), the highest authority for the Dutch *.nl* country code top-level domain (ccTLD), will be used to research whether it is possible to detect malicious domains by looking at spatial characteristics in DNS traffic. By combining existing data with data from common Domain Name System Blacklists (DNSBLs), a model that tries to classify a domain on maliciousness is made. Eventually, the main goal is to research whether it is possible to use interrelations between DNS resolvers and malicious domains to detect more potentially malicious domains.

2 Research Questions

For this research, one main research question is to be answered:

- Is it possible to detect malicious domains by analyzing interrelations between DNS resolvers and blacklisted domains?

Accordingly, four subquestions will be answered:

1. How can the dataset be visualized and analyzed?
2. What is a malicious domain and which sources publicly list them?
3. What interrelations between DNS resolvers and malicious domains exist and what other domains are spatially close within the network?
4. How effective are these interrelations to predict the maliciousness of other domains?

3 Related work

Previous research in this field broadly looked into the effectiveness of the use of existing DNSBLs and analysis of DNS traffic itself. On the side of the DNSBLs, research was done to study whether they could be used to identify spam traffic[6]. Eventually, it turned out almost 80 percent of the hosts caught by the spam traps they used in the research were listed in DNSBLs.

Other researchers looked at botnets sending spam more specifically [11]. They argue that most spam is sent from botnets but found out that it is hard to effectively blacklist bots belonging to a botnet in real-time. Taking this into account, it seems relevant to find out more about these networks and the communication and interrelations of the domains involved. Interestingly, one research pointed out that malicious domains that were automatically generated did not use a wide range of IP addresses to communicate [1]. This might mean the range of DNS resolvers they communicate with is not wide either, making it possible to use these resolvers to detect malicious 'networks' from a higher level.

In 2010, researchers showed a new technique in which DNS lookup failures were visualized as a graph and used to identify malicious hosts [5]. However, the dataset from the SIDN that is available for this research does not contain any information about the successes or failures of any of the queries.

Another research looked into co-occurrence between DNS queries in 2012 [4]. A process quite similar to the one in this paper was used to find out about potentially malicious hosts. What the researchers did not show, however, is what exact dataset was used (i.e. the definition of 'DNS traffic') and how that was representative for more general DNS traffic. The term *DNS Cache Server* used for indicating the traffic source is ambiguous since no context was given about the size of the dataset and the network, and might imply the research lacks a broader overview. Furthermore, they looked at *temporal* characteristics to identify co-relations between *infected hosts*, while this research focuses on identifying co-relations between *DNS resolvers* and *domain names* by looking at *spatial* charcateristics.

4 Approach

4.1 Data gathering

The two most important types of data required for this research were DNS data and blacklists. Further on in the experiments section, these two types of data will be combined and analyzed.

4.1.1 DNS Data

A dataset containing 172,671,762 DNS queries from one of the authorative DNS servers for the *.nl* ccTLD, `ns1.dns.nl`, was shared by the SIDN. This accounted for exactly one day (24 hours) of real-life traffic from January 6, 2016. Each row in the dataset consisted of three fields: *source*, *target* and *timestamp*. For this research the (Unix) timestamp was left out, as only spatial (not temporal) network characteristics were looked at.

It must be noted that this dataset includes *all* DNS requests sent to the server, which means that the validity of the requests is not taken into account. For instance, requests containing invalid characters or syntactical errors appear in the dataset, without any further information about the success or failure of the query. The assumption was made that invalid requests would eventually be filtered out in the analysis automatically, as the likelihood of an invalid request appearing significantly often or matching a malicious domain from any of the blacklists would presumably be quite low.

4.1.2 DNSBL

As the dataset only contained information about the *.nl* ccTLD, the initial blacklists that were to be composed could only contain domain names that shared this feature. Similarly, blacklists containing only IP addresses were of no use. This limited the number of useful blacklisted domain names that were publicly available to a big extent. In this paper, the term *domain (name)* will be used for domains from the *.nl* ccTLD.

Firstly, the SIDN shared a list of domain names that were part of their so-called *sink-hole*. This sink-hole contains 15 domain names that were known to perform malicious activities in the past. By registering these domains themselves, they were taken out of circulation to prevent any further abuse. The reason why these domains were still relevant in this research is that infected machines will still try connect to these domains. A quick check showed that queries to these domains appeared in the dataset relatively often, too.

As research pointed out that most spam was sent from botnets [11], the assumption was made that using known spam domains would be a good way to get into the network of botnets. It turned out that these domains were, indeed, easiest to find. This resulted in a list of 424 malicious domains that were known for spam activity, extracted from the website of anti-spam software development company *joewein.de LLC*[1] .

Then, 6 malicious domains listed on *MalwareDomainList.com*[2] were added to the list, followed by 14 domains from the *Internet Storm Center* from the Sans Technology Institute[3]. For the latter, the 'high' level filtered list was chosen to prevent false positives as much as possible.

Combining these lists resulted in an initial blacklist of 459 domains that were from then on assumed to be true positives.

4.2 Experiments

4.2.1 Initial setup

One way of analyzing a network or dataset is by using any of the Python modules *NetworkX*[4] or *graph-tool*[5]. The problem with these tools, however, is that all graphs are being stored in-memory. Flagging and counting malicious domain names in the graph (i.e. storing attributes along with the nodes and edges) would require far too much memory for a standalone machine with these

[1] `http://www.joewein.net/dl/bl/dom-bl-base.txt`
[2] `http://www.malwaredomainlist.com`
[3] `https://isc.sans.edu/feeds/suspiciousdomains_High.txt`
[4] `https://networkx.github.io/`
[5] `https://graph-tool.skewed.de/`

datasets. *Apache Spark*[6] could offer a solution for these memory problems, but due to the absence of a *Spark cluster* during this research, this was not done.

This, in combination with the fact that no advanced graph theory (such as computationally heavy centrality measures) was required, resulted in the choice to not use dedicated graph visualization tools. Instead, only source-target pairs were stored, counted and analyzed using the Python module *pandas*[7], in which relatively large datasets (such as the ±7.5 Gigabyte CSV file of DNS queries) can be efficiently stored as *dataframes* and worked with in-memory. This is virtually the same as storing edges in a database and offered everything required for the calculations in this research.

4.2.2 Data processing

Malicious domains and suspicious resolvers
First, the dataset was filtered to get only the queries to domains in the initial blacklist. This subset was used to extract all DNS resolvers that did at least one request to a malicious domain. From now on, these resolvers will be called 'suspicious resolvers' as these resolvers have at least some connection to malicious domains.

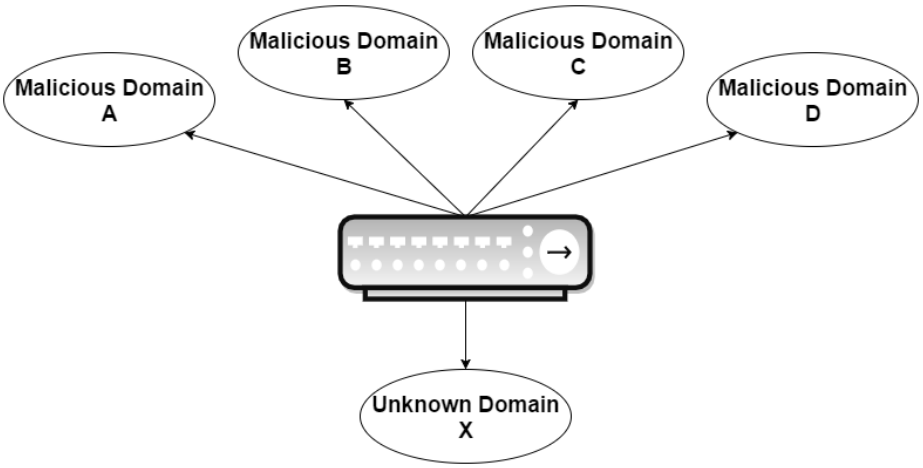

Figure 1: *The concept of suspicious resolvers: If a DNS resolver resolves to four domains (A, B, C and D) that are provably malicious, the fifth domain (X) is considered malicious, too. In this model, the threshold is set at 20 percent (i.e. if one out of five queries by a resolver is provably malicious, the rest is considered malicious, too). In this example, the DNS resolver has a maliciousness ratio (1) of* $\frac{4}{1} = 4.0$.

Then, all other domains these suspicious resolvers tried to resolve were looked up to get the full network of potentially malicious queries (as shown in Figure 1). In short, these are all queries by suspicious resolvers.

Flagging queries and maliciousness ratio
From this new dataset, all source-target pairs were extracted, grouped and

[6] `http://spark.apache.org/`
[7] `http://pandas.pydata.org/`

Table 1: Flagging malicious queries

Resolver	Domain	Classification
Resolver A	Domain A	Malicious
Resolver A	Domain B	Unknown
...	...	...
Resolver Y	Domain B	Unknown
Resolver Z	Domain E	Malicious

A query is considered malicious if the target is in the initial blacklist. Using this definition, each query is flagged as either 'Malicious' or 'Unknown'.

Table 2: Flagging malicious queries

Resolver	Malicious	Unknown	Ratio
Resolver A	2	2	1.0
Resolver B	1	2	0.5
...	...	...	...
Resolver Y	1	4	0.25
Resolver Z	4	300	0.013

For each resolver, the number of 'malicious' and 'unknown' queries is counted. Then, the 'maliciousness ratio' is calculated for each of them. Note: the numbers in this table are an example and do not represent real data.

counted (a resolver can resolve to a domain more than one time). Each source-target pair was then flagged as either *malicious* or *unknown* (see Table 1). This provided the required data for calculating an extra *maliciousness ratio* (1), indicating what the ratio between *malicious* and *unknown* queries is for each resolver (see Table 2).

$$\text{maliciousness ratio} = \frac{\text{number of malicious queries}}{\text{number of unknown queries}} \tag{1}$$

Malicious resolvers and potentially malicious domains

According to Google [3], the Google DNS handles over 400 billion DNS queries per day. Although this number includes queries to all top-level domains (TLDs), it is clear that such a big resolver would presumably at least have some 'malicious queries' in this definition. For this reason, all resolvers with a *maliciousness ratio* of less than 0.25 were stripped out (2). This means that only resolvers for which, for each four 'unknown' queries, there was also 1 provably malicious query in the dataset, were further looked at. The number itself stems from the assumption that a resolver from which 20 percent of the total traffic is provably malicious is significantly active in malicious networks.

After filtering the list of suspicious resolvers based on their *maliciousness ratio*, only the 'malicious resolvers' were left. By looking at all domains they resolved to (excluding the domains from the initial blacklist), a list of 'potentially malicious domains' was constructed.

"A malicious resolver is a resolver for which the maliciousness ratio is greater than or equal to 0.25." (2)

Popular domains
In previous research on co-occurrence of DNS queries [4], researchers mentioned the issue of popular domains. The idea is that popular domain names are likely to co-occur with any other domain name in DNS traffic. This means that websites like `google.nl` and `nu.nl` (a large Dutch news website) are likely to end up in the list of malicious domains when just looking at co-occurrence. To solve this problem, the 100 most popular *.nl* domains (extracted from Alexa[8] indirectly [10]) were assumed to be *not* malicious (3). They were therefore stripped out of the list of potentially malicious domains constructed in the previous step.

"The 100 most popular .nl domain names are not malicious." (3)

Validation
Eventually, an extra check was done to validate the results and identify the number of true positives. This involved querying the VirusTotal API[9]. The assumption made in (4) is important in this verification process. After the VirusTotal scan, the rest of the domain names was classified manually by performing Google searches. In these searches, any references to malicious content (i.e. malware, presence in a blacklist, automatic redirections to malicious domains, et cetera) were looked for. These two steps of validation, together, resulted in a final number of true positives. The rest was classified as either 'non-malicious', 'possibly malicious' or 'unknown'.

"A domain name classified malicious by the algorithm is verified as being a true positive if there is - at least - one scan in the VirusTotal database in which at least one scanner classified the domain as malicious, too." (4)

In the test setup, the classification algorithm was run on a Ubuntu machine, equipped with a Quad-Core Intel i5 processor running at 3.10 GHz with 12 Gigabytes of RAM. The full source code of the algorithm can be found at [10].

5 Results

5.1 Analysis

The algorithm took 13 minutes and 43 seconds to run (see Table 3 for an overview of the produced data files). The initial dataset containing around

[8] `http://www.alexa.com/`
[9] `https://www.virustotal.com/en/documentation/public-api/`
[10] `https://github.com/AukeZwaan/MDNDS`

170 million DNS requests was reduced by roughly 50 percent by only looking at queries done by suspicious resolvers. Doing another round of filtering based these resolvers' maliciousness ratios left only 673 resolvers. These malicious resolvers together resolved to 413 unique domains. Filtering out the top 100 most popular *.nl* domains [10] resulted in the final list of 392 domains that were thus considered potentially malicious.

Table 3: The size of the data files after running the algorithm

Data file	Number of unique rows
Queries to malicious domains	40,649
Suspicious resolvers	8,132
All queries by suspicious resolvers	85,169,973
Malicious resolvers	673
Potentially malicious domains	413
Potentially malicious domains excluding top 100 domains	392

The size of each data file created by the algorithm in the order they were created. Malicious resolvers were DNS resolvers with a *maliciousness ratio* ≥ 0.25. It is important to note that potentially malicious domains did not include domains from the initial blacklist.

5.1.1 Classification of potentially malicious domains

Using the eventual list of potentially malicious domains (see table 3), the VirusTotal API was queried. This resulted in 90 domains being automatically classified as malicious. After the second, manual check, 35 of these unclassified domains could be classified as malicious, too. As it turned out, domains from this last group often contained malicious code on public pages, which were then linked to from Russian and Chinese websites. Some websites immediately redirected to pages containing only advertisements and others were hijacked completely. Combining the two groups, 125 provably malicious domains were found.

The domains that could not be identified as malicious could be roughly divided into two categories: *non-malicious domains* and *possibly malicious domains*. For the *non-malicious domains*, an effective Google search was possible (i.e. clear results, directly referring to the domain were found), but no traces of any malicious behavior were found. Still, domain names classified as non-malicious were sometimes full of advertisements or looked 'shady'. It is not clear whether these domains are administered by people or systems that also administer malicious domains, or are just popular and therefore likely to co-occur with the others.

The other group, the *possibly malicious domains*, consisted solely of domains belonging to hosting providers. 111 of the domains found in the list of potentially malicious domains belonged directly to a hosting provider. For these domains, it was practically impossible to give any classification, as any subdomain of any customer (including any query to one of their DNS servers) could have caused the domain to appear in the list.

As the number of domain names directly belonging to hosting providers

was relatively high (28.32 percent), one possible enhancement would be to add them to a list of trusted domain names. After filtering out these domains, the percentage of true positives goes up from 31.89 percent to 44.48 percent.

For the rest of the domains, it was not feasible to classify them due to external reasons. For instance, it is practically impossible to effectively search for a domain that is only five characters long and contains the word 'I' as Google automatically strips common words from search queries.

Considering the fact that any false positives were identified as such by the absence of clear traces of 'maliciousness' during a Google search, they might still be worth a second look. If they would, for instance, be used for malicious activities behind the scenes and have no public activities yet, they will never end up in the list of true positives in this research. This is something to consider when interpreting the results of the algorithm.

Table 4: Classification of potentially malicious domains

Malicious	Number of domains
Yes	125
No	153
Possibly	111
Unknown	3
Total	392

90 of the 125 domain names classified as "Yes" (malicious) were classified automatically using the VirusTotal API. The rest (35) was classified by hand. The classification "Possibly" was given to domains belonging directly to a hosting provider (such as `argewebhosting.nl`), and "Unknown" meant a manual check was practically infeasible.

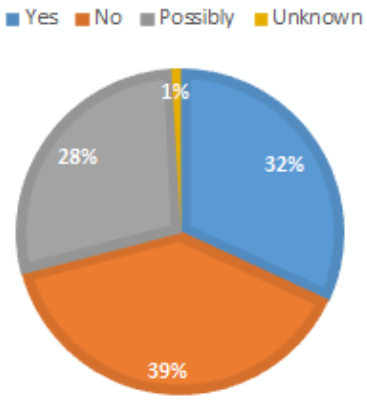

Figure 2: *The classification of domain names on their maliciousness, based on the data from Table 4. Eventually, domains with classification 'Possibly' could be stripped out since they all directly belonged to hosting providers.*

5.2 Evaluation

To test the quality of the algorithm, it was run every 15 minutes for 8 hours, thereby varying the initial blacklist by randomly splitting it up into a training and test set of respectively 90 and 10 percent. It was examined whether the algorithm was able to find any of the domains from the test set, which contained 45 domains that were, just like the training set, provably malicious.

In Table 5, the results of these test rounds are summarized. As it turned out, the model was able to find about 2.6 of the domains of the test set (5.8%) on average. This number shows the algorithm is unable to efficiently find a given set of domain names. With a test set that remains a fixed size over all the tests, a standard deviation of 68.295 is found over the number of potentially malicious domains. This number, combined with an average number of 349.875 potentially malicious domains found in each round underline the algorithm's dependency on the initial blacklist. This is further illustrated in Figure 3.

These results might have been caused by the spatial distribution of the domains from the dataset and the blacklist. If a domain from the test set is in none of the networks looked at, it is by definition impossible to find it using the proposed method. It is thus not entirely clear whether the insignificance of the results of the test rounds say anything about the algorithm or the blacklist.

Table 5: Results of 32 test rounds

	Min	Max	Mean	Std
# Potentially malicious domains	114	400	349.875	68.295
# from test set found	0	5	2.594	1.316

The results of 32 tests run with a random training and test set of respectively 90 and 10 percent of the initial blacklist. On average, 2.6 domains from the test set were correctly identified as potentially malicious domains. The total test set contained 45 domains each round.

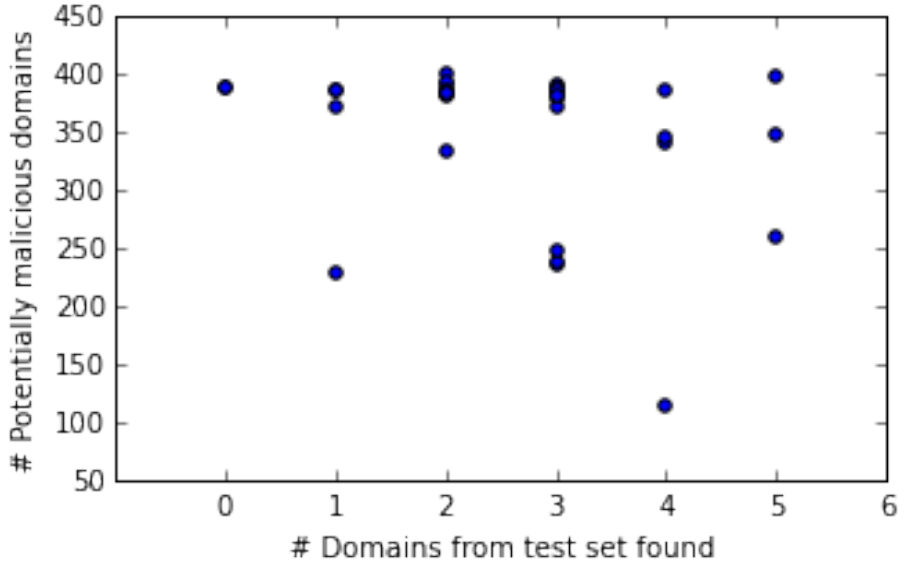

Figure 3: *The results of 32 tests run with a random training and test set of respectively 90 and 10 percent of the initial blacklist. On the X-axis, the number of domains from the test set that appeared in the final list of potentially malicious are plotted. On the Y-axis, the according total number of potentially malicious domains found in that round are shown.*

6 Discussion and conclusion

In this research, a model was proposed to identify malicious domains in a large dataset of DNS traffic. By combining these data with a blacklist of domain names and looking at spatial co-occurrence of DNS queries, a list of 'potentially malicious domains' was constructed. Starting with a dataset containing more that 170 million DNS queries and a blacklist containing 459 .nl domain names, the model was able to filter out 392 domains that it classified as 'potentially malicious'.

The percentage of true positives turned out to be around 32 (or 45 after stripping out domains directly belonging to known hosting providers), which is not high enough to efficiently blacklist malicious domains without negatively affecting legitimate domains. Still, 28 percent of the true positives had not been classified as malicious by any online scanner yet. This shows that the model is able to come up with new valuable information about domain names that was not known before.

Eventually, the results from each DNS traffic analysis done by the model (i.e. the list of potentially malicious domains) can be used for further analysis and can be included as part of a broader detection system.

7 Future work

Enhancing both the trusted domain list (consisting of only the 100 most popular .nl domains in this research) and the initial blacklist is vital to the performance of the model. One way to do this is by running the algorithm recursively. This means that, after one run, the list of true positives would be added to the initial blacklist, after which the algorithm can be run again for a theoretically infinite number of times. Further research could examine what the effect of running the algorithm multiple times is on the eventual ratio between true and false positives.

By running the algorithm on multiple datasets (e.g. each day), commonly found domains can be analyzed. Another next step could be to do a further content-based search for each of the potentially malicious domains. By scanning the contents of each of these domains, malicious content can be looked for. This would enhance the validity of the designation and classification of true positives after each run of the algorithm.

Instead of the free, public VirusTotal API that was used for the classification of true positives, the paid, private VirusTotal API[11] that does not have a rate limit could be used. Alternatively, any offline dataset with similar data would speed up the process.

Another factor that can be included in the algorithm is *whois* data. As the SIDN has access to the ownership information the for each .nl domain name, potentially malicious registrants could be looked for. Using this information and combining it with the information gathered through the model in this research, malicious domains can potentially be spotted when they are just registered. More advanced graph theory would be necessary for this, though.

A last enhancement would be to investigate what the 'most malicious' resolvers are (based on their *maliciouss ratios*), and to use that information to do a reverse analysis (i.e. *"How many times is a domain requested by a malicious resolver, and how does that compare to the rest of the domains?"*). In this setup, the initial blacklist would consist of blacklisted resolvers instead of blacklisted domains.

[11] `https://www.virustotal.com/en/documentation/private-api/`

References

[1] Leyla Bilge et al. "EXPOSURE: Finding Malicious Domains Using Passive DNS Analysis." In: *NDSS*. 2011.

[2] Evan Cooke, Farnam Jahanian, and Danny McPherson. "The Zombie Roundup: Understanding, Detecting, and Disrupting Botnets." In: *SRUTI* 5 (2005), pp. 6–6.

[3] Yunhong Gu. *Google Public DNS and Location-Sensitive DNS Responses*. Dec. 2014. URL: `https://googlewebmastercentral.blogspot.nl/2014_12_01_archive.html`.

[4] Keisuke ISHIBASHI et al. "Extending black domain name list by using co-occurrence relation between dns queries". In: *IEICE transactions on communications* 95.3 (2012), pp. 794–802.

[5] Nan Jiang et al. "Identifying suspicious activities through dns failure graph analysis". In: *Network Protocols (ICNP), 2010 18th IEEE International Conference on*. IEEE. 2010, pp. 144–153.

[6] Jaeyeon Jung and Emil Sit. "An empirical study of spam traffic and the use of DNS black lists". In: *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*. ACM. 2004, pp. 370–375.

[7] Wenke Lee, Cliff Wang, and David Dagon. *Botnet detection: countering the largest security threat*. Springer Science & Business Media, 2007.

[8] P. Mockapetris. *DOMAIN NAMES - CONCEPTS AND FACILITIES*. RFC 1034. Nov. 1987. URL: `https://tools.ietf.org/html/rfc1034`.

[9] P. Mockapetris. *DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION*. RFC 1035. Nov. 1987. URL: `https://tools.ietf.org/html/rfc1035`.

[10] Domain Punch. *Top 1 Million Websites and the Top Level Domains Used to Build Them*. Jan. 2016. URL: `https://domainpunch.com/tlds/topm.php`.

[11] Anirudh Ramachandran, David Dagon, and Nick Feamster. "Can DNS-based blacklists keep up with bots?" In: *CEAS*. Citeseer. 2006.

[12] E Skoudis. *Big honkin'botnet-1.5 million*.