Received: 3 April 2017 | Revised: 31 July 2017 | Accepted: 13 August 2017

DOI: 10.1002/nem.2008

SPECIAL ISSUE PAPER

WILEY

Booter list generation: The basis for investigating DDoS-for-hire websites

José Jair Santanna[1] | Joey de Vries[1] | Ricardo de O. Schmidt[1,2] | Daphne Tuncer[3] | Lisandro Z. Granville[4] | Aiko Pras [1]

[1]University of Twente, Enschede, The Netherlands

[2]SIDN Labs, Arnhem, The Netherlands

[3]University College London, London, UK

[4]Federal University of Rio Grande do Sul, Porto Alegre, Brazil

Correspondence
José Jair Santanna, Design and Analysis of Communication Systems, Faculty of EEMCS, University of Twente, PO Box 217, 7500 AE, Enschede, The Netherlands.
Email: j.j.santanna@utwente.nl

Summary

The expansion of Distributed Denial of Service (DDoS)–for-hire websites, known as *Booters*, has radically modified both the scope and stakes of DDoS attacks. Until recently, however, Booters have only received little attention from the research community. Given their impact, addressing the challenges associated with this phenomenon is crucial. In this paper, we present a rigorous methodology to identify a comprehensive set of existing Booters in the Internet. Before presenting our methodology, we illustrate the benefits of a set of booters on monitoring users from the Dutch NREN, SURFNet, from 2015 to 2017. Our methodology relies on well-defined mechanisms to generate a Booter list, from crawling suspect URLs to characterizing and classifying the collected URLs. The list obtained using the methodology presented in this paper has a classification accuracy of 95.5%, which is 10.5% better compared to previous work.

1 | INTRODUCTION

Distributed Denial of Service (DDoS) attacks have become a daily concern for any service operating in today's Internet. These attacks aim at overloading services and network infrastructures causing temporary degradation or even service unavailability. As a consequence, targets of DDoS face millions of dollars in financial losses, reputation damage, and legal actions.[1] In the past, DDoS was typically performed by technically skilled people, but nowadays, anyone can *hire* DDoS-as-a-service in the Internet through websites known as *Booter* or *Stresser*. Anyone without advanced technical skills can perform sequential attacks towards any target in the Internet for as little as 5 USD.[2] The convenience of DDoS-for-hire helps to boost DDoS popularity as reflected by the constant growth of the phenomenon since 2011.[3]

The majority of Booter customers are teenagers that attack one another's residential connection to gain advantage in online gaming[4]; and most attacks from Booters typically last up to 5 minutes and send traffic at rates up to 10 Gbps.[5] Some Booter outliers, however, are able to deliver significantly more powerful attacks, eg, 100 Gbps,[6] which can cause serious problems on targets that lack proper defense. It is known that a large number of attacks are launched every day from Booters, targeting essential services available online. Example of these events are 3 noticeable series of attacks that occurred in recent years: (1) the multiple attacks preventing tens of millions of customers from connecting to Microsoft Xbox Network, Sony PSN, Instagram, and Tinder for several hours[7]; (2) the attacks against a website of an American police department, preventing citizens from registering crimes[2]; (3) the attack against the Dutch online service (DigID) that stores sensitive information of more than 10 million citizens.[8]

While the Booter phenomenon has been extensively reported by the media and security specialists in blog posts, it has until now been only marginally addressed by the research community. For example, when asked for *booter*, *stresser*, and *DDoS-for-hire*, Google Scholar lists works from 3 research groups only, detailed in Section 7; and most of the available

literature is limited to the investigation of a handful of well-known Booters, ignoring the other hundreds of Booters that exist. We had the hypothesis in Pras et al[9] that since Booters use very similar techniques to perform attacks,[5] in theory, they all have the potential to become a big threat for the Internet. One important evidence to prove this hypothesis was reported by Akamai[10] that affirmed that Booters are responsible for the majority of mega attacks (ie, attacks that exceeded 100 Gbps) against their clients.

The mitigation of the Booters phenomenon is still a big open challenge, and blacklist is a promising approach to address this problem. Previous work[11] has shown that blacklist is an effective solution to mitigate spam-related problems by classifying spam services, using a set of specific characteristics of websites that offer spam as a service. We believe that such approach is also relevant to the Booters context: Booters websites share common characteristics that can be used towards their classification and further generation of Booter blacklists. These blacklists can be used in mitigation strategies, such as for the identification of accesses to Booters from within a network and forecast of potential upcoming attacks. In Section 2, we demonstrate the benefits of a Booter blacklist on detecting users that access booters. In this section, we present the observations of SURFnet (the Dutch NREN) while monitoring users accessing Booters from their network, since 2015.

After demonstrating the value of a Booter blacklist, we introduce our methodology to generate such list. Our methodology consists of 3 steps. We first collect an extensive list of suspect websites in the Internet using a crawler that we implemented (Section 4). Then we scrape and analyze these suspect websites based on 15 characteristics (Section 5). The results of this analysis are finally used to classify whether the suspect website is an actual Booter (Section 6). The blacklist resulted from our methodology contains 435 Booters, and it is publicly available at *http://booterblacklist.com/*. To the best of our knowledge, this is the most comprehensive list of Booters publicly available. Although we update this list on a monthly basis, anyone can create a list of Booters by simply following our methodology (available at *http://github.com/jjsantanna/Booter-black-List*).

Our scientific contribution is the investigation of 8 well-established classification methods to design a methodology for Booter blacklist generation. We strongly believe that before proposing something completely new, it is important to look at other domain areas to find solutions for open problems; and in this case, this approach worked very well. We also propose a new machine-learning algorithm to improve the result of the other well-established methods. Note that in this paper, we extend our previous work in Chromik et al,[12] in which we introduced a straightforward classification heuristic based on 9 characteristics to generate Booter blacklists.

In addition to the direct contributions described above, we aim with this work to raise awareness in the research community and among network operators about the challenges and open issues that still have to be addressed for the mitigation of Booters and their operations.

2 | BOOTER LIST USAGE

In this section, we present a use case to motivate the generation of a comprehensive booter list. Using the methodology described in the remaining of this paper, we periodically create updated lists of Booters openly shared at the Booter Blacklist initiative: http://booterblacklist.com/. We would like to highlight that, currently, the list contains both online and offline Booter websites. While the online Booters can be automatically used for monitoring and blacklisting purpose, the offline Booters can be used for historical analysis of their Market, for example. All the URLs on this list were manually tested to guarantee 100% true positive. When a URL pointed to something different from a Booter Website, we used the Way Back Machine initiative (at http://www.waybackmachine.org) to validate it veracity. This initiative stores screenshots of the majority of websites in the Internet, since 2001.

In this section, we analyze data related to (attempt of) accesses to Booters. The dataset provided by the Dutch NREN (SURFnet) consists of DNS requests originated from within the networks they manage, and DNS responses related to domain names listed in the Booter Blacklist. Our analysis focus on the overall behavior of SURFnet clients that access Booters and, therefore, we only look at DNS requests. The data have been sent to us weekly since June 2015, and SURFnet anonymizes IP addresses of their clients with a SHA-256 encryption algorithm. We analyze a total of 646 days worth of monitoring data (from June 19, 2015, to March 27, 2017), containing 132 335 records and 605 distinct users (ie, IP addresses). The data provided to us by SURFnet cannot be made public; however, our source codes for data analysis are openly available at https://github.com/jjsantanna/booterblacklist_use_cases.

The top graph in Figure 1 shows the number of requests to Booter domain names from SURFnet clients. The lowest amount of queries has been seen during the third quarter both in 2015 and 2016, which was likely caused by the vacations period (SURFnet is a NREN). Another observation is that the overall number of queries has decreased from Q3/2015 to

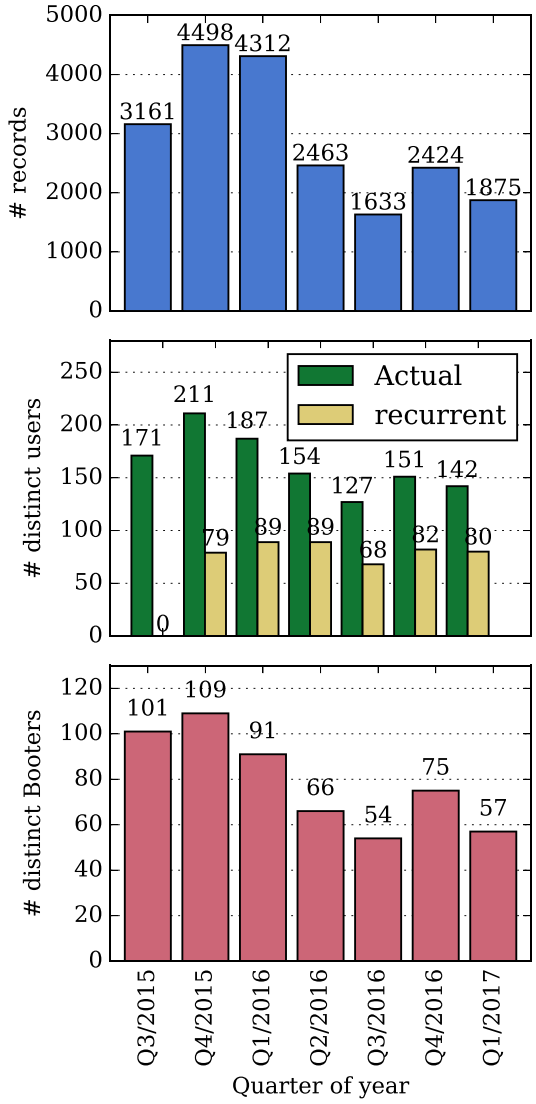


FIGURE 1 Analysis per quarter of year

Q1/2017. A possible explanation is the reduction on the number of "new users" for Booters, ie, IP addresses accessing Booters for the first time. The center graph in Figure 1 shows that while the number of recurrent users (yellow columns) remains roughly the same—with small fluctuations between 68 and 89 users across quarters—the total number of users (green columns) has decreased over time, indicating a reduction on the number of IP addresses accessing Booters for the first time.

Another fact that might have contributed to the reduced number of accesses to Booters is the reduced number of available Booters, as the bottom graph of Figure 1 shows. There is a clear relation between the numbers in the top and bottom graphs of Figure 1. The Booter Blacklist used by SURFnet lists 435 Booters (from which 115 are currently online). Based on the numbers of distinct accessed Booters in the bottom graph, we can conclude that users only access a fraction of the available Booters.

Figure 2 shows for each quarter the cumulative distribution of the number of queries that Booters received. We can clearly see that the "long tail" of the distribution decreases over time, being the Q1/2016 the longest one. Although the tail is reduced, the Booters at the tail are mostly the same, namely, booter.xyz and mostwantedhf.info. The reduced number of accesses to Booters is also visible in Figure 2: While 80% of Booters received 37 queries or less in Q3/2015, in Q1/2016, 80% of Booters accounted for 62 queries or less each and, in Q1/2017, for only 27 queries or less each.

Figure 3 shows the top 10 most accessed Booters for each quarter. We see that many Booters are always present in the top 10: booter.xyz, mostwantedhf.info, quezstresser.com, ipstresser.com, and vbooter.org. Some other Booters have appeared and remained among the top 10 more recently (eg, ragebooter.net and networstresser.com), suggesting that these have

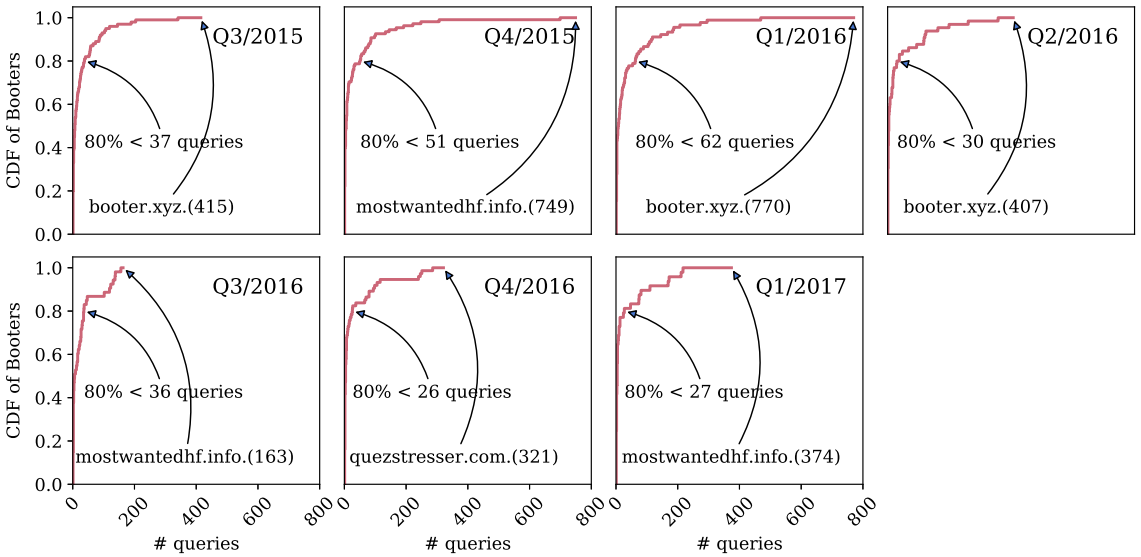


FIGURE 2 CDF of queries to distinct Booters per quarter of year (using the same scale)

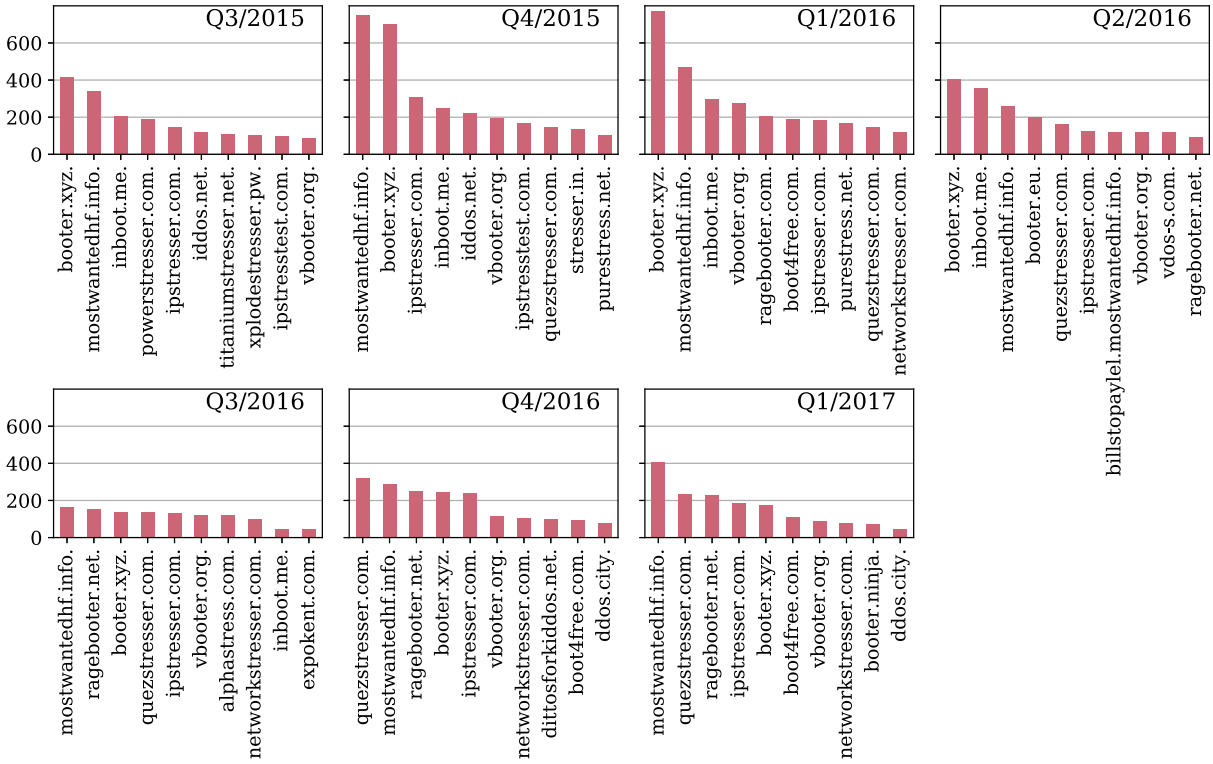


FIGURE 3 Top 10 most accessed Booters per quarter of year (using same scale)

apparently become a preference among users. Note that from all these Booters listed among the top 10 over time, only inboot.me is not available anymore—it disappeared from our logs after Q3/2016; also, to the best of our knowledge, none of these Booters have undergone mitigation actions.

Figure 4 shows the cumulative distribution of requests to Booters by distinct SURFnet users (IP addresses). The distribution "long tail" observed in all quarters, except Q3/2016, shows that some few users perform many more requests to Booter domain names than others. The median of number of requests is quite stable across quarters, with half of users generating up to 5 requests each. We believe that those users accessing Booters only few times are simply curious. Those at the tail of the distribution, however, are likely to be regular clients of "services" provided by Booters.

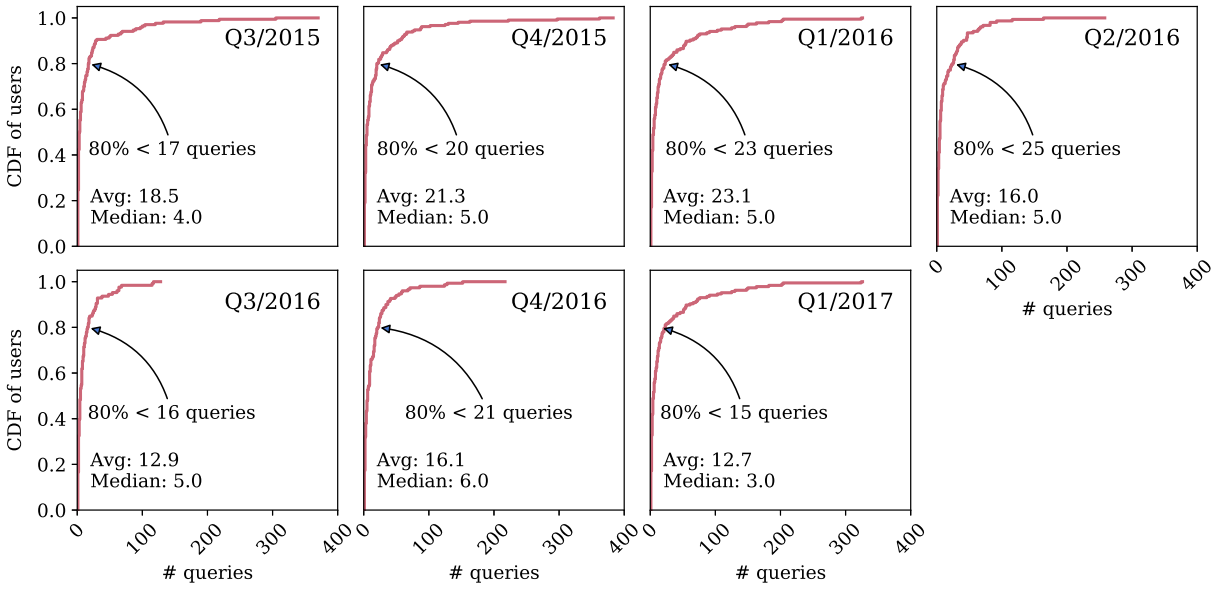


FIGURE 4 CDF of queries performed by users

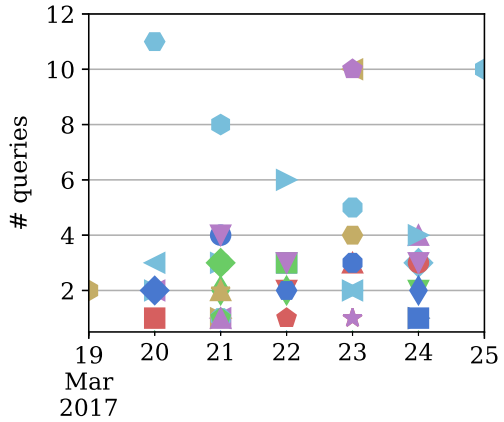


FIGURE 5 Number of access of SURFnet users to Booters

Figure 5 shows the number of queries to Booter domain names as sent by 38 users that accessed Booters during the week between March 19, 2017, and March 25, 2017. In this figure, each symbol represents an individual user (without the need for individual identification—besides the dataset is anonymized). Some of these users (cyan hexagon, cyan triangle, and purple pentagon) performed more accesses than the commonest median of accesses as showed in Figure 2 (ie, 5 accesses). This analysis clearly identifies those users that should perhaps be closely monitored, to prevent them from becoming cybercriminals and, consequently, reducing the number of DDoS occurrences.

The analysis we present in this section helps identify popular Booters, which are likely to be those launching most DDoS attacks among Booters and setting a priority order for mitigation actions. Analysis on users behavior concerning accesses to Booters provides supporting information for organizations, such as SURFnet, to take preventive actions. Such actions are in line with EUROPOL operation named Tarpit, which aims at raising awareness about the illegal character of DDoS attacks and Booters.[13]

The Dutch police took part in the Tarpit operation and approached users that performed DDoS attacks from Booters.[14] The list of Booters used in our work helps the Dutch police identify users that accessed Booters; and such information could be further correlated with (leaked) Booter databased containing records of users that hired DDoS attacks. As expected, not all accesses to Booters result in the hiring and launching of an attack. For example, as reported in Krebs,[15] while online and operational, the vdos-s.com Booter launched more than 170 thousand attacks hired by its users; however, none of the 67 SURFnet users that accessed this Booter had a registered purchase in the leaked database of vdos.com.

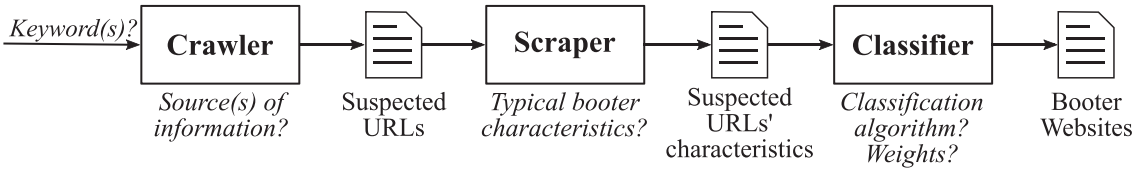


FIGURE 6 Elements and open questions for the development of our methodology

Finally, besides monitoring users from within a network, the Booter Blacklist can be used to identify third party organizations that are (in)directly involved with Booters. In Santanna et al,[16] a list of Booters is used to identify different types of organizations involved with Booters, from Web hosting to payment system and DNS-related operations; all these organizations could play an important role in mitigation strategies against Booters.

3 | METHODOLOGY FOR BOOTER LIST GENERATION

After presenting the benefits that a Booter list can pose (in Section 2), in this section, we describe our methodology for automatic generation of a comprehensive and accurate booter Website list. We define 3 main requirements for our method of creating a list of Booters: automatic, comprehensive, and accurate. Being automatic has the reason on the dynamicity of the Booter phenomenon. Often, new Booters appear and others disappear, and a manual strategy for Booter list generation would not be suitable. The comprehensiveness is required to enable understanding how broad is the booter phenomenon. The third requirement, accurate, is critical because we do not want any non-Booter website to suffer investigation or mitigation on account of our listing method.

To meet those 3 requirements, 3 elements are needed: (1) a crawler, (2) a scraper, and (3) a classifier. The crawler is responsible for collecting URLs that are suspected to be an actual Booter website. The scraper, in turn, collects detailed information on the list of suspected URLs. Finally, the classifier analyzes the characteristics of suspected URLs to categorize whether they point to a Booter Websites or not. Each one of these 3 elements has specific open questions that we address in the remaining of this paper. Figure 6 shows the elements and open questions for the development of our methodology.

While the comprehensiveness requirement is connected to the crawler, accuracy is related to the scraper and the classifier. To meet comprehensiveness, the crawler must be able to retrieve information from a consistent source of information. In addition to the source of information, the crawler must receive a coherent set of keywords to search for the source of information.

In the first step to meet accuracy in the generation of a Booter list, the scrapper must retrieve, from the suspected URLs, the characteristics that define a typical Booter Website, instead of any generic website. The second step is definite; the best algorithm for Booter website classification. There are many algorithms for website classification. We target finding an algorithm that classifies Booters and non-Booter websites based on the set of characteristics collected by the scraper. Finally, the third step to meet the accuracy requirement is to investigate the usage of weights applied to Booter characteristics. In the literature, weighted approaches improve the accuracy of website classification. We, therefore, would like to know if this situation holds for Booter website classification too.

In the next section, we identify the sources of information and the keywords that enable us to collect URLs suspected of being Booter websites. Afterwards, in Section 5, we describe the characteristics that we use to define actual Booter websites. We use these characteristics, in Section 6, to analyze classification approaches and determine which fits best into our objective function for Booter website classification.

4 | CRAWLER: LISTING SUSPECT BOOTERS URLS

The first step of our methodology for generating a blacklist of Booters consists in collecting a list of URLs from suspect websites. To do so, we developed a crawler that retrieves URLs mainly from search engines using 3 keywords relevant to Booters: *booter*, *stresser*, and *ddoser*. Our choice for these 3 keywords derives from the dictionary of keywords used in our previous work.[12] Although the additional keywords *ddos-for-hire* and *ddos-as-a-service* were also used in previous work, we observed that all URLs identified by these 2 keywords were also associated with at least 1 of the 3 other keywords.

Our crawler scrapes results of the Google search engine as the main source for suspect URLs. Using the aforementioned keywords, our crawler is able to retrieve around 800 suspect URLs per search term. This number is a limitation of the maximum number of results per search as fixed by the Google engine. The obtained result is a significant improvement when compared to previous work[12] that was able to retrieve around 370 suspect URLs using the Google search API. In addition to Google search engine, our crawler looks for suspect URLs in the description of around 500 videos returned from searches on YouTube using the same keywords, and also at millions of posts in the *server stress testing (SST)* section of the forum *http://hackerforums.net/*. The total number of *distinct* URLs collected by our crawler (considering Google search engine, Youtube, and hackerforums.net) was 928, which is used in the remainder of this paper.

We would like to highlight there that although the SST section of hackerforums.net was a very useful source of information, it is no longer available.[17] The website booterhub.com was recently created by the hacker community as the explicit substitute version of hackerforums.net SST section. Following these changes, we have adapted our code to be able to retrieve information from this website. However, this is not included in this paper.

Additional sources for searches and keywords could lead to larger lists of suspect URLs than those returned by our crawler. For example, we tested search engines that work only for websites reachable by The Onion Router (TOR) network. The content in this network is not public (ie, reachable, for example, by Google search) but an overlay network that enables anonymous communication. For TOR network, we crawled the results from *ahmia.fi* and *torsearch.es*. However, the few URLs returned from this analysis consisted of a subset of those also identified from the analysis of the *hackerforums.net* posts. In contrast to *Youtube* and *hackerforums.net*, search engines for TOR returned duplicated results only, and it was therefore decided not to include them in the set of sources of information used by the proposed crawler. Our experience suggests that extra suspect URLs are likely to be either duplicates or false positives (not a Booter).

The comprehensiveness and relevance of the list of suspect URLs retrieved by our crawler are features that make it more efficient than simply using existing crawler APIs (a list of public APIs can be found at *https://en.wikipedia.org/wiki/Web_crawler*). The source code of our crawler is available at *https://github.com/jjsantanna/Booter-black-List*.

5 | SCRAPPER: COLLECTING URL INFORMATION

The second step of our methodology for generating a blacklist of Booters consists of acquiring information of each URLs collected by the crawler (described in the previous section). We combine the most relevant set of characteristics found in the general literature of website classification. In contrast to the previous approaches, however, we take all of them into account, which provide more information to use by the classifier (described in the next section). Our set is composed of 15 characteristics, which include the 7 most relevant characteristics proposed in Chromik et al[12] and 8 coming from multiple works. From Chromik et al,[12] we use the following characteristics:

P1. Number of pages: the total number of internal pages in the website;
P2. Time span: the time span of the domain name since its registration;
P3. DDoS Protection Service (DPS) subscription: determines if the suspect Booter website subscribes to DDoS protection services offered by third-party companies;
P4. WHOIS private: determines if sensitive information of a domain name (eg, contact name, address, and e-mail) is retrievable or not using WHOIS protocol;
P5. URL type: defined by the website's landing page, it indicates if the URL of the landing page is the suspect URL itself, or if it is nested within another website, or even within a subdomain of another general high-level domain;
P6. Depth level: indicates the maximum amount of inbound hyperlinks to reach any internal page within the website;
P7. Terms of services page: indicates whether the website contains a page disclaiming the rules to use the service.

The other 8 characteristics we use and their respective source are the following:

A1. Outbound hyperlinks[18]: indicates the amount of outbound hyperlinks (pointing to other domains);
A2. Alexa rank[19]: the website rank within Alexa worldwide ranking (*http://alexa.com*);
A3. Content size[20]: number of words of the visible content in the landing page;
A4. URL length[20]: the number of characters in the URL excluding the domain name;
A5. Domain expiration time[21]: time span between the current date and the expected expiration date for the URL's domain name;

A6. Content dictionary[21]: defined by the ratio between the number of matching words to our defined keywords (Section 4) and the content size;

A7. Login-form depth level[22]: number of links required to reach the login form (every Booter website contains a login form);

A8. Resolver indication[23]: determines whether a website has a service that reveals IP addresses of target systems based on, eg, the domain name, a Skype account, or an online game account.

To determine the relevance of each of the 15 characteristics, we use a list of the 928 suspect URLs collected using our crawler (described in the previous section). From a manual analysis, we identified 113 URLs from this list as being actual Booters. Although very fastidious, manually performing the classification is essential to guarantee the quality of our training dataset (as ground truth). We then scrape each of the Booter URLs collecting data related to each characteristic of interest.

Finally, we determine the characteristic relevance using the odds-ratio metric,[24] commonly used to define weights of characteristics between 2 different elements. Odds ratio is defined as the ratio of the odds of an event occurring in the positive class (booter) to the odds of it occurring in the negative class (not booter). An odds ratio of 1 indicates that a feature is equally useful in identification of both classes. An odds ratio greater than 1 implies that the corresponding feature is more useful in identifying the positive class. We use the odds-ratio metric to find characteristics that are more likely to be related to Booters than to other websites. For example, consider a list of 100 suspect URLs from which 40 are actual Booters; 35 of these Booters have a terms of service (ToS) page that only 12 of non-Booter websites have. The higher ratio (7 : 1) of ToS presence in Booter websites compared to the lower ratio (0.25 : 1) in other websites indicates that the ToS page is a characteristic more relevant to Booters. For this example, the final value of odds ratio for ToS page is 28 (7/0.25).

Table 1 shows the relevance values for the 15 characteristics used to determine whether a suspect URL is a Booter website. The values in the left half are the average results of scrapped values and the normalized values for each characteristic. Normalization is important because of the different scales between the characteristics. The right half shows the odds ratio and respective normalized values. The last column indicates the order of the characteristics, from the most important (highest normalized odds ratio) to the least important (lowest). From the normalized odds ratio, we observe that characteristics from Chromik et al,[12] Table 1, and those from other sources intercalate in terms of relevance. Therefore, we use all the 15 characteristics in the classification approach (Section 6).

The source code of our scraper was merged with the crawler for optimization purposes. It was intended to collect the information of each URL each time a URL is found. The source code is available at *https://github.com/jjsantanna/Booter-black-List*. Note that Booters can potentially change their characteristics. However, these changes can be easily taken into account as the source code is available and any modification straightforwards.

TABLE 1 Relevance of each of the 15 characteristics using a dataset of 928 URLs (113 Booters)

				Normalized values		Normalized	
ID	**Description**	**Booters**	**Non-Booters**	**Booters**	**Non-Booters**	**Odds ratio**	**Odds ratio**
P1	Number of pages	7.88	981.75	0.93	0.23	40.97	1.00
A1	Outbound hyperlinks	0.41	14.10	0.84	0.19	22.83	0.56
P2	Domain age	395.96	3564.29	0.78	0.14	22.19	0.54
A2	Page rank	1.1×10^7	3.2×10^6	0.90	0.30	20.93	0.51
A3	Content size	127.00	679.08	0.70	0.16	12.26	0.30
P3	DPS subscription.	0.73	0.21	0.71	0.21	9.07	0.22
A4	URL length	24.93	53.65	0.36	0.07	7.00	0.17
P4	WHOIS private	0.73	0.28	0.71	0.29	5.98	0.15
P5	URL type	1.04	1.20	0.96	0.80	6.00	0.15
A5	Domain exp. time	310.93	812.22	0.90	0.61	5.77	0.14
P6	Depth level	0.92	1.75	0.87	0.57	5.03	0.12
A6	Content dictionary	0.039	0.014	0.49	0.24	3.00	0.07
A7	Login-form depth level	1.38	2.06	0.52	0.27	2.92	0.07
A8	Resolver indication	0.22	0.19	0.24	0.19	1.39	0.03
P7	Terms of services page	0.47	0.44	0.47	0.44	1.13	0.03

6 | CLASSIFIER: DETERMINING BOOTER WEBSITES

The final step of our methodology is the classification of the potential Booter websites found through the previous 2 steps (Sections 4 and 5). There are many well-established classification methods that can be used to classify Booter websites (eg, the ones proposed in previous studies[25-27]), but none of which can succeed in all the cases. In this section, we evaluate the best classification method among 8 well-established. Firstly, (in Section 6.1), we describe metrics used to measure classification accuracy, which we then apply in our analysis (in Section 6.2) to define the best classification method for Booters.

6.1 | Classification accuracy metrics

The accuracy of a classification approach is measured in terms of successes and errors typically given in a confusion matrix[18]; this matrix implies that a URL is classified in one of the following groups:

- **True positive** (T_P): when a website is correctly classified as a Booter;
- **True negative** (T_N): when a website is correctly classified as a non-Booter website;
- **False positive** (F_P): when a non-Booter website is incorrectly classified as a Booter;
- **False negative** (F_N): when a Booter website is incorrectly classified as a non-Booter website.

The *classification success* is defined by the Classification accuracy rate (CAR) given by

$$CAR = (T_P + T_N)/n, \tag{1}$$

where n is the total number of tested websites. The misclassification (error) rate is given by the *false positive error rate* (FP_{er}) and the *false negative error rate* (FN_{er}), which are given by

$$FP_{er} = F_P/n \tag{2}$$

and

$$FN_{er} = F_N/n, \tag{3}$$

respectively, where F_P is the total number of false positives and F_N the total false negatives.

6.2 | Towards the best Booter classification method

To determine the best classification method for Booter websites, we analyze the 8 most used methods from the literature of website classification. Our goal is to define which of them provides the highest *CAR*. Five of these methods are distance metrics: Euclidean distance,[25] squared Euclidean distance,[28] Manhattan distance,[26] Fractional distance,[29] and Cosine distance.[27] We include the k-Nearest Neighbors (k-NN)[27] and the Naive Bayes[20] classification methods to our investigation. Finally, we propose a supervised machine learning algorithm to improve the results of the best method among the previous described.

Based on the results reported in the literature (summarized in Table 2), we expect that Euclidean, and sqr Euclidean distances will produce unsatisfactory classification rates for Booter websites as we use 15 characteristics ("n") and these 2 metrics are "weak" when used with more than 3 characteristics. For the same reason, instead, Fractional distance, k-Nearest Neighbors, and Naive Bayes are supposed to produce better results than the former 2 approaches, as presented in Table 2. We also present in Table 2 the formulas that we use to calculate the distance metrics. The formulas for k-Nearest Neighbors and Naive Bayes metrics were omitted. For more information, please access their respective references (also presented in Table 2).

The distance metric methods aim at classifying a vector $\vec{v}$, which contains a set of n dimensional features, based on another vector $\vec{p}$ that contains a *perfect* set of features. When the distance between the 2 vectors is smaller than a predefined *threshold*, the vector $\vec{v}$ is classified positively, otherwise negatively. Note that when using distance metric, the objective is to meet the following objective function

TABLE 2 List of classification approaches order by expected accuracy for more than 3 characteristics (n>3)

Classification approach	Formula	Reference	n>3	Complexity	Efficiency
Euclidean distance	$\sqrt{\sum_{i=1}^{n}(a_i - b_i)^2}$	25,30	weak	low	high
Sqr Euclidean distance	$\sum_{i=1}^{n}(a_i - b_i)^2$	28	week	low	high
Manhattan distance	$\sum_{i=1}^{n} \lvert a_i - bi \rvert$	25,26	medium	low	high
Cosine distance	$\frac{\sum_{i=1}^{n} a_i \times b_i}{\sqrt{\sum_{i=1}^{n} a_i^2 \times \sum_{i=1}^{n} b_i^2}}$	27	medium	low	high
Fractional distance	$\sum_{i=1}^{n}(a_i - bi)^f)^{\frac{1}{f}}$	25,29	strong	medium	high
k-Nearest Neighbors	–	27,31,32	strong	low	medium
Naive Bayes	–	19-21	strong	medium	high

$$F_O(threshold) = \begin{cases} \max CAR \\ \min FP_{er} \,|\, FP_{er} \leq FN_{er}, \end{cases} \tag{4}$$

where the "threshold" is such that vector $\vec{v}$ and $\vec{p}$ have the highest *CAR* and lowest FP_{er}.

For our study on Booters, vector $\vec{v}$ is a new list of 465 suspect URLs and their respective 15 characteristics (collected using Sections 4 and 5); vector $\vec{p}$ is our trained data that were acquired by analyzing the 15 characteristics of the 928 URLs manually analyzed (previously presented in Table 1 column "Normalized values" of "Booters"), also showed bellow.

$$\vec{p} = \quad [0.93, 0.84, 0.78, 0.90, 0.70, 0.71, 0.36, 0.71, 0.96, 0.90, 0.87, 0.49, 0.52, 0.24, 0.47] \tag{5}$$

To be able to calculate *CAR*, FP_{er}, and FN_{er}, we manually analyzed the 465 suspect URLs and observed that the set contains 140 Booters and 325 other websites. For the analysis of each one of the 5 distance metric methods, we vary the *threshold* 100 times from 0 to 1 (steps of 0.01). In this range of values, we guarantee that always will be at least one *threshold* that satisfies the objective function, which occurs when the *threshold* is equal to one and FP_{er} is always equal to zero. Then, for each value of these 100 *thresholds*, we look at the resulting *CAR*, FP_{er}, and FN_{er}. The results of these 100 experiments for each metric is presented in Table 3 (column "Unweighted"). Besides of that, in Table 3, we present the best values of *threshold* (*tfls*), *CAR*, FP_{er}, and FN_{er} among all the 100 experiments.

Additionally, it is common that the use weights in the literature to compensate characteristic that have higher impact when classifying elements in groups. Therefore, besides the unweighted approach for classification, we also adopted a weighted methodology to check whether the addition of weights improve the classification rates. We decided to use the odds ratio (described in Section 2) as initial weighting strategy. Therefore, we repeat the unweighted experiments multiplying *weights* (ie, the "Normalized" values of the "Odds ratio" in Table 1) to vector $\vec{p}$.

$$\vec{w} = \quad [1, 0.56, 0.54, 0.51, 0.30, 0.22, 0.17, 0.15, 0.15, 0.14, 0.12, 0.07, 0.07, 0.03, 0.03] \tag{6}$$

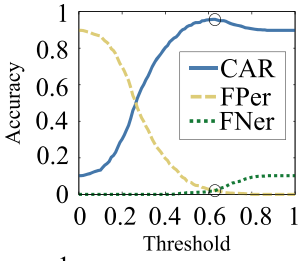
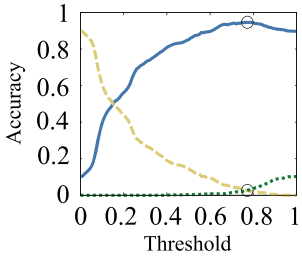
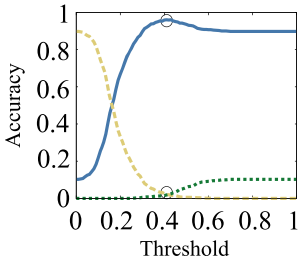
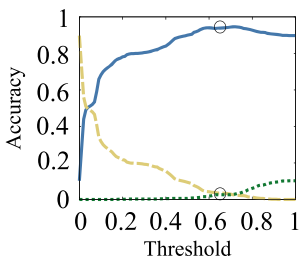
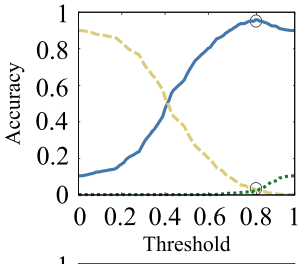
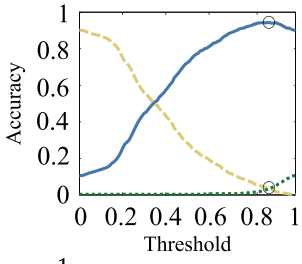
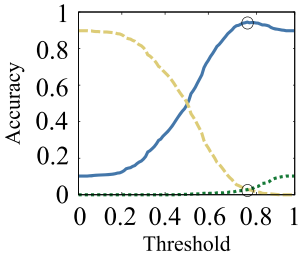
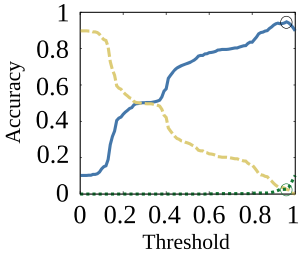
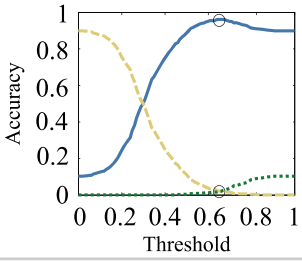
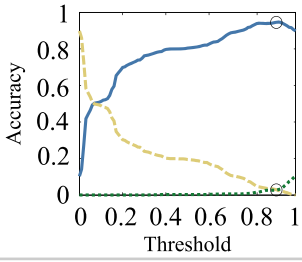
We perform this additional analysis to determine the gain in terms of accuracy of a weighted approach over an unweighted one. The results of the weighted analyses are presented in the column "Weighted" of Table 3. In this table, we also present the values of *CAR*, FP_{er}, and FN_{er} related to the best threshold (thld).

We observed in Table 3 that the variation of the *threshold* generates the same 2 patterns in all the classification methods. The first pattern *CAR* increases with the *threshold* value. The second pattern *CAR* increases proportionally to the decrease of FP_{er}. Both patterns have an exception when $FN_{er} > 0$. From this point, *CAR* has a turnaround and starts decreasing. For this reason, in all the graphs, the value of threshold that best fits the objective function is right before FN_{er} gets equal to FP_{er} (depicted in Table 3 as little circles).

We also observed in Table 3 that the weighted approaches achieve better results in all methods than the unweighted ones. For example, the Cosine distance in the unweighted approach reaches the optimal threshold (0.78) for *CAR*, FP_{er} and FN_{er} equal to, respectively 0.914, 0.049 and 0.037; and for the weighted approach the optimal threshold (0.95) is reached with 0.944, 0.022 and 0.034.

Overall, among all the distance metrics, Manhattan achieves the best result for the unweighted approach, and Cosine for the weighted (highlighted in Table 3 in bold text). Differently from the distance metrics, the inputs for the k-Nearest Neighbors (k-NN) metric are defined using actual distance metrics. The k-NN requires an empirically value for k to decide if a URL is a Booter depending whether the kth closest URLs (using a distance metric) are also Booters. For example, in a

TABLE 3 *CAR*, FP_{er}, and FN_{er} generated for different distance metrics and different threshold values. The best approaches for the weighted and unweighted approaches were Cosine and Manhattan, respectively

	Unweighted	Weigthed	Metric	Unweighted	Weighted
Manhattan dist.			CAR	**0.927**	0.931
			Thld	0.62	0.75
			FPer	0.024	0.019
			FNer	0.049	0.049
Euclidean dist.			FNer	0.049	0.49
			CAR	0.920	0.933
			Thld	0.41	0.71
			FPer	0.03	0.017
Fractional dist.			CAR	0.914	0.923
			Thld	0.82	0.86
			FPer	0.024	0.022
			FNer	0.062	0.056
Cosine dist.			CAR	0.914	**0.944**
			Thld	0.78	0.95
			FPer	0.049	0.022
			FNer	0.037	0.034
Sqr. Euclidean dist.			CAR	0.908	0.940
			Thld	0.66	0.89
			FPer	0.030	0.022
			FNer	0.062	0.039

hypothetical scenario, the k closest elements from the trained dataset to an input URL are Booters, then we consider this URL as a Booter as well. Considering that the value of k is crucial to the classification rate, we varied k from 1 to 15 (steps of 1) using each distance metric as input to k-NN and hoping that the k that gives the best value of *CAR* is lower than 15.

Table 4 shows the resulting *CAR* when using Manhattan, Cosine, and Fractional distances. While the former 2 methods performed better in the previous analysis, the latter achieved the best result of *CAR* using k-NN. For clarity, the other methods are not presented. Our choice of the values of k was judicious given that the best *CAR* for all metrics has the value 10, which is smaller than 15. Note, however, that even the best result of k-NN (considering the Fractional distance, $k = 10$, $CAR = 91\%$, $FP_{er} = 0.073$, and $FN_{er} = 0.017$) does not reach the accuracy of the best metric till this point (ie, Weighted cosine distance, $CAR = 94.4\%$).

Moving forward on defining the best metric for classifying Booter websites, we analyse Naive Bayes. We decided to investigate this metric given it is promising for classifying groups with multiple characteristics (>3), as presented in Table 2.

TABLE 4 *CAR* variation for the Manhattan, Cosine, and Fractional distance together with k-NN

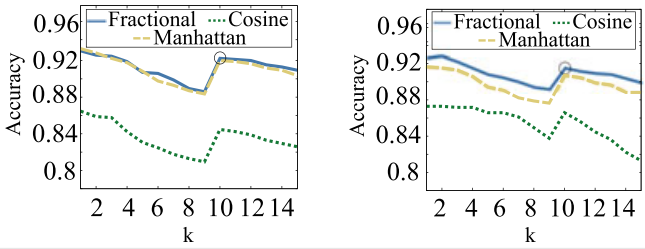
Unweighted	Weigthed	Metric	Unweighted	Weighted
90k-NN.		CAR	0.910	0.914
		k	10	10
		FPer	0.073	0.060
		FNer	0.017	0.026

TABLE 5 Probability of likelihood of each characteristic given outcome X

Characteristic	X = Booter	X=non-Booter
P(Number of pages / X)	0.97	0.23
P(URL type / X)	0.93	0.80
P(Average depth level / X)	0.94	0.67
P(Average URL length / X)	0.37	0.06
P(Domain age / X)	0.89	0.14
P(Domain res. duration / X)	0.89	0.62
P(WHOIS private / X)	0.38	0.28
P(DPS / X)	0.72	0.18
P(Page rank / X)	0.85	0.35
P(Average content size / X)	0.74	0.16
P(Outbound hyperlinks / X)	0.92	0.20
P(Category-specific dict. / X)	0.52	0.19
P(Resolver indication / X)	0.22	0.19
P(Terms of services page / X)	0.44	0.36
P(Login-form depth level / X)	0.82	0.66

TABLE 6 Prior probabilities or base rates

Outcome	Base Rate
P(Booter)	0.1001
P(non-Booter)	0.8999

To analyze this probabilistic approach (instead of distance approach), no threshold parameter is required. This method is entirely dependent on the calculated probabilities of the training dataset (928 URLs). The Naive Bayes classification metric assumes all individual characteristics into be binary, ie, a feature is either true or false. As some of the Booter website characteristics are decimal values in the unit interval, we first have to transform these to their binary equivalents before calculating the individual probabilities. This transformation is accomplished by converting all decimal valued metrics of value higher or equal to 0.5 to 1.0 and, similarly, the other way around. For instance, if we take a normalized number of pages characteristic score of 0.72, which is equal or higher than 0.5, we convert this score to 1.0. Given the whole training dataset of binary characteristic values, we can calculate all individual characteristic probabilities as resulted in Table 5.

Furthermore, Booters and non-Booters are not evenly distributed in the training dataset. Therefore, we calculate the probabilities of any random feature vector being either a Booter or a non-Booter website. We calculate these prior probabilities by dividing the samples of a given outcome by the total number of samples as shown in Table 6. From all the probabilities in Tables 5 and 6, we determine the CAR, FPer, and FNer as presented in Table 7. Note, however, that the results using the Naive Bayes approach (CAR = 91.8%, FPer = 0.049, and FNer = 0.032) do not reach the weighted cosine distance approach (*CAR* = 94.4%).

As presented in Tables 3, 4, and 7, up to this point, the metrics that provided best results were the weighted approaches of Cosine distance (*CAR* = 94.4%), followed by Sqr Euclidean distance (*CAR* = 94.0%), Euclidean distance (*CAR* = 93.3%), Manhattan distance (*CAR* = 93.1%), Fractional distance (*CAR* = 92.3%), Naive Bayes (*CAR* = 91.8%), and k-NN (*CAR* =

TABLE 7 Naive Bayes classification accuracy

	Metric	Unweighted	Weighted
Naive Bayes	CAR	0.912	0.918
	FPer	0.056	0.049
	FNer	0.032	0.032

91.4%). All of them used the same weights; therefore, we consider that changing weights will keep the exact same order of approaches performance. We then decided to further improve the performance of the best metric (Cosine distance). We propose a machine learning algorithm in Algorithm 1.

Algorithm 1 Weight adaptability learning

in: $\vec{v}, \vec{p}, \vec{w}, CAR, FP_{er}, FN_{er}$
in: $max\_interactions, \mu, \sigma$
out: $\vec{w}', CAR', FP'_{er}, FN'_{er}$

1: **procedure** WEIGHADAPT2ABETTERCAR(*input*,*output*)
2: **while** ($i < max\_interactions$) || (CAR=1) **do**
3: **for** $i = 0$ to $len(\vec{w})$ **do**
4: $\vec{w}'[i] \leftarrow \vec{w}[i] * rand\_gauss(\mu; \sigma)$
5: **end for**
6: $CAR', FP'_{er}, FN'_{er} \leftarrow cosine\_dist(\vec{v}, \vec{p}, \vec{w}')$
7: **if** $CAR' > CAR$ **then**
8: $CAR \leftarrow CAR'$
9: **end if**
10: **end while**
11: **end procedure**

The main goal of the algorithm is to update weights towards an optimal weight vector. The inputs are the vectors $\vec{v}$ and $\vec{p}$, which are multiplied by the vector $\vec{w}$ to calculate the original CAR, FP_{er}, and FN_{er}. The values of vector $\vec{w}$ are the "normalized values" of the "odds ratio" in Table 1. At each algorithm interaction, the weights are multiplied by a random number within a Gaussian function with mean $\mu = 0.5$ and standard deviation $\sigma = 0.5$, generating a new weight vector $\vec{w}'$ (line 4). The values of μ and σ are set in such a way to force the new vector $\vec{w}'$ to stay in the interval $[0, 1]$, which is the interval of the original weights $\vec{w}$. After generating $\vec{w}'$, the new values of CAR, FP_{er}, and FN_{er} are computed based on Cosine distance (line 6). Then CAR assumes CAR' if a better value is found. The algorithm runs until CAR achieves the highest possible number (ie, 1) or until the number of max_interactions is reached. We decided to run 1000 times expecting the best value to be reached before this value.

In the 824 iterations of the algorithm, we obtain the following optimal weights

$$\vec{w}' = \quad [1, 0.4, 0.3, 0.47, 0.21, 0.32, 0.17, 0.19, 0.16, 0.18, 0.13, 0.1, 0.04, 0.04, 0.03], \tag{7}$$

where the order of elements follow the order of the 15 characteristics in Table 1. That is, the first element of the vector $\vec{w}'$ corresponds to the weight of the number of pages, and the last element to the ToS page.

The conclusion of our analysis is that the Cosine distance approach is the best one to classify Booter websites based on the 15 characteristics (Section 5). Using the optimal weights vector $\vec{w}'$ and threshold of 0.95, the Cosine distance achieves a classification accuracy of 95.5%. In our most recent analysis, using the Cosine distance and the vector $\vec{w}'$ to classify 10 K URLs resulted in only 7 false positive occurrences, while all the 164 online Booters were classified correctly. In this recent experiment, our classification accuracy was almost 100%. More experiments must be done to determine the stability of our approach. The source codes of the methods presented in this section, including the machine learning algorithm, are publicly available at *https://github.com/jjsantanna/Booter-black-List*.

7 | RELATED WORK

Our work in Chromik et al[12] was the first one to investigate methods for generating Booter blacklists. In that work, we used a set of 9 website characteristics to classify suspect URLs based on a classification heuristic. Instead of improving the proposed heuristic, we investigate which well-known classification approaches is the most suitable for Booter classification, which improved the accuracy of our classification approach from 85% to 95.5%. In addition, compared to Chromik et al,[12] we reduced the number of search terms used to gather suspect URLs, from a larger number of sources (search engines, YouTube, and *hackerforum.net*).

There are many papers that address the generation of blacklists within other areas, such as intrusion detection, spam, phishing, and child pornography. Most of them rely on data classification for improved accuracy. For this work (Section 6), we surveyed several classification methods, highlighting those that could potentially be used for the Booters case, named, Euclidean distance,[25,30] Squared Euclidean distance,[28] Manhattan distance,[26] Fractional distance,[25,29] Cosine distance,[26] K-Nearest Neighbors,[27,31,32] and Naive Bayes.[19-21,31] From all the studied methods, Support Vector,[33,34] Hamming distance,[35] and Genetic Algorithm[36] were not tested in our classification investigation. However, we consider these 3 methods as a future work opportunity to improve our classification accuracy.

There are papers related to Booters that are not related to blacklist generation. Historically, first, there were 2 papers[2,37] that described characteristics of TwBooter, which was a Booter that attracted a lot of attention from security specialists. In these papers, the authors brilliantly analyze TwBooter leaked database containing information of the attack infrastructure, customers, and victims. They also introduced a methodology based on hiring Booter attacks to understand their characteristics. Almost at the same period, we published a positioning paper[38] describing how we believe the Booters' phenomenon should be addressed over time. In that paper, we also introduced the idea of a Booter crawler and a manual classification, which resulted in the paper described in the first paragraph of this section.[12]

In the same year, we scrutinized the characteristics of attacks provided by 14 different Booters.[5] We also performed a thorough analysis of 16 databases of Booters that were hacked and appeared publicly available at *pastebin.org*.[4] Very similar to these previous 2 papers are the works in Bukac et al[39] and Karami et al[40] and both combined the analysis of attacks and database. While in the first, there is not much novelty; the latter presented an impressive payment intervention conducted in collaboration with PayPal and the FBI, which compromised the financial operation of 23 Booters.

Another important work related to Booter is Douglas et al,[41] in which the authors conclude that Booters are illegal (differently from legitimate stress tester). Their main argument is that Booters attack infrastructure mostly consists of compromised/misused machines (eg, botnets and amplifier services), which is also attested by previous works[5,39,40].

Differently from attacks and database analysis, the work in Hutchings and Clayton[42] uses criminology theory to explore how and why offenders begin providing booter services. In our positioning paper,[9] we argue that by observing how Booters perform attacks, we must be prepared for future attacks that could potentially "bring down the Internet." (Un)fortunately, our observation is coming true. Accordingly, in Akamai,[10] the majority of mega attacks performed today (ie, more powerfull than 100 Gbps) have Booters as responsible, while a few years ago, Booters were only responsible for 10 Gbps attacks and lower.

Finally, related to Booters investigation in general (and not related to blacklist generation), there are dozens of blog posts written by Brian Krebs at *http://krebsonsecurity.com*. This journalist is undoubtedly one of the main investigators about the Booters phenomenon. His investigation skills and his freedom to write privacy sensitive details makes his blog posts unique and insightful.

8 | CONCLUSIONS

The DDoS-for-hire phenomenon has been gaining in popularity, and attacks from Booters are becoming a daily threat. However, the power of this phenomenon has been underestimated, and only a handful of academic work, led by 2 research groups, has addressed this problem. In addition, despite these research efforts, previous work has mainly focused on a few known Booters only, overlooking how large the whole phenomenon can be. To pave the way for more comprehensive research on Booters and further mitigation of this threat, in this paper, we proposed a thorough methodology to identify them. The methodology presented in this paper goes from crawling the Internet for suspect URLs to classifying these URLs and finding as many Booters as possible. Our methodology is easily extensible (in terms of key words, characteristics, and classification methods) given that the source code is publicly available at *https://github.com/jjsantanna/Booter-black-List*.

An important note is that Booters are (so far) easy to find by general Internet users, via a Google search, for example, and not hidden in the *dark web*. Therefore, our proposed methodology does not facilitate the discovery of Booters in the Web. Our methodology does not disclose any privacy-related information as it works by using URLs and DNS information only, which is all public. Another note is that although Booters can be in any language (eg, Russian and French), the great majority of them are found to be in English (this follows the goal of Booter owners, ie, earn money by attracting more customers in the public Web).

As future work, we would like to investigate other classification metrics such as hamming distance, logistic regression, and support vector machines, which can potentially improve the classification accuracy.

With the work presented in this paper, we wish to encourage initiatives such as the one of SURFnet (Section 2). We wish our methodology to support large-scale operations to mitigate Booters and their business. For example, that of Paypal on breaking the payment link between Booters and their customers, causing the number of attacks from Booters to reduce[40]; the operation resulting in the prosecution of Booter owners convicted of cyber crimes[43]; and the operation resulting in the prosecution of Booters' customers.[44]

ACKNOWLEDGMENTS

This work is funded by FLAMINGO (EU FP7 ICT-318488) and by D3 (NWO 628001018) projects. Ricardo de O. Schmidt's research is also funded by the SAND (http:www.sand-project.nl/) and DAS (http://www.das-project.nl/) projects. We thank SURFnet and, specially, Roland van Rijswijk-Deij for the support to perform this research.

ORCID

José Jair Santanna http://orcid.org/0000-0002-8361-6729

REFERENCES

1. Kenig R. How Much Can a DDoS attack Cost Your Business? 2013. https://blog.radware.com/security/2013/05/how-much-can-a-ddos-attack-cost-your-business/.
2. Karami M, McCoy D. Understanding the emerging threat of ddos-as-a-service. In: Usenix Workshop on Large-Scale Exploits and Emergent Threats (LEET). Washington D.C.; 2013.
3. Krebs B. DDoS Services Advertise Openly, Take PayPal. 2013. https://krebsonsecurity.com/tag/booter-tw/.
4. Santanna JJ, Durban R, Sperotto A, Pras A. Inside Booters: an analysis on operational databases. In: IFIP/IEEE International Symposium on Integrated Network Management (IM). Washington D.C.; 2015.
5. Santanna JJ, van Rijswijk-Deij R, Sperotto A, et al. Booters—an analysis of DDoS-as-a-service attacks. In: IFIP/IEEE Symposium on Integrated Network and Service Management (IM). Washington D.C.; 2015.
6. Krebs B. The Internet of Dangerous Things. 2015. https://krebsonsecurity.com/2015/01/the-internet-of-dangerous-things/#more-29658.
7. Blake J, Butterly A. Who are Lizard Squad and What's Next for the Hackers? 2015. https://www.bbc.co.uk/newsbeat/article/30306319/who-are-lizard-squad-and-whats-next-for-the-hackers.
8. Softpedia. DDOS Attack on DigiD Impacts 10 Million Dutch Users. https://news.softpedia.com/news/DDOS-Attack-on-DigiD-Impacts-10-Million-Dutch-Users-348791.shtml.
9. Pras A, Santanna JJ, Steinberger J, Sperotto A. DDoS 3.0—how terrorists bring down the internet. 2016. In: International GI/ITG Conference, MMB and DFT. Washington D.C.
10. Akamai. State of the Internet/Security (Q1/2016). https://www.akamai.com/us/en/multimedia/documents/state-of-the-internet/akamai-q1-2016-state-of-the-internet-security-report.pdf. Accessed 4 September 2016.
11. Moura GCM, Sperotto A, Sadre R, Pras A. Evaluating third-party bad neighborhood blacklists for spam detection. In: International Symposium on Integrated Network Management (IM). Washington D.C.; 2013.
12. Chromik JJ, Santanna JJ, Sperotto A, Pras A. Booter websites characterization: towards a list of threats. In: Brazilian Symposium on Computer Networks and Distributed Systems (SBRC). Washington D.C.; 2015.
13. Europol. Cyber Crime vs Cyber Security: What Will You Choose? 2016. https://www.europol.europa.eu/activities-services/public-awareness-and-prevention-guides/cyber-crime-vs-cyber-security-what-will-you-choose.
14. Politie.nl. Ga naar content Politie komt 15 personen op het spoor na gebruik booter- of cryperservice. 2016. https://www.politie.nl/nieuws/2016/december/12/11-politie-komt-15-personen-op-het-spoor-na-gebruik-booter--of-cryperservice.html.
15. Krebs B. Alleged vDOS Proprietors Arrested in Israel. 2016. https://krebsonsecurity.com/2016/09/alleged-vdos-proprietors-arrested-in-israel/#more-36288.
16. Santanna JJ, de O. Schmidt R, Tuncer D, Sperotto A, Granville LZ, Pras A. Quite dogs can bite: what Booters we should go after? and which are our mitigation options? *IEEE Commun Mag (ComMag)*. 2017;55(7):50-56.

17. Krebs B. Hackforums Shutters Booter Service Bazaar. 2016. https://krebsonsecurity.com/2016/10/hackforums-shutters-booter-service-bazaar/.
18. Hammami M, Chahir Y, Chen L. WebGuard: a web filtering engine combining textual, structural, and visual content-based analysis. *IEEE Transactions on Knowledge & Data Engineering*. 2006;18.
19. Kausar F, Al-Otaibi B, Al-Qadi A, Al-Dossari N. Hybrid client side phishing websites detection approach. *Int J Adv Comput Sci Appl (IJACSA)*. 2014;5.
20. Lindemann C, Littig L. Coarse-grained classification of web sites by their structural properties. In: International Workshop on Web Information and Data Management. Washington D.C.: ACM; 2006.
21. Lindemann C, Littig L. Classifying web sites. In: International Conference on World Wide Web. Washington D.C.: ACM; 2007.
22. Ludl C, McAllister S, Kirda E, Kruegel C. On the effectiveness of techniques to detect phishing sites. In: Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA). Washington D.C.; 2007.
23. Krebs B. The World Has No Room For Cowards. 2013. https://krebsonsecurity.com/2013/03/the-world-has-no-room-for-cowards/.
24. Garera S, Provos N, Chew M, Rubin AD. A framework for detection and measurement of phising attacks. In: ACM Workshop on Recurring Malcode. Washington D.C.; 2007.
25. Aggarwal CC, Hinneburg A, Keim DA. On the surprising behavior of distance metrics in high dimensional space. *Database Theory ICDT*. 2001;1973.
26. Hinneburg A, Aggarwal CC, Keim DA. What is the nearest neighbor in high dimensional spaces? In: International Conference on Very Large Data Bases (VLD). Washington D.C.; 2000.
27. Kriegel HP, Schubert M. Classification of Websites as sets of feature vectors. In: International Conference Databases and Applications (IASTED). Washington D.C.; 2004.
28. Kaplantzis S, Mani N. A study on classification techniques for network intrusion detection. In: Conference on Networks and Communication Systems (NCS). Washington D.C.; 2006.
29. Howarth P, Rüger S. Fractional distance measures for content-based image retrieval. In: European Conference on IR Research (ECIR). Washington D.C.; 2005.
30. Chang T, Kuo CCJ. Texture analysis and classification with tree-structured wavelet transform. *IEEE Trans Image Process*. 1993;2(4):429-441.
31. Sun C, Rampalli N, Yang F, Doan A. Chimera: large-scale classification using machine learning, rules and crowdsourcing. *VLDB Endowment*. 2014;7(13):1529-1540.
32. Weinberger KQ, Blitzer J, Saul LK. Distance metric learning for large margin nearest neighbor classification. In: Advances in Neural Information Processing Systems. Washington D.C.; 2005:1473-1480.
33. Joachims T. Text categorization with support vector machines: learning with many relevant features. In: 10th ECML. Washington D.C.; 1998.
34. Zhang Jb, Xu Zm, Xiu Kl, Pan Qs. A Web site classification approach based on its topological structure. *Int J Asian Language Process*. 2010;20:75-86.
35. Manevitz LM, Yousef M. One-class svms for document classification. *J Machine Learn Res*. 2002;2:139-154.
36. Aldwairi M, Alsalman R. MALURLS: a lightweight malicious Website classification based on URL features. *J Emerg Technol Web Intelligence*. 2012;4(2):128-133.
37. Karami M, McCoy D. Rent to Pwn: analyzing commodity Booter DDoS services. *login:: the magazine of USENIX & SAGE*. 2013;38(6):20-23.
38. Santanna JJ, Sperotto A. Characterizing and mitigating the DDoS-as-a-service phenomenon. In: IFIP International Conference on Autonomous Infrastructure, Management and Security (AIMS). Washington D.C.; 2014.
39. Bukac V, Stavova V, Nemec L, Riha Z, Matyas V. Service in denial—clouds going with the winds. In: Network and System Security (NSS). Washington D.C.; 2015.
40. Karami M, Youngsam P, McCoy D. Stress testing the Booters: understanding and undermining the business of DDoS services. In: International Conference on World Wide Web (WWW). Washington D.C.; 2016.
41. Douglas D, Santanna JJ, de O. Schmidt R, Granville LZ, Pras A. Booters: can anything justify distributed denial-of-service (DDoS) attacks for hire? *J Information, Commun Ethics in Society (JICES)*. 2017;15(1).
42. Hutchings A, Clayton R. Exploring the provision of online Booter services. *Deviant Behavior*. 2016;37(10).
43. Tassi P. Lizard Squad Hacker Who Shut Down PSN, Xbox Live, And An Airplane Will Face No Jail Time. https://www.forbes.com/sites/insertcoin/2015/07/09/lizard-squad-hacker-who-shut-down-psn-xbox-live-and-an-airplane-will-face-no-jail-time/.
44. Krebs B. Six Nabbed for Using LizardSquad Attack Tool. 2015. https://krebsonsecurity.com/2015/08/six-nabbed-for-using-lizardsquad-attack-tool/.

José Jair Santanna is an Assistant Professor at the University of Twente, the Netherlands, where he is a member of the Design and Analysis of Communication Systems (DACS) group. His current research aims at a multidisciplinary investigation of Booter Websites. His published works already cover most of aspects related to Booters (from the attack characteristics to the ethical arguments that support the phenomenon). He is interested on the areas of Internet security, measurements, management, and (big) data analysis applied to cyber security.

Joey de Vries graduated in Computer Science from University of Twente in 2015. His research involved the usage of classification algorithms for an extensive collection of Booter Websites. Currently, he works in the field of 3D real-time computer graphics, specifically engaged in the OpenGL and Vulkan graphics API. His research interests are in the areas of computer security and computer graphics.

Ricardo de Oliveira Schmidt is a Postdoctoral Researcher within the chair of Design and Analysis of Communication Systems, at the University of Twente, and a Research Engineer at SIDN Labs, the Netherlands. He obtained his PhD from the University of Twente in 2014. His research interests are in Internet security, management, and measurements.

Daphne Tuncer is a Research Associate in the Communications and Information Systems Group (CISG), Department of Electronic and Electrical Engineering, University College London (UCL), UK. She obtained her PhD in Electronic and Electrical Engineering from UCL in November 2013. Her research interests are in the areas of software-defined networks (in particular applied to network resource management), cache/content management, and adaptive network resource management.

Lisandro Zambenedetti Granville is an Associate Professor at the Federal University of Rio Grande do Sul. He served as TPC Co-Chair of IFIP/IEEE DSOM 2007 and IFIP/IEEE NOMS 2010, and as General Co-Chair of IFIP/IEEE CNSM 2014. Lisandro is Chair of the IEEE ComSoc's Committee on Network Operations and Management (CNOM), Co-Chair of the IRTF's Network Management Research Group (NMRG), and President of the Brazilian Computer Society. His interests include network management, software-defined networking (SDN), and network functions virtualization (NFV).

Aiko Pras is a Professor Internet Security at the University of Twente (UT), the Netherlands, where he is a member of the Design and Analysis of Communication Systems (DACS) group. His research interests include Internet security, measurements, and management. He is chairing the IFIP Technical Committee on "Communications Systems" (IFIP-TC6) and has been chair of the EU Future Internet cluster and coordinator of the European Network of Excellence on "Management of the Future Internet" (FLAMINGO). He serves in many steering committees and editorial boards.

How to cite this article: Santanna JJ, de Vries J, Schmidt R de O, Tuncer D, Granville LZ, Pras A. Booter list generation: The basis for investigating DDoS-for-hire websites. *Int J Network Mgmt*. 2018;28:e2008. https://doi.org/10.1002/nem.2008