Booter Blacklist: Unveiling DDoS-for-hire Websites

José Jair Santanna*, Ricardo de O. Schmidt*, Daphne Tuncer†
Joey de Vries*, Lisandro Z. Granville‡ and Aiko Pras*
*University of Twente
{j.j.santanna, r.schmidt, j.devries-1, a.pras}@utwente.nl
†University College London
d.tuncer@ee.ucl.ac.uk
‡Federal University of Rio Grande do Sul
granville@inf.ufrgs.br

***Abstract*—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. The methodology relies on well-defined mechanisms to generate a Booter blacklist, 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. We also demonstrate the usage of our methodology applied by the Dutch NREN, SURFNet, which started using our blacklist to extend their Booters' activities monitoring.**

I. 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 were 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 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, *e.g.,*100 Gbps [3], which can cause serious problems on targets that lack proper defense. It is known that a large number of attacks is launched every day from Booters, targeting essential services available online. Example of these events are three noticeable series of attacks occurred in recent years: (i) the multiple attacks preventing tens of millions customers from connecting to Microsoft Xbox Network, Sony PSN, Instagram, and Tinder for several hours [6]; (ii) the attacks against a website of an American police department, preventing citizens from registering crimes [2]; (iii) the attack against the Dutch online service (DigID) that stores sensitive information of more than 10 million citizens [7].

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 three research groups only; 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 [8] 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 [9] that affirmed Booters as responsible for the majority of mega attacks (*i.e.,*attacks that exceeded 100 Gbps) against their clients.

The mitigation of the Booters phenomenon is still a big open challenge, and blacklists is a promising approach to address this problem. Previous work [10] has shown that blacklist is an effective solution to mitigate spam-relate 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 on the identification of accesses to Booters from within a network and forecast of potential upcoming attacks.

In this paper we introduce a methodology that uses a set of 15 characteristics to find Booters in a list of suspect URLs. Our classification methodology consists of three steps. We first collect an extensive list of suspect websites in the Internet using a crawler implemented by us (§ II). Then we scrape and analyze these suspect websites based on 15 characteristics (§ III). The results of this analysis are finally

978-3-901882-85-2 © 2016 IFIP

used to classify whether the suspect website is an actual Booter (§ IV). 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/*). SURFnet (the Dutch NREN) has been using our Booter blacklist since 2015 monitoring the accesses to Booters from within their network, while they have been monitoring Booter activities since 2013. We present statistics (§ VI) on what SURFnet has observed in terms of Booters popularity.

Our scientific contribution is the investigation of eight well-established classification methods to design a methodology for Booter blacklist generation. We strongly believe that it is important to look at other domain areas to find solutions for open problems before proposing something completely new; and in this case, this approach worked very well. We also propose our own machine-learning algorithm to improve the result of the other well-established methods. (Note that we extend our previous work in [11], in which we introduced a straightforward classification heuristic based on 9 characteristics to generate Booter blacklists). In addition to the direct contributions described above our goal is to raise awareness, in the research community and network operators, of the challenges and open issues that still have to be addressed for the mitigation of Booters and their operations.

II. 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 three keywords relevant to Booters: *booter*, *stresser* and *ddoser*. Our choice for these three keywords derives from the dictionary of keywords used in our previous work [11]. 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 two keywords were also associated with at least one of the three other keywords.

Our crawler scrappes 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 by the Google engine as the maximum number of results per search. This number is a significant improvement when compared to previous work [11] 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 *Market Place* 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.

Additional sources for searches and keywords could lead to larger lists of suspect URLs than those that our crawler returns. For example, we tested search engines for TOR networks as source of information, such as *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 are therefore not included in the set of sources of information for our 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 retrieved list of suspect URLs by our crawler is what makes 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/tree/master/Crawler/*.

III. 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 used by the classifier (described in the next section). Our set is composed of 15 characteristics, which are the 7 most relevant characteristics proposed in [11] and 8 coming from multiple works. From [11] 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 (*e.g.,* 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:

TABLE I: 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

A1. **Outbound hyperlinks** [12]: indicates the amount of outbound hyperlinks (pointing to other domains);

A2. **Alexa rank** [13]: the website rank within Alexa worldwide ranking (*http://alexa.com*);

A3. **Content size** [14]: number of words of the visible content in the landing page;

A4. **URL length** [14]: the number of characters in the URL excluding the domain name;

A5. **Domain expiration time** [15]: time span between the current date and the expected expiration date for the URL's domain name;

A6. **Content dictionary** [15]: defined by the ratio between the number of matching words to our defined keywords (§ II) and the content size;

A7. **Login-form depth level** [16]: number of links required to reach the login form (every Booter website contains a login form);

A8. **Resolver indication** [17]: determines whether a website has a service that reveals IP addresses of target systems based on, *e.g.,* 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 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 labor, this classification must to be manually performed 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 [18], commonly used to define **weights** of characteristics between two different elements. 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; and that 35 of these Booters have a terms of service (ToS) page, what 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 I shows the relevance values for the 15 characteristics for determining if a suspect URL is a Booter website. The values in the left half (PART I) are the average result of scrapped values and the normalized values for each characteristic. (Normalization is important due to the different scales among the characteristics.) The right half (PART II) 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 [11] (normal font in Table I) and those from other sources (italic font) intercalate in terms of relevance. Therefore, we use all the 15 characteristics in the classification approach (§ IV).

The source code of our scraper was merged with the crawler for optmization purposes. It was intended to collect the information of each URL at the moment that a URL is found. The source code is available at *https://github.com/jjsantanna/Booter-black-List/tree/master/Crawler*. Note that Booters can potentially change their characteristics. However this change can be easily addressed as the source code is available and modifications are straightforwards to be made.

IV. Classifier: Determining Booter Websites

The final step of our methodology is the classification of the potential Booter websites found through the previous two steps (§ II and § III). There are many well-established classification methods that can be use to classify Booter websites (*e.g.,*the ones proposed in [19], [20], [21]). There is not a single classification method that succeed for all the cases. In this section we evaluate the best classification method among 8 well-established. Firstly (in § IV-A) we describe metrics used to measure classification accuracy, which we apply in our analysis (in § IV-B) to define the best classification method for Booters.

A. Classification Accuracy Metrics

The accuracy of a classification approach is measured in terms of successes and errors typically given in a confusion matrix [12]; 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, respectively

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

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

B. 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 [19], squared Euclidean distance [22], Manhattan distance [20], Fractional distance [23], Cosine distance [21]. We include the k-Nearest Neighbors (k-NN) [21] and the Naive Bayes [14] 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.

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 two vectors is smaller than a defined *threshold*, the vector $\vec{v}$ is classified positively, otherwise negatively. Not that when we use distance metric we want to meet the following objective function

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

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

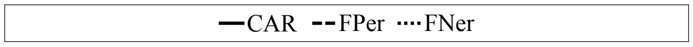
For our study on Booters, vector $\vec{v}$ is a new list of 465 suspect URLs and their respective 15 characteristics (collected using § II and § III); vector $\vec{p}$ is our trained data and is based on the characteristics of 928 URLs (previously presented in Table I column 'normalized values' of 'booters'). 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 $threshold$ from 0 to 1 (steps of 0.01). Then, for each value of the $threshold$ we look at the resulting CAR, FP_{er} and FN_{er}. The results of the analysis are presented in the left graphs of Figure 1 called as *unweighted* analysis. Additionally, we repeat the experiments multiplying *weights* to vector $\vec{p}$. The 'normalized values' of the 'odds ratio' in Table I were chose as weights. 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 right graphs of Figure 1.

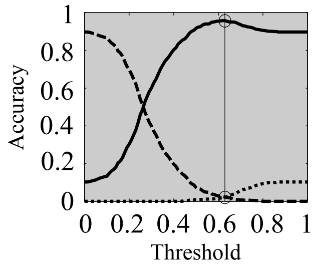
In Figure 1 the variation of the $threshold$ generates the same two patterns in all the classification methods. The first pattern CAR increases with the $threshold$ value. The second pattern CAR increases proportionally to the decrease of FP_{er}. Both patterns have an exception when $FN_{er} > 0$. From 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} equal to FP_{er} (depicted in Figure 1 as a vertical line). 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.

Among all the distance metrics, Manhattan achieves the best result for the unweighted approach, and Cosine for the weighted (highlighted in Figure 1 in a gray background). We also present the best results of CAR, FP_{er} and FN_{er} for both the weighted and unweighted approaches in Table II. This table is sorted by the highest CAR of the unweighted approach.

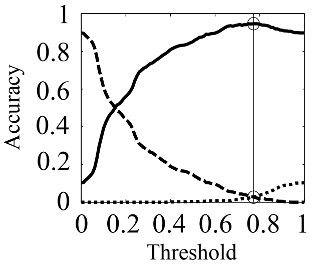
Besides the results for the distance metrics Table II also summarizes the results of the others classification methods (k-NN, Naive Bayes and machine learn). Differently from the distance metrics the inputs for the **k-Nearest Neighbors** metric are defined using actual distance metrics. Another difference is that k-NN requires an empirically value for k,



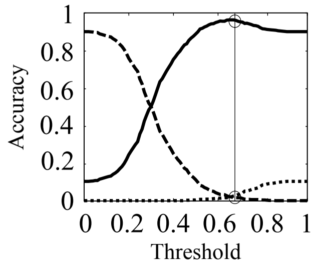
(a) Legend for the graphs bellow



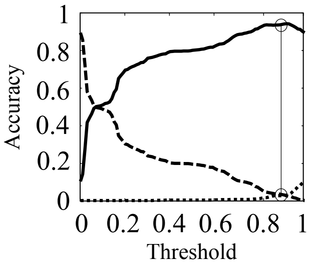
(b) Manhattan Unweighted



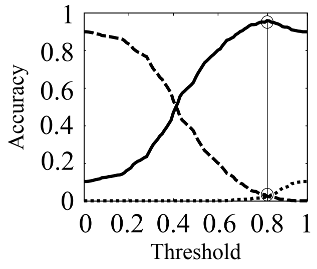
(c) Manhattan Weighted



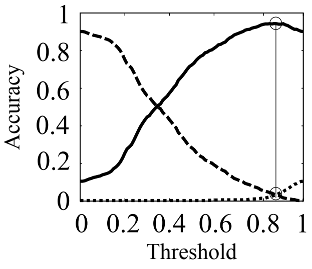
(d) Euclidean Unweighted



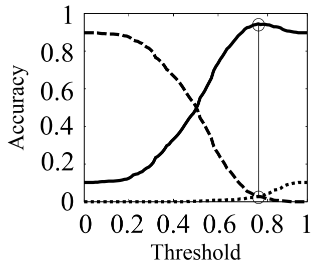
(e) Euclidean Weighted



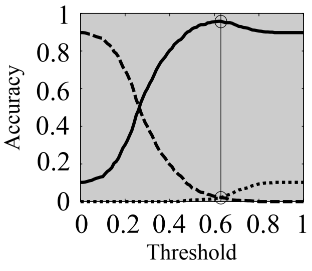
(f) Fractional Unweighted



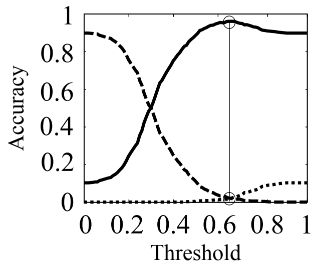
(g) Fractional Weighted



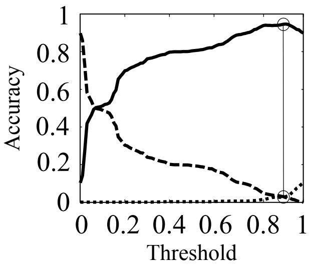
(h) Cosine Unweighted



(i) Cosine Weighted



(j) Sqr. Euclidean Unweighted

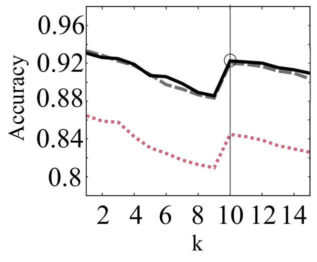


(k) Sqr. Euclidean Weighted

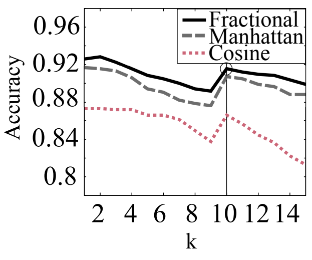
Fig. 1: CAR, FP_{er} and FN_{er} generated for different distance metrics and different threshold values; highlighted (in gray background) the best approaches for the weighted and unweighted approaches, Cosine and Manhattan respectively.

TABLE II: Summary of results for the distance metrics.

	Unweighted			**Weighted**		
Method	CAR	FP_{er}	FN_{er}	CAR	FP_{er}	FN_{er}
Manhattan dst.	0.927	0.024	0.049	0.931	0.019	0.049
Euclidean dst.	0.920	0.030	0.049	0.933	0.017	0.049
Fractional dst.	0.914	0.024	0.062	0.923	0.022	0.056
Cosine dst.	0.914	0.049	0.037	0.944	0.022	0.034
SQR Euclidean dst.	0.908	0.030	0.062	0.940	0.022	0.039
k-NN Fractional	0.910	0.073	0.017	0.914	0.060	0.026
Naive Bayes	0.912	0.056	0.032	0.918	0.049	0.032
Machine Learn				0.955	0.015	0.030



(a) k-NN Unweighted



(b) k-NN Weighted

Fig. 2: CAR variation for the Manhattan, cosine and Fractional distance together with k-NN.

which decides if a URL is a Booter depending the k^{th} closest URLs (from the trained dataset); 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.

Figure 2 shows the resulting CAR when using Manhattan, Cosine and Fractional distances. While the former two methods performed better in the previous analysis, the latter achieved the best result of CAR using k-NN. The other methods are not presented for a better clarity of the results. Our choice of the values of k was very good given that the best CAR for all metrics has the value 10, which is smaller than 15. Note in Table II that even the best result of k-NN (considering the Fractional distance and $k = 10$) does not improve the accuracy of any other previous metric.

To analyze the performance of the probabilistic approach **Naive Bayes** we use the same dataset (928 URLs) from the analysis of the distance metrics, and the additional list of 465 URLs to be tested. Although results for the Naive Bayes (Table II are better than SQR Euclidean and k-NN, they are not as good as those of Cosine Distance.

To further improve the performance of Cosine distance that so far presents the best CAR we propose a straightforward **machine learning** algorithm in Algorithm 1.

The main goal of the algorithm is to updates weights towards an optimal weight vector. The inputs are the vectors $\vec{v}$ and $\vec{p}$, which is multiplied by the vector $\vec{w}$ to calculate the original CAR, FP_{er}, FN_{er}. The values of vector $\vec{w}$ are the

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
```

'normalized values' of the 'odds ratio' in Table I. During every 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 such to force the new vector $\vec{w}$' stays in the interval $]0, 1]$, which is the interval of the original weights $\vec{w}$. After generate $\vec{w}$' the new values of CAR, FP_{er}, FN_{er} 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 (*i.e.,*1) or until the number of $max\_interactions$ is reached. We decided to run one thousand times expecting the best value to be reached before this value.

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

$$\vec{w}' = [1, 0.4, 0.3, 0.47, 0.21, 0.32, 0.17, 0.19, 0.16, 0.18, 0.13, 0.1, 0.04, 0.04, 0.03] \quad (5)$$

where the order of elements follow the order of the 15 characteristics in Table I. 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 (summarized at Table II) is that the Cosine Distance metric is the best one to classify Booter websites based on the 15 characteristics (§ III). Using the optimal weights vector $\vec{w}'$, and threshold of 0.95, the Cosine Distance achieves a **classification accuracy of 95.5%**. Recently using the Cosine Distance and the vector $\vec{w}'$ to classify 10K URLs resulted in only 7 false positive occurrences, while all the 160 reachable 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/tree/master/Classifier/*.

V. Related Work

Our work in [11] was the first one to investigate methods for generating Booter blacklists. In such work we used a set of 9 website characteristics to classify suspect URLs based on a classification heuristic. Instead of increment our heuristic defined in the previous work, 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 [11] 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 (§ IV) we surveyed several classification methods, highlighting those that could potentially be used for the Booters case, named: Euclidean distance [19], [24], Squared Euclidean distance [22], Manhattan distance [20], Fractional distance [19], [23], Cosine distance [20], K-Nearest Neighbors [21], [25], [26], and Naive Bayes [14], [15], [25], [13]. From all the studied methods Support Vector [27], [28], Hamming distance [29] and Genetic Algorithm [30] were not tested in our classification investigation. However, we consider these three 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 two papers [2], [31] that described characteristics of TwBooter, which was a Booter that called too much attention of security specialists. In those papers they brilliantly analyze TwBooter leaked database containing information of the attack infrastructure, customers, and victims. The authors also introduced the methodology based on hiring Booter attacks to understand their characteristics. Almost at the same period we published a positioning paper [32] describing how we consider 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 lately resulted in the paper described in the first paragraph of this section [11].

In the same year that we published [11] we scrutinize 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 this previous two papers there is the work in [33] and [34], 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 minimize the financial operation of 23 Booters.

Differently from attacks and database analysis there is the work in [35] that use criminology theory to explore how and why offenders begin providing booter services. There is also our positioning paper in [8], in which we argue that by observing how Booters perform attacks we must to be prepare for future attacks that would potentially "bring down the Internet". (Un)fortunately our observation are coming true. Accordingly Akamai [9] the majority of mega attacks (*i.e.,*more powerfull than 100 Gbps) have Booters as responsible, while a few years before Booters were restrictively 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.

VI. Booter Blacklist Usage

The blacklist of Booters available at *http://booterblacklist.com/* is public and periodically maintained by us. This blacklist can be used by anyone to, *e.g.,*filter traffic or simply monitor Booter operations within a given network.

In this section we analyze the dataset from an actual deployment of our Booter blacklist. SURFnet (the Dutch NREN) has been seeing many daily (small and large) DDoS attacks to and from network in their domain; and many of these attacks are known to be from Booters. Using our Booters blacklist they are (since June 2015) monitoring the access to Booters from within their network. This monitoring is running for almost a year, and we present next some statistics of what they see.

Measurement dataset. The results presented in this section are based on the monitoring of Booters access across one year at SURFnet. In short, SURFnet records information about who is accessing which Booter (based on DNS records). During this measurement period, we see more than 4.7 K accesses from 441 SURFnet users to 345 Booters; the daily average of is 53 accesses to an average of 14 different Booters. All the scripts used to perform the analyses in this section is available at *https://github.com/jjsantanna/booterblacklist_use_cases*.

Figure 3 shows the statistics of the most accessed Booters. Firstly, in the top plot we present the monthly number of access per Booter (within the top 10 in three different 4-month periods, a quarter) during the measurement period from June 2015 to May 2016. Clearly two Booters detach from the others: *booter.xyz* and *mostwantedhf.info*, with a monthly median number of accesses of 186 and 147 respectively. For all other Booters there were eventual peaks on number of accesses, such as around 200 accesses on Nov 2015 to *ipstresser.com*.

The three middle plots of Figure 3 show the top-10 most accessed Booters in a quarter. A first observation is the remarkable increase on the number of accesses from Q1 compared to Q2 and Q3; in Q2 the top-2 most accessed Booters accounted for 2.5 times more accesses than in Q1. These plots also show

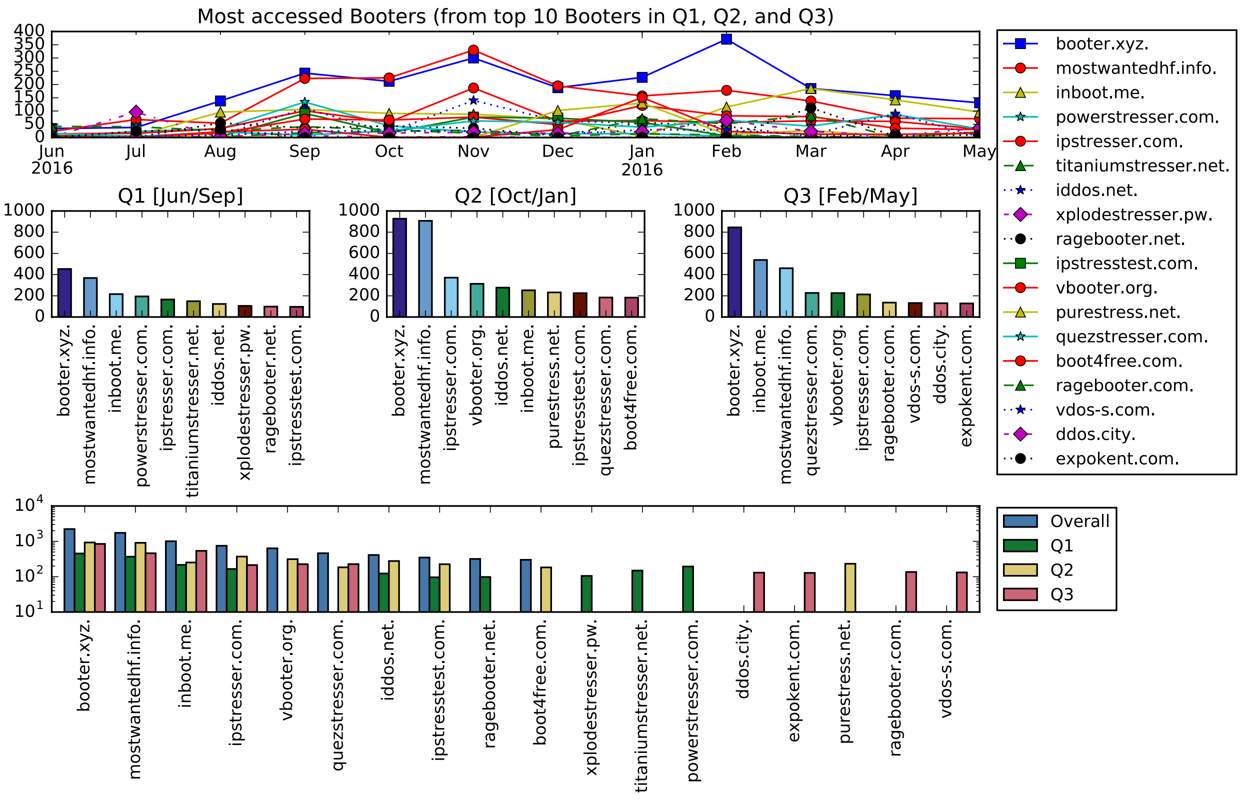


Fig. 3: Monthly number of accesses to the most accessed Booters (top plot) during the whole measurement period; the top-10 most accessed Booters per 4-month period (middle plots); and the aggregation of the access per 4-month period of the most accessed Booters (bottom plot).

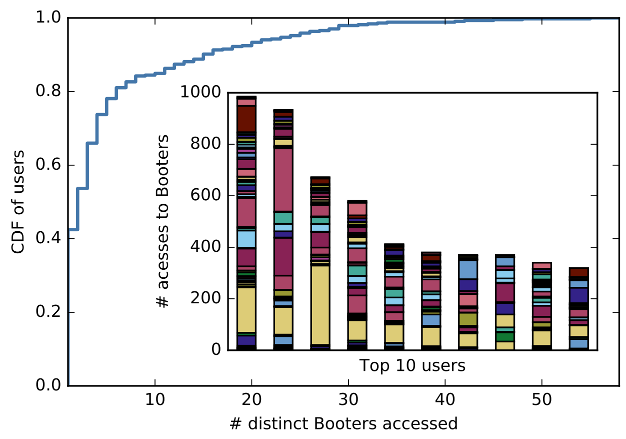


Fig. 4: CDF of the number of distinct Booters accessed by users and the number of accesses by the top-10 users (and the share of accesses for each Booter—identified in different colors.

that there is a variation among the most accessed Booters over time. Although *booter.xyz* is the most accessed Booter across all measurement period, *vbooter.org* that did not rank among the top-10 in Q1 appears as 4th most accessed in Q2 and fifth in Q3; and *inboot.me* ranked third in Q1 drops to sixth in Q2 and later becomes the second most accessed in Q3. In Q3, the last four months of measurements, three Booters from the top-10 most accessed were not observed in Q1 and Q3.

Finally in the bottom plot of Figure 3 we reinforce the observations of the middle plots. In this plot we aggregate the occurrences of Booters per quarter. This aggregation reveal clearly the Booters that were the most accessed (top 10) but appeared only in one of the 4-months period. Note that the top 10 overall (blue bar) are the ten most accessed Booters over the entire measurement period (one year). Such overall hide the other Booters that are becoming more popular, for example the ones that appeared only in the Q3.

Figure 4 shows the access statistics for the top-10 users (/24 IP blocks) that most accessed Booters. All these users accessed many Booters during the measurement period, but the figure indicates that users definitely have preference for a handful of Booters. All users (except number 2) mostly accessed the Booter *booter.xyz*.(yellow); this Booter accounted for half of the accesses from user 3. The other two most accessed Booters *mostwantedhf.info* (pink) and *inboot.me* (blue) followed in order of preference (what is in line with results from Figure 3). These three Booters accounted for an average of 48% of all accesses from the top-10 users.

We present the CDF of distinct Booters accessed per user in Figure 4 to shows that half of all users (62% users) accessing Booters in the measurement period accessed at most 4 distinct Booters. This low number of accessed Booters is because around 40% of users accessed *a single Booter only* (one or multiple times). The CDF shows that roughly 20% of users accessed 10 or more Booters and that very few users ($< 1\%$) accessed 30 Booters or more, for example as the ones presented in the inner plot.

The analysis of access to Booters **does not** necessarily tell us if an attack was ordered. However, it does shed a light on the initial understanding of the relationship between users, which are potential customers, and Booters. By investigating users that access Booter SURFnet together with the Dutch cybercrime unit police successful found and prosecute users that launched attacks. As future work we plan to combine this dataset to DDoS events known to be launched from Booters, and establish a pattern between user behavior and effective Booter hiring.

VII. 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 two research groups, have 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. In order 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 hide 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, which is all public. Another note is that there is the possibility of Booters in others languages (*e.g.,*Russian and French), however by following the goal of Booter owners (*i.e.,*earn money by attracting more customers in the public Web) the great majority of them are found in English.

As future work we aim 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 (§ VI). 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 [34]; the operation resulting in the prosecution of Booter owners convicted of cyber crimes [36]; and the operation resulting in the prosecution of Booters' customers [37].

Acknowledgment

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.

References

[1] R. Kenig, "How Much Can a DDoS attack Cost Your Business?" https://blog.radware.com/security/2013/05/how-much-can-a-ddos-attack-cost-your-business/, 2013.

[2] M. Karami and D. McCoy, "Understanding the emerging threat of ddos-as-a-service," in *USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET)*, 2013.

[3] B. Krebs, "The Internet of Dangerous Things," http://krebsonsecurity.com/2015/01/the-internet-of-dangerous-things/#more-29658, 2015.

[4] J. J. Santanna, R. Durban, A. Sperotto, and A. Pras, "Inside Booters: An Analysis on Operational Databases," in *IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2015.

[5] J. J. Santanna, R. van Rijswijk-Deij, A. Sperotto, R. Hofstede, M. Wierbosch, L. Zambenedetti Granville, and A. Pras, "Booters-An analysis of DDoS-as-a-Service Attacks," in *IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, 2015.

[6] J. Blake and A. Butterly, "Who are Lizard Squad and what's next for the hackers?" http://www.bbc.co.uk/newsbeat/article/30306319/who-are-lizard-squad-and-whats-next-for-the-hackers, 2015.

[7] Softpedia, "DDOS Attack on DigiD Impacts 10 Million Dutch Users," http://news.softpedia.com/news/DDOS-Attack-on-DigiD-Impacts-10-Million-Dutch-Users-348791.shtml.

[8] A. Pras, J. J. Santanna, J. Steinberger, and A. Sperotto, "DDoS 3.0 - How Terrorists Bring Down the Internet," in *International GI/ITG Conference, MMB and DFT*, 2016.

[9] Akamai, "State of the Internet/Security (Q1/2016)," 2016.

[10] G. C. M. Moura, A. Sperotto, R. Sadre, and A. Pras, "Evaluating third-party Bad Neighborhood blacklists for Spam detection," in *International Symposium on Integrated Network Management (IM)*, 2013.

[11] J. J. Chromik, J. J. Santanna, A. Sperotto, and A. Pras, "Booter websites characterization: Towards a list of threats," in *Brazilian Symposium on Computer Networks and Distributed Systems (SBRC)*, 2015.

[12] M. Hammami, Y. Chahir, and L. Chen, "WebGuard: A web filtering engine combining Textual, Structural, and Visual content-based analysis," *IEEE Transactions on Knowledge & Data Engineering*, vol. 18, 2006.

[13] F. Kausar, B. Al-Otaibi, A. Al-Qadi, and N. Al-Dossari, "Hybrid client side phishing websites detection approach," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 5, 2014.

[14] C. Lindemann and L. Littig, "Coarse-grained Classification of web sites by their structural properties," *ACM International Workshop on Web Information and Data Management*, 2006.

[15] ——, "Classifying web sites," *International Conference on World Wide Web*, 2007.

[16] C. Ludl, S. McAllister, E. Kirda, and C. Kruegel, "On the effectiveness of techniques to detect phishing sites," in *Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA)*, 2007.

[17] Krebs, B., "The world has no room for cowards," http://krebsonsecurity.com/2013/03/the-world-has-no-room-for-cowards, 2013.

[18] S. Garera, N. Provos, M. Chew, and A. D. Rubin, "A framework for detection and measurement of phising attacks," in *ACM Workshop on Recurring Malcode*, 2007.

[19] C. C. Aggarwal, A. Hinneburg, and D. A. Keim, "On the surprising behavior of Distance Metrics in high dimensional space," *Database Theory ICDT. Lecture Notes in Computer Science*, vol. 1973, 2001.

[20] A. Hinneburg, C. C. Aggarwal, and D. A. Keim, "What is the nearest neighbor in high dimensional spaces?" *International Conference on Very Large Data Bases (VLD)*, 2000.

[21] H.-P. Kriegel and M. Schubert, "Classification of Websites as Sets of Feature Vectors," in *International Conference Databases and Applications (IASTED)*, 2004.

[22] S. Kaplantzis and N. Mani, "A study on classification techniques for network intrusion detection," in *IASTED Conference on Networks and Communication Systems (NCS)*, 2006.

[23] P. Howarth and S. Rüger, "Fractional Distance Measures for Content-Based Image Retrieval," *European Conference on IR Research (ECIR)*, 2005.

[24] T. Chang and C.-C. J. Kuo, "Texture analysis and classification with tree-structured wavelet transform," *Image Processing, IEEE Transactions on*, vol. 2, no. 4, pp. 429–441, 1993.

[25] C. Sun, N. Rampalli, F. Yang, and A. Doan, "Chimera: large-scale classification using Machine Learning, Rules and Crowdsourcing," *Proceedings of the VLDB Endowment*, vol. 7, no. 13, pp. 1529–1540, 2014.

[26] K. Q. Weinberger, J. Blitzer, and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," in *Advances in neural information processing systems*, 2005, pp. 1473–1480.

[27] T. Joachims, "Text categorization with support vector machines: learning with many relevant features," in *10th ECML*, 1998.

[28] J.-b. Zhang, Z.-m. Xu, K.-l. Xiu, and Q.-s. Pan, "A Web Site Classification Approach Based On Its Topological Structure," *International Journal on Asian Language Processing*, vol. 20, 2010.

[29] L. M. Manevitz and M. Yousef, "One-class svms for document classification," *The Journal of Machine Learning Research*, pp. 139–154, 2002.

[30] M. Aldwairi and R. Alsalman, "MALURLS: A Lightweight Malicious Website Classification Based on URL Features," *Journal of Emerging Technologies in Web Intelligence*, may 2012.

[31] M. Karami and D. McCoy, "Rent to Pwn: Analyzing Commodity Booter DDoS Services," *; login:: the magazine of USENIX & SAGE*, vol. 38, no. 6, pp. 20—-23, 2013.

[32] J. J. Santanna and A. Sperotto, "Characterizing and mitigating the DDoS-as-a-service phenomenon," in *IFIP International Conference on Autonomous Infrastructure, Management and Security (AIMS)*, 2014.

[33] V. Bukac, V. Stavova, L. Nemec, Z. Riha, and V. Matyas, "Service in Denial - Clouds Going with the Winds," in *Network and System Security (NSS)*, 2015.

[34] M. Karami, P. Youngsam, and D. McCoy, "Stress Testing the Booters: Understanding and Undermining the Business of DDoS Services," in *International Conference on World Wide Web (WWW)*, 2016.

[35] A. Hutchings and R. Clayton, "Exploring the Provision of Online Booter Services," *Deviant Behavior*, vol. 37, no. 10, 2016.

[36] P. Tassi, "Lizard Squad Hacker Who Shut Down PSN, Xbox Live, And An Airplane Will Face No Jail Time," http://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/, p. 2015.

[37] B. Krebs, "Six Nabbed for Using LizardSquad Attack Tool," http://krebsonsecurity.com/2015/08/six-nabbed-for-using-lizardsquad-attack-tool/, 2015.