ProxyDetector: A Guided Approach to Finding Web Proxies

Zhipeng Chen, Peng Zhang, Qingyun Liu
Institute of Information Engineering, School of Cyber Security
University of Chinese Academy of Sciences
Beijing, China
{chenzhipeng, pengzhang, liuqingyun }@iie.ac.cn

***Abstract*—With the Internet becoming the dominant channel for business and life, web proxies are also increasingly used for illegal purposes such as propagating malware, impersonate phishing pages to steal sensitive data or redirect victims to other malicious targets.**

In this paper, using thousands of web proxy URLs crawled, we performed a large-scale study on the DOM (Document Object Model) structure features. Our study reveals the existence of the dedicated web proxy DOM among hosts that play orchestrating roles in proxy activities. Motivated by their distinctive features in DOM and URL, we developed an automatical stepping-stone detection system——ProxyDetector. Specially, we explored the potential benefits of considering DOM-based features, which improved 25% recall rate than before. We extensively evaluated ProxyDetector with four methods on a diverse spectrum of corpora with 2,068 web proxy sites and 26,066 legitimate sites. Capable of achieving over 95% precision of web proxy sites with a high recall rate of 96.5% on average, our ProxyDetector has been demonstrated to be an effective solution of detecting the web proxy sites.

Keywords—DOM-based features, web proxy detection

I. Introduction

Today malicious web sites provide new opportunities to criminals who are rapidly industrializing their dark business over the Web [1]. And they are gradually becoming a cornerstone of Internet criminal activities supporting criminal enterprises such as spam-advertised commerce, financial fraud, and as a vector for propagating malware (e.g., so-called "drive-by downloads") [2]. Moreover, more and more attackers always hide their identity by varieties of evasion techniques escaping from censorship systems like "The Great Firewall" of China. As the GWI (global web index) Social report [25] shows, it's Indonesia and Vietnam which lead the way (22% each), followed by China (20%), as shown in Fig.1. This trend is more and more pronounced in fast-growth markets. Over 90 million online adults in China have used one to access restricted social platforms. Many proxy servers steal and track users' information for the sake of profits. And what's more, we find that attackers mainly rely on two methods to conceal their footprints or circumvent the censor. One is encryption-based protocols and anonymous systems, e.g., Tor Browser, etc. So though current systems could meet their needs of providing covertness but unfortunately not deniability. The other method is stepping stones (i.e., proxy): launching attacks not from their own hosts but from intermediary hosts or servers which they previously compromised. So exists the same problem. By proxy servers, attackers could anonymously surf the internet without revealing their own IP addresses. Hence, stepping stone detection is much vital as well as other malicious attack detection because it's quite flexible and can be used to perform any kind of attacks such phishing attacks, DoS (Denial-of-Service) attacks etc., which increasingly become a thorny issue.

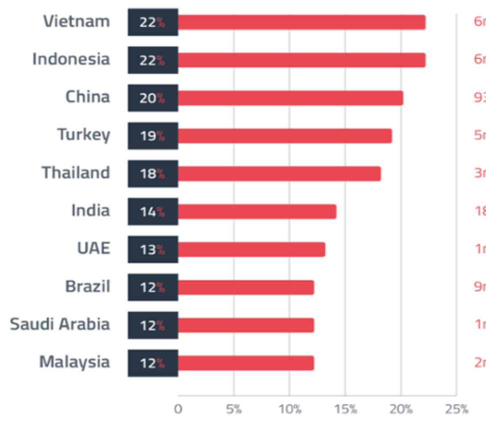
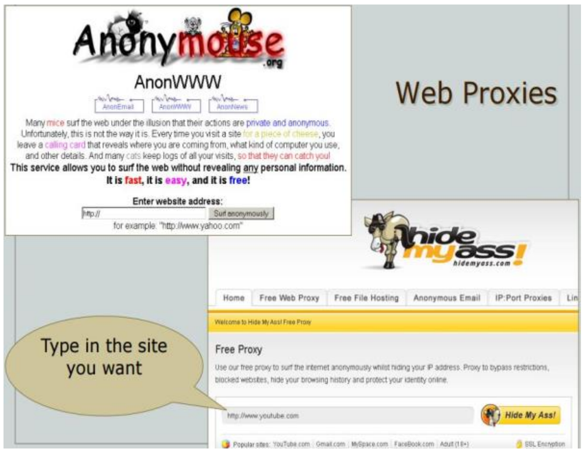


Fig 1. Proxy/VPN users based on the Internet users aged 16-64

Staniford and Heberlein first demonstrated stepping stones detection in [3]. This approach is based on the packet's content and vulnerable to encrypted stepping stones. Many existing solutions rely almost entirely on blacklisting. However, blacklists are not generally effective at preventing new or unknown stepping stones because the proxy sites have to be known before they can be added to a blacklist. Although blacklists are relatively "lowcost" on runtime, it takes a lot of time to add the rules, let alone the changing domains of web proxies regularly. On the other hand, heuristic-based detection employs common characteristics of web proxy sites, such as URL string and domain name information. However, these characteristics could be forged and the mechanism could not detect unknown web proxy sites, the same drawback as the blacklist. Researchers regard the stepping stone detections as the area of intrusion detection system (IDS) by creating Snort [4] rules, however the precision is relatively low.



Many mice surf the web under the illusion that their actions are private and anonymous. Unfortunately, this is not the way it is. Every time you visit a site for a piece of cheese, you leave a calling card that reveals where you are coming from, what kind of computer you use, and other details. And many cats keep logs of all your visits, so that they can catch you!

This service allows you to surf the web without revealing any personal information. It is fast, it is easy, and it is free!

for example: "http://www.yahoo.com"

Use our free proxy to surf the internet anonymously whilst hiding your IP address. Proxy to bypass restrictions, blocked websites, hide your browsing history and protect your identity online.

Fig.2 Web Proxy

Certainly, there are many different types of proxies in different perspectives, such as the Web proxy, HTTP proxy, VPN, Tor and so on. In this paper, based on the captured network traffic, we present ProxyDetector, a novel system to automatically detect the the web proxies in real time. The advantages about online web proxy is that it is free. This means you can enjoy all the benefits offered by the proxy server without having to incur any costs. Just as shown in Fig.2, users only type the url which they want to visit, they will bypass the censorship to surf any resource unlimitedly. ProxyDetector analyzes the relay mechanisms and characteristics, making it a valuable tool to protect a variety of online services. In table 3 (section V) we also compare with some methods used by some commercial product.

Challenges: Due to the great progress in evasion mechanisms, such as web search cloaking, fast fluxing, and domain generation, etc., detecting the stepping stones is becoming more and more challenge. Unlike the normal servers, a large number of stepping stone hosts tend to be invisible, especially for the high anonymity mode as discussed before. In other words, traffic coming through a high anonymity stepping-stone hosts will look just the same as the traffic not using any proxies. In this case, the client can completely hide its identity from the censored institution. What's more, time-based method is limited by the Internet service condition and some malicious activities by attackers. Characteristic-based is to identify traffic characteristics that are invariant or at least highly correlated across stepping stones. That's also vulnerable in varying traffic. On one hand, it is time-consuming on selecting the appropriate features. On the other hand, the features being trained change over time. The classifier should thus be able to learn the evolution of these variations. In machine learning parlance, we need a classifier that is adaptive to "concept drift" [14]. Some active measurement need employ additional packets to measure the the inter-arrival times or the delays on the network. However, such mechanism would not work in most of the proxy servers with default configurations. In the following sections, we will discuss how ProxyDetector copes with these challenges in details.

Form dedicated web proxies. To gain further understanding of web proxies, we study multiplies of web proxy URLs and DOM structures, crawled from different feeds. Our key finding is the existence of a set of form structures that play orchestrating roles in the web proxy infrastructures. From the data we have, it is pretty well confirmed to be web proxy that the form structure has a set of distinctive features. Our approach starts with a relatively small set of known web proxies as seeds and a large number of known legitimate sites from Alexa top websites [26] as references, and then features are extracted combining the merits of URL-based and DOM-based methods. In the end, those classified as 1 by machine learning mechanisms are identified as web proxies.

Contributions. The contributions of the paper are summarized as follows:

- New findings. We conduct the first study on HTML-DOMs dedicated to web proxy sites, and discover their pervasiveness in web proxy and the critical roles they play. Our study reveals their DOM features, especially improves the efficiency of the web feature extraction.
- We implemented our techniques and evaluated it on a large set of web sites, demonstrating the approach is effective and improves the state of the art.

The rest of this paper proceeds as follows. In Section II, we present the problem. Section III gives the ProxyDetector in detail. In Section IV, the user study is shown. The related work is shown in Section V, especially regarding some commercial case. Finally, we conclude this paper in Section VI.

II. PROBLEM FORMULATION

In this section, we describe in detail the goals of our work.

Goal. As mentioned previously, searching for the web proxy sites on the web is a three-step process: Crawl to collect URLs, extract the features, and use a machine learning algorithm to classify the URLs. Our goal is to improve the efficiency of the web feature extraction. More precisely, we have developed techniques that allow us to locate the unique DOM structure features which play important roles in web proxy services. With DOM structure, we refer to the Form field in the HTML DOM structure that points to web proxy pages.

Our techniques are based on the idea of searching for pages with unique features that are similar to ones that are known to be web proxy sites. Intuitively, rather than extracting all web page content, ProxyDetector focuses its extraction from a page features that characterize its proxy nature; pages with similar values for Form field features in the DOM structure are likely to be web proxy sites.

Our core contribution in this paper is to propose, implement, and evaluate a general methodology to identify the web proxies.

Host. In this paper, we call a host a "Client" if an application (e.g., a web browser, a mail agent, or an FTP client) is running on it and asking for certain services from a server. Also, we call a host a stepping stone(proxy node) if it accepts requests from a client and forwards them to another host. And we call a host a "Server" if it provides an Internet service to clients. Following this terminology, when a server accepts a request from a host, our ProxyDetector will be able to determine whether the host is a proxy or not.

Stepping Stone. In this paper, we use the terms stepping stone and the web proxy interchangeably. Both refer to the intermediary host of a connection chain which provides relay service. On the other hand, both benign web sites and legitimate web sites mean non-web-proxy sites in this paper.

We formulate the problem of web proxy detection as a binary classification task for two-class prediction: "web proxy" versus "benign". Specifically, given a data set with N web proxy URL $\{(u_1, y_1), \dots (u_n, y_n)\}$ where u_n for $t = 1,2,\dots,n$ represents a web proxy URL from the training data, and $y_n \in (0,1)$ is the corresponding label where $y_n = 1$ represents a web proxy URL and $y_n = 0$ represents a benign URL. There are two aspects in the web proxy detection:

1) Feature Extraction: For every web proxy page(or URL), we need to construct one dimensional feature vector as the input of the machine learning prediction.

2) Machine Learning: Based on the feature vectors from lots of web proxy URLs, a prediction function will be learned for predicting the class assignment.

Consider a binary classification task, the goal of the web proxy detection is to maximize the predictive accuracy. Both of the two aspects above are important for achieving the goal. While the first part of feature extraction is often based on the domain knowledge and heuristics, the second part focuses on the machine learning methods driven by datasets.

We collect 2,068 web proxy sites and 26,066 legitimate sites. As a result, our system detects 98% of web proxy sites with a false-positive rate of 5% on average.

III. PROXYDETECTOR SYSTEM

In this section, we give a detailed description of our ProxyDetector System. We start an overview of how ProxyDetector works in section III-A. In section III-B, we discuss the ProxyMiner in detail.

A. ProxyDetector System Overview

The process of ProxyDetector can be broken down into two steps. The first step is to extract features as a feature representation of the web page from all relevant information about the URL. This process is the linchpin of our ProxyDetector for the accuracy. The second step is to train classification model via a data driven optimization approach for predictions.

Fig. 3 presents the architecture of the ProxyDetector System. The initial step is to crawl many URLs based on our crawler and stores including URLs and their corresponding web pages. Then based on the Feature Extractor, we continually covert a URL into a feature vector, where several types of information can be considered and different techniques can be used. A feature representation is constructed by crawling all relevant information about the URL. These range from URL-based features (the words used in the URL, WHOIS info, etc.) to DOM-based features (forms info, action, etc.).

```
Algorithm 1:  The Online Learning Procedure
Input: the URLs
Output: the prediction value ( 1: web proxy; 0: benign web sites)
1:  sparse the web pages in terms of the URLs
2:  extract the features from the URLs and their corresponding web
      pages, construct the feature vectors.
3:  train the models using the Scikit-learn platform.
4:  predict the test datasets and update the prediction fuction if
    necessary.
5:  return K
```

Once the information is constructed, it is processed to be in a feature vector. After obtaining the feature vector in the Feature Extractor module for the collections of labelled data, a model can be trained in the Model Training module. And later, the models trained could be used to predict when new URLs arrived in Predictor module. The prediction result would be 1 if the web page is web proxy and 0 otherwise. After the prediction, the true class label is revealed to the model learner, and based on the loss suffered, the learner makes an update of the model in order to improve the predictions in the future. The general framework of our ProxyDetector is outlined in Algorithm 1.

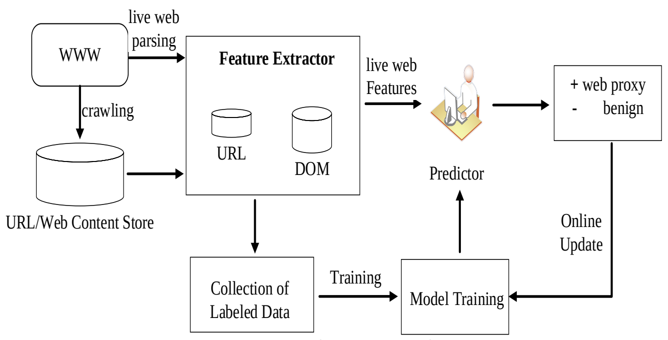


Fig. 3 the process framework of the system

B. ProxyDetector System Details

Fig.3 illustrates the ProxyDetector's working process. We'll present in detail in this module. The main components of the generic module of the ProxyDetector are discussed below.

1) URL/Web Content Store

To precisely analyze and detect web proxy sites, it requires to continuously patrol the web and collect web sites. For this dataset collection task, we mainly rely on a web-scale crawler implemented by ourselves based on the Beautiful Soup [23] to crawl and collect web pages. For the URLs used in our system, we will describe in Section IV.

2) Feature extractor

We categorize the features that we gather for URLs as being either URL-based or DOM-based.

URL-based features: URL-based features are the textual properties of the URL itself (not the content of the page of the page it references). The justification for using URL-based features is that URLs to web proxy sites tend to "look different" in the eyes of the users who see them. Hence, URL-based features provide us for web proxy site classification purposes.

(1) Containing the digits. This feature examines if a URL has some digital numbers. In a more general case, checks if a page's domain name is an IP address.

(2) Number of sensitive words in URLs. We create this feature counting the number of the sensitive words that are found in the URLs. For example, "proxy","block" and "hide".

(3) Suspicious URL. As we observed, dashes(-) is often used by web proxy URLs.

(4) Embedded domain. Web proxies sometimes add users' target's domain in the path segment to their URLs, such as http://rhe.rxxrh.com/http://www.google.com. The first part http is the web proxy, while the latter google is web proxy user's target site.

(5) Suspicious top level domain (TLD). Less common TLD appears in the web proxy sites, such as "xyz", etc.

(6) WHOIS properties. Most of web proxy sites have a short lifetime and change their domain address frequently to avoid being blocked by the blacklist. We use the WHOIS module in python to check the features of the domain's expiration date. The average expiration is no less than 287 days after computing in our datasets. So the expiration time is set 287.

DOM-based Features: Almost all web proxy sites try to trick people into typing their URLs limited by censorship or for concealing their identities through a submit form. Finding a web proxy form in practice is actually by no means trivial. For the high web proxy detection accuracy, we should guarantee that features are actually extracted from web proxy forms rather than other types of forms such as the common search form. To cope with such variations, we designed the following features to declare the existence of a web proxy form.

(1) Bad forms. Web proxies are usually accomplished through HTML forms. Hence, this feature examines whether if a web page contains potential harmful HTML forms. For the definition of harmful, a web page is required to have all of the following: (a) an HTML form; (b) some <input> tags in the form. One type of the <input> is text and the other type checkbox. (c) keywords related to sensitive information like "Encrypt URL", "Encrypt Page", "Allow Cookies", "Remove Scripts" and "Remove Objects" within the scope of the HTML form.

Form recognition is realized by the Beautiful Soup [23] using python parsers——lxml parser [28], binding for the C libraries libxml2 and libxslt. It is unique in that it combines the speed and XML feature completeness of these libraries with the simplicity of a native Python API, mostly compatible but superior to the well-known ElementTree API. Specifically, we defined five form-related keywords as shown above to narrow down our focus to forms that truly request user private information.

(2) Bad action fields. The action attribute points to the server side script (the 'back end') that handles the form submission. Usually, this will be a script (PHP,ASP, Perl) or a CGI program. The methods in the script on legitimate web sites are usually called via absolute URLs in the action field of the HTML form. However, the action field of the web proxy forms is typically a simple file name, or points to a domain different from the web proxy page domain.

(3) Bad on-submit fields. As we know, web proxy sites are usually ephemeral and provide the relay service. The key function is achieved by the on-submit field in the form. Accordingly, this feature is set to 1 if the on-submit field has some sensitive keywords, such as "return updateLocation(this)".

3) *Web sites Classification*

We implemented ProxyDetecor with four classification algorithms, including Logistic Regression, Native Bayes, Support Vector Machines with linear kernel and RBF(Radial Based Function) kernel, respectively in Python Scikit-learn [29].

Logistic Regression: This is a simple model for binary classification where examples are classified in terms of their distance form a hyperplane decision boundary. The decision rule is calculated based on the sigmoid function $\sigma(x) = [1 + e^{-x}]^{-1}$, which converts these distances into probabilities that feature vectors is positive or negative. The function is the following:

$$y = \sigma(\mathbf{w} \cdot x + b), \quad (1)$$

where the weight vector **w** and scalar bias b are parameters to be estimated from training data.

Bayes: A Bayes classifier is most easily trained by computing the conditional probabilities as:

$$P(y = 0,1|x) = \frac{P(x|y) \cdot P(y)}{P(x|y = 1) + P(x|y=0)} \quad (2)$$

The module parameters in Bayes are to maximize the joint log-likelihood of features.

Support Vector Machine (SVM): SVMs are trained to maximize the margin of correct classification, which are robust to slight perturbations. The decision rule in SVMs is based on a kernel function which computes the similarity between features. In our study, we experienced with both linear and RBF kernels.

IV. EVALUATION

In this section, we evaluate the effectiveness of the classifiers on identifying web sites to web proxies. Specifically, we want to answer the following questions: Does using more features lead to more accurate classification? What is the most appropriate classification model for detecting the web proxy? In our experiment, we evaluate our ProxyDetector on a rich corpus with ten thousands of web pages from three feature sets: URL-based, DOM-based and "Full" features. And we conducted a thorough experiment with four different classification algorithms to inspect the generality of our method as well as its well-world performance.

A. *Methodology*

The experiment runs on a machine with Intel(R) Xeon(R) CPU E5-2682 v4 @ 2.50GHz processors. We implemented four classification algorithms, including Logistic Regression, Native Bayes, Support Vector Machines with linear kernel and RBF(Radial Based Function) kernel, respectively in Python scikit-learn [29]. All the train/test was split randomly. To evaluate the impact of the percentage of web proxy sites in the training data on the detection performance, we built a series of randomly selected training sets varying from 20% to 70%. To reduce random variation and avoid lucky train/test splits, we used the average statistics over 10 runs for each dataset in all our experiments.

B. *Data Sets*

This section describes the data sets that we use for our evaluation.

Web proxy sites are usually ephemeral, and most pages will not last more than one week typically because they are taken down by administrators to avoid tracking. To fully study our approach over a larger corpus, we crawled and downloaded the web proxy sites pages when they are still live and conducted our experiment in an offline mode. Our downloader employed the urllib.request module in python to render the web pages.

There're two different sources of data to train our classifier: a benign set of websites and a ground truth stepping-stone blacklist.

Benign sites: For benign URLs, we collect top 26,066 sites from Alexa lists [26]. In this case, our assumption was that these extremely popular web sites are legitimate.

Stepping-stone blacklists. The list consists of some known proxy systems, such as anonymster [27], proxy4free [30], PHPProxy, CGIProxy, and Glype [17], and some others posted by some forums and institutions (i.e, proxy URLs in the BlackLists [16]) .

C. *Feature Comparison*

In our experiments, we adopted precision rate and recall rate as the main evaluation metrics, which are standard metrics in evaluating the classification. We also used the F1 measure, which integrates both TP (True positive rate) and FP (False positive rate) with equal weights into one summary statistic. During tuning the models, we adopted the concept of ROC (receiver operating characteristic curve) and employed the area under the ROC curve (AUC) metric, which is a good summary statistic for model comparison.

Our first experiments on web sites classification were designed to explore the potential benefits of considering different features. Obtaining different features requires different overhead and difficulty. Fig.4 shows the results of the classification as an ROC (receiver operating characteristic curve) graph using three different sets of features: URL-based features, DOM-based features and the Full features, combining URL-based features with DOM-based features. We can see that the full-featured classifier predicts web proxy sites with much better accuracy than with URL-based features and DOM-based features alone, correspondingly. For brevity, we only show detailed results from logistic regression, due to the qualitatively similar results produced by the other classifiers.

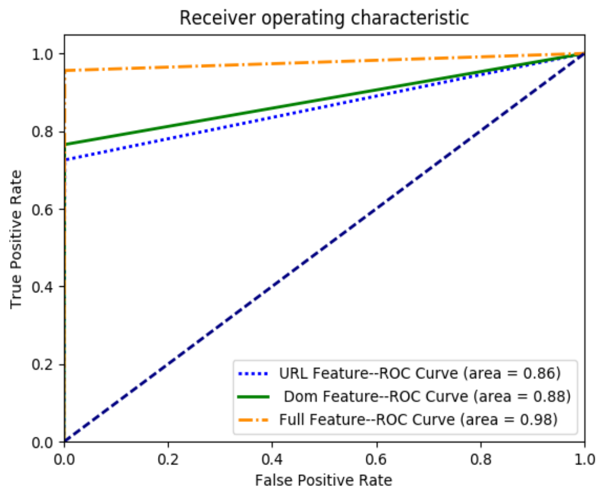


Fig.4. ROC with different sets of features

Referencing the overall rate from the Fig.4, more concretely, Table 1 shows the different feature extraction time-consuming in each feature set. The feature set consists of three features: the URL-based features, the DOM-based features and the "Full" features. We can find obtaining features with DOM-based features acquires a high collection time than that with URL-based features. However, the recall rate and F1-score with DOM-based features are much higher than that only using the URL-based features. Consequently, based on the "Full" feature set, combining the URL-based features and DOM-based features as shown in Table 1, we obtain the best classification performance.

Table 1 compares the classification rates on the three feature sets and shows their effects on classification accuracy.

Feature set	Extract Time (seconds)	Precision (%)	Recall (%)	F1-score (%)
URL	39.53	96	71.5	82
DOM	1099.35	93.75	78.75	85
FULL	1143.24	95	96.5	95.25

D. *Classifier Comparison*

Note that we build four different models that operate on the three different feature sets that we have defined previously (URL-based features, DOM-based features and the FULL features). We have evaluated the effectiveness of individual feature set using one models as the previous section. In this section, we could evaluate with different machine learning models. We train our detection models from datasets using the Scikit-learn machine learning platform. We conducted with four standard models, as shown above. In order to choose a suitable classifier (i.e., the one providing the high precision and recall rate and a reasonably small amount of time-consuming), we trained our datasets to build several models, each with a different classifier and/or different parameters.

Table 2. detection results with different methods

Feature class	Method	TrainingTime (seconds)	Precision (%)	Recall (%)	F1-score
URL	LR	0.02	97	71	0.82
	Bayes	0.01	95	72	0.82
	SVM-linear	1.28	95	72	0.82
	SVM-RBF	2.82	97	71	0.82
	Average	1.03	96	71.5	0.82
	Std.	1.33	1.15	0.577	0
DOM	LR	0.02	97	77	86
	Bayes	0.01	81	82	81
	SVM-linear	1.28	97	78	86
	SVM-RBF	2.82	100	78	87
	Average	1.03	93.75	78.75	85
	Std.	2.33	8.62	2.22	2.71
FULL	LR	0.03	99	96	0.97
	Bayes	0.01	83	98	0.90
	SVM-linear	0.49	99	96	0.97
	SVM-RBF	1.43	99	96	0.97
	Average	0.49	95	96.5	0.95
	Std.	0.66	8	1	0.035

The results for the three feature sets with four machine learning methods are presented in Table 2. It can be seen that the classifiers which produced the best values were the LR and Bayes for the URL-based features (almost the same), LR for the DOM-based features and LR for the "FULL" features. The training time is not including the feature extraction time (we can see the results in Table 1). Hence, Logistic Regression (LR) performed the best consistently among the four algorithms through extensive experiment results. In particular, LR gained an over 16% improvement over Bayes on precision rate.

What's more, combining all features for the four models substantially improves the detection recall rate. Notably, our experiments shows a precision of about 95%, and recall 96.5% in average.

V. RELATED WORK

In this section, we give a detailed description of the related work from two aspects: academic and industry study.

In present academic study, signature-based and characteristic-based methods are two main stepping stones detection mechanism. The former is based on content, such as thumbprints [3] and watermarks [5], etc. Thumbprint creates a signature by matching some attributes of the packets or packet flows to detect the stepping stones. Watermark scheme injects a watermark in the incoming flow at a host connecting to server and checks if it exists on the outgoing flow, if yes this indicates that is a stepping stone host else a normal host. However, that is challenging on the encrypted traffic. The latter approaches are based on analyzing the packet transmission characteristics. Specially, Vahid [6] uses a machine learning based approach on different types of traffic logs to identify the incoming stepping stones base traffic on the server side. Rueimin [7] proposes a server-based scheme to detect whether a host establishes a TCP connection to the server is a stepping stone or not by analyzing RTT (Round-Trip Time). But the RTT is sensitive to network fluctuation and will differ between local traffic and traffic that traverses the WAN (Wide Area Network). There are certain characteristics of network traffic such as packet size, packet timestamp, ON/OFF periods, inter-packet delay, etc., which can help to detect stepping stone hosts [8], [9], [10], [11].

Table 3. Comparison of related applications using various detection mechanisms

method / name	URL blacklist	IP filter	HTTP header filter	Machine Learning	Pre-defined rules	IP geo-location
IP2Proxy			√			√
Snort					√	
MaxMind		√				
CIPAFilter	√					
ProxyDetector	√			√		

Stepping stones detection varies from the applications [6]. Sometimes we could detect the stepping stones by analyzing HTTP headers(X-Forwarded-For), but that doesn't work all the time, because this is an optional header. What's more, Signature-based method is limited by the extraction of the features dynamically. The heuristic method is generally effective at preventing known stepping stones [17]. Hence, the tradeoff between the robustness and extendibility of the signature provides new challenges to the stepping stone detection. Although watermark mechanism is not vulnerable to chaff and timing perturbation, it assumes the message content cannot be encrypted and such active processing would require host access or control in order to make such modifications. That is unrealistic. On the other hand, characteristic-based method is chiefly composed of time-based and content-based. The time-based could have negative impacts on time sensitive traffic and is difficult to compute in a reliable and sufficient way. Many other characteristics are studied by researchers to detect the stepping stones using Machine Learning. However, these identified traffic characteristics are invariant when the classifiers have been trained. Furthermore, these methods couldn't do anything for the encrypted stepping stones. So building a classifier that is adaptive to "feature concept drift" [14] and to encryption (using the HTTPS protocol, etc.) is getting more and more essential.

In industry, there are also some commercial solutions for the stepping stones detection. Lots of examples and a comparison of what methods are used are presented in Table 2. These methods include URL list, IP filters, Packet analysis, HTTP head filters, pre-defined rules and IP geo-location. IP2Proxy [19] analyses the HTTP header X-Forwarded-For for spotting proxy traffic. However this is an optional header. CIPAFilter [20] compares

URLs with a list of known proxy websites and then blocks. The method needs to be updated over the time. MaxMind [21] uses the IP list to offer the detection service. This however runs into the same problem as using a URL list.

Generally, mainstream methods utilize common features for the stepping stone detection, few studies consider the difference features of web proxy sites. We incorporate some methods about malicious web pages detection [24] into web proxy detection, and add the special DOM-based features. And finally, our ProxyDetector is demonstrated to be very effective for web proxy detection by our experiment and is the primary contribution of our work in this paper.

VI. Conclusion

In this paper, we proposed a novel system named ProxyDetector for automatically detecting the stepping stones based on the URLs crawled in real time. There's tremendous value in aiding the forensic analysis of web traffic traces, for example to help in the investigation of the user-browser interactions in real time. To protect end users form visiting abandoned sites or monitor end users using stepping stones, we can attempt to identify suspicious web proxies' URLs by blacklists or their corresponding lexical and host-based features. A particular challenge in this domain is that web proxies are constantly evolving in a dynamic landscape. To prevail in this contest, we experimented with different approaches for detecting the stepping stones.

Through our experiments, we show that ProxyDetector can correctly detecting the stepping stones with high true positive and low false positive, and that it outperforms a previously proposed URL-based approach.

The future work includes how to dig more available abnormal behavior features hiding with malicious surfing using stepping stones. For example, some false negative cases should be considered (i.e, one user may first login on one local CDN, then connect to the proxy node).

Acknowledgment

The research work is supported by Supported by Strategic Priority Research Program of the Chinese Academy of Sciences under Grant (No.XDA06030602), National Natural Science Foundation under Grant (No.61402464, No.61402474, No.61602467) and National Key R&D Program 2016, 2016YFB0801304. The authors thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper.

References

[1] Li Z, Alrwais S, Xie Y, et al. Finding the linchpins of the dark web: a study on topologically dedicated hosts on malicious web infrastructures[C]//Security and Privacy (SP), 2013 IEEE Symposium on. IEEE, 2013: 112-126.

[2] Ma J, Saul L K, Savage S, et al. Beyond blacklists: learning toDetectMaliciousWebsitesfromSuspiciousURLs[C]//Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2009: 1245-1254.

[3] Staniford-Chen S, Heberlein L T. Holding Intruders Accountable on the Internet[C]// Security and Privacy, 1995. Proceedings. 1995 IEEE Symposium on. IEEE, 1995:39-49.

[4] Snort. https://www.snort.org/ ,2017.

[5] Peng P, Ning P, Reeves D S. On the secrecy of timing-based active watermarking trace-back techniques[C]// Security and Privacy, 2006 IEEE Symposium on. IEEE, 2006:15 pp.-349.

[6] Aghaei-Foroushani V, Zincir-Heywood / N. A Proxy Identifier Based on Patterns in Traffic Flows[M]. IEEE, 2015.

[7] Lin R M, Chou Y C, Chen K T. Stepping stone detection at the server side[C]// Computer Communications Workshops. 2011:964 - 969.

[8] He T, Venkitasubramaniam P, Tong L. Packet Scheduling Against Stepping-Stone Attacks with Chaff[J]. Proc IEEE Military Communications Conf, 2006:1 - 7.

[9] Kumar R, Gupta B B. Stepping Stone Detection Techniques: Classification and State-of-the-Art[M]// Proceedings of the International Conference on Recent Cognizance in Wireless Communication & Image Processing. Springer India, 2016.

[10] Shullich R, Chu J, Ji P, et al. A SURVEY OF RESEARCH IN STEPPING-STONE DETECTION[J]. International Journal of Electronic Commerce Studies, 2011, 2(2).

[11] Zhang Y, Paxson V. Detecting stepping stones[C]// Conference on Usenix Security Symposium. USENIX Association, 2001:263--279.

[12] Google Protocol Buffer. http://developers.google.com/protocol-buffers/ docs/overview?csw=1

[13] Seleniumwebdriver. http://docs.seleniumhq.org/projects/webdriver/,2017

[14] Gama J, Žliobaitė I, Bifet A, et al. A survey on concept drift adaptation[J]. ACM Computing Surveys (CSUR), 2014, 46(4): 44.

[15] Chen Z, Zhang P, Zheng C, et al. CookieMiner: Towards real-time reconstruction of web-downloading chains from network traces[C]//Communications (ICC), 2016 IEEE International Conference on. IEEE, 2016: 1-6..

[16] URL blacklist. http://urlblacklist.com ,2017

[17] J. Brozycki. Detecting and preventing anonymous proxy usage, SANS Inst, 2008

[18] Miller S, Curran K, Lunney T. Traffic Classification for the Detection of Anonymous Web Proxy Routing[J]. International Journal for Information Security Research, 2015, 5(1): 538-545.

[19] IP2Proxy, http://www.fraudlabs.com/ip2proxy.aspx, 2017

[20] CIPAFilter, https://cipafilter.com/, 2017

[21] MaxMind, https://www.maxmind.com/, 2017

[22] Trestian I, Ranjan S, Kuzmanovi A, et al. Unconstrained endpoint profiling (googling the internet)[C]//ACM SIGCOMM Computer Communication Review. ACM, 2008, 38(4): 279-290.

[23] Beautiful Soup, https://www.crummy.com/software/BeautifulSoup/,2017.

[24] Invernizzi L, Benvenuti S, Cova M, et al. EvilSeed: A Guided Approach to Finding Malicious Web Pages[C]// Security and Privacy. IEEE, 2012:428-442.

[25] Global Web Index Q4,2013-Q3,2014 based on the Internet users aged 16-64 http://insight.globalwebindex.net/chart-of-the-day-90-million-vpn-users-in-china-have-accessed-restricted-social-networks?ecid=

[26] Alexa. http://www.alexa.com/, 2017

[27] anonymster. https://anonymster.com/best-free-web-proxy-sites-list/, 2017

[28] lxml parser, http://lxml.de/, 2017.

[29] Scikit-learn, http://scikit-learn.org/stable/ ,2017,

[30] Proxy4free, http://www.proxy4free.com/list/webproxy1.html, 2017.