A Novel Semantic-Aware Approach for Detecting Malicious Web Traffic

Jing Yang[1,2], Liming Wang[1](✉), and Zhen Xu[1]

[1] State Key Laboratory of Information Security,
Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{yangjing,wangliming,xuzhen}@iie.ac.cn

[2] School of Cyber Security, University of Chinese Academy of Sciences,
Beijing, China

Abstract. With regard to web compromise, malicious web traffic refers to requests from users visiting websites for malicious targets, such as web vulnerabilities, web shells and uploaded malicious advertising web pages. To directly and comprehensively understand malicious web visits is meaningful to prevent web compromise. However, it is challenging to identify different malicious web traffic with a generic model. In this paper, a novel semantic-aware approach is proposed to detect malicious web traffic by profiling web visits individually. And a semantic representation of malicious activities is introduced to make detection results more understandable. The evaluation shows that our algorithm is effective in detecting malice with an average precision and recall of 90.8% and 92.9% respectively. Furthermore, we employ our approach on more than 136 million web traffic logs collected from a web hosting service provider, where 3,995 unique malicious IPs are detected involving hundreds of websites. The derived results reveal that our method is conductive to figure out adversaries' intentions.

Keywords: Web security · Malicious web traffic · Semantic analysis
Unsupervised learning

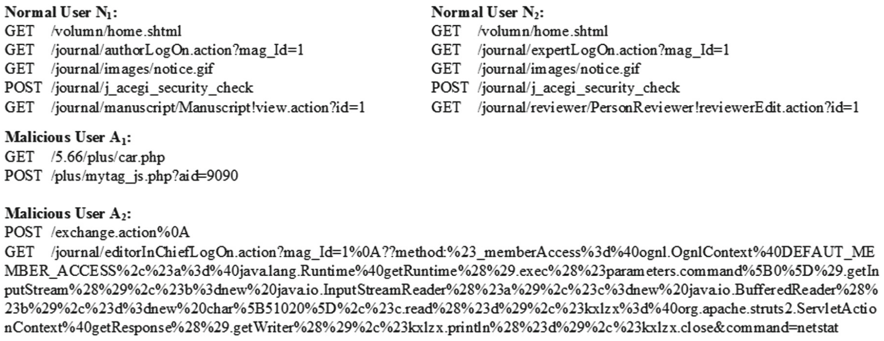
1 Introduction

Compromised websites have become increasingly attractive targets for attackers who exploit them to commit cyber crimes, such as distributing malware, controlling botnets and implementing watering hole attacks [1–3]. Due to current security mechanisms (e.g., Google Safe Browsing) for protecting users against malicious sites, attackers have to find more known and clean hosting sites [1]. Consequently any openly accessible website on the Internet may become their prey.

For a webmaster, it is meaningful to distinguish malicious web traffic generated by users who access the website not for its inherent services but for some malicious targets, such as web application vulnerabilities, uploaded malware and malicious advertising web pages. If a website is compromised, the webmaster can be notified immediately and make a remedy. If not, understanding attackers' intentions in advance can help the webmaster strengthen security strategies to prevent the site from being compromised.

Generally malicious web traffic involving web compromise can be categorized as three types. As common threats on the Internet, web scan focuses on probing websites for known weaknesses, and web penetration emphasises on finding and exploiting web application vulnerabilities with elaborately crafted web requests. Web abusing traffic, proposed in our work, refers to visits for malicious resources uploaded on a compromised site by attackers. Although extensive works [4–10] on detection of malicious web traffic have been proposed, they can only apply to some types of web attacks with respective prerequisites. In the literature there is no a generic detection model which can distinguish different malicious web traffic without depending on training data or priori knowledge, as different malicious web visits vary greatly.

In this work, we introduce a semantic-aware methodology to distinguish malicious web traffic in an unsupervised-learning way. Our key observation is that, for a website, most requests from normal users are semantically similar while different from malicious users. Regardless of human or bots, users send requests to a website for acquiring its provided services, which makes most network behavior look similar. The malicious send requests for finding or exploiting vulnerabilities, and accordingly their requests are surely different from those of the normal. As illustrated in Fig. 1, for a journal's website, almost all of requests sent from normal users are semantically relevant, and different users may send same requests with a great probability. However, requests from malicious users vary greatly, and most of them are totally irrelevant to normal requests.



```
Normal User N1:
GET   /volumn/home.shtml
GET   /journal/authorLogOn.action?mag_Id=1
GET   /journal/images/notice.gif
POST  /journal/j_acegi_security_check
GET   /journal/manuscript/Manuscript!view.action?id=1

Normal User N2:
GET   /volumn/home.shtml
GET   /journal/expertLogOn.action?mag_Id=1
GET   /journal/images/notice.gif
POST  /journal/j_acegi_security_check
GET   /journal/reviewer/PersonReviewer!reviewerEdit.action?id=1

Malicious User A1:
GET   /5.66/plus/car.php
POST  /plus/mytag_js.php?aid=9090

Malicious User A2:
POST  /exchange.action%0A
GET   /journal/editorInChiefLogOn.action?mag_Id=1%0A??method:%23_memberAccess%3d%40ognl.OgnlContext%40DEFAUT_ME
MBER_ACCESS%2c%23a%3d%40java.lang.Runtime%40getRuntime%28%29.exec%28%23parameters.command%5B0%5D%29.getIn
putStream%28%29%2c%23b%3dnew%20java.io.InputStreamReader%28%23a%29%2c%23c%3dnew%20java.io.BufferedReader%28%
23b%29%2c%23d%3dnew%20char%5B51020%5D%2c%23c.read%28%23d%29%2c%23kxlzx%3d%40org.apache.struts2.ServletActio
nContext%40getResponse%28%29.getWriter%28%29%2c%23kxlzx.println%28%23d%29%2c%23kxlzx.close&command=netstat
```

Fig. 1. Examples of requests from normal users and malicious users

Based on the observation, our methodology profiles a web user's visiting behavior and measures the degree of abnormality of a visit by utilizing a modified term frequency and inverse document frequency (TF-IDF) algorithm. A dynamic threshold is derived to distinguish abnormal users. Unlike existing works, our detection results include not only anomaly scores but also the summary information of malicious activities, which makes the detection results more understandable. We evaluate our approach with a manually labeled dataset consisting of four different websites. It is shown that our method is effective in discovering various malicious web users, as the average precision is 90.8% and the recall is averagely 92.9%. Furthermore, we employ our approach on a large dataset with more than 136 million web traffic logs from a web hosting service provider, where 3,995 unique malicious IPs are detected involving hundreds of websites. In the results, it is impressive that the semantic representation of malicious visits can help webmasters or security analysts understand malice intuitively, which is helpful to improve network defense strategies and identify compromised sites.

We organize this paper as following: Background and related work are introduced in Sect. 2. We present our approach and give details on the anomaly score computation in Sect. 3. Evaluation of our approach and results in the wild are presented in Sect. 4. Finally we make a discussion and conclude the paper in Sect. 5.

2 Background and Related Work

2.1 Classification of Malicious Web Traffic

With regard to web compromise, malicious web traffic refers to requests sent from users who browsing the website not for its inherent services but for malicious targets. We summarize various malicious web traffic into three categories according to the adversary's intention.

- **Web Scan.** In web scanning, attackers probe a website to determine whether it contains certain exploitable web resources. Typical targets in web scan include URLs of known vulnerable third-party components (e.g., `Wordpress`, `CKEditor`), URLs of known web shells (e.g., `R57`, `c99`), and sensitive file links (e.g., `www.zip`, `.htaccess`). Web scan is one of the most common web attacks and usually performed in a large scale on the Internet. In other words, most of web scanning are indiscriminate. Requests in these attacks are very similar to each other.
- **Web Penetration.** In web penetration, actual web hacking attacks, including SQL/Command injection, Cross-Site scripting and others, are performed to locate web vulnerabilities and exploit them in order to compromise a site. Furthermore, web shells would be uploaded to the compromised websites and used to control the sites by attackers. Contrary to web scanning, the requests used for web application penetration are usually elaborately crafted manually or with special web vulnerability exploitation tools. Typical features of request in this traffic are the different distribution of characters and the different structural information from normal requests.
- **Web Abuse.** After successfully subverting control a website, attackers may abuse it for further illicit activities, such as distributing malware and advertising illicit contents (e.g., drug, adult and gambling). These unwanted resources,

for a webmaster, may attract a plenty of visits from bots and human to the website accompanied with a large volume of abusing traffic. Requests in web abusing traffic usually do not contain harmful information to web applications, but the resources they ask for are never seen in previous traffic.

2.2 Related Work About Malicious Web Traffic Detection

Xie et al. [4] introduced Scanner Hunter to detect HTTP scanning. Their key assumption is that web requests of different scanners are similar with each other. Accordingly, it does not work on a site with small traffic volume, since it is not common that many scanners probe a small website.

Kruegel and Vigna [5] proposed a supervised learning based method to detect malicious web requests with a combination of six different anomaly detection models. Many following works [6–9] introduced improved detection approaches. Aiming at the problem of requiring training data, Lampesberger et al. [10] presented an online-learning detection method by transforming a request into a fixed-length symbol sequence. All of them emphasis on detection of malicious requests with crafted parameters, which only probably occur in web penetration attacks.

A web shell is an important tool used in web penetration. Canali and Balzarotti [12] analyzed communication behavior between attackers and web shells after they have compromised a website. Starov et al. [13] gave a more comprehensive study of web shells. Both of them do not involve any detection method. FireEye [14] reported a detailed analysis of the popular shell *China Chopper* and explained how to detect it through network traffic with Snort rules. For web abusing traffic, there is no detection work proposed in the literature as well, and only Alrwais et al. [2] referred to the abusing traffic and utilized it to find compromised websites.

Overall, all of the above works focus on identifying some types of malicious web traffic with respective prerequisites. There is no a generic method to detect all three types of malice without depending on training data or priori knowledge. For a webmaster, it is most concerned that how to distinguish malicious visits from massive complex web traffic by employing a direct and simple method. The challenge is that different types of malicious traffic vary greatly in terms of requests' volume, structure and semantics.

2.3 Related Work About Semantic Analysis in Network Security

Recently, semantic analysis has been more popular in the area of network security. Zhang et al. [11] proposed SBotScope to analysis large-scale malicious bot queries received by a known search engine. Liao et al. [15] introduced a technique *semantic inconsistency search* to detect illicit advertising content. For detection of malicious web traffic, semantic analysis can help avoid too much dependencies on structural information, which requires strong background knowledge on specific attacking techniques.

3 Methodology

To generically detect various malicious web traffic, we employ the literal similarity in requests from normal visits, and introduce a novel semantic-aware detection approach. Since our detection is conducted on the level of a single visit, it is crucial that how to characterize users' dynamic web visiting activities. In our approach, a user's visiting profile is represented in two word sets, and the anomaly score of each word is computed by using a modified TF-IDF algorithm. Furthermore, to avoid interference of normal but infrequent words, a global normal word dictionary is automatically generated. The anomaly score of a user is educed from scores of the two word sets. Then we derive a dynamic threshold to classify abnormal users. The concept of our methodology is shown in Fig. 2.

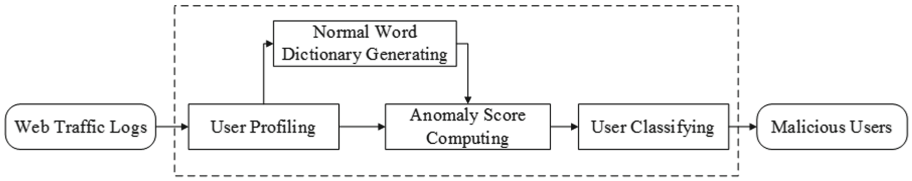


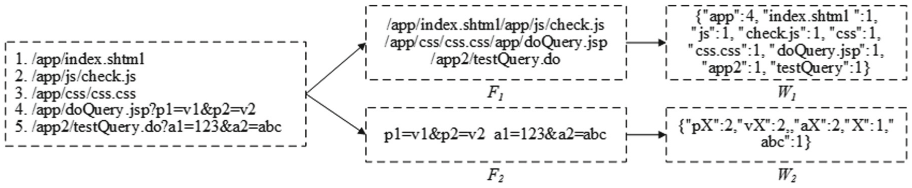
Fig. 2. The overview of our approach

3.1 User Profile

The intentions of web users are directly tied to their requests sent to websites. We derive a method to represent massive requests in aggregation to get a user's profile.

A HTTP request begins with a method token and a Request-URI which includes a resource identifier and an optional query string. After URL decoding, we extract all N_r requests for a user U, and put all resource identifiers and query strings together into two independent text files: F_1 and F_2. The method token is added in F_1 as well.

For F_1, we use `/` and `SPACE` as separators to split the file into a word set $W_1 = \{(w_{1i} : n_{1i})\}$, where n_{1i} is the occurrences of the word w_{1i} in the file F_1. For F_2, the separators are `/`, `?`, `=`, `&`, `,`, `@`, `-`, `;` and `SPACE`, and the word set is $W_2 = \{(w_{2i} : n_{2i})\}$. In addition, we replace all continuous numbers with the character `X`. An example of the user profiling process is shown in Fig. 3.



1. /app/index.shtml
2. /app/js/check.js
3. /app/css/css.css
4. /app/doQuery.jsp?p1=v1&p2=v2
5. /app2/testQuery.do?a1=123&a2=abc
/app/index.shtml/app/js/check.js /app/css/css.css/app/doQuery.jsp /app2/testQuery.do
F1
{"app":4, "index.shtml ":1, "js":1, "check.js":1, "css":1, "css.css":1, "doQuery.jsp":1, "app2":1, "testQuery":1}
W1
p1=v1&p2=v2 a1=123&a2=abc
F2
{"pX":2,"vX":2,,"aX":2,"X":1," abc":1}
W2

Fig. 3. An example of user profiling

3.2 Normal Word Dictionary

The normal word dictionary is a word set automatically derived from the words of all users. It is based on the assumption that a word is more likely to be normal if it occurs in more visits. Furthermore, even if a word is infrequent, but it is structurally similar with normal words, it is probably normal. We use n-gram to measure the structural similarity. Empirically we choose 4-gram.

To derive the dictionary set D_{nw}, firstly a global word set $G_w = \{(w_i : m_{w_i})\}$ is maintained, in which m_{w_i} is the number of users whose request contains the word w_i, and the number of total users is M. Then, we define a percentage threshold T_{fw} to distinguish frequent words, which means a word w_i is normal if it is in the first T_{fw} percent of words in G_w in descending order by m_{w_i}. All the frequent words are put in D_{nw}, and the n-gram items of them are put in NG_{nw}. For other words in G_w, if more than half of their n-gram items are in NG_{nw}, they are added into D_{nw}.

3.3 Anomaly Score

To measure the degree of abnormality of a user, we first compute the anomaly score of each word in the two files with a modified TF-IDF algorithm. TF-IDF is one of the most popular scheme in the area of information retrieval and text mining, which uses a numerical statistic to reflect how important a word is to a document in a corpus. In our method, a word's abnormality is reflected by its importance in the document F_1 or F_2, where the corpus is the global word set G_w. A word is defined as a normal word in F_b, where $b \in \{0, 1\}$, if its anomaly score equals to 0.

For a word w_{bi} in F_b the anomaly score a_{bi} is defined as following:

$$a_{bi} = tf(w_{bi}, F_b) \times idf(w_{bi}, G_w). \tag{1}$$

The computation of the inverse document frequency is the same for the both files:

$$idf(w_{bi}, G_w) = \log(\frac{M}{m_{w_{bi}}}). \tag{2}$$

Since F_1 contains words in resource identifiers, some words may occur repeatedly. Directly using the number of times that a word occurs in F_1 may amplify the score of a frequent word. Here we compute the term frequency for F_1 or F_2 separately as following:

$$tf(w_{1i}, F_1) = 1 + \frac{n_{1i}}{N_r}, \tag{3}$$

and

$$tf(w_{2i}, F_2) = 1 + \log(n_{2i}). \tag{4}$$

If a word stratifies any of following conditions, the anomaly score is directly set to 0.

- the word is in the normal word dictionary D_{nw};
- the word is the character `X`;
- the word is a non-English word (e.g., a Chinese word);
- the word ends with a postfix of some embedded static resources (e.g., `a.jpg`, `b.css` and `c.js`).

The anomaly score of F_b is the summation of all words' scores as $A_{Fb} = \sum a_{bi}$. Owing to that the abnormality of a user lies on malicious requests sent from the user, it is probable that the anomaly score of a user with more requests is higher than that with less requests, which would make that some small-scale malicious activities are ignored. To avoid it, we introduce penalty factors ω_1 and ω_2 for each file, along with an adjustment function $P(x)$. The user's anomaly score is derived as bellow:

$$A = P(\omega_1)A_{F1} + P(\omega_2)A_{F2}, \tag{5}$$

where ω_1 and ω_2 respectively indicate the proportion of normal words in F_1 and F_2. $P(x)$ is a piecewise linear function, defined as following:

$$P(x) = \begin{cases} x & \text{if } x <= 0.5 \\ 3x - 1 & \text{if } x > 0.5. \end{cases} \tag{6}$$

3.4 Dynamic Threshold

Existing works involved with anomaly scores usually use a given threshold provided externally. For example, Kreugel and Vigna [5] set the threshold of each web request to a certain percentage of the highest anomaly score. In our method, anomaly scores of different malicious users and different websites may vary greatly, and it is impractical to specify a threshold externally. Here we employ a dynamic threshold to distinguish abnormal users.

Ideally in a given set $\{A_i\}$, where A_i is the anomaly score of the user U_i, scores of normal users should be as small as possible, while malicious users are opposite. If the gap between scores of normal users and malicious users is large enough, the threshold is easily to confirm, where the gap is not necessarily the global maximum. Hence, we transfer the problem of determining the threshold to finding a local maximum gap for a monotonously non-decreasing sequence.

An intuitive method is given as following: the sequence of ordered scores $\{\tilde{A}_k | A_i^{min} \leq \tilde{A}_k \leq \tilde{A}_{k+1} \leq A_i^{max}\}$ is derived from the original set $\{A_i\}$; If $\tilde{A}_{\hat{k}}$ is the first score where the gap between $\tilde{A}_{\hat{k}}$ and $\tilde{A}_{\hat{k}-1}$ is greater than a gap threshold T_{gap}, which means that $\tilde{A}_{\hat{k}}$ is distinctly larger than $\tilde{A}_{\hat{k}-1}$, and $\tilde{A}_{\hat{k}}$ is determined as the threshold. For the set $\{A_i\}$, if A_i is larger than $\tilde{A}_{\hat{k}}$, the user U_i is determined as malicious. Empirically the gap threshold T_{gap} is set as $\log M$.

3.5 Semantic Representation

The detection result for an abnormal user is represented with the anomaly score A as well as the malicious word sets MW_1 and MW_2. The two malicious word sets are derived from W_1 and W_2, where $MW_1 = \{(w_{1i} : a_{1i}) | a_{1i} \geq a_{1i+1}\}$ and MW_2 is the same. Based on the semantic representation, the overview of a malicious user's web activities can be directly identified without the need for reviewing the raw traffic logs.

4 Evaluation and Results

We use a labeled dataset D_L to evaluate our approach and an unknown dataset D_U to analysis its detection ability in the wild. All of them are web traffic logs collected with the open-source network monitor Bro [16]. For simplicity, a user is identified with the source IP address of a request. For the proxy traffic, the original IP is extracted from the HTTP header `X-Forwarded-For`.

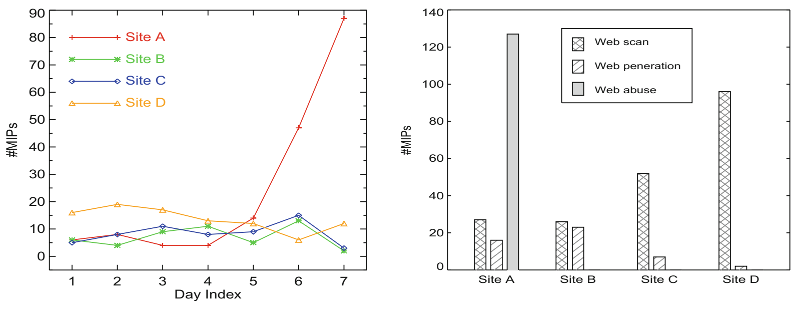
4.1 Dataset

The labeled dataset D_L involves four different websites, which are named from Site A to Site D. We respectively collected traffic logs from the four sites in different seven days, and labeled them manually with several free web security log analyzers as auxiliary tools, including Apache-scalp [17] and 360 xingtu [18]. Among total 43,504 IPs, we found 376 malicious IPs, and more than two-thirds of them are not detected by our auxiliary tools. The summary of D_L is shown in Table 1. The average ratio of malicious IPs to the total for each day is listed in the last column. For all malicious IPs, the number of web requests in their malicious traffic ranges from one to more than ten thousands.

Table 1. Summary of the labeled dataset D_L

No.	Website	Site type	Date range	#Requests	#IPs	#MIPs	Ratio
1	Site A	Aspx	Apr. 01–07	27,544	998	170	13.2%
2	Site B	Php	May 17–23	32,272	2,479	49	2.1%
3	Site C	Java	May 17–23	115,440	7,759	59	0.8%
4	Site D	Html	May 17–23	386,725	32,268	98	0.3%

The details of malicious IPs of four sites are illustrated in Fig. 4. Figure 4(a) shows the occurrence of malicious IPs in each day, and Fig. 4(b) presents the number of malicious IPs in three types of malice. Significantly, the number of malicious IPs for Site A greatly increases in the last two days, while the situations of other three sites are almost stable. At the same time, web abuse only occurs in the traffic of Site A, and for other sites web scan is relatively in the majority, which is in line with expectations. It is owing to that Site A was compromised in the fifth day by attackers through exploiting the PUT method vulnerability, and uploaded several web shells, such as



(a) Malicious IPs in each day (b) Malicious IPs in different malice

Fig. 4. Malicious IPs in D_L

/xwvkk11544.txt, /byHmei7.txt and /miao.xx.txt. Subsequently, some malicious pages were added to the site, which lead to a large number of visits for malicious URLs (e.g., /guiling/hotel/goto.aspx?read_LdLdnApFWm.html). The great majority of the abusing traffic are from web robots of known search engines. Additionally, the most popular attack in web penetration traffic for Sites B and C is SQL injection.

The unknown dataset D_U consists of 136 million web traffic logs involving roughly 3,000 fully qualified domain names (FQDNs) from Jul. 22, 2016 to Aug. 2, 2016, which were collected from a web hosting service provider. The distinct IPs totally count for more than one millions. Most of websites in D_U are portal sites generated by a commercial Content Management System (CMS). Consequently, the most visited web resources are static web pages with .html as postfixes of resource identifiers.

4.2 Evaluation

For D_L, we use two metrics **Precision (P)** and **Recall (R)** to measure the effectiveness of our methodology. Given the numbers of True Positive (TP) and False Positive (FP), the precision is calculated as: $P = TP/(TP + FP)$, and the recall is $R = TP/(TP + FN)$.

Many existing works use True Positive Rate (TPR) and False Positive Rate (FPR) as metrics to evaluate an anomaly detection method. However, in our labeled dataset D_L, the negative samples of four web site vary greatly, which may cause too much deviation in FPR.

With the detection window as one day and T_{fw} for the normal word dictionary as 45%, the dataset actually is separated as 28 groups. The detection results for each group are listed in Table 2. In the results, the precision of 12 groups are up to 100%, and 18 groups achieve 100% recall. Overall, the average precision achieved by our approach is 90.7%, and the recall is averagely 92.9%.

It is worthy of attention that the recall is only 15.1% for the seventh day of Site A. The reason is that among 264 IPs of Site A in Day 7 there are 78 IPs who queried uploaded malicious pages, which makes the normal word dictionary

Table 2. The precision and recall on four sites

	Site A (**P**/**R**)	Site B (**P**/**R**)	Site C (**P**/**R**)	Site D (**P**/**R**)
Day 1	1.000/0.833	1.000/1.000	1.000/0.800	0.947/1.000
Day 2	0.800/1.000	1.000/1.000	0.889/0.889	0.864/1.000
Day 3	1.000/1.000	0.889/0.889	0.750/0.900	0.850/1.000
Day 4	1.000/1.000	1.000/1.000	0.778/0.875	0.722/1.000
Day 5	1.000/1.000	1.000/1.000	0.900/1.000	0.857/1.000
Day 6	1.000/0.894	1.000/0.909	0.833/1.000	0.875/0.875
Day 7	0.925/0.151	1.000/1.000	0.750/1.000	0.750/1.000
Avg	0.961/0.840	0.984/0.971	0.843/0.923	0.838/0.982

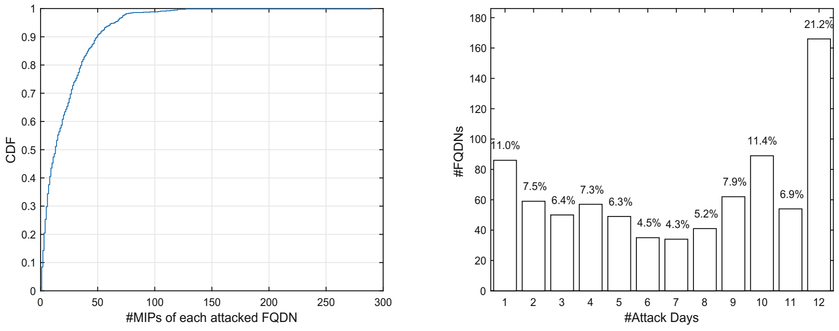
polluted. Actually, our approach is mainly used to provide a direct and effective way to analyze web traffic afterwards. For analysts, it is an obvious indicator of a possible web compromise that the number of malicious IPs in Day 6 increased roughly five times than the average of the previous four days. In such situation, the normal word dictionaries generated in the following detection windows are not trustable anymore, which should be replaced by dictionaries in previous detection windows. Here, we replace the normal word dictionary of Day 7 with Day 1, and the recall rises to 96.5% with 100% precision.

4.3 Results in the Wild

For D_U, we set the detection window as one day as well. In order to reduce false positives as much as possible, we set T_{fw} as 30% and filter out the FQDNs whose number of visiting IPs in a detection window is less than 20. As a result, there are altogether 1,413 FQDNs and 969,731 distinct IP addresses left.

We totally find 3,995 unique malicious IPs involving 782 attacked FQDNs. In Fig. 5(a) it is presented that for almost 90% of attacked FQDNs, there are no more than 50 distinct malicious IPs. However, there are about two in five FQDNs attacked in more than 10 days. It is indicated that for websites in D_U, web attacks frequently occur but not burst.

The top six malicious IPs sorted by cumulative anomaly scores are presented in Table 3. In the last column, we list the top abnormal words of each user. From the words, the intention of each attacker can be directly identified. The first two IPs attacked four different sites in different days, while the words show that they utilized the same tool to carry out targeted web scans. The next three IPs attacked more than two hundreds FQDNs respectively, and the difference is that *.*.40.135 and *.*.153.20 carried out web scan persistently while *.*.154.104 only used one day. Different from the first five malicious web scanners, the last one is an attacker who intended to discover vulnerabilities of the target website with an automatic web penetration tool, since the total number of request sent from the IP is more than 1 millions. From its typical words, it is obvious that



(a) CDF of MIPs for attacked FQDNs (b) Days for attacked FQDNs

Fig. 5. Detection results in D_U

the malicious IP at least conducted large amounts of SQL injection attacks. Due to the site only containing static web resources, the injection strings were added into the resource identifiers of requests and consequently occurs in MW_1.

Table 3. Top 6 malicious IPs in D_U

IP	#F	#D	#R	Top abnormal words
..195.144	3	1	513,903	MW_1: `head spacecp_index.asp indignation aspxspyX.jsp` MW_2: `etc cdir id c+dir type action rif connector file`
..220.206	3	3	548,520	MW_1: `head spacecp_index.asp indignation aspxspyX.jsp` MW_2: `c+dir type action rif etc cdir .htr action type`
..40.135	397	7	123,957	MW_1: `post a.php newsletters login-wall-ypasv fuck.php` MW_2: `aid eval($_post[expdoor]); cmd {${eval($_post)}}`
..153.20	271	12	259,688	MW_1: `post plus mytag_js.php zdqd.php manage xianf.asp` MW_2: `aid X`
..154.104	241	1	2,063,842	MW_1: `classes web-inf .idea uc_server database.sql` MW_2: `index.jsp file servletpath contextpath inputfile`
..122.133	1	1	1,171,289	MW_1: `tX%'andX+X-X-X=X+X+X+Xand'cwlf'!='cwlf%_X.html` `if(now()=sysdate(),sleep(X),X)` MW_2: `.jpg_X.html_X.html .jpg.html .jpg_X.html .html`

Statistically, the top 20 most popular malicious words existing in our detected malicious traffic include `get`, `plus`, `index.php`, `admin`, `mytag_js.php`, `convert`, `utility`, `bbs`, `post`, `data`, `asp`, `include`, `install`, `editor`, `config`, `plugins`, `templates`, `templets`, `ckeditor`, `config.inc.php`. It is revealed that most of malicious visits look for typical weaknesses of some `php` and `asp` web applications. For example, `plus` and `mytag_js.php` belong to the request `/plus/mytag_js.php?aid=9090`, which is a web shell link existing in the known CMS application *DedeCMS* in China. In addition, we analyze the top 30 most attacked FQDNs and find that their popular abnormal words are almost the same as the above. However, there is only one exception which consists of a unique word `m.yonjizz.com`. With querying it in a search engine, we find that it is a online video site related to adult. There are more than twenty malicious IPs who visited the FQDN with requests like `/n/M.yonjizz.com/szh/1` and `/l/Www.58porn.com/es/1`. Such malicious request are not discovered from traffic of other FQDNs in D_U, which may be a clue that the site was possibly compromised.

5 Discussion and Conclusions

Understanding cybercrimer's network behavior in the wild is extremely important. In the paper, we introduce a semantic-aware methodology to distinguish malicious web traffic in an unsupervised-learning way.

Compared with Scanner Hunter proposed in [4], our approach does not depend on the mutual similarity between different attackers and can be directly employed on massive raw web traffic logs even that there is only one malicious request. Lampesberger et al. [10] also introduced an unsupervised-learning method to detect malicious requests. Utilizing a statistical representation of bytes between two separators, their approach loses the semantic information of original requests and can only detect requests with obvious changes in the statistical distribution of characters. Our method preserves the semantic information at the maximum extent and is able to distinguish malicious traffic with almost the same character distribution as the normal, like web abuse.

In conclusion, we firstly extend the scope of malicious web traffic by importing web abuse in this work. And then to model dynamic malicious web activities, we propose a generic detection method to identify malicious web users with a modified TF-IDF algorithm. We evaluate our approach on a manually labeled dataset, and the results reveal that it can effectively distinguish various malicious web traffic. Furthermore, as shown in the results derived from the unknown dataset, the semantic representation can help to understand malice directly, which is valuable for improving network defense strategies and identifying web compromise.

Acknowledgments. This paper is supported by the National Key R&D Program of China (2017YFB0801900).

References

1. StopBadware and CommTouch: Compromised Websites: An Owner's Perspective. https://www.stopbadware.org/files/compromised-websites-an-owners-perspective.pdf
2. Alrwais, S., Yuan, K., Alowaisheq, E., Liao, X., Oprea, A., Wang, X., Li, Z.: Catching predators at watering holes: finding and understanding strategically compromised websites. In: Proceedings of the 32nd Annual Conference on Computer Security Applications, pp. 153–166. ACM (2016)

3. Li, F., Ho, G., Kuan, E., Niu, Y., Ballard, L., Thomas, K., Bursztein, E., Paxson, V.: Remedying web hijacking: notification effectiveness and webmaster comprehension. In: Proceedings of the 25th International Conference on World Wide Web, pp. 1009–1019. ACM (2016)
4. Xie, G., Hang, H., Faloutsos, M.: Scanner hunter: understanding http scanning traffic. In: Proceedings of the 9th ACM Symposium on Information, Computer and Communications Security, pp. 27–38. ACM (2014)
5. Kruegel, C., Vigna, G.: Anomaly detection of web-based attacks. In: Proceedings of the 10th ACM Conference on Computer and Communications Security, pp. 251–261. ACM (2003)
6. Valeur, F., Mutz, D., Vigna, G.: A learning-based approach to the detection of SQL attacks. In: Proceedings of the Conference on Detection of Intrusions and Malware and Vulnerability Assessment (DIMVA), pp. 123–140 (2005)
7. Robertson, W., Vigna, G., Kruegel, C., Kemmerer, R.A.: Using generalization and characterization techniques in the anomaly-based detection of web attacks. In: Annual Network and Distributed System Security Symposium (NDSS) (2006)
8. Song, Y., Keromytis, A.D., Stolfo, S.J.: Spectrogram: a mixture-of-Markov-chains model for anomaly detection in web traffic. In: Annual Network and Distributed System Security Symposium (NDSS) (2009)
9. Krueger, T., Gehl, C., Rieck, K., Laskov, P.: TokDoc: a self-healing web application firewall. In: Proceedings of the 2010 ACM Symposium on Applied Computing, pp. 1846–1853. ACM (2010)
10. Lampesberger, H., Winter, P., Zeilinger, M., Hermann, E.: An on-line learning statistical model to detect malicious web requests. In: SecureComm, pp. 19–38 (2011)
11. Zhang, J., Xie, Y., Yu, F., Soukal, D., Lee, W.: Intention and origination: an inside look at large-scale bot queries. In: Annual Network and Distributed System Security Symposium (NDSS) (2013)
12. Canali, D., Balzarotti, D.: Behind the scenes of online attacks: an analysis of exploitation behaviors on the web. In: Annual Network and Distributed System Security Symposium (NDSS) (2013)
13. Starov, O., Dahse, J., Ahmad, S.S., Holz, T., Nikiforakis, N.: No honor among thieves: a large-scale analysis of malicious web shells. In: Proceedings of the 25th International Conference on World Wide Web, pp. 1021–1032. ACM (2016)
14. FireEye. Detecting and Defeating the China Chopper Web Shell. https://www.fireeye.com/content/dam/fireeye-www/global/en/current-threats/pdfs/rpt-china-chopper.pdf
15. Liao, X., Yuan, K., Wang, X., Pei, Z., Yang, H., Chen, J., Duan, H., Du, K., Alowaisheq, E., Alrwais, S., Xing, L., Beyah, R.: Seeking nonsense, looking for trouble: efficient promotional-infection detection through semantic inconsistency search. In: IEEE Symposium on Security and Privacy, pp. 707–723 (2016)
16. Paxson, V.: Bro: a system for detecting network intruders in real-time. In: Proceedings of 7th USENIX Security Symposium (1998)
17. Apache-scalp. https://github.com/nanopony/apache-scalp
18. 360 Xingtu. http://wangzhan.360.com/Activity/xingtu