# Scalable Detection of Promotional Website Defacements in Black Hat SEO Campaigns

Ronghai Yang*,1, Xianbo Wang*,2, Cheng Chi1, Dawei Wang1,
Jiawei He1, Siming Pang1, and Wing Cheong Lau2

1*Sangfor Technologies Inc.*, 2*The Chinese University of Hong Kong*

## Abstract

Miscreants from online underground economies regularly exploit website vulnerabilities and inject fraudulent content into victim web pages to promote illicit goods and services. Scalable detection of such promotional website defacements remains an open problem despite their prevalence in Black Hat Search Engine Optimization (SEO) campaigns. Adversaries often manage to inject content in a stealthy manner by obfuscating the description of illicit products and/or the presence of defacements to make them undetectable. In this paper, we design and implement DMoS - a Defacement Monitoring System which protects websites from promotional defacements at scale. Our design is based on two key observations: Firstly, for effective advertising, the obfuscated jargons of illicit goods or services need to be easily understood by their target customers (*e.g.*, sharing similar shape or pronunciation). Secondly, to promote the underground business, the defacements are crafted to boost search engine ranking of the defaced web pages while trying to stay stealthy from the maintainers and legitimate users of the compromised websites. Leveraging these insights, we first follow the human convention and design a jargon normalization algorithm to map obfuscated jargons to their original forms. We then develop a tag embedding mechanism, which enables DMoS to focus more on those not-so-visually-obvious, yet site-ranking influential HTML tags (*e.g.*, *title*, *meta*). Consequently, DMoS can reliably detect illicit content hidden in compromised web pages. In particular, we have deployed DMoS as a cloud-based monitoring service for a five-month trial run. It has analyzed more than 38 million web pages across 7000+ commercial Chinese websites and found defacements in 11% of these websites. It achieves a recall over 99% with a precision about 89%. While the original design of DMoS focuses on the detection of Chinese promotional defacements, we have extended the system and demonstrated its applicability for English website defacement detection via proof-of-concept experiments.

* First two authors contributed equally to this work.

## 1 Introduction

With the development of online underground economies, compromising high-ranking vulnerable websites and injecting fraudulent content to promote illicit goods/ services, *e.g.*, unlicensed drugs, pornography, counterfeit, and illegal gambling operations, appears to be a lucrative strategy and thriving practice [33, 35]. Since search engines drive the majority of web traffic [26], miscreants often leverage this as a Black Hat Search Engine Optimization (SEO) technique to promote their products. In this paper, we refer to such *profit*-motivated campaigns as *promotional website defacements*, or *defacements* in short. Promotional defacements can inflict significant harm on the compromised websites, causing reputational damage, traffic hijacking, search engine penalty/ blockade, and thus revenue loss. According to the IMF, the US Internet underground economy is estimated to contribute to 5.4% of its GDP [28], for which defacements are one of the major promotional channels. Given the substantial economic incentives, defacements have become increasingly prominent: Between 1998 and 2016, the number of reported incidents per year grew from a few thousand to more than one million [36]. In some countries, there are regulatory requirements for the involved parties, *e.g.* search engines, to detect promotional defacements so as to thwart the spread of illegal goods and services. Unfortunately, promotional defacements remain elusive and difficult to root out despite their prevalence and considerable damages. For example, Google has to update its page ranking algorithm against defacements for more than 500 times per year [40].

To tackle this challenge, we decide to build an easy-to-deploy cloud-based defacement monitoring service. In contrast, end-host-based solutions would require software modification/ installation on customer web-servers, which hinders usability and deployability. Note also that one cannot detect defacements by merely tracking changes with respect to a “gold” (reference) copy of a webpage. This is due to the dynamic nature of modern web pages, especially those with user-generated content like posting and follow-up comments. Meanwhile, promotional defacers have advanced beyond the use of old black hat SEO tricks such as keyword stuffing and link farming [23]. For example, they have introduced **Stealthy Defacement** techniques by carefully selecting a few site-ranking influential regions in a web page and make tiny modifications in these regions only. By doing so, the defacer can keep the illicit content unnoticeable from law-enforcement officials as well as the maintainers/ legitimate users of the compromised website while being indexed and ranked highly by search engines. Worse still, promoters of illicit products have evolved to conspire with consumers searching for such goods to circumvent content-based detection via **Keyword Obfuscation**: Since it is increasingly difficult to advertise or search for illicit products based on their standard names or related keywords, defacers and illicit goods seekers have developed a vocabulary of jargons, *i.e.*, obfuscated keywords of illicit products, to bypass the blockade of search engines.

These new evasion techniques have rendered conventional Natural Language Processing (NLP)-based detection schemes ineffective due to the following reasons: 1) Although human subjects can readily understand the obfuscated and constantly evolving jargons for illicit goods, standard NLP models cannot recognize those jargons and their intent without retraining. For instance, while typical NLP models can tell "MARK SIX" is related to gambling, they cannot infer the context of its obfuscated form "M4RK SIX". 2) The context can be very different and diverse across various content elements (identified by their corresponding HTML tags) within a single web page. For example, a news article within a web page can be interlaced with not-directly-relevant but legitimate advertisements. Without explicit modeling of the different semantics associated with each tag, it is easy for a conventional NLP-based scheme to incorrectly flag legitimate content, *e.g.*, embedded advertisements, as defacements. It is also difficult to pinpoint the small yet illicit snippets introduced by stealthy defacements without understanding the semantics of each tag. As a result, it remains challenging, even for first-tier search engines, to detect imperceptible promotional defacements. A case in point: 36% of popular searches still return at least one malicious link [29].

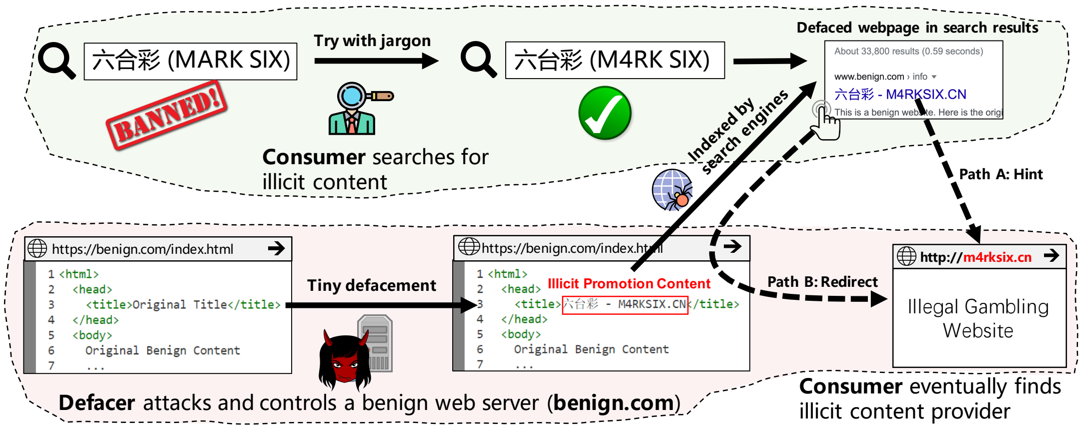
In this paper, we present DMoS, an industrial-strength, scalable monitoring/ detection solution for promotional website defacements. Notice that whatever actions were taken by the defacers, their objective is to promote the illicit content to their target consumers via legitimate search engines. This key observation motivates us to develop a solution based on the following techniques:

- **Tag Embeddings**. To advertise the illicit content, defacers need to optimize stealthy defacements (by modifying a few important tags only) to boost the ranking of the resultant page in search results. Towards this end, we propose a new tag-embedding model, which follows search-engine indexing algorithms to focus on text and tags in the imperceptible yet index-impacting regions of a web page.
- **Jargon Normalization Algorithm (JNA)**. While defacers and their target customers can craft arbitrary jargons in theory, they need to be readily understood by human subjects in practice. Therefore, these jargons often share similar shapes or pronunciation with the original forms. We therefore build the first full-featured jargon normalization algorithm to facilitate NLP models to understand the authentic meaning of such obfuscated jargons.

With these two techniques, we design and implement DMoS to automatically detect promotional defacements in a scalable manner. Unlike proof-of-concept studies, DMoS is production-ready. In fact, DMoS has already been deployed as a security monitoring service by a key Chinese cybersecurity service provider for its enterprise customers. During the initial months of the deployment of DMoS, it has served to monitor 7000+ commercial websites by analyzing more than 38 million web pages and detected defacements among over 11% of those sites. We have manually verified the results, reported them to the site owners, and received their confirmation. It also enables the uncovering of new types of defacement techniques. In summary, this paper has made the following technical contributions:

- **New Techniques.** We present the first industry-ready tool, DMoS, to enable scalable detection of promotional website defacements. In particular, we have developed a set of techniques to neutralize the new evasion tricks performed by defacers. Our approach of building a tag-embedding model and normalization of obfuscated characters/jargons can effectively and precisely identify defaced web pages. We have conducted large scale testing with DMoS on more than 38 million Chinese web pages and performed proof-of-concept experiments on over 30 thousand English web pages.
- **Measurement Study.** Through collaboration with a key industrial company, we systematically conduct a large-scale measurement study on the security of commercial websites to reveal the characteristics of promotional website defacements and quantify their pervasiveness.
- **New Findings.** We uncover new types of defacement techniques/ preferences that enhance our understanding of the online underground ecosystem. By effectively catching defacements at their early stage, we substantially raise the bar to thwart large-scale promotional defacement campaigns.
- **New Dataset.** We present a semi-automatic approach to collect defacement datasets and release an English defacement dataset [17] to the research community.

**Roadmap.** The remainder of this paper is organized as follows: Section 2 provides the background of our work. Section 3 elaborates the design of DMoS. Section 4 evaluates the performance of DMoS. Section 5 presents the measurement studies and reports new findings. Section 6 discusses the lim-


六合彩 (MARK SIX)
BANNED!
Try with jargon
六台彩 (M4RK SIX)
Consumer searches for illicit content
Defaced webpage in search results
About 33,800 results (0.59 seconds)
www.benign.com › info
六台彩 - M4RKSIX.CN
This is a benign website. Here is the origi
Indexed by search engines
Path A: Hint
Path B: Redirect
https://benign.com/index.html
Tiny defacement
Illicit Promotion Content
http://m4rksix.cn
Illegal Gambling Website
Defacer attacks and controls a benign web server (benign.com)
Consumer eventually finds illicit content provider

```
<html>
  <head>
    <title>Original Title</title>
  </head>
  <body>
    Original Benign Content
    ...
```

```
<html>
  <head>
    <title>六台彩 - M4RKSIX.CN</title>
  </head>
  <body>
    Original Benign Content
    ...
```

Figure 1: Conspiring Acts of a Defacer and an Illicit Content Seeker via Promotional Website Defacements

itations and further research directions. Section 7 describes the related work and Section 8 concludes this paper.

## 2 Background

In this section, we first explain why defacements are widely employed for black hat SEO. We then introduce existing countermeasures taken by search engines against malicious content and black hat SEO. We also show how seekers of illicit goods use jargons as search keywords. We then describe the overall process of promotional website defacements and illustrate the conspiring acts between the defacers and their target consumers. Finally, we discuss other conventional defacement/ black hat SEO techniques.

### 2.1 Defacements for Black Hat SEO

Search Engine Optimization (SEO) is the process of making a website more visible to its target audience by boosting its ranking among the relevant results returned by mainstream search engines. It is expensive and slow to increase the search-engine-visibility of a website through legitimate, "organic" content enhancement. An illegal short-cut is to hack into popular websites and plant promotional content and referring links for the illicit pages. Due to its cost-effectiveness, promotional website defacements have been widely adopted as a means of black hat SEO.

Defacers can take various approaches to hack into target websites, including but not limited to SQL injection, remote command execution (RCE), and social engineering. They often use recently disclosed vulnerabilities (0-day or N-day) for batch scanning and exploitation. Sometimes, attackers only crack the login credentials of the content management system (CMS) to escalate their privilege to edit articles. Worse still, defacers may gain command execution access to the hacked web server and have more flexible or full control of the website. In the latter case, it is possible to evade detection even when a defacement monitoring system is running on the webserver.

### 2.2 Search Engine Policies

Although the primary goal of search engines is to make web pages universally accessible, they need to cleanse the search results to comply with legal/ regulatory requirements as well as to improve user experience. Certain types of content are commonly censored by most search engines. These include sexually explicit materials, pirated contents, solicitation of illegal services, as well as sensitive personal information [27]. Search results belonging to these categories are either removed upon valid requests or pruned automatically [30].

It is a common practice for search engines to sanitize their results by critically reviewing any web pages containing blocked keywords. As such, defacers often use obfuscated jargons in their promotional content and expect their target customers to search for their goods/ services using the same jargons. This can be observed by, *e.g.*, comparing the search results of MARK SIX (*i.e.*六合彩) and M4RK SIX (*i.e.*六台彩) returned by Microsoft Bing. The first term is a blocked gambling keyword, and the returned search results are all normal websites. In contrast, the second term is an obfuscated jargon with a similar shape. In this case, Bing returns a long list of illegal gambling websites.

### 2.3 Active Consumers Search for Jargons

Active consumers are users who actively look for the illicit products they demand. For underground markets, active consumers are prevalent due to the difficulty of promoting products via legal channels and scarce supply. Most people find content on the Internet through search engines with relevant keywords. However, illicit content is not trivially searchable since search engines usually block related keywords. Active consumers, who are aware of blocking, have evolved to search with obfuscated keywords to replace the blocked ones. There are also cases where consumers need to transform the keyword in various forms by trial and error until they find the desired content. During this process, new obfuscated keywords, *i.e.*, jargons, are coined. Since the rules that consumers use to obfuscate keywords should be natural and straightforward to a human subject, most coined jargons can be categorized into two types: One is homophonic jargon, which replaces the original word/ character with another of similar pronunciation. Another is homomorphic jargon, which uses a word/ character with a similar shape. We discuss these two types of jargons in detail in Section 3.2.2.

## 2.4 An Example of a Defacement-Driven Promotional Campaign

Figure 1 shows the walk-through of a defacement-driven promotional campaign for illicit goods/ services. It helps readers understand both actors' motivation and rationale: the illicit-goods-seeking consumer and the defacer who promotes/ offers the goods. The upper part of Figure 1 shows the typical behavior of a consumer. As mentioned in Section 2.2, the original keywords related to illicit content are likely to be blocked by search engines. The consumer's natural reaction is to try other keywords that are not blocked, namely, the so-called jargons which are obfuscated words/ terms with similar shape or pronunciation.

Expecting such behavior from their consumers, defacers would use those jargons for black hat SEO. As described in Section 2.1, defacers hack into popular websites and perform tiny modification to insert their promotional content, hoping that the stealthy defacement can evade detection and indexed by search engines. When consumers search the jargons and find the defaced website among the search results, there are usually two ways for the consumer to reach the soliciting website. Firstly, marked as "Path A" in the figure, the consumer can infer the illicit content information based on the hints left by the defacer, *e.g.*, website URL or contact information. Secondly, the consumer can click on the search results and visit the defaced web page, as illustrated by "Path B".

## 2.5 Other Defacement/ Black Hat SEO Techniques

Below we discuss other existing defacement/ black hat SEO techniques:

- **Keyword stuffing.** The number of keywords contained in a web page is important for the page rank score. Defacers can thereby include a spectrum of trending keywords to associate the illicit website with many hot words.
- **Link farm [23].** To accumulate a large number of incoming links pointed to the illicit websites, defacers can compromise high page-ranking sites to inject referring hyperlinks.
- **Cloaking [43].** To evade detection, cloaking techniques are widely used to serve different content to different visitors or search engines dynamically. User-Agent, Referer header, IP address, *etc.*, are inspected to present the illicit content only to the target customers.

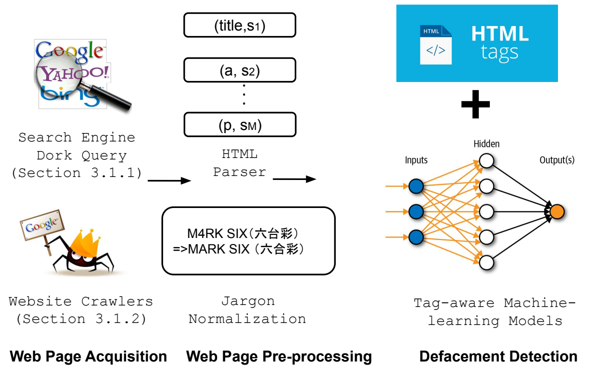


Figure 2: System Overview of DMoS

Since illicit solicitations are being vigorously censored by search engines, defacers have been forced to evolve their techniques. For example, there is a phase-out of old techniques like keyword stuffing as it involves extensive modifications of the web page and thus can be easily detected by search engines. In contrast, modern defacers mostly perform stealthy defacements and use obfuscated jargons, which tend to be more elusive yet effective. Nevertheless, whatever techniques defacers use, they are limited to promote products to their customers via search engines. Based on this observation, DMoS can detect defacements generally and robustly.

## 3 Design of DMoS

Figure 2 depicts the overall design of DMoS. At a high level, the workflow of DMoS consists of 3 phases. In the Web Page Acquisition phase, we apply the search engine dork query [15], which uses advanced search operators (*e.g.*, *site:*, *inurl:*, *etc.*), to find suspicious web pages under a given domain. We then pre-process and normalize the obfuscated jargons in the Web Page Pre-processing phase so that we can leverage NLP techniques to detect defacements in the third phase. Due to the target market of our collaborating industrial partner, we focus on defacement detection for Chinese websites in this section. Note however that, our approach is general: in Section 4.4, we will demonstrate how DMoS can be extended to detect defacements of web pages in other languages.

### 3.1 Web Page Acquisition for Training

It is a complex, labor-intensive process to collect and label a large dataset of web pages to cover a wide range of defacement techniques. Below, we first discuss how to collect representative defaced and legitimate web pages. We then present our efficient data annotating techniques for training DMoS.

### 3.1.1 Collecting Defaced Dataset

To avoid detection by the authorities, defacers have evolved their techniques, making it more challenging to collect defaced web pages. We note that defacers' primary goal is to make illicit products rank high in relevant search results so that they can be readily accessible. Based on this observation, we leverage the web-crawling capabilities of search engines and use advanced search engine queries to efficiently retrieve the most suspicious web pages. Specifically, we search a list of illicit keywords (like "MARK 6", "M4RK SIX") on search engines. Since these illicit keywords[1] are often meaningless and irrelevant to outsiders due to their obfuscation (*e.g.*, homophonic/ homomorphic jargons in Section 3.2.2), it is therefore reasonable to believe that the pages indexed under jargons are more likely to be defaced web pages. We then use Selenium [11], a web browser automation tool, to follow the links in the search results and crawl the corresponding web pages.

**Jargon-mining algorithm.** The next issue is to collect a large number of illicit keywords to cover different defacement scenarios. Sole reliance on manual keyword crafting is not viable due to its labor-intensive nature and limited/ uncertain coverage. To overcome this challenge, we leverage the "related search" function offered by all major search engines [35, 45] to expand our illicit keyword list based on a small set of seed jargons. The procedure is detailed as follows:

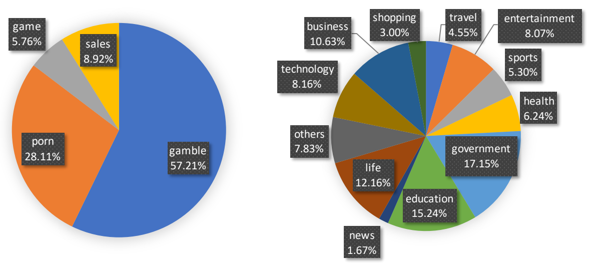
- We first manually collect a small list of seed jargons through manual crafting or keyword extraction from illicit websites (*e.g.*, adult-content website). We also extract jargons from cybercrime marketplaces (*e.g.*, Silk Road, a breeding ground for illegal drug business [47])
- We input seed jargons as search terms on mainstream search engines such as Google, Baidu, 360 Search, *etc.*
- By collecting the "related search" keywords, we form a candidate list of keywords that are likely to be illicit.
- We then manually review the keywords in the candidate list to filter out those not related to defacements.

While the above process may still miss some illicit keywords, thanks to the built-in generalization power of its Jargon Normalization module, DMoS can still achieve high detection performance in our large-scale empirical studies. Refer to Section 6.1 for more detailed discussions on this issue.

### 3.1.2 Collecting Legitimate Dataset

Collecting legitimate data, *e.g.* legitimate web pages that have not been compromised/ defaced, is relatively straightforward: We first crawl those top-ranked websites across multiple categories in the China Webmaster list [8]. Since these websites are expected to be less susceptible, we then filter and label these web pages using a light-weight scheme to be described in Section 3.1.3.

[1]For the ease of presentation, we use the terms illicit keywords and jargons interchangeably.



(a) Defaced Dataset Categories (b) Legitimate Dataset Categories

Figure 3: Topic Distribution in Defaced/Legitimate Web Pages

Upon further examination, we observe that web pages from the same website are more similar to each other. If we directly feed all the crawled web pages as training data for our machine-learning model, it will lead to overfitting, which in turn will result in poor identification/ detection performance for never-before-seen data. To ensure the diversity within the dataset, we use ssdeep [4], a fuzzy hashing algorithm, to compute the fuzzy hash value. The hash distance is then used to measure web pages' similarity, and only the Top-1000 dissimilar web pages per website are used for training.

### 3.1.3 Data Annotation

The training of DMoS requires labeled data, which is very time consuming and resource-intensive.

**Label defaced dataset.** We manually examine and label the suspicious pages collected from search engines. If, in the collected dataset, there is only one suspicious page under a domain, we manually examine the page by looking for illicit keywords and checking their context. If there are multiple suspicious pages under a domain, we randomly sample 10% pages under this domain for manual verification. We observe that defacements are often conducted in batch for all web pages from a website with similar jargons. Therefore, if all the sampled web pages are defaced, we assume all the collected web pages under the same domain are defaced. Otherwise, we manually check the remaining suspicious pages under this domain. For defaced web pages, we also label the categories of the promoted illicit products.

**Label legitimate dataset.** While our legitimate dataset was crawled from first-tier websites, we were surprised to find defacements in some of the web pages. For example, we found 163.com, ranked Top-140 worldwide by Alexa, was injected with gambling materials in March 2019. To ensure the correctness of labels, we need to design a light-weight scheme to remove any defaced web pages from the legitimate dataset. Specifically, we first develop a list of alarming keywords, which are broad enough so that any defaced web page should contain at least one alarming keyword with a very high probability. In other words, it is difficult for defacers to craft a defaced web page without any alarming keywords. Then, we use the Aho-Corasick string-searching algorithm [6] to search for alarming keywords among web pages. By admitting the vast majority (95%) of the legitimate web pages with this pre-screening step, we can afford to check the remaining ones manually to ensure they have not been compromised. To generate the list of alarming keywords, we compare the frequency difference of a keyword in the legitimate dataset and defaced dataset, respectively. More details can be found in Appendix A.

#### 3.1.4 Details of Our Final Dataset

Based on 439 seed jargons, we collected 216,708 illicit keywords, where keywords related to gambling, adult content (porn), and sales comprise the majority of the data. Given these illicit keywords, we collected 294,393 suspicious web pages from 4,931 websites in total. After verifying and labeling the web pages as discussed in Section 3.1.3, we filter out web pages of incorrect labels to yield 147,754 of defaced web pages across 2,103 websites. The dataset covers common defacement categories, whose proportion is shown in Figure 3a. Note that the category of sales includes different illicit goods and services such as fake credentials/ certificates, counterfeits, surrogacy, and unlicensed drugs, *etc.*

For the legitimate dataset, we collected 389,438 web pages from 5,121 websites in total, covering most website categories (*e.g.*, government, sports, health, *etc.*). Their statistics are depicted in Figure 3b. We randomly split 70% of the legitimate and defaced datasets for training and use the remaining 30% for offline testing (see Section 4.1).

## 3.2 Web Page Preprocessing

As mentioned earlier, modern defacement campaigns often *obfuscate* jargons (also called "illicit keywords” in this paper) using similar shape or pronunciation to bypass detection. Existing NLP techniques cannot adequately learn or understand these jargons for two reasons. Firstly, the vocabulary of jargons is constantly evolving and differs significantly from the standard vocabulary. Secondly, many illicit keywords are created in ungrammatical and obfuscated forms (*e.g.*, letter "l" can be replaced by digit "1") while the NLP techniques are geared towards the properly-written text. To leverage the technical advances of NLP for detecting defaced web pages, we need to recover transformed jargons to their base forms. Below, we first introduce how to parse HTML and then discuss how to identify and recover these jargons.

#### 3.2.1 Parsing HTML into Tag-Sentence Pairs

Web pages are parsed to extract tag-sentence pairs. In particular, there exist special HTML tags which can contain text not only in the body but also in specific attributes, *e.g.*, *keywords* of *meta* tag and *alt* of *image* tag, *etc.* One example snippet is presented in Listing 1, for which the parsed result is [(‘a.title’, ‘MARK SIX’), (‘a’,‘benign content’)].

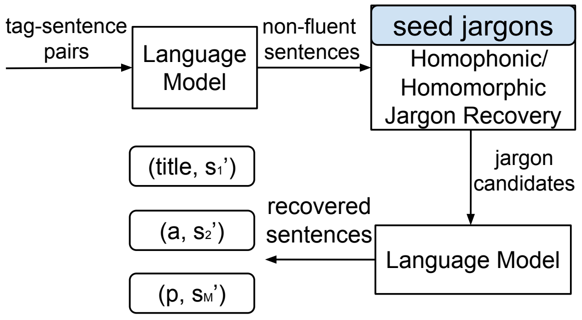


Figure 4: The Procedure for Jargon Normalization

Listing 1: An Example of Special HTML Tags

```
<a href="http://benign.web.com"
  title="MARK␣SIX"> benign content </a>
```

For a formal definition, suppose *page* is a web page, its parsed result is $page = [(tag_1, s_1), (tag_2, s_2), \cdots, (tag_M, s_M)]$, where $M$ indicates the maximum length of *page*. Let $N$ be the maximum length of a sentence $s_i$. When the length of a sentence exceeds this parameter $N$, we split this sentence into sub-sentences, and add these tag-sub-sentence pairs to *page*.

#### 3.2.2 Jargon Normalization

Recovering obfuscated jargons is of great importance for machine-learning models to understand the semantics of the jargons and thus can precisely recognize defaced web pages. Since the obfuscated jargons should be easily understood by human subjects, these jargons often share similar pronunciations (*i.e.*, homophonic jargons) or similar shapes (*i.e.*, homomorphic jargons) with their original forms.

- **Homophonic jargon.** There is a system called Pinyin to notate the pronunciation of Chinese characters. The Pinyin-to-character mapping is a one-to-N mapping. Therefore, homophonic characters are common and have been widely abused by defacers. For example, instead of directly using MARK SIX (*i.e.*, 六合彩), defacers may use MARC SIX (*i.e.*, 六和彩), a transformed jargon with the same Pinyin “LiuHeCai".
- **Homomorphic jargon.** Some Chinese characters have similar shapes. These homomorphic characters are also commonly used to obfuscate jargons. For example, defacers can use M4RK SIX (*i.e.*, 六台彩), as shown in Figure 1, to replace the blocked keyword of MARK SIX (*i.e.*, 六合彩) based on their similar appearance.

To identify the obfuscated keywords, we develop the following jargon normalization procedure, as illustrated in Figure 4:

1. Language Modeling (LM) is a classical task for assigning probabilities to sentences drawn from a corpus. It

is often used to measure the smoothness/ idiomaticness of sentences. Since sentences with obfuscated jargons should not be smooth, we can use LM to filter out smooth sentences, which should not contain any jargons. For efficiency considerations, we adopt the unigram LM [10] approach. We then build the model using a corpus consisting of the Chinese wiki [9], the legitimate dataset we collected (see Section 3.1.2), and illicit web pages[2], but not the defaced victim pages.

2. We use unigram LM to assign a probability for each sentence. High-probability sentences are believed to be similar to the corpus. In contrast, the minority of low-probability sentences are assumed to be non-fluent and thus can contain obfuscated jargon candidates.
3. We identify the homophonic/ homomorphic jargons from these non-fluent sentences and transform them back to their base forms (*i.e.*, jargon candidates), on which we expand later.
4. For jargon candidates, we use LM to measure the fluentness again. Only high-probability sentences are used and recovered, while the others are abandoned since they have never been seen in our large corpus.

**Homophonic Jargon Recovery.** To identify and recover homophonic jargons from non-fluent sentences, we take the following steps:

- We collect a list of seed jargons (*e.g.*, MARK SIX). As described in Section 3.1.1, this is a relatively straightforward process of extracting keywords from illicit websites, *etc.*
- We use a sliding window to traverse non-fluent sentences and convert Chinese characters within the window to their Pinyin representations.
- We then calculate the editing distance (*e.g.*, Levenshtein distance [49]) of Pinyin between the characters in the window and seed jargons.
- If the distance is under a threshold, it means the characters in the window and the seed jargons are similar. Therefore, we replace the characters with the matched seed jargon. The latter is treated as jargon candidates.

**Homomorphic Jargon Recovery.** We need an encoding system, similar to Pinyin, but can encode Chinese characters according to their shapes instead of pronunciations. Fortunately, we find the Four-Corner System [1] that can suit this requirement. It can be seen as a perceptual hashing algorithm that can embed the shape information of Chinese characters. With this encoding, the distance between homomorphic jargons should be small.

The Jargon Normalization Algorithm (JNA) has achieved an outstanding accuracy. We manually check the results of the offline testing data. JNA identifies 50,855 obfuscated jargons, out of which only 5 are false positives. More details are given in Section 5.2. Due to the high cost of manual false negative checking, we can only afford to sample around 50 defaced web pages without detected obfuscated jargon. When checking all text content of these sampled pages, we could not find any undetected obfuscated jargon. However, limited by the sample size, our estimated recall and false negative rates are expected be lower than the actual rates. By identifying and recovering obfuscated jargons, DMoS can increase its precision from 91.29% to 94.89%, which can reduce manual efforts on processing false alarms by over 41%. More details on the effectiveness of JNA are discussed in Section 4.1.3.

[2]To crawl illicit web pages, we can visit illegal websites directly or follow the links of defaced pages. They should not contain obfuscated jargons.

## 3.3 Detection of Web Page Defacements

Stealthy defacements have been widely used in modern defacement campaigns. On the one hand, injecting tiny illicit content can make the defacement unnoticeable for detection. On the other hand, with minor modifications of the site-ranking influential HTML tags (*e.g.*, *title*, *meta*, *etc.*), defacers can effectively advertise illicit content via search engines, which drive the majority of web traffic today. Below, we first discuss the limitation of existing approaches. We then illustrate how to encode the tag information on top of different neural networks to better identify stealthy defacements.

### 3.3.1 Limitation of Existing Solutions

The following plausible approaches turn out to be ineffective:

- **Classical text classification.** Stealthy defacements only introduce small illicit snippets. However, normal content can be very long and embedded with confusing content. Without context and hierarchy awareness, it is difficult for classical text classification models like SVM or XGBoost to pay attention to suspicious regions to accurately capture the defacement.
- **Reference (Gold) copy**. Modern web pages contain many dynamically generated or even user-generated content. Tracking changes with respect to a reference ("gold") copy of the web page is not as easy as for static web pages. To get reasonable results, the service needs to be deployed on customers' servers. However, such a requirement goes against our goal of providing a cloud-based service that does not require any software modification or installation on customer's machines, which gives lower cost and better usability.
- **Checking a few important tags only**. Although stealthy defacements typically occur in a few tags, defacements generally can appear in other tags (Section 5.1 shows 44 different tags). Furthermore, there are intricate relationships among the tags. Checking separate tags can result in the loss of essential hints. A straightforward example is that the context of the text in <*title*> and <*meta*> should be consistent. Otherwise, it may indicate defacements. This necessitates considerable efforts to model the properties and relationship of tags.

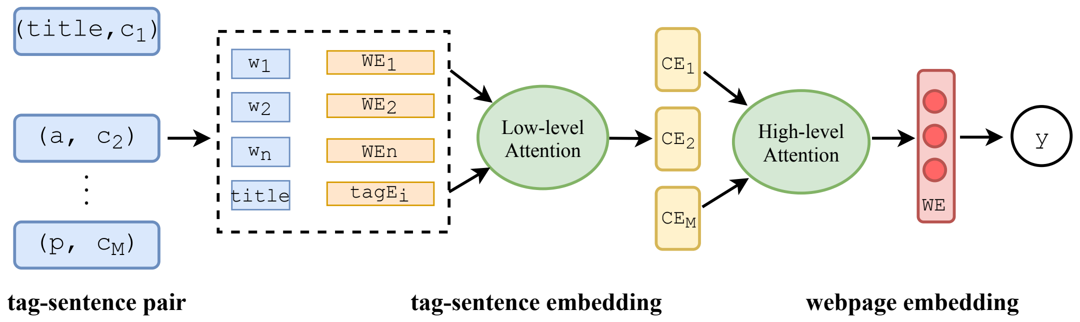


Figure 5: Architecture of the THAN Model

### 3.3.2 Key Insights

It is challenging to pinpoint the stealthy defacement. Enterprise websites are rather diverse. They integrate code and resources from dozens of third-party service providers, ranging from personalized ads, marketing tags, CDNs, to third-party JavaScript libraries and many others. Legitimate web pages can therefore contain seemingly illicit "noises" including legitimate advertisements which promote legitimate products (*e.g.*, drugs, luxury, red wine, *etc.*), ambiguous yet legitimate website content (*e.g.*, love novels and movies which may have on-the-verge description of adult content, *etc.*), and others. In other words, stealthy defacements sometimes do not exhibit any differences from these noises.

Fortunately, due to their different purposes, the "noisy" but legitimate content and the illicit/ fraudulent ones (planted by defacers) should appear in different regions of a web page. For defacements, illicit snippets are typically injected into those more informative HTML tags so that the promoted products can appear not only in popular search results, but also look innocent from the perspective of legal authorities. In contrast, the noises of legitimate web pages are carefully designed for *better human viewing*, for example, 1) legitimate ads often appear in the banner and should not occur in *title*, *meta* tags; 2) ambiguous website content (*e.g.*, love novels, legitimate lottery websites, *etc.*) often appears in the main body part. In summary, *promotional defacements essentially show distinctive location and tag preferences, compared to the noisy legitimate content.*

According to this observation, promotional defacement identification requires a model to follow the page-ranking algorithms of search engines to focus more on important tags while avoiding the noises induced by normal regions that may cause the web page to be misclassified. Towards this end, we introduce a tag embedding scheme, which can be used on top of different state-of-the-art neural networks, including Hierarchical Attention Network (HAN) [46], BERT [24], and others, to automatically learn the information and importance of HTML tags. As shown in Table 2, such tag information can significantly improve the performance of these state-of-the-art networks in detecting defacements.

### 3.3.3 Design of Tag-Aware HAN Model

We design the so-called Tag-aware Hierarchical Attention Network (THAN for short) by introducing the tag embeddings on top of the HAN [46] network. Figure 5 depicts the network architecture of THAN. Following the design of HAN, THAN also includes a two-layer attention mechanism (one at the word layer and another at the sentence layer), which enables the model to adjust the attention paid to individual words and sentences. However, unlike the sentence embedding scheme in HAN, we develop the tag-sentence embedding in the second layer instead. This is to better capture the differences in tag-preference between stealthy defacements and noises.

As introduced in Section 3.2.1, let $(tag_i, s_i)$ be the $i$-th tag-sentence pair of *page*. We first split $s_i$ to a word sequence $[w_1, w_2, \cdots, w_n]$. Using the word2vec model [37, 38], we can learn the word embeddings [3] from the Chinese wiki corpus [9] and the dataset collected in Section 3.1. Each word $w_t$ is mapped to a $D$-dimensional embedding vector $we_t$. Since not all words contribute equally to a sentence, we follow HAN [46], in the first layer, to give more weight to more informative words (*e.g.*, jargons). We then aggregate these weights to form sentence embeddings $SE_i$ as follows:

$$u_{it} = tanh(W_w GRU(h_{t-1}, we_t) + b_w) \tag{1}$$

$$\alpha_{it} = \frac{\exp(\langle u_{it}, u_w \rangle)}{\sum_t \exp(\langle u_{it}, u_w \rangle)} \tag{2}$$

$$SE_i = \sum_t \alpha_{it} h_{it} \tag{3}$$

where $W_w$, $b_w$, $u_w$ are trainable parameters and $\alpha_{it}$ indicates the attention weight of the $t$-th word for sentence $s_i$.

**Tag-Sentence Embeddings.** To learn the importance of each tag, we map $tag_i$ to a trainable vector $tagE_i \in R^d$, *i.e.*, the tag embedding. We then concatenate the tag embedding with the weighted sentence vector to yield the embedding of the tag-sentence pair $(tag_i, s_i)$ as follows:

$$TSE_i = [tagE_i; SE_i] \tag{4}$$

[3]For Chinese application, we actually learn the embeddings for each character, which can generalize better than word embeddings. For ease of presentation, we use word embedding to represent character embedding, if not specified otherwise.

Compared to sentence embeddings, such tag-sentence embeddings ($TSE_i$) can reflect the differences in tag and location preferences between stealthy defacements and noises which promote similar products. To highlight those more suspicious tag-sentence pairs, we use the second layer attention network to automatically assign more weight for the defaced content when constructing the web page embedding $WE_i$. The second attention layer shares the same structure as the first one. Finally, we feed the web page embedding to a fully connected network to predict whether a web page has been defaced or not.

$$y = sigmoid[(W \times WE_i) + b] \tag{5}$$

**Training THAN model.** The THAN model is trained by TensorFlow [13]. In our experiments, each web page is split into tag-sentence pairs, out of which we randomly sample 150 tag-sentence pairs as the input of our model. We map words to 256 dimension continuous vectors of real numbers. Note that the dimension of the tag embedding cannot be too high. Otherwise, the neural network will give too much weight to the tag and vice versa. Finally, we set the dimension of the tag embedding to 32, at which point we can obtain a stable model. We use the Adam optimization algorithm for training, with an initial learning rate of 0.001. The dropout rate is set to 0.3. The THAN model is trained for 3 epochs and the batch size is set to 64.

### 3.3.4 Design of Tag-Aware BERT Model

To demonstrate the broad applicability of our method, we integrate our proposed tag embedding to the state-of-the-art NLP model, namely, Bidirectional Encoder Representations from Transformers (BERT) to obtain Tag-aware BERT (T-BERT for short). BERT is based on a multi-layer bidirectional Transformer [41] structure and is the first fine-tuning based representation model that achieves state-of-the-art results for a wide range of NLP tasks. We use the open-source implementation of BERT [19] to conduct experiments. Unlike HAN and THAN, in addition to the token embedding of each word, BERT also utilizes segment embedding and position embedding to enrich the information of the sentence representation. These three kinds of embedding are aggregated to obtain the sentence representation (*i.e.*, $SE_i$). Following the practice in THAN, the tag embedding is concatenated with the sentence vector to obtain the embedding of the tag-sentence pair. Then an attention layer is applied to obtain the representation of the entire web page. The hyperparameters and experiment setting of training T-BERT are consistent with those of THAN.

### 3.3.5 Demystifying the Tag Embeddings

We use THAN as an example to visualize how and why tag embeddings can improve detection performance. Firstly, we would like to identify which tags are believed to be more important by the tag embeddings. Generally, an embedding with a large $L2$ norm can activate a large weight in a neural network. Therefore, we use the $L2$ norm to measure the importance of different tags. Table 1 presents top tag embeddings with larger $L2$ norms of a sample defaced page. We can see that tags which are more likely to be defaced have larger $L2$ norms, *e.g.*, *title*, *meta*, and so on. Refer to Section 5.1 for tag preferences of defacers. Observe that, tags that are often associated with noise (*e.g.*, ads or ambiguous content) are automatically learned to be trivial. This experiment demonstrates that THAN, given the tag embedding, can assign a heavier weight to suspicious tags while avoiding noises.

Table 1: Top-8 Tag Embeddings

| Tag | title | meta | meta. description | meta. content | marquee | a | span | div |
|---|---|---|---|---|---|---|---|---|
| L2 Norm of Tag Embedding | 0.73 | 0.58 | 0.42 | 0.26 | 0.24 | 0.18 | 0.18 | 0.16 |

**Case study of learned features.** To verify the ability of THAN in steering more attention towards informative tag-sentence pairs and words in a web page, we visualize the hierarchical attention layers for one random web page. Note that the features are automatically learned, and no domain knowledge is required. As shown in Figure 6, every line is a tag-sentence pair. The bar represents the weight of the tag-sentence pair, and the colored words denote they are more informative in this sentence. To avoid clutter, we only show the Top-10 most important tag-sentence pairs and only present important words by normalizing their weights. As expected, those informative tags (*e.g.*, *title*, *meta* and *a*) dominate the list. Upon closer examination, it is revealed that illicit keywords contribute more weights for sentence embeddings. This experiment also demonstrates the effectiveness of the THAN model.

## 4 Evaluation of DMoS

We have implemented DMoS in Python using 12,500 lines of code. Unlike other Proof-of-Concept research, DMoS is a full-featured tool and has been deployed as a commercial service in the real world. To determine the effectiveness of DMoS, we evaluate it based on large-scale fresh data in the wild. In this section, we analyze the performance of DMoS on different datasets and compare it against classical machine-learning models as well as commercial products by other leading companies.

### 4.1 Offline Experiments

For the dataset collected in Section 3.1, we randomly split 70% of websites for training and use the remaining 30% for offline testing. Instead of sampling web pages, we split by websites since web pages of different websites are more diverse. This is to stress test the generalization power of DMoS. For this reason, the number of web pages for offline testing

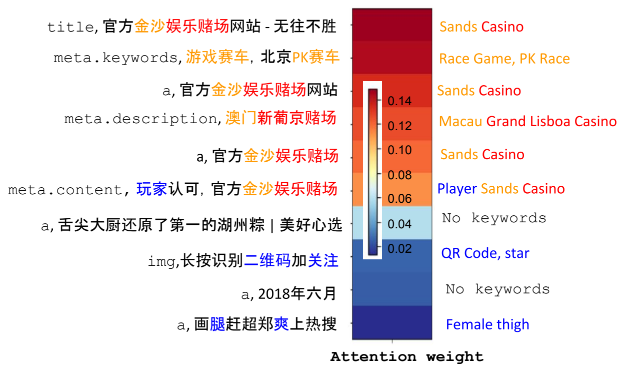


Figure 6: Features Learned by THAN. On the left are tags and the corresponding sentences in Chinese. On the right are the English translation of the keywords highlighted in the same color.

is not exactly 30% of the dataset. The offline testing dataset consists of 20,958 defaced pages and 40,426 legitimate ones.

### 4.1.1 Comparison with Other Detection Schemes

As our baselines, we select the following state-of-the-art solutions in the area of malicious web content detection:

- WAF: Most WAF vendors utilize a keyword-searching approach to detect defacements. We acquired one of such WAF devices from a major Chinese vendor and directly used it for testing without further training.
- Saxe *et al.* [39]: This work divides the documents into multiple sub-regions and then utilizes a deep learning approach to aggregate these sub-regions for representing and classifying web pages. Based on the feature extraction function partially described in the paper, we try our best to re-implement its detection scheme.
- BoW model: The bag-of-words model is well known. We first extract the most important words based on TF-IDF [14] and then use the popular XGBoost model [22] to fit the training data.
- HAN [46]: HAN is designed for structure-free document classification. Here, we reproduce the network to classify the defaced web pages.
- FastText [31]: FastText is a simple and efficient neural network text classification model. We use its open-source library in our evaluation.
- BERT [24]: BERT is a multi-layer bidirectional Transformer [41]. It is the first fine-tuning based representation model that achieves state-of-the-art results for a wide range of NLP tasks. The training process is discussed in Section 3.3.4.

For fair comparisons, all the baseline models (except the black-box WAF) are trained with the same training dataset, and we then fine-tune them for the best performance. As shown in Table 2, DMoS shows superior performance across the board.

Table 2: Classification Performance in Offline Experiments

| Methods | Precision | Recall | F1 |
|---|---|---|---|
| WAF | 93.32% | 9.76% | 17.67% |
| Saxe *et al.* [39] | 82.53% | 85.62% | 84.05% |
| BoW | 86.56% | 91.87% | 89.14% |
| HAN [46] | 86.95% | 93.86% | 90.28% |
| FastText [31] | 81.47% | 88.21% | 84.71% |
| BERT [24] | 93.41% | 97.64% | 95.48% |
| THAN | 91.29% | 96.89% | 94.00% |
| T-BERT | 95.66% | 98.76% | 97.19% |
| DMoS (JNA + THAN) | 94.89% | 97.40% | 96.13% |
| DMoS_v2 (JNA + T-BERT) | **97.53%** | **99.37%** | **98.44%** |

### 4.1.2 Effect of Tag Information

To evaluate the effectiveness of the HTML tag information, we first compare THAN with HAN. The former introduces the information of HTML tags and automatically learns the weight of different tags. Table 2 shows that the use of tag-derived information by THAN results in a considerable improvement of 4.34% in precision and 3.03% in recall over HAN. Similarly, T-BERT can further improve the already-high performance of BERT to 95.66% (precision) and 98.76% (recall). Unfortunately, it is impractical to apply BERT and its variations (*i.e.*, DMoS_v2), or even its lightweight version (*e.g.*, ALBERT [32]) in practice. This is due to the huge number of model parameters (108 million parameters in TBERT compare to 1.6 million in THAN) and slow inference speed. In our experiments with hardware specified in Section 4.2.2, THAN takes 0.28s while TBERT takes 9.87s to process each web page on average. As a result, we deploy DMoS (with JNA + THAN) for the online experiments. Note that DMoS already outperforms all the state-of-the-art solutions, including BERT. For the remainder of the paper, DMoS means DMoS (with JNA + THAN), unless specified otherwise.

### 4.1.3 Effect of Jargon Normalization

To demonstrate the effectiveness of the proposed jargon normalization algorithm, we compare THAN and BERT to their DMoS counterparts, which normalize jargons before applying the THAN/ BERT model for classification. By identifying and learning the semantics of jargons, both DMoS and DMoS_v2 achieve better performance, as shown in Table 2: With JNA, the precision of THAN increases from 91.29% to 94.89%, which can reduce manual efforts for verifying false alarms by over 41%. Similarly, JNA can improve the precision of T-BERT from 95.66% to 97.53%, thus cutting false positives by over 43%.

## 4.2 Online Experiments

In this subsection, we introduce the business model of the DMoS service and then analyze its performance based on a

Table 3: Online Testing Results for Five Months

| No. of Total Pages (Sites) | No. of Legitimate Pages (Sites) | No. of Defaced Pages (Sites) | Schemes | True Positive | False Positive | False Negative | Precision | Recall | F1 |
|---|---|---|---|---|---|---|---|---|---|
| 38,526,989 (7298) | 37,994,394 (6474) | 532,595 (824) | WAF | 19,958 | **1634** | 511,342 | **92.43%** | 3.76% | 7.23% |
| | | | DMoS | **532,021** | 65485 | **574** | 89.04% | **99.89%** | **94.15%** |

* Web pages reported in this table have been de-duplicated by MD5 hash values. Websites have been de-duplicated by top-level domains.
* WAF failed in handling 1,239,717 pages due to various exceptions, which are not counted when evaluating its performance.

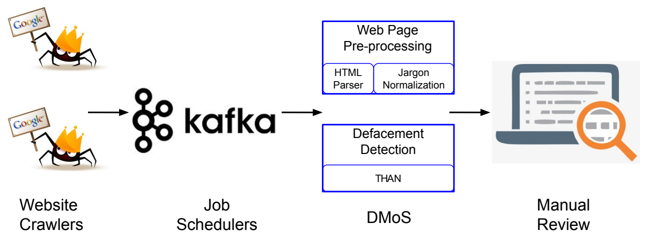


Figure 7: Online Deployment of DMoS.

five-month long online experiment.

### 4.2.1 Usage of DMoS

It is straightforward for site owners/ administrators to use DMoS. Specifically, they just need to submit a URL to register the defacement monitoring service for their website. DMoS then crawls the website from this URL regularly, as described in Section 4.2.2. They also need to provide contact information for defacement notification. Depending on the confidence in the notification, site owners can manually investigate or directly remove the illicit content.

### 4.2.2 Deploying DMoS as a Cloud-Based Service

Figure 7 illustrates how DMoS is deployed as a commercial cloud-based service. Firstly, the provider utilizes website crawlers to acquire web pages covered by the service. It then uses Kafka [16], a distributed event streaming, to distribute the collected web pages to available DMoS instances. Finally, the defaced pages identified by DMoS are examined manually by the provider who would notify the site owners with true positives.

**Website crawlers.** We run website crawlers on 42 machines, each with an 8-core CPU, 16GB Memory, and a 400 Mbps network connection. A machine is configured to run 7 instances of crawlers in parallel. Knowing a defaced website may, by design, serve different pages to visitors originated from different geographical regions, we leverage a nation-wide cloud infrastructure to deploy crawlers in parallel across different provinces to eliminate the impact of page-fetch location. A crawler also mimics various search-engine bots and normal browsers by switching its User-Agent field in the request to ensure it can receive the defaced content. Note that we do not limit DMoS to get web pages only via website crawlers. Any web page acquisition channel (*e.g.*, search engines, or firewalls, *etc.*) can be readily fed into DMoS.

**Scheduler.** The sheer volume of web pages makes real-time detection challenging. We need a reliable, high-throughput, low-latency platform for handling such data feeds. We use Kafka to receive pages from crawlers and distribute them to the available DMoS instance for detection.

**Defacements Detection.** We run 50 instances of DMoS on a CentOS 7 machine with a 16-core CPU and 100GB memory. Each instance consumes 1.2G memory. The classification of each web page, on average, can be completed within 0.3 second. Such runtime efficiency enables DMoS to crawl and analyze millions of web pages for thousands of websites every day.

### 4.2.3 Detection Accuracy

We collect performance statistics of DMoS for five months. As presented in Table 3, DMoS has classified $38,526,989$ web pages (after deduplication) across $7,298$ websites in total. It has discovered $532,021$ defaced web pages among 824 websites. Except for precision, DMoS outperforms the widely-used WAF device in all other metrics, especially the overall metric of $F1$. Note that we can only afford to compare with WAF since it utilizes simple string search algorithms and thus is efficient enough to handle our large volume online dataset in real-time. Although the precision of DMoS is slightly lower, we notice that more than 90% of false positives occur in 67 websites only. Furthermore, these websites typically share similar error-prone sentences. For practical deployment, we can develop a whitelist based on such sentences. More specifically, for any defaced pages identified by DMoS, we can compute the string similarity of the believed-to-be illicit sentences with the whitelist of this website. If they are similar, we treat such defaced pages as false positives and will not manually examine them. In this way, we can afford to perform manual-review for pages reported online.

**Manual verification.** Given the large volume of web pages, it is very resource-consuming to obtain the ground truth for our online experiment. We run the online experiment with DMoS and a WAF device simultaneously and manually check every page reported by either of the systems. Hence, we have accurate true/false positives for both. To estimate false negatives,

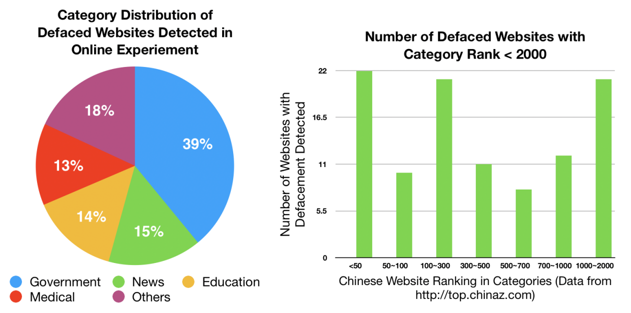


Figure 8: Topic Distribution of Defaced Sites in Online Tests

we manually check every page exclusively reported by the WAF. This gives us $FN_{\overline{DMoS}}$, *i.e.*, the number of pages missed by DMoS but captured by the WAF. For pages reported neither by DMoS nor the WAF, we use alarming keywords (described in Appendix A) to filter out about 95% of innocent pages. We then perform a sampled manual checking [4] for the remaining suspicious pages. This gives us an estimated $FN_{\overline{both}}$, *i.e.*, the number of defaced pages missed by both of the systems. With $TP$ being the true positives for DMoS, the total false negatives and recall can be determined by:

$$FN_{DMoS} = FN_{\overline{DMoS}} + FN_{\overline{both}}\ ,\ Recall_{DMoS} = \frac{TP}{TP + FN_{DMoS}}$$

A similar procedure is applied to compute $FN_{WAF}$ and $Recall_{WAF}$.

This manual verification process was continuously conducted by specialized data annotators of our collaborating industrial partner on a daily basis during our five-month online experiment. Nevertheless, it is still possible to omit defaced web pages. As such, the actual recall of DMoS can be lower.

#### 4.2.4 Statistics of Defaced Websites

Figure 8 presents the categorical distribution of the victim websites with defacements detected. The wide-ranging categories of defaced websites detected by DMoS demonstrate its ability to generalize beyond the training dataset. In particular, the categories of government, news media, education, and medical institutions occupy the majority of defacement targets. A significant portion of defaced websites are quite popular: Amongst 824 websites where defacements are detected, 2 websites belong to the Top 5 list of the overall category ; 22 websites rank within the Top 50, and 105 rank within 2000 in their own categories.

We also categorize defaced web pages by the type of illicit goods and services they promoted. Gambling and porn are found to be the most popular topics in defacement campaigns, accounting for 61% and 27% of the promoted targets, respectively. The game and sales categories account for 9% and 3% respectively. These results are consistent with the distribution within the defaced dataset depicted in Figure 3a.

[4]We uniformly sample 10 pages from each website every day. If any page is defaced, we manually check all pages of the website with similar alarming keywords.

#### 4.2.5 A Longitudinal Study

As the data pattern evolves, a model can become obsolete over time. This is referred to as concept drift in the field of machine learning. Under an adversarial setting, such as website defacements, concept drift can be introduced intentionally by the attacker to impede the detection performance of DMoS. To measure the impact of concept drift, we perform a five-month long longitudinal study using our online dataset. The result is shown in Figure 9. It turns out DMoS performs stable over time, which proves its generality: The recall is always above 99% every month. The precision also stabilizes around 88% after a slight decrease at first. Under such gradual and relatively minor drift, it is possible to maintain (or even enhance) the performance of DMoS at its high level by periodical fine-tune its neural network parameters using new data.

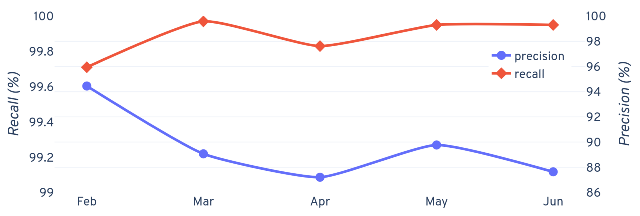


Figure 9: Variation of Performance of DMoS over Time

### 4.3 Comparison with Online URL Checkers

We also compare the performance of DMoS with popular online URL safety-checking tools from major commercial vendors including Baidu [7], Tencent [12] and VirusTotal [5]. Due to the API query limits of these commercial tools, we can only use them to check hundreds of URLs. For fair comparisons, we select different categories of URLs in similar proportions within our dataset. Given the fast-changing nature of defaced web pages[5], we complete our experiments within two days to guarantee consistency. Table 4 shows that DMoS significantly outperforms the other tools according to various metrics. We also observe that these commercial products incorrectly assume websites with ICP (Internet Content Provider) license to be always secure, regardless of their content, which results in their high false negative rate.

[5]Site owners will remove the illicit content immediately after detection.

Table 4: Comparison with other Online URL Checkers

| Schemes | No. of Legitimate Pages | No. of Defaced Pages | Precision | Recall | F1 |
|---|---|---|---|---|---|
| Tencent | 235 | 190 | 50.00% | 6.73% | 11.86% |
| VirusTotal | 235 | 190 | 57.14% | 15.38% | 24.24% |
| Baidu | 235 | 190 | 63.41% | 50.49% | 56.22% |
| DMoS | 235 | 190 | **98.37%** | **97.32%** | **97.85%** |

Table 5: Detection Performance for English Dataset

| Methods | $D_d^{real}$ dataset | | $D_d^{craftS}$ dataset | | $D_d^{craftL}$ dataset | |
|---|---|---|---|---|---|---|
| | Precision | Recall | Precision | Recall | Precision | Recall |
| HAN | 90.39% | 91.21% | 86.43% | 90.12% | 93.54% | 95.86% |
| BERT | 91.51% | 94.55% | 88.23% | 93.72% | 95.81% | 96.43% |
| DMoS (THAN) | 92.54% | 92.46% | 90.65% | 91.63% | 94.72% | 95.98% |
| DMoS_v2 (T-BERT) | **94.10%** | **95.35%** | **92.11%** | **95.42%** | **96.33%** | **97.12%** |

## 4.4 Adapt DMoS for Other Languages

Since our collaborating industrial partner's main market is in China, we have focused on detecting defacements in Chinese web pages so far. Note however that, our approach is general in nature and can be applied to other languages with the following steps:

- Apply similar approaches to collect datasets for training.
- Replace the jargon normalization algorithm with language-specific de-obfuscation techniques if any.
- Take the language-specific word embeddings as the input and train the THAN model as usual.

In what follows, we demonstrate how DMoS can be adapted to detect defacements of English web pages.

### 4.4.1 Collecting the English Dataset

As a proof of concept, we focus on the category of porn-related defacements without considering other types. Following the same data acquisition approach described in Section 3.1.1, we first collect a list of adult-content websites. We then crawl their web pages and extract illicit keywords from the titles. This is based on the observation that titles of adult-content pages typically contain a large portion of illicit keywords. By manually examining these candidates, we finally collect $10,656$ English illicit keywords related to adult content.

To collect defaced pages, we develop two approaches as follows:

- As discussed in Section 3.1.1, we query search engines with illicit keywords and crawl returned web pages. After manual checking of crawled pages, we collect $5,928$ defaced pages among 22 websites. We refer to this dataset as $D_d^{real}$.
- We carefully craft a list of defaced pages by following defacers' procedures. Specifically, we notice that defacers typically utilize off-the-shelf tools [18] to generate defaced pages based on a list of illicit keywords. Leveraging these tools, we generate two datasets with different amounts of defacements. Each dataset includes $12,817$ defaced pages. The first one is the stealthy defacement dataset ($D_d^{craftS}$), where each page contains 3 to 7 illicit keywords. The second one has more defacements with more than 20 illicit keywords in each page, which we refer to as $D_d^{craftL}$.

The legitimate web pages are collected from top international websites including education, news and government websites. In total, we collect $31,946$ web pages, which we refer to as $D_l$. We use 80% of the defacement dataset (*i.e.*, $D_d^{real}$, $D_d^{craftS}$ or $D_d^{craftL}$) and $D_l$ for training. We then use the remaining 20% of defacement dataset and $D_l$ for testing.

To facilitate further research and reproducibility, we have released a subset of the English defacement dataset at [17], which includes 500 real defaced web pages, over 25,000 crafted web pages with defacements, and over 1,400 illicit keywords. Interestingly, during collection of real defaced pages, we identified defacements in some well-known institutional domains including *nyu.edu*, *mit.edu* and *cuhk.edu.hk*. We included all of them in our real testing set, and the detection accuracy is reported in the first column in Table 5. Their snapshots are also available in our released dataset. Most of the defacements in the affected websites had been removed by their administrators over the course of our experiments.

### 4.4.2 Detecting Defaced English Web Pages

The training procedure is similar except for the following differences:

- We do not notice the usage of obfuscated jargons in English defacements. If any, interested readers can directly apply the language-specific jargon normalization algorithms. Therefore, we only make Lemmatization with NTLK to map words with similar meaning into one word (*e.g.*, better $\rightarrow$ good, doing $\rightarrow$ do).
- Similar to the Chinese scenario, we then train the HTML classification model (DMoS/DMoS_v2). Here we use word2vec as the initial embedding.

The experiment results for the three datasets are shown in Table 5. It shows our proposed tag-aware method can improve the performance of HAN and BERT under the English scenario as well. This demonstrates the applicability of our approach to other languages.

A closer examination reveals that our approach brings more improvements for the detection of stealthy defacements. Specifically, the improvement over HAN is: 4.22% (stealthy defacements) *vs* 1.18% (large defacements) in precision, and 1.51% (stealthy defacements) *vs* 0.12% (large defacements) in recall. The improvement over BERT is: 3.88% (stealthy defacements) *vs* 0.52% (large defacements) in precision, and 1.70% (stealthy defacements) *vs* 0.69% (large defacements) in recall. These results are consistent with our previous findings on Chinese defacement detection.

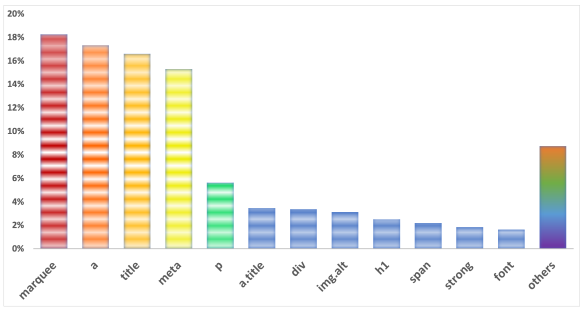


Figure 10: Relative Frequency of Infected HTML Tags. Duplicated tags in one web page are not counted to avoid bias.

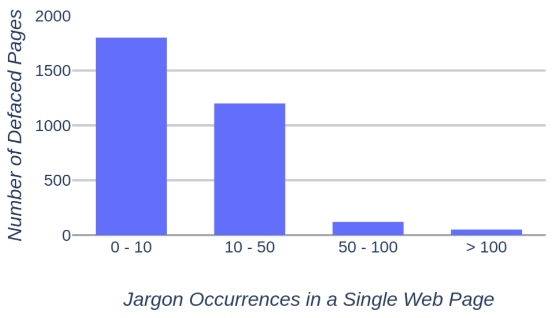


Figure 11: Distribution of Jargon Occurrences

## 5 New Discoveries and Measurements

Based on the detection results of DMoS on Chinese dataset, we have performed a measurement study to better understand the preferences of website defacement techniques. This study has led to a set of surprising findings and new insights.

### 5.1 Prevalence of Stealthy Defacements

Our study confirms that most defacers tend to keep the defacements to a minimum. By counting the injected/altered tags in each defaced web page, we find that 26% of defaced web pages are injected with only one tag. 70% of defaced web pages contain fewer than 10 injected tags. Only 5% of web pages contain more than 50 injected tags. This can explain why THAN outperforms the standard HAN model: it focuses more on the suspicious tags while avoiding the noise introduced by the majority of normal regions.

Analyzing the frequency distribution of HTML tags in defaced web pages, we find that while 44 different tags are used to inject illicit promotional text, $marquee$, $a$, $title$, $meta$ are the most preferred tags. Together, they comprise 67% of the data. Figure 10 reports the corresponding statistics. The *marquee* tag is widely used because it is convenient to visually hide its content with the *scrollamount* and *scrolldelay* properties. While some other styling techniques are also used for cloaking, *e.g.*, tiny font or white color, they have been known to and detected by most search engines for a long time [35]. The prevalence of the $a$ tag is simply because hackers need to put links to lead users or search engine crawlers to their promoted websites. Lastly, $title$ and $meta$ tags are preferred as they have higher priority when search engines index the web page.

### 5.2 Jargon Normalization Results

Our proposed Jargon Normalization Algorithm (JNA) has achieved a high precision. To compute the precision, we manually review the jargons identified by JNA in the offline testing dataset. JNA identifies 50,848 obfuscated jargons among 3,147 pages out of 20,958 defaced pages. We manually check these jargons and do not find false positives. For the 40,426 legitimate pages, JNA recognizes 7 obfuscated jargons across 4 pages. Particularly, 2 jargons are, in fact, typos within original content. The other 5 jargons are false positives of JNA. Notice that, even in the cases where JNA has made a mistake, the corresponding pages can still be correctly classified by DMoS.

Figure 11 shows the distribution of the number of jargons found in defaced pages. Although most defaced pages contain fewer than 50 obfuscated jargons, some pages surprisingly include hundreds of them. Table 6 lists an example keyword of MARK SIX (*i.e.*, "六合彩") as well as the frequency of its Top-8 transformed jargons. The *Equivalence in English* column explains the Chinese jargon obfuscation through similar representations in English[6]. Usually, the initial keyword is widely used by defacers, and the ones that evolved from it also receive attention, albeit less than the initial one. Given the large volume of obfuscated jargons, we need to revert the jargons to their base forms so that THAN can understand their semantics to enhance the correct classification of defaced pages. As shown in Table 2, the jargon normalization algorithm improves DMoS by 3.60% in precision and 0.51% in recall.

### 5.3 Differences in Search Engines

Search engines have different policies regarding black hat SEO. Leveraging this observation, defacers sometimes perform website defacements targeting some specific search engines. As observed, a significant portion of defaced web pages that DMoS detected applies the cloaking techniques, by checking the User-Agent and Referer header, to return illicit content to certain search engine bots only. More illicit content appearing in search results indicates weaker black hat SEO tackling of the corresponding search engine.

To improve user experience, most modern search engines display warnings next to defaced or malicious websites in their search results so that users will not visit these potentially dangerous websites. Figure 12 illustrates one example

[6]These English equivalent jargons are for illustration purpose only. They have not been used by defacers in practice.

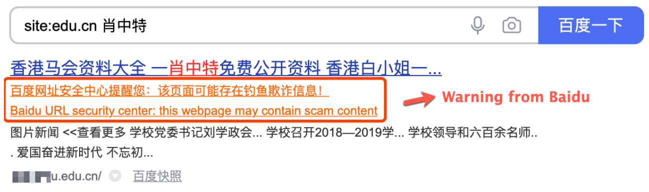
Table 6: Top 8 Transformed Jargons Related to “MARK SIX”

| No. | Keywords | Equivalence in English | Frequency |
|---|---|---|---|
| 1 | 六合彩 | MARK SIX | 620356 |
| 2 | 六和彩 | MARC SIX | 145759 |
| 3 | 6和彩 | MARK 6 | 48991 |
| 4 | 六合采 | MARK SYX | 26509 |
| 5 | 六合财 | MARK SIKS | 4852 |
| 6 | 六台彩 | M4RK SIX | 3378 |
| 7 | liuhecai | N.A. | 3357 |
| 8 | 六盒彩 | MARCK SIX | 3181 |

Table 7: Search Result Warnings and Defaced Pages Statistics

| Search Engines | Defaced Pages | With Warning Tips |
|---|---|---|
| baidu.com | 4451 | 29.8% |
| m.baidu.com | 231 | 31.1% |
| so.com | **10300** | **11.3%** |
| m.so.com | **8770** | **7.7%** |
| sogou.com | 5499 | 17.9% |
| m.sogou.com | 3430 | 21.4% |

of such warning tips. The higher percentage of warnings indicates that the search engine has more powerful mitigation against black hat SEO. During defaced dataset collection for offline testing (Section 3.1.1), one batch of data (22,939 web pages over 2,563 domains) is collected simultaneously from three major Chinese search engines: Baidu (baidu.com), 360 Search (so.com), and Sogou (sogou.com), together with their mobile versions (m.baidu.com, m.so.com, and m.sogou.com). By comparing the proportion of defaced web pages and the number of search engine warnings, we can deduce which search engine is preferred by defacers and has less effective black hat SEO mitigation techniques. Table 7 presents our findings. It shows that 360 Search is a preferred target for defacers, as it has a significantly higher portion of defaced pages with the least warnings. Meanwhile, given a large number of defaced pages from mobile search results (especially for 360 Search and Sogou), it seems the mobile versions of search engines have already attracted defacers’ attention for exploitation.


site:edu.cn 肖中特

香港马会资料大全 一肖中特免费公开资料 香港白小姐一...

百度网址安全中心提醒您：该页面可能存在钓鱼欺诈信息！

Baidu URL security center: this webpage may contain scam content

图片新闻 <<查看更多 学校党委书记刘学政会... 学校召开2018—2019学... 学校领导和六百余名师..

. 爱国奋进新时代 不忘初...

[redacted]u.edu.cn/ 百度快照

Figure 12: A Warning Generated by the Search Engine.

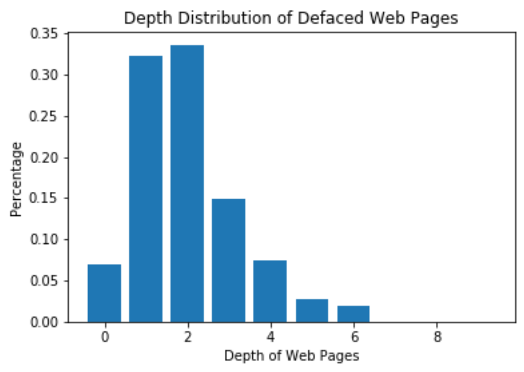


Figure 13: Depth of Defaced Pages from Site Landing Page

## 5.4 Hijacking of Expired DNS Domains

We have observed a recent trend of attackers abusing expired DNS domains during our manual post-detection analysis, especially for those domains previously held by government agencies. It is easy to understand the attackers’ motivation: these privileged domains usually rank high in search engines. After the domain expired, its ranking retains for some period before search engines adjust it. Furthermore, it is easier for defacers to take over the expired domains (*e.g.*, by direct purchase or exploiting vulnerabilities in outdated, no-longer-maintained websites). Attackers can thus promote their illicit content more effectively. An interesting note: at the beginning of 2019, we observed a burst of abuse on expired domains initially owned by the Chinese government. Upon further investigations, we learned that the Central Office of China had issued a document requiring government organizations to start using the .gov.cn domain in September 2018 [3]. As a result, many government websites that were not under the .gov.cn domain had to migrate to the new domains and thus released the control of their previous domains.

## 5.5 Path Depth of Defaced Web Pages

We are interested in whether the web pages within a shallower depth of a site are more likely to be defaced. Here, the depth means the number of hops from the landing page to the defaced page, and zero-depth means the landing page itself. To this end, we examine the defaced web pages discovered by DMoS. Although we limit the depth of our crawler to be 12, as presented in Figure 13, most defacements occur within 4 hops from the landing page, and the overall percentage for page-depth greater than 8 is negligible. Knowing the depth distribution of the defaced pages, we can reduce the depth of our crawler without sacrificing the recall of DMoS.

## 6 Discussions

Our research shows that DMoS offers a highly effective solution to the threat of promotional defacement. In this section, we discuss its limitations and future research directions.

### 6.1 Robustness under Incomplete Keyword List

It is difficult, if not impossible, to collect a complete list of illicit keywords. Fortunately, our approach has performed effectively even under an incomplete illicit keyword list due to the following reasons:

- Promotional defacements mainly aim at SEO. By leveraging the "related keyword" support by search engines, our method can already harvest the majority of illicit keywords used by defacements.
- Although the set of jargons is extensive, their original forms are limited. For example, all the obfuscated jargons listed in Table 6 are evolved from a single initial jargon, *i.e.*, Mark Six. Given the Jargon Normalization algorithm, we can map unknown obfuscated jargons to the limited set of original jargons.
- As shown in Section 5.1, adversaries can deface multiple HTML tags in one page. Therefore, we may miss multiple keywords/tags, but can still have a good chance to catch others.
- THAN takes word embeddings as the input. Words can have similar embeddings if they appear in similar contexts in the corpus. As such, DMoS has generalization power and can understand never-before-seen jargons to a certain extent since jargons often appear in similar contexts.

### 6.2 Robustness against Other Evasion Schemes

Defacers can adjust their strategy to keep defacements hidden from DMoS. They can hide the illicit content by embedding it within images. Worse still, these images can be obfuscated by introducing distortions and noises [48]. However, such evasion techniques cannot make illicit content appear in the search results and limit their impact. An adversary may use other evasion tricks by targeting specific search engines only. For example, defacers may leverage JavaScript to hide illicit content from search engines except for Google, since the latter can somehow index JavaScript framework [2]. However, DMoS does not limit itself to get web pages from crawlers only. Instead, it can be used together with other web page acquisition channels (*e.g.*, firewalls), making such evasion tricks less effective.

## 7 Related Work

In this section, we review existing efforts in website defacement detection from three perspectives.

### 7.1 Defacement Detection

Davanzo *et al.* [25] treat web defacement detection as an anomaly detection problem of web pages. However, the evaluation is based on only 620 hand-picked web pages, which undermines its general applicability. Delta [20] designs a framework to detect whether a website change is malicious or benign based on clustering. By approaching the problem visually, Meerkat [21] renders web pages using headless browsers and applies computer vision techniques to identify the malicious content changes among screenshots of the website. However, these works are more suitable for detecting politically-motivated defacements, where websites' visual appearance is deliberately changed. They are unlikely to achieve similar performance for stealthy promotional defacements.

The most related work of DMoS should be [39], which utilizes the hashing trick to create a hierarchical representation of web documents (*e.g.*, HTML, JavaScript, CSS, *etc.*) and then feed such input to neural networks for general web documents classification. However, this work does not utilize the information carried by HTML tags and thus cannot distinguish stealthy defacements from legitimate noises in many cases. Furthermore, [39] uses the web content reported to VirusTotal [5], a platform aggregating many anti-virus products, to train the network. As discussed in Section 4.3, most vendors in VirusTotal fail to detect promotional defacements, which affects the performance of the trained model.

### 7.2 Black Hat SEO Investigation

Black hat SEO has become the most popular channel for advertising illicit goods and services. Wang *et al.* [44] show that at least 0.48% of Google autocomplete results are polluted. Nektarios *et al.* [34] construct a representative list of 218 drug-related keywords for Google search and automatically gather 40,000 malicious search results to quantify the prevalence of search-redirection attacks, especially for unlicensed pharmacies. Following a similar approach, Nektarios *et al.* [33] further perform a four-year long longitudinal investigation to reveal the evolution of the attacking strategies and technologies. Advanced offensive techniques, like using jargons, turn out to be effective and, as reported by [42], can often evade detection to bypass blockade by search engines. Towards this end, Liao *et al.* [35] develop a semantics-based approach to facilitate search engines to detect those compromised sites by identifying the semantic gaps between the malicious keywords and the infected sponsored Top-Level Domains (sTLD). In contrast, DMoS supports the detection of promotional defacements in any domain without limiting to specific sTLD.

### 7.3 Natural Language Processing

Besides defacement detection literature, researchers have made substantial progress in the area of natural-language-processing-based text classification, *e.g.*, BERT [24], FastText [31], *etc.* Our scheme differs from typical document classification work, *e.g.* sentiment analysis, in that we seek to detect an active adversary who tries hard to evade detection and is willing to obscure the semantics of their text. While inspired by Hierarchical Attention Network [46], we have developed new techniques to encode the HTML tag information into the neural network and examine content at hierarchical spatial scales. As a result, we can capture locality and thus achieve higher accuracy compared to the direct application of state-of-the-art NLP models.

## 8 Conclusion

In this paper, we propose DMOS, a scalable cloud-based detector for promotional website defacements. Using DMOS, we examine 38,526,989 web pages over 7,298 websites, finding 11% of which, to our surprise, have been defaced at least once. Our study demonstrates the pervasiveness of conspiring acts between the seekers and promoters of illicit goods and services over the Internet. By effectively detecting website defacements at their early stage, DMOS has substantially raised the bar to thwart the negative impact of promotional defacements as the site owners can quickly remove the illicit content to contain its damage. The findings of our measurement study also reveal new interesting phenomena in the Internet underground ecosystem.

## Acknowledgements and Ethical Considerations

We thank our shepherd Prof. Bimal Viswanath and the anonymous reviewers for their valuable comments, which help to improve the paper considerably. We also thank our colleagues Shangcheng Shi, Zuoye Gong, and Zhiyi Cheng for their help in this work. This research is supported in part by the CUHK Project Impact Enhancement Fund (Project#3133292) and the CUHK MobiTeC R&D Fund. For all the Chinese websites in our experiments, crawling and detection of websites are authorized by their owners with legal agreements. The crawled pages are kept privately and cannot be shared with the public. We have informed owners of all the 824 defaced websites detected in our online experiment and provided them a complete list of defaced page URLs. All the site owners had subsequently confirmed our findings, removed the illicit content, and deployed further security measures. For the defacements we identified throughout our English dataset collection process, most affected websites had removed the illicit content by themselves during our experiment. We have notified four websites where defacements remain.

## References

[1] Four-corner system, 1995.

[2] Can google properly crawl and index javascript frameworks? a javascript seo experiment. online, 2017. https://www.onely.com/blog/javascript-seo-experiment/.

[3] Central office notice: Government websites domain policy. online, 2018. http://www.gov.cn/zhengce/content/2018-09/06/content_5319675.htm.

[4] ssdeep project. online, 2018. https://ssdeep-project.github.io/ssdeep/index.html.

[5] Virustotal. online, 2018. https://www.virustotal.com.

[6] Aho–corasick algorithm. online, 2019. https://en.wikipedia.org/wiki/Aho%E2%80%93Corasick_algorithm.

[7] Baidu url security center. online, 2019. https://bsb.baidu.com.

[8] China webmaster. online, 2019. http://top.chinaz.com/.

[9] Chinese wiki corpus. online, 2019. https://dumps.wikimedia.org/zhwiki/latest/zhwiki-latest-pages-articles.xml.bz2.

[10] Language model. online, 2019. https://en.wikipedia.org/wiki/Language_model.

[11] Selenium. online, 2019. https://www.seleniumhq.org/.

[12] Tencent url security center. online, 2019. https://urlsec.qq.com.

[13] Tensorflow. online, 2019. https://www.tensorflow.org/.

[14] Tfidf. online, 2019. https://en.wikipedia.org/wiki/Tf%E2%80%93idf.

[15] Exploring google hacking techniques. online, 2020. https://securitytrails.com/blog/google-hacking-techniques.

[16] Kafka: A distributed streaming platform. online, 2020. https://kafka.apache.org/intro.

[17] Sample defacement dataset of english webpages. online, 2020. http://mobitec.ie.cuhk.edu.hk/DMoS/.

[18] Some example tools for defacement and black hat seo. online, 2020. http://www.zylou.cn/hmseo.

[19] Transformers: State-of-the-art natural language processing for pytorch and tensorflow 2.0. online, 2020. https://github.com/huggingface/transformers.

[20] BORGOLTE, K., KRUEGEL, C., AND VIGNA, G. Delta: Automatic identification of unknown web-based infection campaigns. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*.

[21] BORGOLTE, K., KRUEGEL, C., AND VIGNA, G. Meerkat: Detecting website defacements through image-based object recognition. In *24th USENIX Security Symposium*.

[22] CHEN, T., AND GUESTRIN, C. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (2016), ACM, pp. 785–794.

[23] CHUNG, Y.-J., TOYODA, M., AND KITSUREGAWA, M. A study of link farm distribution and evolution using a time series of web snapshots. In *Proceedings of the 5th international workshop on Adversarial information retrieval on the Web* (2009), ACM, pp. 9–16.

[24] DEVLIN, J., CHANG, M.-W., LEE, K., AND TOUTANOVA, K. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*, pp. 4171–4186.

[25] G. DAVANZO, E. M., AND BARTOLI, A. Anomaly detection techniques for a web defacement monitoring service. In *Expert Systems with Applications* (2011).

[26] GESENHUES, A. Study: Organic search drives 51% of traffic, social only 5%. *Search Engine Land* (2014).

[27] GOOGLE. Policies for content posted by users on search. online, 2018. https://support.google.com/websearch/answer/7408270?hl=en.

[28] IMF. The countries with the largest shadow economies. online, 2017. https://tinyurl.com/x0b0r79y.

[29] JOHN, J. P., YU, F., XIE, Y., KRISHNAMURTHY, A., AND ABADI, M. deseo: Combating search-result poisoning. In *USENIX security symposium* (2011), pp. 20–35.

[30] JOSLIN, M., LI, N., HAO, S., XUE, M., AND ZHU, H. Measuring and analyzing search engine poisoning of linguistic collisions. In *Proceedings-IEEE Symposium on Security and Privacy* (2019), IEEE.

[31] JOULIN, A., GRAVE, E., BOJANOWSKI, P., AND MIKOLOV, T. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers* (2017), pp. 427–431.

[32] LAN, Z., CHEN, M., GOODMAN, S., GIMPEL, K., SHARMA, P., AND SORICUT, R. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations* (2019).

[33] LEONTIADIS, N., MOORE, T., AND CHRISTIN, N. A nearly four-year longitudinal study of search-engine poisoning. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pp. 930–941.

[34] LEONTIADIS, N., MOORE, T., AND CHRISTIN, N. Measuring and analyzing search-redirection attacks in the illicit online prescription drug trade. In *USENIX Security Symposium* (2011), vol. 11.

[35] LIAO, X., YUAN, K., WANG, X., PEI, Z., YANG, H., CHEN, J., DUAN, H., DU, K., ALOWAISHEQ, E., ALRWAIS, S., ET AL. Seeking nonsense, looking for trouble: Efficient promotional-infection detection through semantic inconsistency search. In *2016 IEEE Symposium on Security and Privacy (SP)*, pp. 707–723.

[36] MAGGI, F., BALDUZZI, M., FLORES, R., GU, L., AND CIANCAGLINI, V. Investigating web defacement campaigns at large. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*.

[37] MIKOLOV, T., CHEN, K., CORRADO, G., AND DEAN, J. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[38] PENNINGTON, J., SOCHER, R., AND MANNING, C. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (2014), pp. 1532–1543.

[39] SAXE, J., HARANG, R., WILD, C., AND SANDERS, H. A deep learning approach to fast, format-agnostic detection of malicious web content. In *2018 IEEE Security and Privacy Workshops (SPW)*, pp. 8–14.

[40] SEOMOZ. Google algorithm change history. online, 2018. https://moz.com/google-algorithm-change.

[41] VASWANI, A., SHAZEER, N., PARMAR, N., USZKOREIT, J., JONES, L., GOMEZ, A. N., KAISER, Ł., AND POLOSUKHIN, I. Attention is all you need. In *Advances in neural information processing systems* (2017), pp. 5998–6008.

[42] WANG, D. Y., DER, M., KARAMI, M., SAUL, L., MCCOY, D., SAVAGE, S., AND VOELKER, G. M. Search+ seizure: The effectiveness of interventions on seo campaigns. In *Proceedings of the 2014 Conference on Internet Measurement Conference*, pp. 359–372.

[43] WANG, D. Y., SAVAGE, S., AND VOELKER, G. M. Cloak and dagger: dynamics of web search cloaking. In *Proceedings of the 18th ACM conference on Computer and communications security* (2011), pp. 477–490.

[44] WANG, P., MI, X., LIAO, X., WANG, X., YUAN, K., QIAN, F., AND BEYAH, R. Game of missuggestions: Semantic analysis of search-autocomplete manipulations. In *25th Annual Network & Distributed System Security Symposium (NDSS)* (2018).

[45] YANG, H., MA, X., DU, K., LI, Z., DUAN, H., SU, X., LIU, G., GENG, Z., AND WU, J. How to learn klingon without a dictionary: Detection and measurement of black keywords used by the underground economy. In *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 751–769.

[46] YANG, Z., YANG, D., DYER, C., HE, X., SMOLA, A., AND HOVY, E. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pp. 1480–1489.

[47] YUAN, K., LU, H., LIAO, X., AND WANG, X. Reading thieves' cant: automatically identifying and understanding dark jargons from cybercrime marketplaces. In *27th USENIX Security Symposium* (2018), pp. 1027–1041.

[48] YUAN, K., TANGY, D., LIAO, X., WANG, X., FENG, X., CHEN, Y., SUN, M., LU, M., AND ZHANG, K. Stealthy porn: Understanding real-world adversarial images for illicit online promotion. In *Proceedings-IEEE Symposium on Security and Privacy* (2019).

[49] YUJIAN, L., AND BO, L. A normalized levenshtein distance metric. *IEEE transactions on pattern analysis and machine intelligence 29*, 6 (2007), 1091–1095.

## A Collecting Alarming Keywords

Compared to illicit keywords, developing the alarming keywords turns out to be relatively straightforward. While defacers often obfuscate keywords, it noteworthy that many obfuscated jargons share the same stem (*i.e.*, commonly-used sequence). For example, there are many illicit phrases derived from "mark six", like "buy mark six in Hong Kong" or "Hong Kong mark six". However, the stem "mark six" is seldom changed. As such, the number of stems is limited, making them suitable as alarming keywords. Specifically, we build the alarming keywords as follows:

- We build a dictionary of stop words (*e.g.*, "the", "Hong Kong"), which are used in the legitimate dataset as well.
- For each phrase, we remove the stop words. For example, "buy mark six in Hong Kong" becomes "buy mark six".
- We extract all *n*-gram sequences where *n* is larger than a threshold $\theta_n$ (set to 2), *e.g.*, "buy mark six" is splitted into "buy mark", "mark six", and "buy mark six".
- For each sequence, we compute its frequency in the defaced and legitimate dataset, $f_d$ and $f_l$, respectively.
- Finally, we compute the frequency difference $f_d - f_l$ for all substrings, starting from the shortest one. If it is more than a threshold $\theta_{alarm}$, we include this sequence as an alarming keyword.

Note that we will extract at least one sequence from every illicit phrase to ensure the completeness of alarming keywords.