computers

MDPI

Article

Detecting Website Defacements Based on Machine Learning Techniques and Attack Signatures

Xuan Dau Hoang * and Ngoc Tuong Nguyen

Posts and Telecommunications Institute of Technology, Hanoi 100000, Vietnam; tuong.nguyenng@gmail.com

* Correspondence: dauhx@ptit.edu.vn; Tel.: +84-904-534-390

Received: 23 February 2019; Accepted: 7 May 2019; Published: 8 May 2019

Abstract: Defacement attacks have long been considered one of prime threats to websites and web applications of companies, enterprises, and government organizations. Defacement attacks can bring serious consequences to owners of websites, including immediate interruption of website operations and damage of the owner reputation, which may result in huge financial losses. Many solutions have been researched and deployed for monitoring and detection of website defacement attacks, such as those based on checksum comparison, diff comparison, DOM tree analysis, and complicated algorithms. However, some solutions only work on static websites and others demand extensive computing resources. This paper proposes a hybrid defacement detection model based on the combination of the machine learning-based detection and the signature-based detection. The machine learning-based detection first constructs a detection profile using training data of both normal and defaced web pages. Then, it uses the profile to classify monitored web pages into either normal or attacked. The machine learning-based component can effectively detect defacements for both static pages and dynamic pages. On the other hand, the signature-based detection is used to boost the model's processing performance for common types of defacements. Extensive experiments show that our model produces an overall accuracy of more than 99.26% and a false positive rate of about 0.27%. Moreover, our model is suitable for implementation of a real-time website defacement monitoring system because it does not demand extensive computing resources.

Keywords: defacement attacks of websites; defacement monitoring and detection; anomaly-based defacement detection; signature-based defacement detection; machine learning-based defacement detection

1. Introduction

Defacement attacks are a type of attacks that amend the website's content and as a result change the website's appearance [1,2]. Figure 1 is a web page of Tuy-Hoa (Vietnam) airport's website, that was defaced in March 2017 and Figure 2 is the home page of an Australian government's website that was defaced by Indonesian hackers with a message "Stop Spying on Indonesia". According to some reports, the number of defacement attacks reported in the world escalated from 2010 to 2011 and from 2012 to 2013 [2,3]. However, the number of defacement attacks has reduced in recent years [2,3]. Nevertheless, there are still thousands of websites and web applications that are defaced everyday all over the world [2–4]. Here are some of the most popular defacement attacks reported in the world recently [5]:

- In 2011, the home page of the website for Harvard University was replaced by the photo of the Syrian President, Bashar Al-Assad.
- In 2012, there were about 500 Chinese websites defaced by an anonymous hacker group.
- In 2013, the whole website of MIT University, USA was defaced after the death of the well-known hacker, Aaron Swartz.

- In 2014, about 100 Singaporean websites were defaced. Most of these websites were operated by the opposition Reform Party.
- In 2015, an ISIS propaganda website on the dark web was defaced and its contents were replaced with online medicine advertisements for selling Prozac and Viagra.

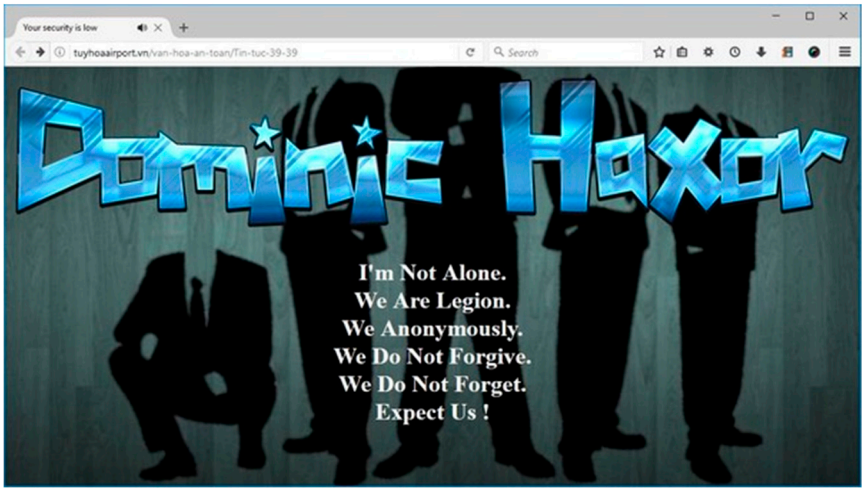

Figure 1. A web page of Tuy-Hoa, Vietnam airport's website defaced in March 2017.

Figure 2. The home page of an Australian government's website defaced by Indonesian hackers.

Although many causes of defacement attacks have been pointed out, the main cause is websites, web applications, or hosting servers with severe security vulnerabilities that are exploited and permit attackers to initiate defacement attacks [1,2,4]. Common security vulnerabilities that exist in websites and web applications include SQLi (SQL injection), XSS (cross-site scripting), CSRF (cross-site request forgery), inclusion of local or remote files, inappropriate account management, and software that is not updated [1,2,4].

Defacement attacks to websites, web portals, or web applications can result in critical consequences to their owners. The attacks can cause an interruption of the website's normal operations, damage to the owner reputation, and possible losses of the valuable data. In turn, these may lead to a huge financial loss. A defacement attack to a website immediately interrupts its normal operations because the organization's staff and customers are not able to access features or services provided by the website. Furthermore, if appropriate countermeasures are not applied timely there may be more new attacks to the website in the near future because the details of the website's security vulnerabilities are

leaked. The reputation damage to the website owner, and in the long term, potential data losses are also serious. Due to the scope of this paper, interested readers may find the detailed discussion about this in [5].

Because defacement attacks to websites and web applications are widespread and there are serious impacts, many countermeasures have been proposed and deployed in practice. Current countermeasures against defacement attacks include: (1) scanning and fixing security vulnerabilities that exist in websites, web portals, or web applications and (2) installing defacement monitoring tools, such as VNCS web monitoring [6], Nagios web application monitoring software [7], website defacement monitoring [8], and WebOrion defacement monitor [9].

This paper proposes a hybrid website defacement detection model that is based on the combination of the machine learning-based detection and the signature-based detection. We extend the machine learning-based detection method proposed in our previous work [10] and use it in the proposed hybrid defacement detection model. The advantages of the machine learning-based detection are (1) the detection profile can be inferred from the training data automatically and (2) the high overall detection accuracy and the low false positive detection rate. The signature-based detection is used to boost the processing speed of the proposed model for common forms of defacement attacks.

The remainder of the paper is structured as follows: Section 2 discusses some related works, Section 3 presents the machine learning-based detection model, Section 4 presents the hybrid website defacement detection model, and Section 5 provides the paper's conclusion.

2. Related Works

There are many website defacement monitoring and detection methods and tools that have been proposed and implemented in practice. These solutions can be divided into two categories which are the signature-based detection approach and the anomaly-based detection approach [11,12]. The signature-based detection approach first creates a set of known attack signatures from defaced web pages. Attack signatures are usually encoded in the form of rules, or string patterns. Then, the approach looks for attack signatures in the monitored web pages. If a match is found, a defaced attack is detected. The signature-based approach is fast and efficient for detecting known attacks. However, it is not able to detect new-form or unknown attacks.

On the other hand, the anomaly-based detection approach first constructs a "profile" from the information of monitored pages of a website that is in normal working conditions. Then, the pages are observed to extract the information, and then the page information is compared with the profile to look for a difference. A defacement attack is detected if any notable difference is found and an attack alarm is raised. The major advantage of this approach is that it has the potential to detect new or unknown attacks. However, it is very hard to decide the detection threshold between the monitored web page and the profile because the content of dynamic web pages changes regularly.

Anomaly-based techniques for the defacement monitoring and detection of websites and web applications consist of those based on traditional comparison methods as well as advanced methods. While traditional comparison methods include checksum comparison, diff comparison, and DOM tree analysis, advanced methods are based on complicated or learning techniques, such as statistics, data mining, machine leaning, genetic programming, and analysis of page screenshots [11,12]. The following parts of this section will provide a description of these methods. In addition, some website defacement monitoring tools widely used in practice are also discussed.

2.1. Defacement Detection of Websites Based on Traditional Comparisons

Defacement detection of websites and web applications using the comparison of checksums is one of the simplest methods to find changes in web pages. Firstly, the checksum of the web page's content is computed using hashing algorithms, such as MD5 or SHA1, and stored in the detection profile. Secondly, the web page is monitored and a new checksum of the web page's content is calculated and then compared with its corresponding checksum saved in the detection profile. If the two checksum

values are not the same, an alarm is raised. This technique seems to work fine for static web pages. However, the technique is not applicable for dynamic pages, for instance web pages of e-commerce websites because their content changes frequently [11,12].

Diff comparison method uses the DIFF tool which is commonly available on Linux and UNIX environments. DIFF is used to compare the current content of the web page and its content stored in the profile to find the changes. The most difficult thing to do is to decide on an anomaly threshold as the input for the monitoring process of each web page. In short, the Diff comparison technique is relatively effective for most dynamic pages if the anomaly threshold is chosen correctly [11,12].

DOM is an API that determines the logical structure of web pages, or HTML documents. DOM can be used to scan and analyze the web page structure. The DOM tree analysis can be used to detect changes in the web page structure, rather than changes in the web page content. First, the page structure is extracted from the page content in the normal working condition and stored in the profile. Then, the page structure of the monitored page is extracted and then compared with the stored page structure in the profile to look for the difference. If a notable difference is found between the page structures an alarm is raised. Generally, this technique works fine for stable structured web pages [11,12]. However, it is not able to detect unauthorized modifications in the content of the web pages.

2.2. Defacement Detection of Websites Based on Advanced Methods

This section presents a survey of some defacement detection proposals based on advanced methods, including Kim et al. [13], Medvet et al. [14], Bartoli et al. [15] and Borgolte et al. [16].

Kim et al. [13] proposed the use of a statistical method for the web page defacement monitoring and detection. The proposed method consists of the training stage and the detection stage. In the training stage, the HTML content of each normal web page is first divided into features using the 2-gram method, and then the occurring frequency of each 2-gram or feature is counted. On the basis of the results of a statistical survey, they conclude that 300 2-grams at the highest occurring frequencies are sufficient to represent a web page for the defacement detection. The detection profile contains all normal web pages of the training dataset, each of which is transferred to a vector of 300 2-grams and their occurring frequencies. In the detection stage, as shown in Figure 3, the monitored web page is first retrieved, and then its HTML content is processed and converted to a vector using the same technique done for training web pages. Next, the monitored page's vector is compared with the corresponding page vector stored in the detection profile using the cosine distance to compute the similarity. If the computed similarity is less than the abnormal threshold, an attack alarm is raised. The abnormal threshold is generated initially and then dynamically updated for each web page periodically. The strong point of the proposed method is that it can create and adjust dynamic detection thresholds, and thereby can reduce the false alarm rate. However, the major shortcomings of this approach are (1) the periodic adjusted thresholds are not appropriate for monitored web pages where the content is frequently changed, and therefore the proposed method still generates more false alarms and (2) it demands high computing resources for the dynamic threshold adjustment for each monitored page.

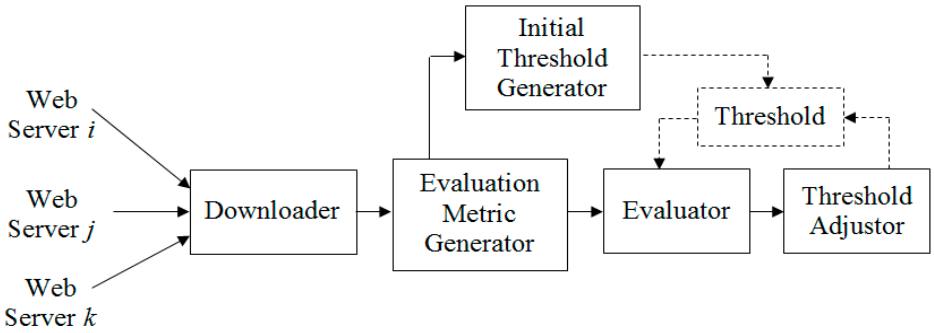

Figure 3. Detection process for web page defacements proposed by Kim et al. [13].

Medvet et al. [14] and Bartoli et al. [15] proposed building the website defacement detection profile by using genetic programming techniques. First, in order to collect data of web pages, they use 43 sensors for monitoring and extracting the information of monitored pages. The next step is the vectorization process, where the gathered information of each page is transformed into a vector of 1466 elements. The proposed method includes two stages of training and detection. In the training stage, web pages of normal working websites are retrieved and vectorized to construct the detection profile based on genetic programming techniques. In the detection stage, the information of the monitored web page is retrieved, vectorized, and then compared with the detection profile to look for the difference. If any significant difference is found an attack alarm is raised. The main drawbacks of this approach are that (1) it demands highly extensive computing resources for the building of the detection profile due to the large-size vectors of web pages and (2) it uses expensive genetic programming techniques.

Borgolte et al. built Meerkat [16] which is a system for the defacement detection of websites and web applications that is based on the image analysis and recognition of screenshots of web pages using computer vision techniques. Figure 4 shows Meerkat's architecture that is based on the deep neural network. The inputs to Meerkat are a list of web addresses (URL) of monitored pages. For each URL, the system first loads the web page and then takes a screenshot of the page. The page screenshots (images) are used as the inputs to the system instead of the original web pages for the defacement analysis and detection. Similar to other learning-based systems, Meerkat also has the training stage and the detection stage. In the training stage, it gathers screenshots of monitored web pages that are in normal working conditions. The training screenshots are processed to extract high level features using advanced machine learning methods, such as the stacked autoencoder and the deep neural network. The set of features of monitored pages are then stored in the detection profile. In the detection stage, the same processing procedure used in the training stage is applied to each monitored web page to create the current set of page features. The page's current feature set is compared with its feature set stored in the detection profile to find the difference. If any notable difference is found an attack alarm is fired. Meerkat was tested on a dataset of 2.5 million normal web pages and 10 million defaced web pages. The tested results show that the system achieves high detection accuracies from 97.42% to 98.81% and low false positive rates from 0.54% to 1.52%. The advantages of this proposed system are that the detection profile can be constructed from the training data and the system was experimented on a large dataset. Nevertheless, this method's main disadvantage is that it demands extensive computing resources for highly complicated image processing and recognition techniques. Moreover, Meerkat's processing may also be slow because a web page must be fully loaded and displayed in order to take its high-quality screenshot.

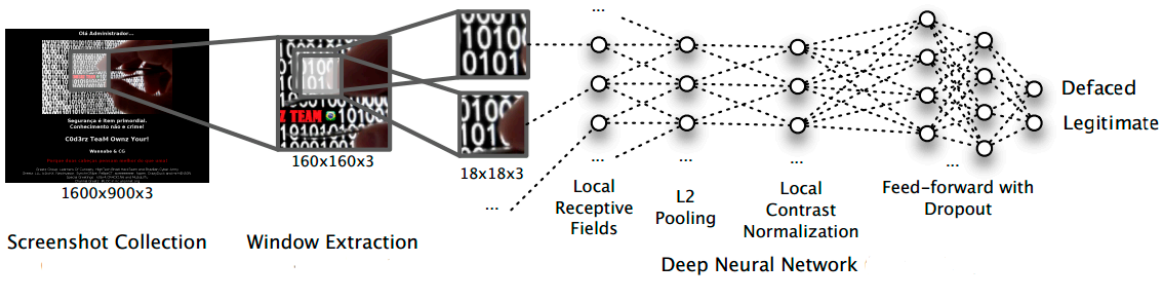

Figure 4. Meerkat's architecture based on the deep neural network [16].

2.3. Defacement Monitoring and Detection Tools

This section introduces some popular tools for website defacement monitoring and detection, which include VNCS web monitoring [6], Nagios web application monitoring software [7], Site24x7 website defacement monitoring [8] and WebOrion defacement monitor [9].

2.3.1. VNCS Web Monitoring

VNCS web monitoring [6] is a security solution of the Vietnam Cybersecurity (VNCS) that can be used to monitor websites, web portals, and web applications based on the real-time collection of web logs and the Splunk platform [17]. The solution's monitoring agents are installed on target systems to collect and transfer web logs to the central server for processing. Splunk is used for storing, indexing, searching, analyzing, and management of web logs. The main features of VNCS web monitoring consist of unified web log management and automatic analysis of web logs to detect website issues and attacks, including web page defacements, SQLi attacks, XSS attacks, and real-time site status alerts.

The solution's disadvantages are (1) its monitoring agents need to be installed on monitored systems to collect and transfer web logs, (2) its set-up and operation costs are high because it is a commercial solution, and (3) it only uses checksum and direct comparison of web page contents, which in turn may generate a high level of false alarms for websites with dynamic contents, such as e-shops and forums.

2.3.2. Nagios Web Application Monitoring Software

Nagios web application monitoring software [7] is a commercial solution for monitoring websites, web portals, and other web applications. There is a range of monitoring tools provided for different requirements of customers, such as Navgios XI which is a recently published version of the solution. Typical features of the solution include URL monitoring, HTTP status monitoring, website availability monitoring, website content monitoring, and website transaction monitoring.

The shortcomings of the Nagios solution are (1) its set-up and operation costs are high because it is a commercial tool and (2) it only uses checksum and direct comparison of web page contents, which may generate a high level of false alarms for websites with dynamic contents, such as e-stores and forums.

2.3.3. Site24x7 Website Defacement Monitoring

Site24x7 website defacement monitoring [8] is a service for monitoring and detecting website defacement attacks. This service provides the following features:

- Early identification of website security issues, including unauthorized insertion, or modification of web page's HTML elements, such as text, script, image, link, iframe, and anchor;
- Scans the entire website to find attacking links and other issues related to web page quality;
- Identifies changes in the href or src attributes of each HTML tag, that points to not-being used domains;
- Early identification of violations to security policies;
- Minimizes every effort to take control of the monitored website.

The advantage of this service is simple installation and low initial setup cost. However, the service is only suitable for static web pages and not suitable for dynamic web pages, such as e-commerce websites or forums.

2.3.4. WebOrion Defacement Monitor

WebOrion defacement monitor [9] is a solution for website defacement detection that can be provided as a service or it can be installed as software on the clients' site. The major features of WebOrion defacement monitor [9] include:

- The content analysis engine is responsible for analyzing and comparing the elements of the web pages with thresholds to detect unauthorized modifications.
- The advanced integrity validation engine is responsible for validating the integrity of the page elements using hash calculation. The final decision of the page status is determined based on an advanced decision algorithm.

- The image analysis engine converts the web page into an image and it is analyzed and compared with thresholds to detect changes.
- The monitoring is agentless which means that there is no requirement to installat monitoring agents on the target systems.
- The intelligent baseline determination means that baseline or threshold values for comparison are determined in a smart way using page analysis.
- Active warning and reporting are provided because it supports sending email and SMS alerts automatically to predefined email addresses and phone numbers when unauthorized changes are detected.

The advantage of this system is that it can monitor and detect defacements comprehensively without installing monitoring agents on the target systems. However, the option of installation as software on the clients' site requires high initial setup costs.

2.4. Comments on Current Techniques and Tools

From the survey of website defacement monitoring and detection methods and tools, some remarks can be given as follows:

- Defacement detection techniques using the comparison of checksums, or diff tools and the analysis of DOM trees can only be used effectively for static websites. Furthermore, the calculation of a suitable detection threshold for each monitored page is difficult and computationally expensive.
- Defacement detection methods based on machine learning and data mining have potential because the detection profile or the threshold can be "learned" from the training data.
- Kim et al. [13] proposes an algorithm to dynamically generate and adjust the detection threshold for each monitor page to reduce false positive alarms. However, this method only works well for web pages that have fairly stable content. For highly dynamic websites, such as e-shops or forums it is not effective.
- The common shortcoming of Medvet et al. [14], Bartoli et al. [15] and Borgolte et al. [16] is the extensive computing requirements because they use either large-size feature sets [14,15] or highly complicated algorithms [14–16]. This may restrict their implementation and deployment in practice.
- Commercial website monitoring tools like VNCS web monitoring [6], Nagios web application monitoring software [8], Site24x7 website defacement monitoring [8] and WebOrion defacement monitor [9] have two common drawbacks: (1) they are expensive because they are commercial solutions and (2) they only use checksum and direct comparison of web page contents, which may generate a high volume of false alarms on dynamic websites.

In this paper, we extend our previous work [10] by proposing a hybrid defacement detection model based on machine learning and attack signatures. In this work, we first carry out extensive experiments on a larger dataset of English and Vietnamese web pages using the machine learning-based defacement detection to verify the detection performance. Then, we combine the machine learning-based detection and the signature-based detection to build the hybrid defacement detection model. The combination of the two detection techniques aims at improving the detection rate and boosting the processing speed for common forms of defacement attacks. Our proposed defacement detection model does not require extensive computational resources because we only use low-cost supervised learning algorithms, such as Naïve Bayes, or Random Forest (RF) for the classification of web page HTM code. Furthermore, the building of the detection classifier and the attack signatures from the training data is done offline. This in turn makes the proposed detection model more efficient because it is not necessary to generate and update the dynamic detection threshold for each monitored page.

3. Defacement Detection Based on Machine Learning

3.1. The Machine Learning-Based Defacement Detection Model

In this section, we extend our previous work [10], in which the machine learning-based defacement detection model is tweaked and tested with a larger dataset. Figures 5 and 6 present the training stage and the detection stage of the model, respectively. All processing steps in both the training and detection stages are the same as those followed in [10].

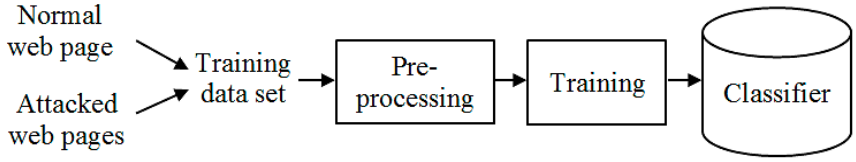

Figure 5. The machine learning-based defacement detection model: the training stage.

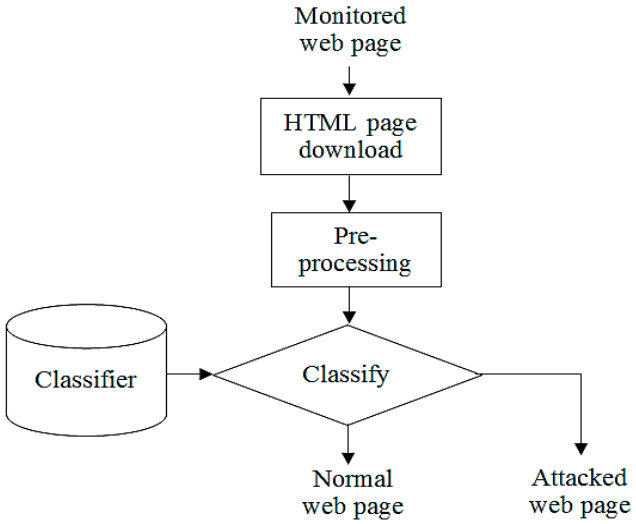

Figure 6. The machine learning-based defacement detection model: the detection stage.

3.2. Experiments and Results

3.2.1. The Dataset for Experiments

The dataset for experiments includes subsets of normal web pages and defaced web pages as follows:

- Normal web pages consist of 1200 web pages in English, which are manually gathered from many websites over the world. The normal web pages collected are from common fields, such as news sites, websites of universities and schools, online stores, real estate sites, etc. We name this subset of pages as N1;
- Normal web pages include 217 web pages in Vietnamese, which are manually collected from Vietnamese websites, such as news sites, university websites, online stores, private and government organization sites, etc. This subset is used to test the detection performance of the classifier constructed using the dataset of English web pages. We name this subset of pages as N2;
- Defaced web pages contain 1200 defaced pages, which are extracted from Defacer.ID [18] using a script. We name this subset of pages as D1.

3.2.2. Preprocessing of Dataset

The two preprocessing tasks are (1) extraction of page features using the n-gram method and (2) vectorization of these features using the term frequency (TF) method. The preprocessing is done using the same steps as those followed in [10]. However, we select 2-gram to extract page features because 2-gram is simpler and much faster than 3-gram, or higher-order n-gram.

3.2.3. Training

Two supervised machine learning algorithms, including Multinomial Naïve Bayes and Random Forest, which are supported by the Python Sklearn machine learning library, are used in the training stage to construct the classifiers. These algorithms are selected because they are relatively simple and fast. Based on the experimental results, the machine learning algorithm that produces the higher overall detection accuracy will be chosen for use in the proposed hybrid defacement detection model. In addition, the training stage can be done offline, and therefore it does not affect the processing speed of the detection stage.

3.2.4. Measurements of Experiments

The measurements used in our experiments contain PPV, FPR, TPR, FNR, ACC, and F1. PPV is positive predictive value, or precision; FPR is false positive rate; TPR is true positive rate, or recall, or sensitivity; FNR is false negative rate; ACC is the overall accuracy; and F1 is the F1 score. These measurements are computed as follows:

$$\mathrm{PPV} = \mathrm{TP}/(\mathrm{TP} + \mathrm{FP}) \times 100\% \quad (1)$$

$$\mathrm{FPR} = \mathrm{FP}/(\mathrm{FP} + \mathrm{TN}) \times 100\% \quad (2)$$

$$\mathrm{TPR} = \mathrm{TP}/(\mathrm{TP} + \mathrm{FN}) \times 100\% \quad (3)$$

$$\mathrm{FNR} = \mathrm{FN}/(\mathrm{FN} + \mathrm{TP}) \times 100\% = 100 - \mathrm{TPR} \quad (4)$$

$$\mathrm{ACC} = (\mathrm{TP} + \mathrm{TN})/(\mathrm{TP} + \mathrm{FP} + \mathrm{TN} + \mathrm{FN}) \times 100\% \quad (5)$$

$$\mathrm{F1} = 2\mathrm{TP}/(2\mathrm{TP} + \mathrm{FP} + \mathrm{FN}) \times 100\% \quad (6)$$

where TP (true positives), FP (false positives), FN (false negatives) and TN (true negatives) are described on the confusion matrix of Table 1.

Table 1. The confusion matrix of true positives (TP), false positives (FP), false negatives (FN), and true negatives (TN).

		Actual Class	
		Attacked	**Normal**
Predicted class	**Attacked**	TP	FP
	Normal	FN	TN

3.2.5. Experimental Setup and Results

We use parts of N1 and D1 subsets to form datasets for training to build classifiers and cross-validation tests. We take 100 pages from N1 and 100 pages from D1 to form dataset T1. Then, we increase the number of pages by 100 from each subset N1 and D1 to form other datasets from T2 to T12. Table 2 shows all experimental datasets and their components. For each dataset, we randomly take 75% of pages for training to build the classifier and 25% of pages for testing. The ratio between normal pages and defaced pages of training data and testing data is roughly 1:1. The final result is the average of the 10-fold cross-validation test. This experiment aims at measuring the

classification performance of machine learning algorithms and the effect of the training data amount on the classification measurements.

Table 2. Datasets for training to build classifiers and 10-fold cross-validation testing.

Dataset	Number of Pages from N1	Number of Pages from D1
T1	100	100
T2	200	200
T3	300	300
T4	400	400
T5	500	500
T6	600	600
T7	700	700
T8	800	800
T9	900	900
T10	1000	1000
T11	1100	1100
T12	1200	1200

Tables 3 and 4 show the classification performance using Multinomial Naïve Bayes and Random Forest algorithms, respectively.

Table 3. Classification performance using Multinomial Naïve Bayes.

Dataset	PPV (%)	FPR (%)	TPR (%)	FNR (%)	ACC (%)	F1 (%)
T1	96.5747	3.5814	95.6881	4.3119	96.0000	95.9954
T2	97.9902	2.0508	95.5854	4.4146	96.8000	96.7482
T3	98.3567	1.6000	96.1413	3.8587	97.2667	97.2288
T4	97.8548	2.1041	95.8800	4.1200	96.8500	96.8433
T5	98.0212	1.8963	96.1849	3.8151	97.1600	97.0829
T6	98.0310	1.9213	96.1121	3.8879	97.1000	97.0522
T7	97.7806	2.1669	96.1124	3.8876	96.9429	96.9334
T8	97.7487	2.1818	96.2953	3.7047	97.0500	97.0129
T9	97.9393	1.9655	96.9642	3.0358	97.4889	97.4454
T10	98.4216	1.5800	96.1154	3.8846	97.2400	97.2514
T11	97.4154	2.4698	95.9256	4.0744	96.7273	96.6593
T12	98.5138	1.4162	96.4528	3.5472	97.5333	97.4704

Table 4. Classification performance using Random Forest (with 50 trees).

Dataset	PPV (%)	FPR (%)	TPR (%)	FNR (%)	ACC (%)	F1 (%)
T1	95.9133	3.6742	97.3052	2.6948	96.6000	96.4839
T2	97.1385	3.0295	97.6748	2.3252	97.3000	97.3579
T3	97.3198	2.6195	97.9871	2.0129	97.6667	97.6381
T4	98.5832	1.3679	98.5746	1.4254	98.6000	98.5695
T5	98.7423	1.1475	97.9278	2.0722	98.4000	98.3262
T6	98.8561	1.1283	98.3318	1.6682	98.6000	98.5906
T7	99.3547	0.6147	98.6756	1.3244	99.0286	99.0111
T8	99.4397	0.5439	98.7407	1.2593	99.1000	99.0883
T9	99.1603	0.8489	98.9860	1.0140	99.0667	99.0715
T10	99.4483	0.5712	98.8961	1.1039	99.1600	99.1692
T11	99.4826	0.5020	98.9510	1.0490	99.2182	99.2142
T12	99.7362	0.2700	98.8197	1.1803	99.2667	99.2738

We then use the classifier trained using the dataset T12 and the Random Forest algorithm for the classification of the N2 subset to validate the detection capability for the other language web pages than English. The result is shown in Table 5.

Table 5. Detection result using classifier of T12 dataset and Random Forest for classification of normal Vietnamese web pages.

Dataset	TP	FP	FN	TN
N2 (217 Vietnamese normal pages)	0	0	0	217

3.2.6. Discussion

The experimental results shown in Tables 3–5 confirm that:

- Overall, the Random Forest algorithm outperforms the Multinomial Naïve Bayes algorithm. While Random Forest produces stable-increased detection accuracies when the amount of training data increases, Multinomial Naïve Bayes' accuracies and other measurements are not stable.
- The training of the detection model converses quickly and it does not require a lot of training data. As the result shown in Table 4, the detection accuracy (ACC and F1) becomes very stable from dataset T7 and larger datasets. This confirms that our machine-learning based model is reliable although it is trained with a small amount of data.
- Generally, the machine learning-based defacement detection based on Random Forest produces a high level of the overall accuracy (ACC and F1 above 99.0% for the datasets T7 to T12) and a low level of false positives (FPR is less than 0.62% for the datasets T7 to T12).
- The result given in Table 5 confirms that the detection model trained using English web pages can be used to monitor Vietnamese web pages for defacements. This result is promising for a language independent defacement detection model, although, more experiments may be needed to validate the multi-language defacement detection capability.

The experimental results also confirm that the proposed model based on machine learning is able to detect defacement attacks on both static and dynamic web pages because the detection profile or the classifier is trained from multiple normal and defaced web pages and it does not depend on the content of a specific web page.

4. Hybrid Defacement Detection Based on Machine Learning and Attack Signatures

4.1. The Hybrid Defacement Detection Model

Although the machine learning-based detection model described in Section 3 produces high detection accuracy, its detection stage is still relatively slow because of the feature extraction and classification of the HTML page. This may be a serious issue when there are a large number of web pages to be monitored. In order to address the processing speed issue, we propose a hybrid defacement detection model that combines the machine learning-based detection and the signature-based detection. As mentioned in Section 2, the signature-based detection is fast and efficient for known attacks, and therefore it is used to improve the processing speed for common types of known defacement attacks.

The proposed hybrid detection model includes two stages: (1) the training stage and (2) the detection stage. The training stage as depicted in Figure 7 consists of the following steps:

1. Collect the dataset for training. The training dataset is built from a set of normal web pages and a set of defaced web pages. The training dataset T12 is used as described in Section 3.
2. For web pages to be monitored, external files embedded in the web pages (such as CSS, JavaScript, and image files) are extracted and downloaded. Next, the hash value of each file content is calculated using the MD5 hashing algorithm. Then pairs of external file names and their hash values of monitored web pages are stored in the "file hash database" for change detection in the detection stage.
3. The training dataset is preprocessed for the feature extraction using the 2-gram method and is then vectorized using the term frequency (TF) method. Then, the Random Forest algorithm is used to build the classifier from the training dataset.

4. Attacked or defaced web pages are processed manually to extract common attack patterns or attack signatures.

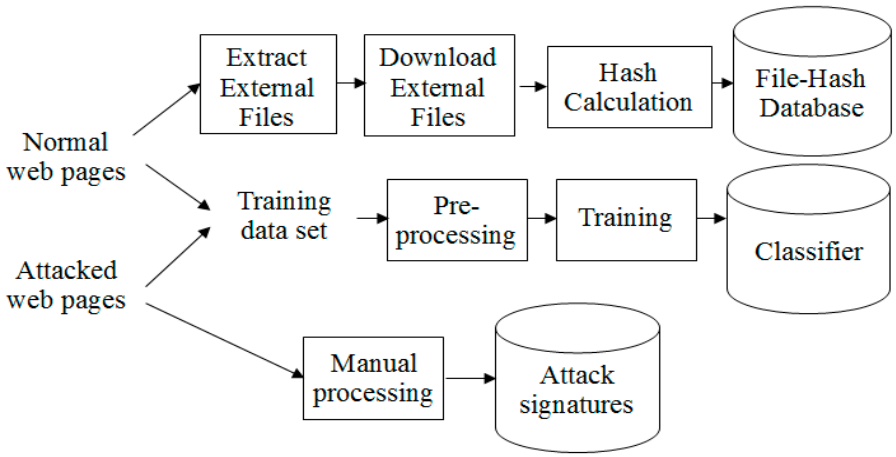

Figure 7. The hybrid defacement detection model: the training stage.

The detection stage as shown in Figure 8 consists of the following steps:

1. The HTML code of the monitored web page is downloaded from the provided URL.
2. The page's HTML code is matched against the attack signatures created in the training stage. If a signature is found, the page status is assigned as "attacked" and the detection process is completed. Otherwise, the page is forwarded to next step for further processing.
3. The HTML code is then preprocessed using the same method done for each web page in the training stage. Then, it is classified using the classifier built in the training stage. The output of the detection stage is the web page's status of either "normal" or "attacked".
4. Checking for changes in external files embedded in the monitored page. The integrity checking procedure is as following:

    a. From the HTML code of the monitored web page, the embedded external files, such as CSS, JavaScript, and image files are extracted and downloaded.
    b. Each external file name is matched against the monitored web page's external file names stored in the file hash database. If the file is found go to the next step. Otherwise a change alert is raised.
    c. The new hash value of the external file is calculated using the MD5 hashing function.
    d. The new hash value is compared against the file's original hash value stored in the "file hash database". If the two hash values are not the same (not matched), a change alert is raised.

The change alerts are manually reviewed by system administrators. If a change is authorized the new hash value of the external file is updated to the "file hash database". Otherwise, if the change is not authorized, the "attacked" status is assigned to the monitored web page.

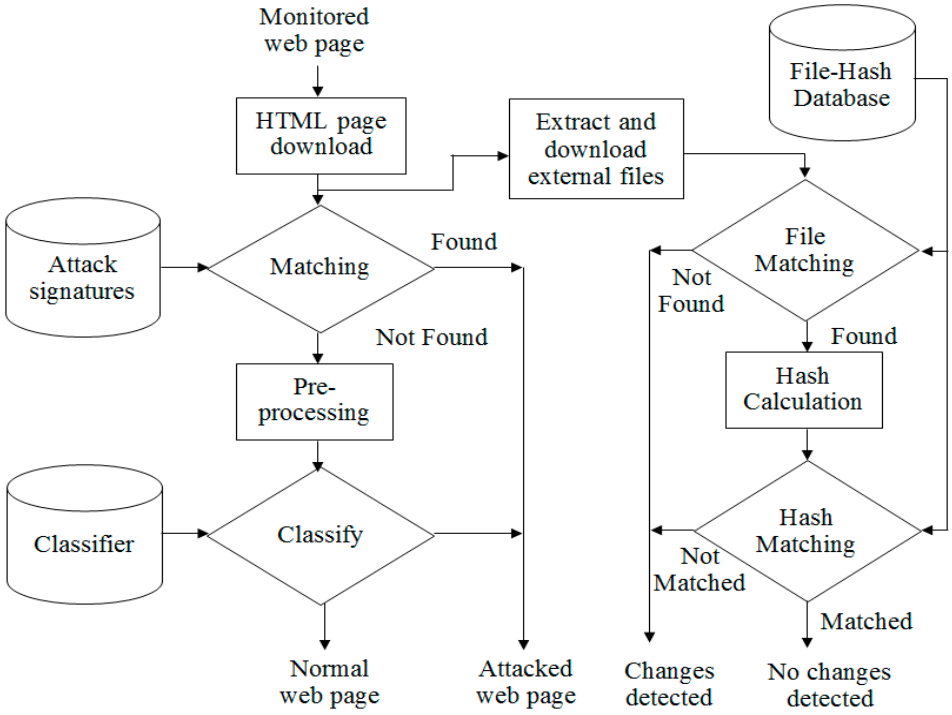

Figure 8. The hybrid defacement detection model: the detection stage.

4.2. Signature-Based Detection

4.2.1. Construction of Attack Signatures

As mentioned in Section 4.1, the attack signatures are manually extracted from defaced web pages. We manually review the HTML code of each defaced web page to find patterns that commonly appear in defaced pages to construct the list of attack signatures. The attack signatures are stored and can be updated when new defaced pages are detected. In our work, we have created an initial set of 50 attack signatures for experiments. Table 6 shows some sample attack signatures extracted from defaced web pages.

Table 6. Sample attack signatures extracted from defaced web pages.

Sample Attack Signatures
By Cyberpunks
XrillZed004
Mr.Plug1n
MrMoonz
ABD3LOS
GO1B 1D1OT
./LoliSecID
IND CYBER ARMY
Mr.Joker366
ManiAc_BD
./sT0ry_mB3m
HACKED By STUDENT'S
Cyb3rCl4y
xNot_RespondinGx
DB999ZeCs

4.2.2. Experimental Results

We test our initial set of 50 attack signatures on the N1, N2, and D1 components of the experimental dataset described in Section 3.2.1. The 50 attack signatures are used to scan for defacements in normal

pages of the N1 and the N2 sets to find the false positives. On the other hand, we gradually increase the number of attack signatures from 10 to 50 and use them to scan for defacements in defaced pages of D1 set to find the detection rate. Tables 7 and 8 show the performance of the signature-based detection.

Table 7. The performance of signature-based detection on normal pages (50 signatures).

Dataset	TP	FP	FN	TN
N1 (1200 English normal pages)	0	0	0	1200
N2 (217 Vietnamese normal pages)	0	0	0	217

Table 8. The performance of signature-based detection on defaced pages (D1, 1200 pages).

Number of Signatures	TP	Detection Rate
10	270	22.50%
20	345	28.75%
30	499	41.58%
40	534	44.50%
50	564	47.00%

Experiment results show that our signature-based detection produces no false positives on the N1 and the N2 sets and the detection rate is from 22.50% to 47.00% for our initial attack signatures from 10 to 50 signatures on the D1 set. Although the detection rate is not very high, it is used to boost the processing speed for the proposed hybrid detection model by reducing the load for the machine learning-based detection. In addition, the detection rate can be improved when more attack signatures are added.

4.3. Comparison of the Proposed Hybrid Detection Model with Previous Methods

4.3.1. General Comparison

In this section, we provide a general comparison of our hybrid detection model with works proposed by Kim et al. [13], Medvet et al. [14], Bartoli et al. [15], and Borgolte et al. [16] given in Table 9.

Table 9. Comparison of the proposed hybrid detection model with previous methods.

Comparison Factors	Kim et al. [13]	Medvet et al. [14], Bartoli et al. [15]	Borgolte et al. [16]	Proposed Hybrid Model
1. Detection methods	Anomaly	Anomaly	Anomaly	Hybrid
2. Used algorithms	Statistics, cosine distance	Genetic programming	Image recognition based on deep learning	Supervised learning and pattern matching
3. Number of page features	300 2-grams	1,466	Not applicable	300 2-grams
4. False positive rate	Not available	0.71% with GP-1466-F3	from 0.547% to 1.528%	Less than 0.62%
5. Experiment dataset	185 pages	125 normal pages and 75 defaced pages	10,053,772 defacements and 2,554,905 normal web pages	1200 normal English pages, 217 normal Vietnamese pages and 1200 defaced pages
6. Page data input	HTML code	Multiple information items of page	Page screenshot	HTML code
7. Update detection thresholds	Update frequently	No need	No need	No need
8. Work on highly dynamic pages	Ineffective	Yes	Yes	Yes
9. Major drawback	Not effective for highly dynamic pages	Computational expensive due to large feature set	Highly computational expensive due to image processing techniques	The processing of change alerts is done manually.

4.3.2. Computational Overheads

In this section, we give a qualitative comparison in terms of computational overheads of our hybrid detection model with works proposed by Kim et al. [13], Medvet et al. [14], Bartoli et al. [15], and Borgolte et al. [16]. We cannot provide a quantitative comparison in terms of computational overheads because we do not have enough information from the previous works.

Our proposed model has almost the same level of computational overheads as that of Kim et al. [13] because two methods use the same 300 2-gram features for the vectorization of web pages, and they both use relatively low-cost algorithms (refer to line two of Table 9) for training and detection stages. However, our model's major advantage over Kim et al. [13] is it can work well on static and dynamic web pages without the need to dynamically update the detection thresholds.

On the other hand, our hybrid detection model requires much less computational resources as compared with that of Medvet et al. [14], Bartoli et al. [15], and Borgolte et al. [16]. Medvet et al. [14] and Bartoli et al. [15] use a very large feature set of 1466 elements and the generic programming that is generally considered slow-convergence. Moreover, the generic programming algorithm does not guarantee finding the global maxima [19]. Similarly, Borgolte et al. [16] uses image processing and recognition techniques based on deep learning, which are always much more expensive than the text classification based on the classical supervised learning. Furthermore, they need 125 machines to collect web page screenshots and a powerful GPU for the neural network deep learning in the training stage [16].

5. Conclusions

This paper proposed a hybrid website defacement detection model that is based on machine learning techniques and attack signatures. The machine leaning-based component is able to detect defaced web pages with a high level of accuracy and the detection profile can be learned using a dataset of both normal pages and defaced pages. The signature-based component helps boost the processing speed for common forms of defaced attacks. Experimental results showed that our defacement detection model can work well on both static and dynamic web pages and that it has an overall detection accuracy of more than 99.26% and a false positive rate of lower than 0.62%. The model is also able to monitor web pages in other languages than the web page language of training data.

Although the proposed model works well on both static and dynamic web pages in different languages, the model does have some limitations. One limitation is related to the fact that we use the MD5 hashing algorithm to detect changes in embedded external files of monitored pages. Because this method is sensitive to any changes of these files, the model may generate many change alerts. The other issue with our model is the processing of change alerts is done manually and this in turn may cause some delays in the processing flow.

For future works, we will carry out more experiments to validate the independent language capability of the proposed detection model. In addition, more attack signatures will be added into the initial set. Moreover, our next task is to implement a real-time monitoring and detection system for website defacements based on the proposed hybrid defacement detection model.

Author Contributions: Conceptualization, X.D.H.; methodology, X.D.H., N.T.N.; software, N.T.N.; validation, X.D.H.; formal analysis, X.D.H.; investigation, X.D.H.; resources, X.D.H.; data curation, X.D.H., N.T.N.; writing—original draft preparation X.D.H.; writing—review and editing, X.D.H., N.T.N.; visualization, X.D.H.; supervision, X.D.H.; project administration, X.D.H.; funding acquisition, X.D.H.

Funding: This research was funded by the Ministry of Science and Technology, Vietnam grant number KC.01.05/16-20.

Acknowledgments: This work has been done in the Cybersecurity Lab, Posts and Telecommunications Institute of Technology, Hanoi, Vietnam and funded by the Ministry of Science and Technology, Vietnam grant number KC.01.05/16-20.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. DIGISTAR. What is Web Defacement Attack and Defensive Measures? Available online: https://www.digistar.vn/tan-cong-giao-dien-deface-la-gi-va-cach-khac-phuc/ (accessed on 20 January 2019).
2. Romagna, M.; van den Hout, N.J. Hacktivism and Website Defacement: Motivations, Capabilities and Potential Threats. In Proceedings of the 27th International Conference of Virus Bulletin, Madrid, Spain, 4–6 October 2017.
3. Wei, W. Rise in Website Defacement Attacks by Hackers around the World. Available online: https://thehackernews.com/2013/11/rise-in-website-defacement-attacks-by.html (accessed on 20 January 2019).
4. Banff Cyber Technologies. Best Practices to Address the Issue of Web Defacement. Available online: https://www.banffcyber.com/knowledge-base/articles/best-practices-address-issue-web-defacement/ (accessed on 20 January 2019).
5. Banff Cyber Technologies. Business Implications of Web Defacement. Available online: https://www.banffcyber.com/knowledge-base/articles/business-implications-web-defacement/ (accessed on 20 January 2019).
6. Vietnam Cyberspace Security Technology. VNCS WEB MONITORING. Available online: http://vncs.vn/en/portfolio/vncs-web-monitoring/ (accessed on 20 January 2019).
7. Nagios Enterprises, LLC. Web Application Monitoring Software with Nagios. Available online: https://www.nagios.com/solutions/web-application-monitoring/ (accessed on 20 January 2019).
8. Site24x7. Website Defacement Monitoring. Available online: https://www.site24x7.com/monitor-webpage-defacement.html (accessed on 20 January 2019).
9. Banff Cyber Technologies. WebOrion Defacement Monitor. Available online: https://www.banffcyber.com/weborion-defacement-monitor/ (accessed on 20 January 2019).
10. Hoang, X.D. A Website Defacement Detection Method based on Machine Learning. In Proceedings of the International Conference on Engineering Research and Applications (ICERA 2018), Thai-Nguyen, Vietnam, 1–2 December 2018.
11. Davanzo, G.; Medvet, E.; Bartoli, A. A Comparative Study of Anomaly Detection Techniques in Web Site Defacement Detection. In Proceedings of the IFIP TC 11 and the 23rd International Information Security Conference, Milano, Italy, 7–10 September 2008; pp. 711–716.
12. Davanzo, G.; Medvet, E.; Bartoli, A. Anomaly detection techniques for a web defacement monitoring service. *J. Expert Syst. Appl.* **2011**, *38*, 12521–12530. [CrossRef]
13. Kim, W.; Lee, J.; Park, E.; Kim, S. Advanced Mechanism for Reducing False Alarm Rate in Web Page Defacement Detection. In Proceedings of the 7th International Workshop on Information Security Applications (WISA 2006), Jeju Island, Korea, 28–30 August 2006.
14. Medvet, E.; Fillonand, C.; Bartoli, A. Detection of Web Defacements by means of Genetic Programming. In Proceedings of the Third International Symposium on Information Assurance and Security (IAS 2007), Manchester, UK, 29–31 August 2007.
15. Bartoli, A.; Davanzo, G.; Medvet, E. A Framework for Large-Scale Detection of Web Site Defacements. *ACM Trans. Internet Technol.* **2010**, *10*, 10. [CrossRef]
16. Borgolte, K.; Kruegel, C.; Vigna, G. Meerkat: Detecting Website Defacements through Image-based Object Recognition. In Proceedings of the 24th USENIX Security Symposium (USENIX Security), Washington, DC, USA, 12–14 August 2015.
17. Splunk Enterprise. Available online: https://www.splunk.com/en_us/software/splunk-enterprise.html (accessed on 20 January 2019).
18. Defacer.ID. Available online: https://defacer.id (accessed on 10 August 2018).
19. Eiben, A.E.; Smith, J.E. *Introduction to Evolutionary Computing*; Springer: Berlin/Heidelberg, Germany, 2003.