A HYBRID APPROACH FOR DETECTING MALICIOUS WEB PAGES USING DECISION TREE AND NAÏVE BAYES ALGORITHMS

Onashoga, S. A.[1], Abayomi-Alli [2], A., Idowu, O. [3], Okesola, J. O. [4]

[1,2,3] Department of Computer Science, Federal University of Agriculture, Abeokuta, Nigeria.
[1]onashogasa@funaab.edu.ng, onashogasa@gmail.com; [2] abayomiallia@funaab.edu.ng; [3] idowuoluwasanmi@gmail.com
[4] School of Computing, University of South Africa, South Africa, 48948535@mylife.unisa.ac.za

Abstract

Recent advances in computers and computer networks have made webpages a potential target for malicious activities by hackers. In this study, a hybrid malicious URL detection system using Decision Tree and Naïve Bayes was proposed and developed. The framework, which is a detection model consists of three major parts, namely: innate mechanism, feature extraction and classification modules, aims at classifying webpages as benign or malicious. PhishTank and Alexa ranking websites were collected for the URL corpus with 3000 benign and 355 malicious webpages while 12 HTML document features were extracted from each URL using JSoup Web Content feature extractor. Classification experiments conducted within the WEKA environment showed that the system respectively classified URLs as benign and malicious with 96.6% and 83.7% accuracy; while the Hybrid URL Detection Model, Decision Tree and Naïve Bayes respectively had Detection Rate (DR) of 93.1%, 83.1% and 66.1%; False Positive Rate (FPR) of 6.7%, 16.9% and 33.9%. Finally, the Ensemble Classifiers showed an Accuracy of 97.7% and 93.1% on the training and testing datasets, respectively.

Keywords: *Algorithms, Classification, Decision Trees, Machine learning, Naïve Bayes, WebPages, WEKA.*

I. Introduction

The advances in computers, computer networks and smart devices have increased the number of services that are available on the Internet with many people accessing services via web applications. A variety of these web applications provides convenient services to users, such as using online commerce, communicating through social network application and services or surfing for information online [1].

A malicious web page refers to one containing harmful content that can exploit a client-side computer system. This type of attack is termed web-based client-side attack. The attack is delivered as part of the web page itself and is designed to exploit client-side vulnerabilities such as flaws in the implementation of browser functionality, interpreters of active content within WebPages or scriptable client-side components such as HTML components. Simple modification of source codes can create new types of malicious web pages with numerous users falling victim. That is, the users visiting the vulnerable web pages can become victims of attacks [2].

Detection techniques used to defend users against malicious web pages are classified into three categories, namely: blacklisting, static analysis and dynamic analysis. Blacklisting is the most common approach among these three approaches; previously identified lists of known malicious properties such as URLs, domain names, and IP addresses can be blocked from users' access [3] [4] [5]. Blacklisting is a simple and precise detection method for identified malicious web pages; it cannot evaluate new web pages that have not been blacklisted yet, even if they are malicious.

Dynamic approach attempts to capture unusual behaviors, such as launching attacks and destroying user computers, when the page is explored in a controlled environment, such as client honey pots and virtual machines. Dynamic analysis techniques scrutinize the web-based scripts associated with web pages to detect if the page is malicious; these techniques have a high detection rate but are resource intensive [1].

In order to overcome the limitations of blacklisting and dynamic analysis, approaches on static detection techniques using machine learning to detect malicious WebPages have been proposed. By learning previously observed patterns of normal and malicious web pages, the machine can extract the static features and make predictions in real-time. In machine learning-based techniques, the detection time is faster than in dynamic analysis techniques, but the detection accuracy is lower in dynamic analysis [6].

In Section II of this paper, recent research advances in detecting malicious webpages based were reviewed. The advantages and challenges encountered in each of the techniques were examined. In Section III, the proposed methodology stating with the features selection and database overview were considered. Section IV details the components in the proposed detection framework, while Section V highlights the steps taken in the evaluation of the proposed approach. Section VI presents the study conclusions.

II. Related Work

In this section, previous studies for detecting malicious web pages were reviewed. Examples of detection techniques include blacklisting technique, signature technique, rule-based technique and machine-learning techniques.

Blacklisting was and still is a popular technique. Whittaker [7] analyzed millions of pages daily from the noisy Google's phishing blacklist. The main contribution was achieving 90% classification accuracy for phishing pages after three weeks training. Approximate pattern matching algorithm was used to match URL components against black list entries. Though the techniques tried to automatically manage blacklists and increase classification accuracy, the approach has cost limitation due to growing size and incorrect listing.

In the signature approach, detection systems use known signature to detect malicious web pages. Signatures can be from some well-known Intrusion Detection Systems (IDS) or anti-virus applications. The approach was used in malicious URLs detecting system using low interaction client honeypot. Snort signature is used to detect malicious web pages in their Honey C system [8]. The HTTP responses from web servers are constructed under XML format, and then analyzed against Sport signatures. In Monkey-Spider system [9], signature approach was used to detect malicious websites. The contents of websites are crawled and stored in files. The crawled contents are then scanned by ClamAV – an antivirus application [10].

The state-change approach is used for detecting systems using high interaction client honeypot, which is one of the efficient instruments for detecting malicious web pages. The main idea of the approach is monitoring the state change in the client system when visiting a URL in real-time. If there is any unauthorized state change during the visitation, the visited URL is classified as malicious. In [11] the Strider Honey Monkeys system, a monkey program loads a browser, instruct it to visit each URL and wait for a few minutes for downloading process. The state changes in the system are then detected against unauthorized creation of executable files or registry entries in the system [12].

To detect drive-by-download attack, [12] used event triggers. Trigger conditions were created to track unauthorized activities in process creation, file system and registry system. The trigger conditions also include any event that makes the browser or the system crash. During visitation, if a URL makes a trigger fire, it is classified as unsafe. The state change approach is also used by [13] in their client honeypot system to collect Internet-based malware. A behavior monitoring module is conducted to track malicious behavior. It hooks native Application Program Interface (API), Dynamic-Link Library functions and Total Degree of Influence (TDI) in order to monitor all

activities causing buffer overflow, accessing system resources such as process, network, file, and registry.

[6] Introduced the Prophiler that used HTML tag counts, percentage of the JavaScript code in the page, percentage of whitespace, entropy of the script, entropy of the strings declared, number of embed tags, presence of Meta refresh tags, the number of elements whose source is on an external domain and the number of characters in the page. While improving accuracy the Prophiler significantly increased the number of features to 88. In addition to the increased overhead from statistically processing the page content, this technique suffered from the inherent danger of accessing malicious pages, downloading the content before deciding whether it is malicious or not.

[8] Proposed a classification mechanism to detect malicious web pages based on analysis of HTTP responses from potential malicious web servers. The method was implemented in a hybrid system in which all URLs are classified by static heuristic method and sent to high interaction client honeypot for verification. To classify URLs by static heuristics method, some common attributes were chosen based on three proposed elements in malicious web pages: exploit delivery mechanism and obfuscation. The first step in this method is collecting malicious and benign web pages and then extracting potential attributes from these web pages. The focus of the research is making the choice of features according to the DHML knowledge usage. Four classification algorithms used in the experiments comparison are decision tree, Naïve Bayes, SVM and boosted decision tree. The result showed that the boosted decision tree got the best performance with high true positive rate and low false positive rate.

To detect malicious web pages, [14] proposed the concept of abnormal visibilities. Malicious web pages usually change their display modes in order to be invisible or almost invisible. The authors showed three main forms of abnormal visibility. The first one is changing the width and height attributes of iframe in order to make embedded malicious codes invisible or almost invisible. Setting the display style of iframe display: 'none' is the second form of abnormal visibility. The last form is generating iframe tag dynamically for obfuscation. Abnormal visibility fingerprints are created and used to detect malicious web pages. Each web page is scanned to detect any form of abnormal visibility. The detected value in any kind of abnormal visibility is compared with a threshold value. If the detected value is less than the threshold value, the web page has an abnormal visibility and is considered as a possible malicious page.

[15] Used anomaly detection and emulation to identify malicious JavaScript Code. The features were chosen based on sequence of carrying out an attack: redirection and cloaking, de-obfuscation, environment preparation, and exploitation. The study posits that not all of the features were necessary for identifying an attack and classified the features into two groups: useful features and necessary features. To extract features, they used emulated HTML browser Html Unit (Gargoyle) while experts flag suspicious web pages for further investigation.

III. Proposed Methodology

This section explains the methodology used in the design of the hybrid system for malicious WebPages detection, with background knowledge on the feature selection and data collection processes described in the subsequent subsections.

III (a) Feature Selection

The first step on feature selection is to identify potential malicious features, which can distinguish between benign web pages and malicious one. According to analysis, there are three main groups of malicious contents web pages as follows:

- Foreign contents are contents that are loaded from outside along with suspicious web pages by some malicious HTML tags such as frame, iframe, image source etc. According to [16],

iframe is especially known as very common method to load outside malicious web pages along with suspicious one.

- Exploit code contents are referred to as core contents of malicious web pages. They target specific vulnerabilities in web browsers, plug-ins or operating systems. Some of HTML tags known as delivery of potential malicious codes are applet, object, and embed. Exploit codes are encoded in scripts with obfuscations techniques to hide from detection devices in most cases [2].
- Script contents, known as the most common malicious contents of malicious web pages, are used for two main purposes: delivering and hiding malicious codes by obfuscations. Features from scripts that could distinguish between benign web pages and malicious web pages are script size, string size, word size, argument size, character distribution [2].

For this study, 12 potential features were selected from the main malicious contents. Analysis of selected features extracted from the web page is the basis for determining if a page is malicious or benign. A Java class library named JSoup [17] was used to build a feature extractor in the Java environment. The extractor receives the domain name to be classified as input; it then extracts the web content to get the specified HTML document features. This is forwarded to the detection algorithms for classification, which analyzes and stores the features in the detector database. Table 1 shows the selected features used for the hybrid malicious web pages detection system.

Table 1: Selected Features

Feature name	Number of Features	Features
HTML document features	12	1. Number of “divs” 2. The iFrame’s size 3. Number of input tag that has type attribute = “hidden” 4. Meta tags 5. Number of included URLs: href 6. Number of words 7. Number of words per line 8. Line count 9. Average word length 10. Null space count 11. Number of delimiters 12. Presence of “img” “src”

III (b) Data Collection

For this study, benign URLs and malicious URLs from Alexa ranking website [18] and PhishTank [19] were collected, respectively. A crawler with the top 3000 websites on the Alexa ranking website was seeded and then searched for malicious database to find recently discovered malicious URLs. JSoup, a Java tool for extracting web content to build a feature extractor was used to collect benign and malicious web pages. 3000 normal web pages and 355 malicious web pages were collected to verify the performance of the proposed detection method.

IV. Proposed Hybrid Detection Method

A malicious web page detection method that hybridized the Decision Tree algorithm and Naïve Bayes is proposed. The process of detecting malicious web pages is illustrated in Figure 1.

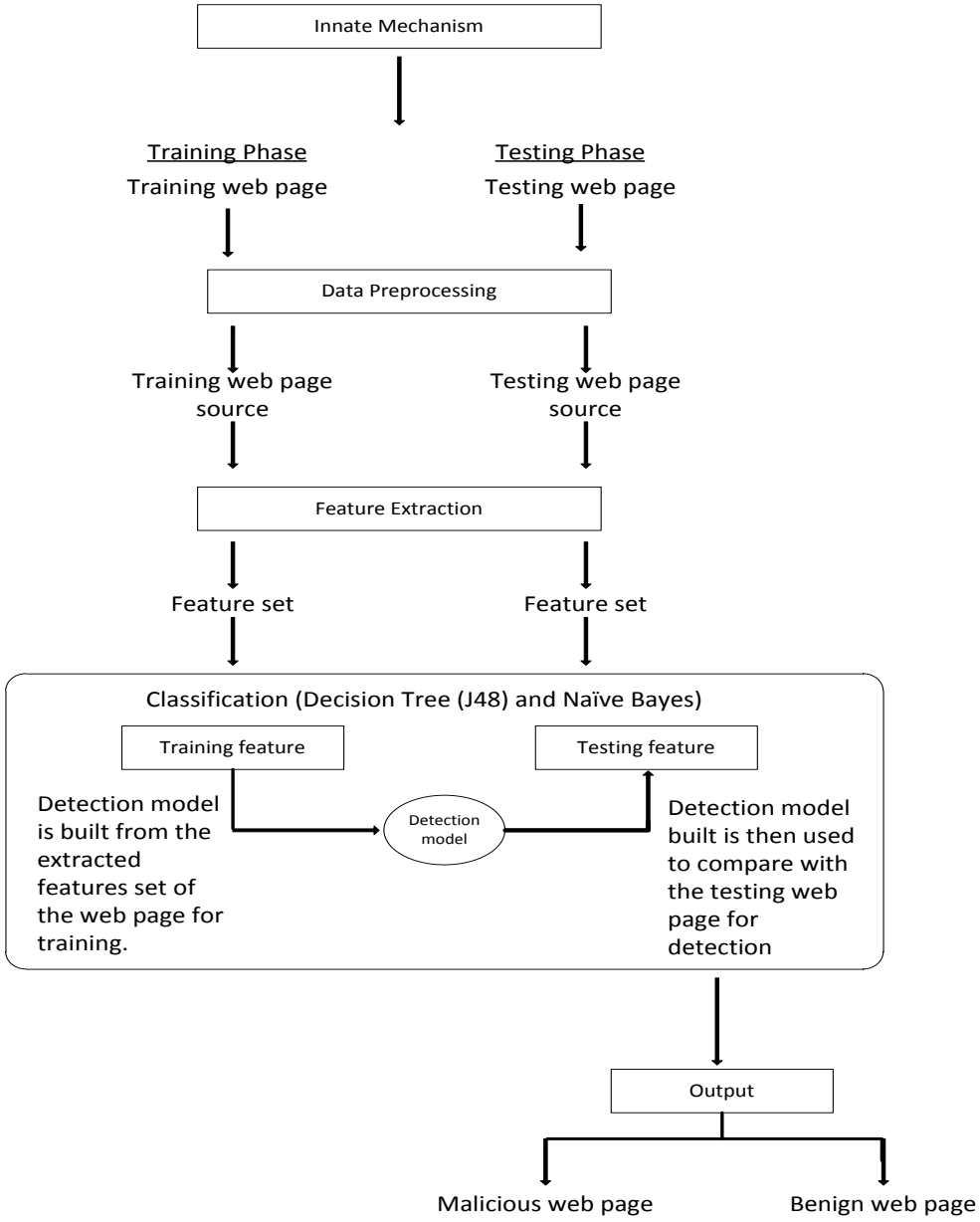

Figure 1: Framework of the hybrid malicious URL detection system

The framework consists of the following fundamental components described as:

a. Innate mechanism
b. Feature Extraction
c. Classification Phase

IV (a) Innate Mechanism

The innate mechanism module receives a domain name and tries to check with the blacklist to determine if the domain is malicious. If the domain exists in the blacklist, client request is denied to access the requested services. If the domain is resident in the whitelist, the access is granted. In a case where the domain does not exist in both blacklist and whitelist, thus the domain is forwarded to the feature extraction module.

The actions performed by the innate mechanism are listed below:

i. Deny Access to domain Ad, if (D ∈ {B}) //Malicious
ii. Grant Access to domain Ag, if (D ∈ {W}) //Benign
iii. Send domain to Feature Extraction module F, if (D ∉ {B} && {W})

Where:
D = Domain name, B = Blacklist, W =Whitelist, F = Feature Extraction module

IV(b) Feature Extraction

JSoup which is a Java class library was used to build a Feature Extractor. This extractor receives the domain name to be classified as input; it extracts the web content to get the specified HTML document features and forwards its output to the detection algorithms for classification, which analyzes the features and stores all the features in the detector database. In building this system, twelve (12) relevant features were used. The number of selected features used is listed in Table 1.

IV(c) Classification Phase

This is the final phase where the web page is categorized as either malicious or benign. The detector tries to predict the category to which a website belongs based on the extracted HTML document features. The detection engine which is the ensemble classifier that consists of Decision Tree and Naïve Bayes algorithms.

Classification using the Ensemble classifier: To detect the malicious WebPages, the ensemble classifier initially classifies the web pages using J48 algorithm, which is known as a WEKA implementation of the C4.5 algorithm [20] and Naïve Bayes, which receives the output generated by J48 and then used it to store its concept description as the prior probability of the class of the features.

The Detector Database: This is the permanent storage part of the system that stores data for both detectors. The data stored in the database is in two categories, namely;

i. Training data: Features of the known instances of Malicious and Benign domains are stored in the detector database to enable the detector to effectively detect their WebPages.
ii. Testing data: These are data that has been specifically identified for use to affirm the effectiveness of the proposed hybrid system.

V. Experiments and Results

The proposed hybrid detection method that ensembles the J48 algorithm and Naïve Bayes algorithm are evaluated. In this section, the experimental environment is described and the results are compared with the results from other detection methods.

V(a) Experimental Environment

There were 3000 benign web pages and 355 malicious web pages collected in order to verify the performance of the proposed hybrid detection method as shown on Table 2. All experiments were performed using WEKA (3.6.13) [20].

Table 2: Experimental environment

Number of benign web pages	3000
Number of malicious web pages	355
Data collection tool	JSoup
Classification tool	WEKA 3.6.6

In order to test the performance of the detection method, the detection rate and false positive rate, were recorded and compared with the conventional Decision tree and Naïve Bayes. The decision rate is the rate of detected malicious web pages from the total number of malicious web pages; the false positive rate is the rate of web pages misclassified as malicious from the total number of normal web pages. The proposed hybrid method uses the approach that focuses on achieving high detection rates.

V (b) Experimental Results

The experimental results are discussed here. The purpose of the proposed method is to detect malicious web pages while achieving a high detection rate. In order to increase the detection rate, a hybrid detection method that combines the Decision Tree and Naïve Bayes algorithm were introduced. The experimental results of the classification rate are presented in Table 3.

Table 3: Malicious web pages classification results

	Classified Malicious (%)	Classified Benign (%)
Real Benign	0.036	0.964
Real Malicious	0.837	0.163

Table 4 shows the comparison of the hybrid method with the Decision tree only and Naïve Bayes only. Decision tree had a detection rate (DR) of 83% and a false positive rate (FPR) of 1.3% while Naïve Bayes had DR of 66% and a FPR of 34% approximately. It was observed that the proposed hybrid approach classified malicious web pages accurately than Decision trees and Naïve Bayes algorithms, respectively.

Table 4: Comparison of the proposed method results with the single J48 algorithm and single Naïve Bayes algorithm

	Proposed method	Decision tree	Naïve Bayes
DR	93.1%	83.1%	66.1%
FPR	6.9%	16.9%	33.9%

In Table 5, the ensemble classifiers show the correctly classified instances as 551 with accuracy of 97.965% and incorrectly classified instances as 13 with accuracy of 2.305% for the training dataset. The true positive rate (TPR) = 0.977 while FPR = 0.043. The Kappa statistic = 0.9448 showing a high statistical dependence. The ensemble classifiers showed the correctly classified instances as 355 with accuracy =94.4149% and incorrectly classified instances as 21 with accuracy = 5.5851% for the testing dataset. The TPR = 0.944 while FPR = 0.11. The Kappa statistic is 0.8537 showing a high statistical dependence.

Table 5: Performance evaluation of the proposed system

Dataset	Accuracy (%)	TPR	FPR	Precision	Recall	F-Measure	ROC
Training	97.695	0.977	0.043	0.977	0.977	0.977	0.971
Testing	93.122	0.944	0.110	0.944	0.944	0.943	0.934

VI. Conclusion

As deployment and usage of web-based services go on the rise, hackers also attack computer users by injecting codes into their web pages, usually for malicious purposes such as information theft. This paper presents a detection method for the classification of malicious web pages with a

hybrid method based on machine learning algorithms. The proposed hybrid approach is composed of Decision Tree and Naïve Bayes algorithm to detect both known and unknown malicious web pages.

The experimental results showed that the proposed hybrid method exhibited a significantly improved detection of 98%. However, the proposed method had a relatively low false rate of 4.3%. Future work will be on application of the hybrid detection method in Intrusion Detection System.

References

1. Suyeon, Y. and Sehun, K. 2014. Two-Phase Malicious Web Page Detection Scheme Using Misuse and Anomaly Detection, International Journal of Reliable Information and Assurance, 2014, 2(1): 23.
2. Van L., Ian W., Xiaoying G., Peter K. 2011. Identification of Potential Malicious Web Pages, Proceedings of the International Conference on Web Information Systems and Mining, Shanghai, China, 2011, (2):150.
3. Eshete, B., Adolfo, V., and Komminist, W. 2011. Malicious Website Detection: Effectiveness and Efficiency Issues. 1st IEEE SysSec Workshop (SysSec), 6th July, 2011. Amsterdam, pp. 123-126.
4. Felegyhazi, M., Christian K., and Vern, P. 2010. On the potential of proactive domain blacklisting. Proceedings of the 3rd USENIX conference on Large-scale exploits and emergent threats: botnets, spyware, worms, and more. USENIX Association Berkeley, CA, USA, 3(32),1-8.
5. Bannur, S., Sushma, N., Lawrence, K., and Stefan, S. 2011. Judging a site by its content: learning the textual, structural, and visual features of malicious web pages. Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence (AISec' 11), 2011, October 21, 2011, Chicago, Illinois, USA. 4:67-75.
6. Canali, U. and Davide, T. 2011. Prophiler: a fast filter for the large-scale detection of malicious web pages. Proceedings of the 20th international conference on World Wide Web, 2011. ACM 20: 44.
7. C. Whittaker, B. Ryner, and M. Nazif, 2010. Large-Scale Automatic Classification of Phishing Pages, In Proceedings of the 17thAnnual Network and Distributed System Security Symposium (NDSS'10), San Diego, CA, Mar 2010, Vol 17, pg 231.
8. Seifert, C., Welch, I. and Komisarczuk, P. 2006. HoneyC - The Low-Interaction Client Honeypot. Proc. NZCSRSC, 2011. Hamilton. pp. 1-9
9. Stuart E. S., Rachna, D., Andy, O., and Ian, F. 2007. The emperor's new security indicators". SP '07: Proceedings of the 2007 IEEE Symposium on Security and Privacy, 2007, 7:51-65.
10. Ikinci, A., Holz, T. and Freiling, F. 2008. Monkey Spider: Detecting Malicious Websites with Low Interaction Honey clients. Proc. Sicherheit, Saarbruecken. pp 1-15.
11. Y.-M. Wang, D. Beck, X. Jiang and R. Roussev, 2006. Automated WebPatrol with Strider Honey Monkeys: Finding Web Sites that Exploit Browser Vulnerabilities, In NDSS, 2006.
12. A. Moshchuk, T. Bragin, D. Deville, S. Gribble, and H. Levy. 2007. SpyProxy: Execution-based Detection of Malicious Web Content. In Proceedings of the USENIX Security Symposium, 2007, 4:19.
13. Xiaoyan, S., Yang, W., Jie, R., Yuefei, Z. and Shengli, L. 2008. Collecting Internet Malware Based on Clientside Honeypot". Proc. the 9th International Conference for Young Computer Scientists. ICYCS 2008. pp. 1493-1498.

14. Bin, L., Jianjun, H., Fang, L., Dawei, W., Daxiang, D. and Zhaohui, L. 2009. Malicious Web Pages Detection Based on Abnormal Visibility Recognition". Proc. International Conference on EBusiness and Information System Security (EBISS '09) 2009. 1-5.
15. Cova, M., Kruegel, C. and Vigna, G. 2010. Detection and Analysis of Drive-by-Download Attacks and Malicious JavaScript Code. Proc. WWW2010, Raleigh NC, USA.
16. Provos, N., Mavrommatis, P., Abu, M.and Monrose, R. F. 2008. All your iframes point to us. Google Inc, 2008.
17. JSoup. Available online at Available: http://www.jsoup.org, Accessed July, 2015
18. Alexa. Available online at http://www.alexa.com, Accessed August, 2015.
19. PhishTank. Available online at http://www.phishtank.com, Accessed August, 2015.
20. WEKA. Available online at http://www.cs.waikato.ac.nz/ml/weka/, Accessed August, 2015.

Article received: 2016-02-08