A Comparison of Machine Learning Attributes for Detecting Malicious Websites

A K Singh
dept. of CSIS
BITS Pilani
Pilani, India
aksingh2411@gmail.com

Navneet Goyal
dept. of CSIS
BITS Pilani
Pilani, India
goel@pilani.bits.ac.in

***Abstract*—The number of Malicious Websites has increased manifold in the past few years. As on start of year 2018, 1 in every 13 URL was malicious, amounting to 7.8% URLs identified as malicious [1]. These figures have increased by 2.8%, thereby showing an increasing trend of attack vectors through Malicious Websites. These statistics clearly highlight the need to detect Malicious Websites on the Internet. Many research works have suggested Machine Learning techniques to detect Malicious Websites. Research has also been done to compare Machine Learning algorithms for their detection. However, the aspect of attribute selection for detecting Malicious Websites using Machine Learning has not been delved in detail. In Machine Learning techniques, attribute selection outweighs the importance of any other aspect in the process. Thus, there is a need to compare and analyze the various attributes that can help find Malicious Websites faster and better. This paper is focused to address this research gap, so that, fewer and optimal attributes can do a better job.**

***Index Terms*—Data Mining, Web Mining, Malicious Websites, Machine Learning.**

I. Introduction

MANY researchers have proposed the use of machine learning for detecting malicious web sites. The most important facet in machine learning techniques is the identification of right attributes that can be used for classification of malicious websites. A right selection of attributes ensures better classification accuracy with minimum computational resources.

This paper seeks to identify Attributes that can classify malicious websites with best accuracy and with least computational resources. A typical machine learning process to detect malicious websites is shown schematically in Figure 1. The steps involved in such a process are- crawling , attribute extraction from crawled websites and processing, classifier training using a training dataset[1] and then finally running the classifier on the test dataset to detect malicious websites.

In this paper, twenty five various possible attributes that are used for Malicious Website detection have been considered. These attributes have been discussed with respect to classification accuracy and computational requirements (during extraction and pre-processing). Based on this analysis, the most suitable attributes are recommended.

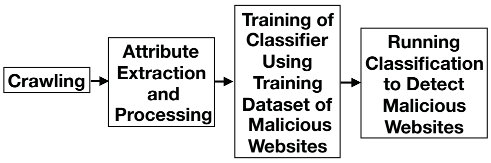
Crawling
Attribute Extraction and Processing
Training of Classifier Using Training Dataset of Malicious Websites
Running Classification to Detect Malicious Websites

Fig. 1: Malicious Website Detection with Machine Learning.

Various tools and platforms have been used as part of this research. MalCrawler [2] has been used to crawl websites. Extraction and pre-processing has been done using HTML Unit Browser Emulator [3] and custom written Java code. Rhino JavaScript Emulator [4] has been used for running the JavaScript code in a sandboxed environment. Machine Learning tasks have been carried out using WEKA tools [5].

II. Related Work

Significant research has been carried out on Malicious Website detection using Machine Learning Techniques. However, these papers have restricted themselves to few attributes for machine learning. Ma et al. in their paper on detecting malicious websites from suspicious URLs, have considered only URL related attributes [6]. While, Wressnegger et al. have discussed Flash based malwares [7], Mavrommatis et al. have discussed iFrames [8], Ganesh et al. have analyzed Java Applet based malwares [9], Mao et al. have discussed HTML content based analsysis of malicious websites [10], Cova et al. have analyzed few JavaScript based attributes [11] and Gorji et al. have discussed obfuscated JavaScript code based attacks [12]. Thus, we find that a holistic analysis of all possible attributes that are required for Malicious Website detection is lacking. An effort has been made in this paper to address this gap.

In this paper, apart from analyzing the attributes, we will also rank them based on two critera. Firstly, with respect to their classification accuracy in malicious website detection. Secondly,with respect to their requirement of computational resources during extraction and processing. Attribute selection is an important aspect, which improves classification results by reducing both overfitting and underfitting. Hall et al.

[1] Training Dataset is prepared from a known list of malicious websites.

978-1-5386-7902-9/19/$31.00 ©2019 IEEE

have discussed few attribute selection techniques generally applicable to all data mining processes [13]. In this paper we have used two techniques[2] of attribute selection for machine learning in order to ensure better selection.

III. Attributes Considered

Various attributes that can be used to detect malicious websites, were considered. Amongst the list of all possible attributes, twenty five suitable attributes were selected for further analysis. The attributes considered for detecting malicious websites are shown below in Table I.

TABLE I: List of Attributes Considered

No	Attribute Category	Attribute Name	Attribute Description
A1	Location of Website	Geographical Location	IP address is used to find geographical location of the web server using the Geo IP [14] database.
A2	URL Properties	URL	Keywords extracted from URL are analyzed.
A3	-do-	HTTPS Enabled Website	Whether the website uses the secure HTTPS or unsecured HTTP.
A4	-do-	Domain Name	Keywords are extracted from domain name and analyzed.
A5	-do-	DNS WHOIS Information	It is checked whether the DNS record of website exists.
A6	-do-	Redirection from the website	It is checked whether the website redirects or not.
A7	Website Behavior	Cloaking	It is checked whether website is cloaking or not.
A8	Web Page Semantics (HTML and Content)	Presence of an iFrame on the webpage	The presence of iFrame is checked.
A9	-do-	The Number of Java Applet Tags	The number of these tags are recorded.
A10	-do-	The Number of Flash Script Tags	-do-
A11	-do-	The Top 10 keywords of the Website	These keywords are analyzed against set of keywords generally found on malicious websites.
A12	-do-	Meta Tag Values	Keywords are extracted from Meta Tags and compared as in A11.
A13	-do-	HTML Title Tag Values	Keywords are extracted from HTML and compared as in A11.
A14	Java Script Behavior and Semantics	Redirection of URL using document. location() function	Presence of redirection using JavaScript is checked.
A15	-do-	XML HTTP Request (XHR)	It is checked if XHR is meant for the same or different domain.
A16	-do-	Communication with Flash components	It is checked if the Flash component establishes communication with the browser.
A17	-do-	Communication with Java Applet Components	It is checked if the Java Applet component establishes communication with the browser.
A18	-do-	URL in XHR	It is checked whether the URL in XHR is encoded or not.
A19	JavaScript Code Content	Presence of the eval() function	Presence of JavaScript eval() function is checked.
A20	-do-	Presence of the unescape() function	Presence of JavaScript unescape() function is checked.
A21	-do-	Presence of Popups using Window.open() function	Checks the presence of Window.open() function.

[2] 'Ten Fold Cross Validation' and 'Gain Ratio Method' have been the two methods of attribute selection that have been used in this paper. They have been discussed in detail in later sections.

No	Attribute Category	Attribute Name	Attribute Description
A22	-do-	Presence of the find() function	Checks the presence of JavaScript find() function.
A23	-do-	Presence of obfuscated code	Obfuscated code is the Encrypted JavaScript code. The presence of such code is checked.
A24	-do-	Size of the JavaScript Code	Size of JavaScript code length is recorded.
A25	-do-	Size of Obfuscated Code	Size of Obfuscated JavaScript code is recorded.

IV. Attribute Extraction and Pre-Processing

The details of the attributes listed in the previous section, along with the details of preprocessing and extracting these attributes in given in succeeding paragraphs. MalCrawler [2] has been used for crawling websites in this paper. Wherever a known list of malicious sites was required, the malware domain list online was utilized [14]. For building up a training dataset of malicious website detection, the Google Safe Browsing API was used [15].

A. Location Region- Based on IP Address

The IP address of the website is detected and recorded by the crawler. Geographical location is then determined from the IP address using the GeoIP database [16]. The geographical location is stored as country name, a nominal type attribute.

B. URL

The URL is extracted by the crawler. A Bag-of-words representation of the URL [6] is generated and then compared with the top 100 words found in URLs of malicious websites. The number of the matches is then stored as a numerical type attribute.

C. HTTPS Enabled Website

HTTPS is more secure than HTTP and websites running on HTTPS are less likely to host malicious content [17]. Therefore, as part of this research we have crawled to check whether the website is using HTTP or HTTPS. This value is then stored as a boolean attribute.

D. Domain Name

The Domain Name is known to have been used as an attribute in Detection of Malicious Websites using Machine Learning [18] [19]. Keywords are extracted from the Domain Name and are then compared with few common keywords that are linked to malicious behaviour. The number of matches is then stored as a numerical type attribute.

E. DNS WHOIS Information

The DNS WHOIS information provides the registration details of the website. Presence and absence of fields (for example address of domain owner) in DNS WHOIS information has been found to be linked to maliciousness [19]. The presence and absence of this information is stored as a boolean attribute.

F. Redirection from Website

Redirection has often been linked to malicious behaviour [20]. In this paper the MalCrawler [2] has been used for redirection detection and the detection results are stored as a boolean attribute.

G. Cloaking

Cloaking is the phenomenon in which the website shows different pages based on the client platform. Malcrawler [2] is used to detect actions of cloaking by a website by serving different HTTP requests having different user-agent fields in HTTP header. Presence of cloaking has been found to be linked to maliciousness [21] [22]. The presence and absence of cloaking is saved as a boolean attribute.

H. Presence of iFrame on Web Page

The iFrame HTML tag is known to have been utilized to download malicious JavaScript exploits [8] in numerous attack vectors. In this paper, the presence and absence of iFrame tag has been checked by parsing[3] websites. The presence or absence is stored as a boolean attribute.

I. Number of Java Applet Tags

Malicious exploits through Java Applets are known to have been reported on many websites [7]. Thus, its presence is recorded by parsing the website and is stored as a boolean attribute.

J. Number of Flash Script Tags

Just like Java Applets, numerous cases of exploit injection through Flash Scripts have been reported [8] [23]. Thus, its presence is recorded by parsing the website. The value is stored as a boolean attribute.

K. Top 10 Keywords of the Website

The website is parsed to extract the Term Frequency - Inverse Document Frequency (TF-IDF)[4]. The TF-IDF of this website is compared against the TF-IDF [11] [24] of know malicious websites. The match of top 10 TF-IDF keywords of the website against the TF-IDF of malicious websites is saved as a numerical value attribute.

[3] Parsing is the process of reading the complete HTML content, including the JavaScripts. As part of this research, it has been done using customized libraries in Java that have been used in conjunction with MalCrawler [2]

[4] TF-IDF is a statistical method of showing the importance of a word in a document

L. Meta Tag Values

The TF-IDF of Meta Tag is computed separately. This TF-IDF is compared against the TF-IDF of known malicious websites [24]. The value of this match is saved as numerical value attribute.

M. HTML Title Tag Values

The Bag-of-words is extracted from the HTML title tag. It is compared against the Bag-of-words of malicious websites [24]. The match of this comparison is saved as a numerical attribute.

N. Redirection Using document.location()

The presence of JavaScript function document.location() is checked by parsing the website. This function has been found to be linked to malicious redirects [12]. The presence or absence of this function is saved as a boolean attribute.

O. XML HTTP Request (XHR)

XHR is the core of AJAX [5] technology. However, XHR can be used to inject exploits [25]. The website is parsed to take a count of number of XHR instances. This value is saved as a numerical attribute.

P. Communication with Flash Components

Flash components communicating with the browser are indicative of an exploitative behavior [23]. This feature is checked using the HTML Unit Browser Emulator [3]. The presence or absence of such a behavior is stored as a boolean attribute.

Q. Communication with Java Applets

Like the Flash component, the communication of Applet with the browser can be used to inject exploits [7]. This is also checked using the HTML Unit Browser Emulator. This behavior is stored as a boolean value.

R. URL in XHR

A URL in XHR outside the domain is an indicator of malicious behavior [25]. This can be analyzed using the HTML Unit Browser Emulator. The presence or absence of this behavior is recorded as a boolean value.

S. Presence of eval() Function

Malicious websites use eval() function to generate malicious code at runtime to thwart detection [12]. The JavaScript code on the website is parsed to detect the eval() function. The number of eval() functions in JavaScript code is stored as a numerical attribute.

T. Presence of unescape() Function

Hackers generally encode malicious code and use the unescape() function to decode it. Thus, the number of unescape() function calls in JavaScript is a strong indicator of malicious activity [26]. For our Machine Learning analysis, the number of unescape() functions in JavaScript code is stored as a numerical attribute.

U. Presence of Windows.open() Popups

The JavaScript Windows.open() Popups are used for ads and also to inject exploits [27]. For our Machine Learning analysis, the presence or absence of this Popup is stored as a boolean value.

V. Presence of find() Function

The find() JavaScript is used along with unescape() and eval() to decrypt malicious code at runtime [26]. Thus, the occurrences of find() function in JavaScript code is noted and stored as a numerical attribute.

W. Presence of Obfuscated Code

Obfuscated code is the encrypted JavaScript code. Generally, obfuscation is done to thwart detection of malicious code [28]. Thus, its presence is a strong indicator of malicious activity. The presence or absence of obfuscated code is recorded as a boolean attribute.

X. Size of JavaScript Code

Generally, malicious JavaScript code have been found to be of relatively larger sizes [28]. Thus, the size of JavaScript code is a good indicator for maliciousness. In our research, we have used it as a numerical attribute that has a value equal to the size of JavaScript code in Kilo Bytes (KB).

Y. Size of Obfuscated Code

Large obfuscated code indicates the presence of an exploit [28]. Thus, the size of obfuscated code (in KB) is captured and stored as a numerical value for Machine Learning.

V. Classification - Predicting Malicious Websites

Classification as part of this research was carried out using two algorithms - C4.5 and Naive Bayes. These two algorithms were chosen as they represent two different approaches to classification. While C4.5 uses a decision tree, Naive Bayes uses a probabilistic learner. The WEKA [5] data mining software was used for training and running the classifiers. The training dataset was built from a known list of malicious and benign websites using the Malware Domain List [14] and Google Safe Browsing API [15]. These malicious and benign websites were crawled, parsed and processed using MalCrawler [2] and customized Java Code in order to create this training datset. A copy of this dataset that has been created as part of this research has been published online [29]. For checking the attribute predictability two techniques were used. Firstly, as we are looking to rank the attributes based on their ability to predict malicious websites, we used the Gain Ratio method [13] of attribute selection. In this method each attribute

[5] AJAX (Asynchronous JavaScript and XML) is a technology to create asynchronous web applications.

Ai is assigned score based on the information gain between itself and the class. If C is the class and A is the attribute, equations (1) and (2) below give the Entropy H before and after observing the attribute.

$$H(C) = -\sum_{c\epsilon C} p(c) log_2 p(c) \tag{1}$$

$$H(C/A) = -\sum_{a\epsilon A} p(a) \sum_{c\epsilon C} p(c/a) log_2 p(c/a) \tag{2}$$

Secondly, Ten Fold Cross-validation was run, one attribute at a time, to assess the ability of each attribute to predict malicious websites. The Confusion Matrix produced by this Ten Fold Cross-validation is given in Table II. The table elucidates the ability of an attribute to predict malicious websites.

TABLE II: RESULTS OBTAINED DURING CLASSIFICATION

Attribute No	Naive Bayes Classifier				C 4.5 Classifiers			
	TN	FN	FP	TP	TN	FN	FP	TP
A1	11%	12%	10%	67%	12%	14%	9%	65%
A2	10%	12%	13%	75%	7%	6%	7%	80%
A3	10%	9%	4%	77%	7%	8%	6%	79%
A4	3%	6%	4%	87%	6%	5%	4%	85%
A5	8%	7%	9%	76%	6%	8%	9%	77%
A6	3%	3%	5%	89%	3%	4%	5%	88%
A7	2%	4%	3%	91%	2%	3%	3%	92%
A8	3%	2%	3%	92%	3%	2%	2%	93%
A9	10%	12%	13%	65%	16%	14%	14%	56%
A10	11%	10%	12%	67%	15%	7%	13%	65%
A11	2%	3%	9%	86%	4%	3%	9%	84%
A12	2%	18%	3%	77%	5%	17%	3%	75%
A13	5%	6%	8%	81%	4%	14%	4%	78%
A14	5%	6%	4%	85%	9%	3%	6%	82%
A15	7%	8%	7%	78%	8%	8%	7%	77%
A16	10%	12%	11%	67%	12%	14%	11%	63%
A17	20%	18%	15%	47%	17%	17%	10%	56%
A18	9%	11%	5%	75%	8%	14%	7%	71%
A19	8%	6%	10%	76%	9%	6%	6%	79%
A20	6%	8%	6%	80%	6%	2%	10%	82%
A21	1%	2%	3%	94%	2%	3%	2%	93%
A22	7%	9%	8%	76%	8%	10%	8%	74%
A23	2%	4%	3%	89%	4%	6%	7%	83%
A24	6%	7%	6%	81%	5%	8%	5%	82%
A25	1%	1%	2%	96%	2%	1%	3%	94%

VI. Analysis of Results Obtained

The results obtained in the previous section have been analysed in succeeding paragraphs to find the most suitable set of attributes for detecting Malicious Websites.

A. *Classification Accuracy of Attribute*

The comparison of the 25 attributes considered for detection of malicious website using machine learning is shown graphically in Figure 2. The bar graph shows the classification accuracy for each single attribute, running Ten Fold Cross-validation, using the two classification algorithms- C4.5 and Naive Bayes.

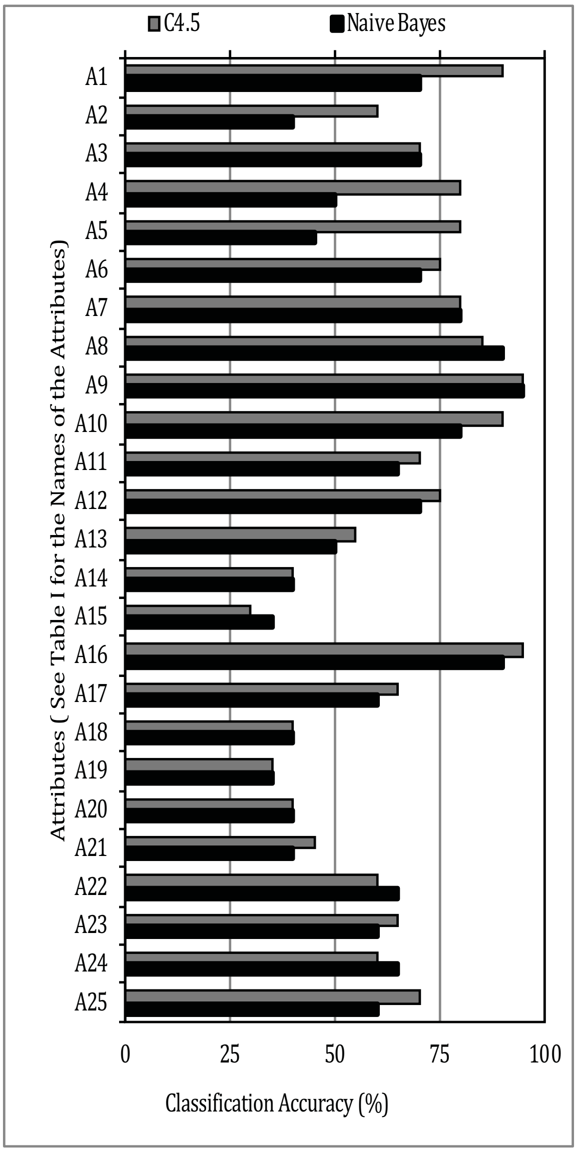

Fig. 2: Classification Accuracy of Attributes.

B. *Computational Resources Used*

The computational resources (memory and CPU cycles) utilized for attribute extraction and pre-processing is an important factor to rank the attribute. The computational resources used by attributes were assessed using the Netbeans Profiler[6], when running the extraction and pre-processing java code. The values obtained were normalized to show on a scale of 0 to 1. The result obtained is shown as a chart in Figure 3.

[6] Netbeans is a software development and testing platform for Java. Netbeans Profiler was used to measure the CPU cycles and memory utilization while running the Java code for Attribute extraction and Pre-processing.

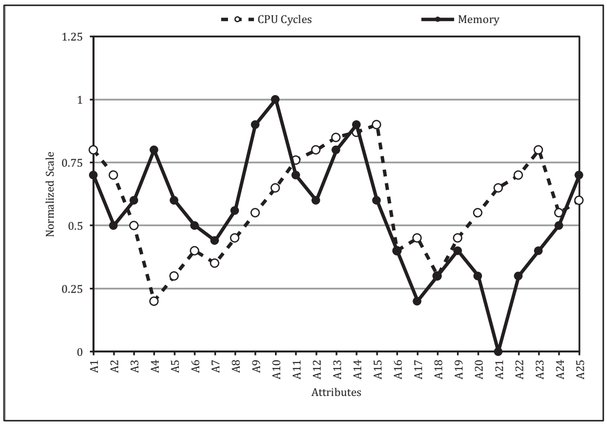

Fig. 3: Computation Resources Used by Attributes.

C. The Top 5 Attributes

Based on the experiments conducted above, the top 5 attributes to predict malicious websites based on the classification accuracy and the requirement of computational resources are - A7 (Cloaking), A8 (Presence of iFrame), A6 (Redirection from Website), A25 (Size of Obfuscated Code) and A21 (Popups using Window.open() Function). The 10 fold cross-validation result using these five attributes alone is given in Table III.

TABLE III: Confusion Matrix of Classification (Top 5 Attributes)

	Predicted Non-malicious	**Predicted Malicious**
Non-malicious	TN = 2%	FP = 4%
Malicious	FN = 3%	TP = 91%

VII. Conclusion

The paper tried to compare Machine Learning Attributes for detecting malicious websites. A total of 25 attributes were considered, which are generally used to detect malicious websites. These attributes were analysed with respect to the computational resources required for extraction and processing, and also, the classification accuracy in predicting malicious websites. Based on the analysis, top 5 attributes were identified for detecting malicious websites. The paper made a unique effort to analyse holistically the attributes required for detecting a malicious website through a Machine Learning process. Previous studies related to detection of malicious websites using Machine Learning had considered only a few attributes, and had not compared the relative importance of using an attribute with respect to others. The comparative and analytical model discussed in this paper may also be used by researchers to carryout other forms of malware analysis that use machine learning.

References

[1] Symantec, “ISTR Internet Security Threat Report Volume 23,” Symantec, Tech. Rep., 2018. [Online]. Available: https://www.symantec.com/content/dam/symantec/docs/reports/istr-23-2018-en.pdf.

[2] A. K. Singh and N. Goyal, “Malcrawler: A crawler for seeking and crawling malicious websites,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2017.

[3] “HTML Unit Browser Emulator,” 2018. [Online]. Available: http://htmlunit.sourceforge.net/.

[4] “Rhino-Mozilla,” 2018. [Online]. Available: https://developer.mozilla.org/en-US/docs/Mozilla/Projects/Rhino.

[5] “WEKA 3- Open Source Data Mining and Machine Learning Software,” 2018. [Online]. Available: https://www.cs.waikato.ac.nz/ml/weka/.

[6] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, “Beyond Blacklists : Learning to Detect Malicious Web Sites from Suspicious URLs,” *World Wide Web Internet And Web Information Systems*, 2009.

[7] C. Wressnegger, F. Yamaguchi, D. Arp, and K. Rieck, “— Technical Report — Analyzing and Detecting Flash-based Malware using Lightweight Multi-Path Exploration,” *University of Göttingen, Germany*, no. December, 2015. [Online]. Available: http://christian.wressnegger.info/content/projects/gordon/2015-tr.pdf.

[8] P. Mavrommatis and N. Provos, “All Your iFRAMEs Point to Us,” *Symposium A Quarterly Journal In Modern Foreign Literatures*, 2008.

[9] N. Ganesh, F. Di Troia, V. A. Corrado, T. H. Austin, and M. Stamp, “Static Analysis of Malicious Java Applets,” *Proceedings of the 2016 ACM on International Workshop on Security And Privacy Analytics*, 2016.

[10] J. Mao, W. Tian, P. Li, T. Wei, and Z. Liang, “Phishing-Alarm: Robust and Efficient Phishing Detection via Page Component Similarity,” *IEEE Access*, vol. 5, pp. 17 020–17 030, 2017. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8015116/.

[11] M. Cova, C. Kruegel, and G. Vigna, “Detection and analysis of drive-by-download attacks and malicious JavaScript code,” in *Proceedings of the 19th international conference on World wide web - WWW ’10*, 2010.

[12] A. Gorji and M. Abadi, “Detecting Obfuscated JavaScript Malware Using Sequences of Internal Function Calls,” in *Proceedings of the 2014 ACM Southeast Regional Conference on - ACM SE ’14*, 2014.

[13] M. A. Hall and G. Holmes, “Benchmarking Attribute Selection Techniques for Discrete Class Data Mining,” *IEEE Transactions on Knowledge and Data Engineering*, 2003.

[14] “Malware Domain List,” 2018. [Online]. Available: https://www.malwaredomainlist.com/.

[15] “Google Safe Browsing,” 2018. [Online]. Available: https://safebrowsing.google.com.

[16] “GeoIP Database - GeoLite 2,” 2018. [Online]. Available: https://www.maxmind.com/en/geoip2-databases.

[17] A. P. Felt, R. Barnes, A. King, C. Palmer, C. Bentzel, and P. Tabriz, “Measuring https adoption on the web,” in *26th USENIX Security Symposium*, 2017, pp. 1323–1338.

[18] K. A. Messabi, M. Aldwairi, A. A. Yousif, A. Thoban, and F. Belqasmi, “Malware detection using dns records and domain name features,” in *Proceedings of the 2nd International Conference on Future Networks and Distributed Systems*. ACM, 2018, p. 29.

[19] M. Klatt, B. W. Roberts, and T. C. Helming, “Domain reputation evaluation process and method,” Apr. 6 2017, uS Patent App. 14/872,191.

[20] H. Mekky, R. Torres, Z.-L. Zhang, S. Saha, and A. Nucci, “Detecting malicious http redirections using trees of user browsing activity,” in *INFOCOM, 2014 Proceedings IEEE*. IEEE, 2014, pp. 1159–1167.

[21] M. Hirotomo, Y. Nishio, M. Kamizono, Y. Fukuta, M. Mohri, and Y. Shiraishi, “Efficient method for analyzing malicious websites by using multi-environment analysis system,” in *2017 12th Asia Joint Conference on Information Security (AsiaJCIS)*. IEEE, 2017, pp. 48–54.

[22] R. Wang, Y. Zhu, J. Tan, and B. Zhou, “Detection of malicious web pages based on hybrid analysis,” *Journal of Information Security and Applications*, vol. 35, pp. 68–74, 2017.

[23] C. Wressnegger and K. Rieck, “Looking back on three years of flash-based malware,” in *Proceedings of the 10th European Workshop on Systems Security*. ACM, 2017, p. 6.

[24] B. Altay, T. Dokeroglu, and A. Cosar, “Context-sensitive and keyword density-based supervised machine learning techniques for malicious webpage detection,” *Soft Computing*, pp. 1–15, 2018.

[25] N. Jagpal, E. Dingle, J.-P. Gravel, P. Mavrommatis, N. Provos, M. A. Rajab, and K. Thomas, “Trends and lessons from three years fighting malicious extensions.” in *USENIX Security Symposium*, 2015, pp. 579–593.

[26] S. Morishige, S. Haruta, H. Asahina, and I. Sasase, “Obfuscated malicious javascript detection scheme using the feature based on divided url,” in *Communications (APCC), 2017 23rd Asia-Pacific Conference on*. IEEE, 2017, pp. 1–6.

[27] O. Sivan and Y. Lavi, “Web page and web browser protection against malicious injections,” Jul. 18 2017.

[28] H. Kikuchi, D. Yu, . A Chander - US Patent 9, 686, and U. 2017, “Method and apparatus for constructing security policies for web content instrumentation against browser-based attacks,” 2017. [Online]. Available: https://patents.google.com/patent/US9686288B2/en.

[29] A. K. Singh, “Dataset for Comparison of ML Attributes for Detecting Malicious Websites,” 2018. [Online]. Available: http://dx.doi.org/10.17632/stctg82wsf.1.