2020 12th International Conference on Communication Systems & Networks (COMSNETS)

Machine Learning & Concept Drift based Approach for Malicious Website Detection

Siddharth Singhal, Utkarsh Chawla
Dept. of Computer Science & Engineering
Delhi Technological University
New Delhi, India
{siddharthsinghal_bt2k16, utkarshchawla_bt2k16}@dtu.ac.in

Rajeev Shorey
TCS Research and Innovation &
Dept. of Computer Science & Engineering
Indian Institute of Technology
New Delhi, India
rajeev.shorey@tcs.com

***Abstract*—The rampant increase in the number of available cyber attack vectors and the frequency of cyber attacks necessitates the implementation of robust cybersecurity systems. Malicious websites are a significant threat to cybersecurity. Miscreants and hackers use malicious websites for illegal activities such as disrupting the functioning of the systems by implanting malware, gaining unauthorized access to systems, or illegally collecting personal information. We propose and implement an approach for classifying malicious and benign websites given their Uniform Resource Locator (URL) as input. Using the URL provided by the user, we collect Lexical, Host-Based, and Content-Based features for the website. These features are fed into a supervised Machine Learning algorithm as input that classifies the URL as malicious or benign. The models are trained on a dataset consisting of multiple malicious and benign URLs. We have evaluated the accuracy of classification for Random forests, Gradient Boosted Decision Trees and Deep Neural Network classifiers. One loophole in the use of Machine learning for detection is the availability of the same training data to the attackers. This data is exploited by the miscreants to alter the features associated with the Malicious URLs, which will be classified as benign by the supervised learning algorithms. Further, owing to the dynamic nature of the malicious websites, we also propose a paradigm for detecting and countering these manually induced concept drifts.**

***Index Terms*—URL Feature Extraction, Malicious Website Detection, Concept Drifts, Feature Vectors, Gradient Boosted Trees, Random Forest, Feedforward Neural Networks**

I. Introduction

We have witnessed an exponential growth in the number of users browsing the web. This is due to several reasons such as, for example, majority of businesses shifting online, an increase in usage of highly reliable cloud storage and computation devices, and availability of high-speed connectivity to the internet across the globe including remote areas at low prices. As systems get connected to the internet, they become vulnerable to a wide range of cyber threats such as spam-advertising, financial fraud, malware implanting, and information theft. Malicious websites are the primary attack vector used by hackers for propagating these attacks. Miscreants provide these URLs as baits to naive users who are unable to distinguish between malicious and benign websites, which necessitates the development of robust solutions for validating an unfamiliar URL or suspicious website.

A variety of methodologies have been proposed and deployed for detecting websites as well as mitigating the cyber attacks. One of the most widely used approaches is to maintain a centralized blacklist of suspicious as well as malicious sites, IP addresses, and domains [5]. The solution, though elegant and straightforward, is not scalable as it is tedious or impossible to validate and maintain a list of all these websites.

To overcome the scalability issues, we propose a system for automatic feature extraction and real-time classification using the input URL. Singh et al. [13] analyse the effectiveness of various machine learning attributes for detecting malicious websites. However, their work is limited to the use of Host-Based and Content-Based features. In our approach, we extract the Lexical features along with Host-Based, and Content-Based features and store these attributes as feature vectors for each URL. We feed these feature vectors as input to the supervised learning algorithm, which classifies these URLs as malicious or benign. The supervised learning algorithms used in our study are Random Forests, Gradient Boosted Trees, and Feed Forward Neural Networks. Our models are trained on a list of malicious and benign URLs curated from a variety of sources. While the proposed approach is agnostic to a variety of attacks and is scalable, it does not conform to the dynamic nature of websites. Miscreants who have access to the same training data and are capable of identifying patterns in detection algorithms attempt to alter certain features of malicious sites with the motive of bypassing the security. The scenario where the relation between the input data and the target variable changes over time is referred to as Concept Drift [8]. Tan et al. [7] propose machine learning techniques such as Linear Support Vector Machines, Logistic Regression, Random Forests, and Naive Bayes for detecting malicious websites along with statistical method for detecting Concept Drifts in websites. In this paper, we define a novel approach for detecting concept drifts by computing the distance between old and new feature space. The novelty of our proposed approach lies in the use of Neural Networks for detecting Malicious websites.

II. Dataset

In an effort to validate our approach, we collect a set of malicious URLs and a set of benign URLs. For each URL in both the datasets, we extract the features listed in Table I.

978-1-7281-3187-0/20/$31.00 ©2020 IEEE

We then pre-process these features to be suitable for our supervised learning models.

- **Phase 1: Data Collection:** We used the following public blacklists to collect malicious URLs: PhishTank [12], a collaborative platform operated by OpenDNS to distribute and verify phishing websites and Malware Domain List (MDL) [1], which maintains an archive of malware-infected websites. Benign URLs are retrieved from majestic.com [9], which lists the top one million URLs with the most referring subnets. Our final dataset has a total of eighty thousand unique URLs, out of which forty thousand are malicious, and the remaining forty thousand are benign.
- **Phase 2: Feature Extraction:** We categorize the URL features into three groups: lexical, host-based, and content-based. For host-based and content-based, we use the features which are considered to be important by Singh et al. [13]. We also include lexical based features, which have been known to improve the performance of the classifier [2] [3].
  - **Lexical Features:** Lexical features are extracted from the URL (Uniform Resource Locator) string. The idea is to classify a malicious website from a benign one by identifying the differences in various aspects of their URL strings.
    - *URL Length:* Length of the Uniform Resource Locator.
    - *Host Length:* Length of the hostname.
    - *Host token count:* Number of tokens in the hostname delimited by (".").
    - *Path Length:* Length of the pathname.
    - *Number of symbols:* Number of special characters such as '&', '%', '$', '#', '~', etc.
  - **Host-Based Features**
    - *Location:* The Geographical location of the website is determined from the IP address through the GeoIP database.
    - *Autonomous System Number:* This is a unique number that is available globally to identify an autonomous system and which enables the system to exchange exterior routing information with other neighbouring autonomous systems.
  - **Content-Based Features:** Content-Based features are obtained by analysing the HTML and the JavaScript code of the website.
    - *HTTPS Enabled:* The presence of the Secure Socket Layer (SSL) protocol ensures that the website has a basic level of privacy and integrity [4].
    - *Applet Count:* The number of java applet tags. Java applets are used to perform malicious exploits [15].
    - *Presence of Eval() function:* Eval() function is used to generate malicious code at runtime, which prevents detection [6].

Feature	Description
URL Length	Length of the Uniform Resourse Locator
Host Length	Length of the hostname.
Host Token Count	Number of tokens in the hostname (delimited by '.')
Path Length	Length of the pathname.
Number of symbols	Number of special characters like '', '%', '$', '', '=', etc.
Location	Geographical Location of the website
ASN	Autonomous System Number of the website
HTTPS Enabled	Presence of Secure Socket Layer
Applet Count	Number of Java Applet tags
Eval() function	Presence of eval() function
XHR	Number of XHR tags
Popups	Presence of windows.open() function
Redirection	Check if the website redirects
unescape() function	Presence of unescape() function

TABLE I: Summary of features extracted from the URL

    - *Count of XML HTTP Requests (XHR) tag:* XHR, although being at the core of AJAX, can be used to inject malicious exploits. The number of XHR tags is stored as a numerical value.
    - *Presence of popups:* The windows.open() function is generally used for advertisements, but can also be used to inject malicious code.
    - *Redirection:* Redirection in a website can be used as a means to redirect from a benign URL to a malicious one [10]. Redirection is checked through the Requests library in python, and the result is stored as a boolean variable.
    - *Presence of unescape() function:* The unescape() function is usually used for decoding encoded malicious code [11].

Table I provides a summary of all the aforementioned features.

III. Proposed Framework

In this section, we define the design of our proposed framework which classifies the website as malicious or benign. Further, the system is equipped with the ability to retrain the supervised machine learning model in the presence of concept drifts. Our system consists of three modules, namely feature extraction, supervised learning and concept drift detection. The high-level architecture of the system is shown in Figure 1.

- **Feature extraction:** This module makes use of open-source python libraries such as Requests, WHOIS, BeautifulSoup and open-source GeoIP database to extract features from a given URL. All the features given in Table I are obtained using this module.
- **Supervised Learning model:** This module is responsible for training the supervised learning model followed by predicting the degree of maliciousness for unseen URLs. We analysed three algorithms, namely Gradient Boosted Trees, Random Forest and Feedforward Neural Networks as candidates for our learning model.
  - ***Gradient Boosted Trees:*** Gradient boosting is a machine learning technique which uses an ensemble of decision trees to form the prediction model. It

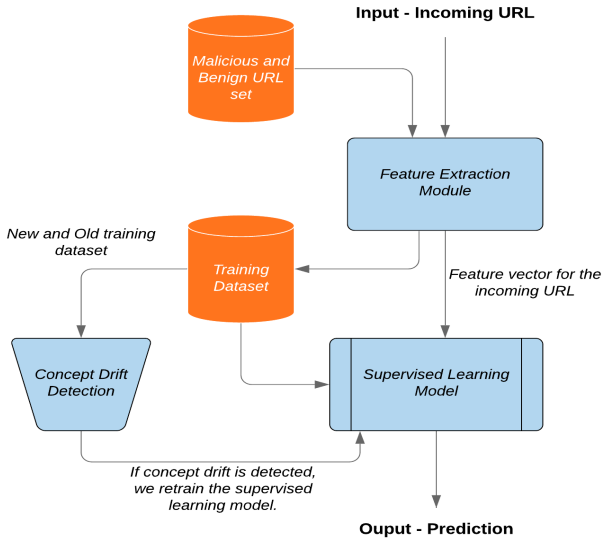

Fig. 1: System Architecture

sequentially builds the model such that the succeeding predictors can learn from the mistakes of the preceding predictors

- ***Random Forest:*** Random Forest is a machine learning technique used for classification and regression, which is an ensemble of decision trees. It uses bagging and feature randomness for creating a forest of uncorrelated trees having accuracy higher than the individual trees.
- ***Feedforward Deep Neural Networks:*** This is an artificial neural network where links between the different layers do not form a cycle. Thus, in this network, data only moves in the forward direction.

- **Concept Drift Detection:** One shortcoming of our model is that hackers and miscreants are aware of the standard features that are obtained from the URL. Hence, this provides them with the opportunity to modify the properties of malicious URLs to escape detection. This phenomenon, where the relationship between the input data and the target variable changes over time, is referred to as concept drift [8].
We propose an algorithm to detect concept drifts in real-time efficiently. During concept drift detection, we again collect the feature vectors for all the malicious and benign URLs in our training dataset. The algorithm focuses on finding the difference in data distribution between the old training data and the newly collected feature vectors data. For every malicious URL in the new training dataset, we find the least distant malicious URL feature vector in the old dataset. We repeat the same procedure for all the benign URLs in the new dataset. If the average of all the distances calculated for this time period is more than a pre-set threshold value, then we claim that concept drift has occurred. We use the Heterogeneous Euclidean-Overlap Metric (HEOM) [14] for calculating the distance between two feature vectors. HEOM has been known to perform well with vectors consisting of both numerical and categorical attributes, hence it fits perfectly with the dataset discussed in the paper. Once concept drift is detected, we retrain our supervised learning model on the new data to improve the model's ability to detect URLs with manipulated features. Occurrence of a concept drift can also be validated by a steep fall in the accuracy of our classification model on our validation set.

Algorithm 1 Concept Drift Detection

Inputs:
Mal_old ← Array of feature vectors for all malicious URLs (in training data) created before concept drift
Ben_old ← Array of feature vectors for all benign URLs (in training data) created before concept drift
Mal_new ← Array of feature vectors for all malicious URLs (in training data) created after concept drift
Ben_new ← Array of feature vectors for all benign URLs (in training data) created after concept drift
Ouput:
1 : Concept Drift detected
0 : No Concept Drift detected
Algorithm:
total_distance ← 0
total_count ← 0
for each item e **in** Mal_new **do**
  distance ← infinite
  for each item f **in** Mal_old **do**
    distance ← min(distance, **HEOM(e,f)**)
  end for
  if $distance$!= 0 **then**
    total_distance ← total_distance + distance
    total_count ← total_count + 1
  end if
end for
for each item e **in** Ben_new **do**
  distance ← infinite
  for each item f **in** Ben_old **do**
    distance ← min(distance, **HEOM(e,f)**)
  end for
  if $distance$!= 0 **then**
    total_distance ← total_distance + distance
    total_count ← total_count + 1
  end if
end for
mean_distance ← total_distance / total_count
if $mean\_distance > threshold$ **then**
  return 1
end if
return 0

IV. Evaluation

We first evaluate the performance of our supervised classification algorithms in detecting malicious websites. Table II provides an overview of the classification performance obtained on our test set for three algorithms. To evaluate the performance, we used the following metrics:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN},$$

$$Precision = \frac{TP}{TP + FP},$$

$$Recall = \frac{TP}{TP + FN}.$$

Where TP, TN, FP and FN denotes the number of True Positives, True Negatives, False Positives and False Negatives respectively. We observe that Gradient Boosting Algorithm is the most effective in classifying malicious and benign URLs with an accuracy of 96.4% for a maximum depth (longest path from root to leaf for any estimator tree in the ensemble) of 4. Figure 2 depicts the trends in accuracy achieved by Gradient Boosting for different hyperparameters. As illustrated in the figure, we observe that for fewer estimators in the ensemble, the accuracy is low due to underfitting. The accuracy increases with an increase in the number of estimators reaching a maximum value of 96.4% for 80 estimators. Thereafter, the accuracy falls steeply due to overfitting on the training dataset. We observe that the effectiveness of these algorithms in detecting malicious website begins to decrease when a concept drift occurs. In order to assess the performance of the concept drift detection algorithm, we create an artificial concept drift in our dataset. We start interchanging the target labels of our data such that a fraction of the malicious URLs become benign, and the respective benign URLs become malicious. We interchange the labels of 10% of the URLs in the database and then apply our algorithm to the artificial dataset. Our detection algorithm returns a value of 0.32, which is higher than our pre-set threshold value of 0.20. Thus, we can claim that our algorithm is able to successfully recognize concept drifts.

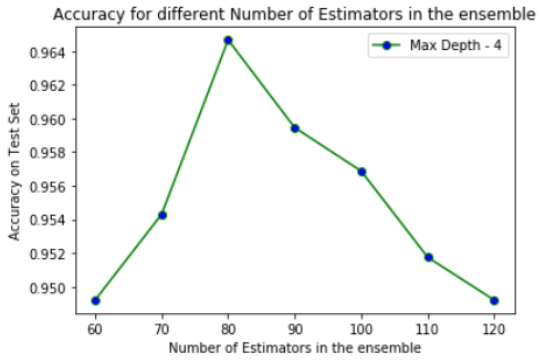

Fig. 2: Accuracy curve for Gradient Boosted Decision Trees at Maximum Depth 4

Model	Accuracy(%)	Precision(%)	Recall(%)
Random Forest	93	92.4	91.8
Gradient Boosting	**96.4**	**96.2**	**96.3**
Neural Network	95.4	94.5	93.8

TABLE II: Classification performance for different supervised learning models

V. Conclusion

In this paper, we have proposed a robust and novel approach for assisting users in detecting malicious and harmful websites and URLs. The dataset used for training and testing has been curated from a variety of sources. Our algorithm achieves a superior accuracy of 96.4% on the test dataset. We have proposed a novel approach for detecting and mitigating attempts by attackers for circumventing malicious website detection algorithms. Results show that the proposed algorithm is able to successfully detect concept drifts.

References

[1] "Malware domain list," https://www.malwaredomainlist.com/.
[2] G. Chakraborty and T. T. Lin, "A url address aware classification of malicious websites for online security during web-surfing," *2017 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*.
[3] H. Choi, B. B. Zhu, and H. Lee, "Detecting malicious web links and identifying their attack types," *WebApps'11, Proceedings of the 2nd USENIX conference on Web application development, pp.11-11, Berkeley, USA (2011)*.
[4] A. P. Felt, R. Barnes, A. King, C. Palmer, C. Bentzel, and p. Tabriz, "Measuring https adoption on the web," *26th USENIX Security Symposium, 2017, pp. 1323–1338.*
[5] Y. Fukushima, Y. Hori, and K. Sakurai, "Proactive blacklisting for malicious web sites by reputation evaluation based on domain and ip address registration," *2011 IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications*.
[6] A. Gorji and M. Abadi, "Detecting obfuscated javascript malware using sequences of internal function calls," *Proceedings of the 2014 ACM Southeast Regional Conference on - ACM SE '14, 2014.*
[7] T. Guolin, Z. Peng, L. Qingyun, L. Xinran, Z. Chunge, and D. Fenghu, "Adaptive malicious url detection: Learning in the presence of concept drifts," *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications*.
[8] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang, "Learning under concept drift: A review," *IEEE Transactions on Knowledge and Data Engineering, 2018.*
[9] Majestic, "Majestic million," https://majestic.com/reports/majestic-million.
[10] H. Mekky, R. Torres, Z.-L. Zhang, S. Saha, and A. Nucci, "Detecting malicious http redirections using trees of user browsing activity," *INFOCOM, 2014 Proceedings IEEE. IEEE, 2014, pp. 1159–1167.*
[11] S. Morishige, S. Haruta, H. Asahina, and I. Sasase, "Obfuscated malicious javascript detection scheme using the feature based on divided url," *Communications (APCC), 2017 23rd Asia-Pacific Conference on. IEEE, 2017, pp. 1–6.*
[12] OpenDNS, "Phishtank," https://www.phishtank.com.
[13] A. K. Singh and N. Goyal, "A comparison of machine learning attributes for detecting malicious websites," *2019 11th International Conference on Communication Systems Networks (COMSNETS)*.
[14] D. R. Wilson and T. R. Martinez, "Improved heterogeneous distance functions," *Journal of Artificial Intelligence Volume 6 Issue 1, January 1997 Pages 1-34*.
[15] C. Wressnegger, F. Yamaguchi, D. Arp, and K. Rieck, "Technical report - analyzing and detecting flash-based malware using lightweight multi-path exploration," *University of Gottingen, Germany, no. December, 2015. [Online]*.