2017 2nd IEEE International Conference On Recent Trends in Electronics Information & Communication Technology (RTEICT), May 19-20, 2017, India

Malicious Web Content Detection Using Machine Leaning

Anand Desai,
Sardar Patel Institute of Technology,
ananddesai71195@gmail.com,

Janvi Jatakia
Sardar Patel Institute of Technology,
jatakiajanvi12@gmail.com,

Rohit Naik
Sardar Patel Institute of Technology,
rohit.naik246@gmail.com

Nataasha Raul
Sardar Patel Institute of Technology,
nataasharaul@spit.ac.in

Abstract—Naive users using a browser have no idea about the back-end of the page. The users might be tricked into giving away their credentials or downloading malicious data. Our aim is to create an extension for Chrome which will act as middleware between the users and the malicious websites, and mitigate the risk of users succumbing to such websites. Further, all harmful content cannot be exhaustively collected as even that is bound to continuous development. To counter this we are using machine learning- to train the tool and categorize the new content it sees every time into the particular categories so that corresponding action can be taken.

***Keywords*—Machine learning, Chrome, Extension, Phishing, Malicious**

I. INTRODUCTION

Malicious web pages are those, which contain content that can be used by attackers to exploit end-users. This includes web pages with phishing URLs, spam URLS, JavaScript malware scripts, Adware, and many more. Nowadays, it is becoming very difficult to detect such vulnerabilities due to the continuous development of new techniques for carrying out such attacks. In addition, not all the users are aware of the different type of exploits which attackers can take advantage of. Therefore, when there is a vulnerability in a web page, which the user is not aware of, this tool will help him stay safe despite his lack of knowledge of the website. In addition, if the URL itself has signs of being a phishing URL, then the user will be protected from that website.

Python, being open source, along with the help of the variety of support libraries, simple to understand syntax, and abundant resources, proves to be the best option for implementing machine learning. The alternate methodology is to check the entered URL in the list of websites which are declared malicious by a trusted source. But the drawback of this method is that the list is non-exhaustive i.e. it increases every day. Also, because of such a large list, the latency time of the system will always increase which can frustrate the user. Hence we use the Machine Learning approach, where the URL can be tested against a trained classifier. A Google Chrome extension is a very good way of ensuring easy access to the tool for the user. Since Google Chrome is the most widely used web browser throughout the world and with its popularity only increasing month-by-month, it is the best option for implementing this tool to ensure the maximum outreach.

The goal is to ensure safe browsing irrespective of the website which the user wishes to visit. Even if the user decides to visit a phishing website, measures will be taken to protect the user from being harmed.

II. LITERATURE SURVEY

A. Related Papers

According to the survey conducted in “Survey on Malicious Web Pages Detection Technique”, the authors mention that the web attacks are on a rise. In the year 2012, each month noted a total of approximately 33,000 phishing attacks, which summed to a loss of \$687 million. With such a hike in the number of web attacks, there is a dire need of a system which prevents such types of webpage attacks [1]. The user goes to a website unknowingly which might not be safe for the user and ends up either giving his credentials to other intruders and hackers or downloading malicious data. So to prevent this type of attacks, a tool is needed which examines the URL entered by the user and checks if the website is malicious or not. In the paper "Identifying Vulnerable Websites by Analysis of Common Strings in Phishing URLs" [2], the authors mention about the rise in the phishing websites and the method they used to detect and prevent it. For the detection of a safe or a phishing webpage, they implemented Largest Common Substring (LCS) method. They prepared a database of phishing websites of their own and tested the LCS on the new website. In the paper “On URL Classification”, the authors mention about how URLs can be used to use the victim's computer resources for different attacks like phishing, denial of service. Also, they compared various methodologies including traditional ones and the recent trends in the field of machine learning. The results show that machine learning techniques are better for detection [4]. So instead of using traditional techniques, we plan to integrate the concept of machine learning in our tool.

Now, as we have to examine whether the website is malicious or benign, we will have to extract the features of the website. In the paper “Feature Extraction Process: A Phishing Detection Approach”, the author talks about how the features can be extracted from a URL. The author finds 17 features which can be extracted from the URL based on which a URL can be declared as phishing or no [5]. In our project, we plan to extract 23 features which will enable more accuracy in declaring a URL as malicious or benign. In the paper "Feature extraction and classification phishing websites based on URL"

978-1-5090-3704-9/17/$31.00 © 2017 IEEE

[6], M. Aydin and N. Baykal used the feature extraction technique to form a feature matrix using which they classified the URL. They extracted 133 features and used only a subset of it which they considered as prominent. They have not specified the reason for choosing the parameters using which they declare a website as malicious or no. They used different parameters and different algorithms to test the efficiency obtained. We plan to select a single algorithm and use all the parameters and features as given in the dataset we selected. In "A Comparison of Machine Learning Techniques for Phishing Detection", Abu-Nimeh et al [7] have done a comparative study of six different classifiers to find which classifier works the best. They have showed that Random Forest outperforms other classifiers by having the lowest error rate.

In the paper "Detection of phishing URLs using machine learning techniques", the author [8] discusses about the rise of phishing websites and give techniques to extract features and implement machine learning algorithms to classify the same. They have extracted features like traffic rank details, lexical features, page rank etc. They have presented a study of different machine learning algorithms. A fixed result showing the best algorithm is not done in the paper, we will give statistical analysis of all the algorithms and even the accuracy of the chosen algorithm to prove the result.

As the technology advances, the number of possible malicious attacks would also increase. It would be impossible to prevent all the new malicious and phishing websites using the traditional methodology of storing the list of malicious URLs and checking directly from the database. So, "Malicious Web Content Detection using Machine learning" aims at improvising the traditional methodology by adding machine learning for the same. The paper uses four different classification algorithms to detect Dynamic HTML malicious codes and states that Boosted Decision Tree gives the best output. The prototype can determine whether webpage is malicious or not, but cannot block or prevent the malicious content [9]. Now, there can be an instance that the URL might be benign but it may contain certain JavaScript code or iframe which will lead to download of malicious content. So taking this possibility in consideration, we will scrape the entire webpage to detect malicious contents even if the website is rendered safe by the algorithm.

B. *Existing Systems*

- Phish Detector - A powerful extension to detect phishing attacks in Online-Banking web sites. It is a rule-based system that analyses the webpage content to identify phishing attacks. Phish Detector has the ability to detect online-banking phishing scams quickly with zero false negative alarm. For accurate result, it is recommended to use this extension for your Online-Banking web pages only [12]. This extension works only for banking websites and does not take into account the other domains.
- Netcraft Extension - The Netcraft Extension is a tool allowing easy lookup of information relating to the sites you visit and providing protection from Phishing. The Netcraft anti-phishing community is effectively a giant neighbourhood watch scheme, empowering the most alert and most expert members to defend everyone within the community. As soon as the first recipients of a phishing mail report it, we can block it for all users of the extension providing an additional level of protection from Phishing [13]. This extension waits for the first victim to report and accordingly notifies the other users for the same.
- Cascaded Phish Detector - The online cascaded phish detector is composed of a client-side component and a server-side component. The client side is implemented as a Chrome extension, which injects content script to web pages and extracts the corresponding HTML DOMs [14]. This extension only considers the HTML DOMs for identifying the phishing components and not other parameters.

III. METHODOLOGY

Phishing attacks being on a rise, using a direct search from the phishing website database is not enough. To provide protection against new attacks, machine learning provides the best alternative for the same. We used the UCI Dataset of Phishing Website to train the classifier. Later, whenever a user enters the URL, the features are extracted and the URL is tested on the trained classifier to obtain the result. Following were the major steps involved in the implementation:

A. *Obtaining Dataset*

The dataset was obtained from the UCI - Machine Learning Repository [16] which contains the Phishing Web Site Dataset. This dataset is composed of 11055 entries of websites which are classified as phishing and benign. These entries each have 30 features of the website used.

B. *Feature Selection*

From the dataset, out of the 30 features present, it was infeasible to extract all the features. This is because many features used some standard databases which are not accessible to us. Also, extracting some of the features seemed not possible as they demanded the extraction of data from the server of the website, which is not possible. Hence, we narrowed down our dataset to contain 22 features. Some of them were:

- URL Length - Long URLs are generally used by phishers to hide the doubtful parts. Hence, the length can be calculated of the URL. If the length of the URLs turn out to be more than 52 characters then the URL is considered as phishing.
- Google Index - This feature examines whether a website is in Google's index or not. When a site is indexed by Google, it is displayed on search results. Usually, phishing webpages are merely accessible for a short period and as a result, many phishing webpages may not be found on the Google index.

C. Choosing Classification Algorithm

For classifying the URL entered, as either safe or malicious, we considered the following three algorithms:

- K-Nearest Neighbours (kNN) - kNN algorithm can be used for both classification as well as regression problems. However, mostly it is used for classification problems. 'k' in the kNN algorithm stands for the number of nearest neighbors we wish to take vote from. When predicting for a new data sample, this algorithm will run a search on the training dataset to find the closest k-samples. The predicted class of those similar samples is then found out and the summary of that is given out as the class label for this new data sample. Also, for different types of data, the similarity measure varies. The similarity measure for real-valued data can be Euclidean distance, whereas for other data like categorical or binary, Hamming distance is helpful [26].
- Support Vector Machines (SVM) - They are supervised learning models used for both classification and regression. The SVM algorithm uses a dataset where the input samples are divided into two classes with labels either 0 or 1. The methodology includes finding a line (in two-dimension space) or a plane (in multi-dimension space) also called as a hyperplane which will most efficiently separate the two classes.

$$B0 + (B1 * X1) + (B2 * X2) = 0 \qquad (1)$$

  In equation (1) [27], the coefficients (B1 and B2) give the slope of the line and the intercept (B0) is calculated by the algorithm. Also, X1 and X2 are the two input data points. By plugging in these data points into the line equation, you can calculate whether a new point is on which side of the line and the class will be predicted [27].
- Random Forest - This algorithm makes use of decision trees for classification. It creates multiple decision trees at the time of training. Random Forests are a combination of tree predictors. The formation of each tree is based on the values of a random vector sampled independently and each tree in the forest has the same distribution. The main principle is that a group of "weak learners" can combine and form a "strong learner". Random Forests do not over fit data due to the law of large numbers and therefore are a wonderful tool for making predictions. When the right kind of randomness is introduced, they become accurate classifiers and regressors. Single decision trees often have high variance or high bias. Random Forests attempt to alleviate this problem by taking the average thereby finding a natural balance between the two extremes. Due to the fact that Random Forests have less number of parameters that can be tuned and that they can be used directly with default parameter settings, they are a simple tool to use without having a model or to produce a reasonable model in a fast and efficient manner [28].

For the comparative study of these algorithms, we not only consider the accuracy, but also other metrics which are used to select the best classifier. We calculated the accuracy, precision, recall, F1-Score for our dataset for all the algorithms. As per the result of these metrics, we got the best scores when the RF algorithm was used. Hence, we decided to train the classifier with the RF Algorithm. Below is the screenshot depicting the results of all the algorithms with the different performance metrics.

```
 Support Vector Machine Results
              precision    recall  f1-score   support

          -1       0.96      0.89      0.92       460
           1       0.92      0.97      0.94       594

avg / total       0.94      0.94      0.94      1054

The accuracy is: 0.935483870968
[[408  52]
 [ 16 578]]


 K-Nearest Neighbours Algorithm Results
              precision    recall  f1-score   support

          -1       0.95      0.89      0.92       460
           1       0.92      0.96      0.94       594

avg / total       0.93      0.93      0.93      1054

The accuracy is: 0.930740037951
[[408  52]
 [ 21 573]]


 Random Forest Algorithm Results
              precision    recall  f1-score   support

          -1       0.96      0.95      0.96       460
           1       0.96      0.97      0.97       594

avg / total       0.96      0.96      0.96      1054

The accuracy is: 0.96110056926
[[436  24]
 [ 17 577]]
```

Figure 1. Accuracy for different algorithms

Also, for Random Forest, it is possible to use different values for minimum split size. Minimum split size denotes the least number of samples which are needed to split an internal node. The default number of nodes required for splitting an internal node is 2. By considering the different split sizes of 3 to 11, we got the results as shown in Table 1. In the table, TP-True Positive, TN-True Negative, FP-False Positive, FN-False Negative:

TABLE I. Accuracy for different split size

Split Size	Accuracy	TP	TN	FP	FN
3	95.73	431	579	15	30
4	95.26	434	571	23	27
5	95.83	432	579	15	29
6	95.63	435	574	20	26
7	96.11	435	579	15	26
8	95.82	434	577	17	27
9	96.01	432	581	13	29
10	95.45	429	578	16	32
11	95.83	430	581	13	31

On observing the above table, minimum split size was selected as 7. So we applied the Random Forest Algorithm with the minimum split size of 7.

D. Google Chrome Extension

Extensions are add-ons to the browser which help in adding more features and making browser usage easier for the user [25]. So, for our case, when a user enters the URL of the website, we display whether the URL entered is benign or phishing using this Chrome Extension. HTML, JavaScript, CSS are used for coding a Google Chrome Extension.

IV. RESULTS

When the user enters a URL, the extension takes the URL using the GET method and passes the same to the python code using the Java script of the extension. The python code then extracts all the features from the URL and forms an array. We then test this on the trained classifier of Random Forest.

Below is the screenshot of a safe website i.e. http://www.stackoverflow.com. Since major of the features of the website were safe, the net result was obtained as safe by the Chrome extension. We display the complete web and future scope site to the user.

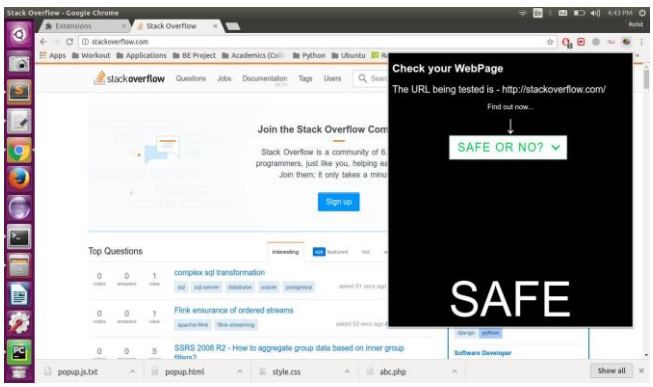

Figure 2. Result of the extension for a safe website

Now, for the case of phishing websites, if the website is rendered as phishing by the extension, the user is prompted about it. Below is the screenshot for the phishing websites i.e http://www.fyccenter.com/drive/auth/view/share/index.html.

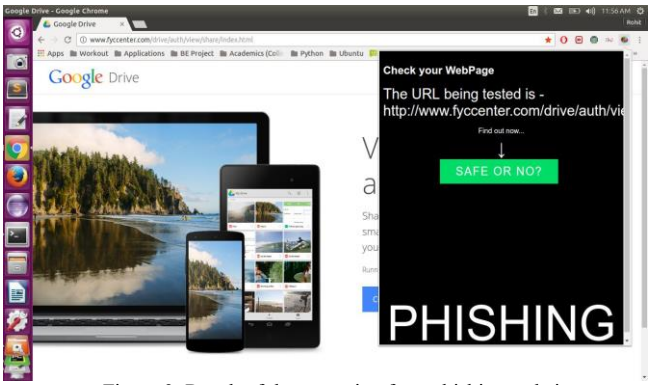

Figure 3. Result of the extension for a phishing website

CONCLUSION AND FUTURE SCOPE

This paper proposes the development of a Chrome Extension for identifying phishing websites. This concept displays the safety component of a website to keep the user safe. Otherwise, the user might end up giving his credentials to the phishers which can lead to huge losses.

The future scope of this idea is very broad. Some websites only have a few components (for example, a form, or an image etc.) which make the website unsafe. So for such websites, one can block this malicious part and display the complete safe webpage to the users. This can be implemented in a similar fashion as to that of AdBlock Extension which blocks a particular part of the webpage and displays the rest.

REFERENCES

[1] D. R. Patil, J. B. Patil, "Survey on Malicious Web Pages Detection Techniques, Science and Technology", 2015 International Journal of u- and e- Service.

[2] B. Wardman, G. Shukla and G. Warner, ;"Identifying vulnerable web-sites by analysis of common strings in phishing URLs,"; 2009 eCrime Researchers Summit, Tacoma, WA, 2009, pp. 1-13.

[3] A. B. Sayambar, A. M. Dixit, "On URL Classification, International Journal of Computer Trends and Technology" 2014.

[4] Xiang et al., "A Feature-Rich Machine Learning Framework for Detecting Phishing WebSites, ACM Transactions on Information and System Security "2011.

[5] A. Abunadi, O. Akanbi and A. Zainal, "Feature extraction process: A phishing detection approach," 2013 13th International Conference on Intellient Systems Design and Applications, Bangi, 2013, pp. 331-335. doi: 10.1109/ISDA.2013.6920759

[6] M. Aydin and N. Baykal, "Feature extraction and classification phishing websites based on URL," Communications and Network Security (CNS), 2015 IEEE Conference on, Florence, 2015, pp. 769-770.

[7] Abu-Nimeh S., Nappa D., Wang X., & Nair S. (2007)." A Comparison of Machine Learning Techniques for Phishing Detection". APWG eCrimes Researchers Summit, (pp. 60-69). Pittsburgh, PA.

[8] James, Joby, L. Sandhya, and Ciza Thomas, "Detection of phishing URLs using machine learning techniques," Control Communication and Computing (ICCC), 2013 International Conference on. IEEE, 2013.

[9] Hou et al., "Malicious Web Content Detection by Machine learning, Expert Systems with Applications", International Journal 2010.

[10] G. Venkataraman and A. Ravichandran, "Adaptive Semantic Search: Re-Ranking of Search Results Based on Webpage Feature Extraction and Implicitly Learned Knowledge of User Interests," Semantics,

Knowledge and Grids (SKG), 2014 10th International Conference on, Beijing, 2014, pp. 75-78.

[11] Khamis et al, "Characterizing A Malicious Web Page", Australian Journal of Basic and Applied Sciences 2014.

[12] Curtsinger et al., "Zozzle: Fast and Precise In-Browser JavaScript Malware Detection", SEC'11 Proceedings of the 20th USENIX conference on Security 2011.

[13] G. Lu and S. Debray, "Automatic Simplification of Obfuscated JavaScript Code: A Semantics-Based Approach," Software Security and Reliability (SERE), 2012 IEEE Sixth International Conference, Gaithersburg, MD, 2012, pp. 31-40.

[14] M. Aldwairi, R. Alsalman MALURLS: "A Lightweight Malicious Website Classification based on URL features, Web Intelligence", 2012 Journal of Emerging Technologies.

[15] Eshete et al., Malicious Website Detection: Effectiveness and Efficiency Issues, SysSec Workshop (SysSec) 2011.

[16] "UCI Machine Learning Repository: Data Set", Archive.ics.uci.edu, 2017. [Online].Available:http://archive.ics.uci.edu/ml/datasets/Phishing+Websites UCI Phishing Websites Data Set. [Accessed: 30- Mar- 2017].

[17] Ma et al., Beyond Blacklists:" Learning to Detect Malicious Web Sites from Suspicious URLs, Knowledge discovery and data mining", 2009 15th ACM SIGKDD international conference.

[18] Liu Wenyin, Guanglin Huang, Liu Xiaoyue, Xiaotie Deng and Zhang Min, "Phishing Web page detection," Eighth International Conference on Document Analysis and Recognition (ICDAR'05), 2005, pp. 560-564 Vol. 2.

[19] Canali et al., Prophiler: "A Fast Filter for the Large-Scale Detection of Malicious Web Pages", Web Security, 2011, WWW'11.

[20] Cova et al., "Detection and Analysis of Drive-by- Download Attacks and Malicious JavaScript Code", WWW'10 Proceedings of the 19th international conference on World wide web, 2010.

[21] Ma et al., "Identifying Suspicious URLs: An Application of Large-Scale Online Learning, Machine Learning", 2009 ICML '09 Proceedings of the 26th Annual International Conference.

[22] Y. Huang and L. Li, "Naive Bayes classification algorithm based on small sample set," 2011 IEEE International Conference on Cloud Computing and Intelligence Systems, Beijing, 2011, pp. 34-39.

[23] Zhou et al., "Malicious Websites Detection and Search Engine Protection", Journal of Advances in Computer Network 2013.

[24] Choi et al., "Detecting Malicious Web Links and Identifying their attack types, Web application development", 2011 2nd USENIX conference.

[25] "What are extensions? - Google Chrome", Developer.chrome.com, 2017. [Online]. Available: https://developer.chrome.com/extensions. [Accessed: 30- Mar- 2017].

[26] J. Brownlee, "Tutorial To Implement k-Nearest Neighbors in Python From Scratch - Machine Learning Mastery", Machine Learning Mastery, 2017.[Online].Available:http://machinelearningmastery.com/tutorial-to-implement-k-nearest-neighbors-in-python-from-scratch/. [Accessed: 30- Mar- 2017].

[27] J. Brownlee, "Support Vector Machines for Machine Learning - Machine Learning Mastery", Machine Learning Mastery, 2017. [Online]. Available: http://machinelearningmastery.com/support-vector-machines-for-machine-learning/. [Accessed: 30- Mar- 2017].

[28] "Random Forests Algorithm", Datasciencecentral.com, 2017. [Online]. Available:http://www.datasciencecentral.com/profiles/blogs/random-forests-algorithm. [Accessed: 30- Mar- 2017].