Chrome Extension for Detecting Phishing Websites

Bhavya Shah[1], Karan Dharamshi[2], Mihir Patel[3], Dr.Vaishali Gaikwad[4]

[1,2,3] *Student* [4]*Associate Professor*
[1,2,3,4] *Dept. of Information Technology, K.J. Somaiya Institute of Engineering & Information Technology, Mumbai, Maharashtra, India.*

---***---

Abstract - *Phishing is a social engineering attack in which a hacker or an attacker attempts to steal user's private, sensitive data and tries to manipulate by using a malicious fake link which exactly looks like a legitimate link. When a user clicks on that link, he/she is redirected to the hacker's webpage instead of the legitimate one. User thinking that it's legitimate, provides them the sensitive data and user's data gets compromised. Due to this, there is a huge loss of one's individual data and there can also be a loss of the company's data when the employee of the organization is compromised. The traditional approach to phishing detection was to make a database with blacklisted websites list and the phishing links associated with it and was checked against the entered link in to check the entered or visiting link is present in the phishing database or not. But there is a problem with heuristic exhaustive search where if the phishing link is not present in the database then there will be a problem and the user may visit the website. To overcome the exhaustive search problem we use a machine learning algorithm where we use the website's link and will validate them based on the URL features and will decide them as legitimate or phished. We tested various algorithms and selected the Random Forest algorithm to apply on our dataset and created a chrome browser extension or also called as chrome based plugin.*

Key Words: Chrome Plugin, Phishing Recognition, Random Forest Algorithm, URL Feature Extraction, Web based Security

1. INTRODUCTION

Phishing is a very common social engineering attack these days. It has disrupted people's lives as well as affected several organizations by stealing sensitive information using different manipulation techniques. There are various methods to perform phishing using E-mail, Click baiting ads, Whaling, Spear-Phishing and many more. Attackers clone an official website into a website that looks legitimate to the user which is actually managed by the attacker. The individuals end up providing their credentials and other sensitive data to the cloned website which is then used by the attacker for malicious purposes.

In [1] 2013, cybersecurity studies found that one-third of all phishing attacks are aimed at stealing your money. More than 30 percent of attacks used the names of leading banks, payment systems and online stores, including MasterCard, Visa, and American Express. The most common brand name used that year was Amazon.com, with Apple and eBay rounding out the top-three.

Phishing is not limited to email and website pop-ups. Links in online ads, status updates, tweets, and Facebook posts can lead you to criminal portals designed to steal your financial information. Various methods have been implemented to avoid phishing but have not been very effective against it. Many organizations and companies have been organizing campaigns on cybersecurity measures to be taken against phishing. This traditional approach hasn't been very successful and has led to the rise of various phishing detection software and plugins developed with different kinds of technology year after year.

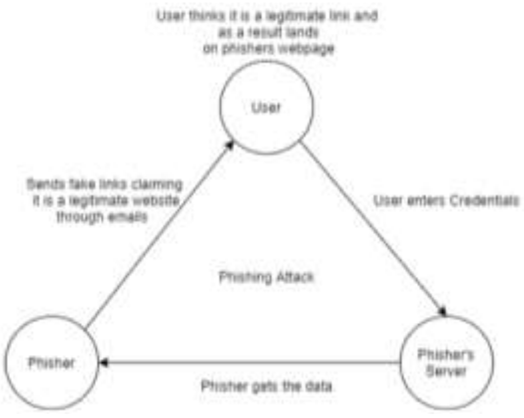

User thinks it is a legitimate link and as a result lands on phishers webpage
User
Sends fake links claiming it is a legitimate website through emails
User enters Credentials
Phishing Attack
Phisher
Phisher's Server
Phisher gets the data

Fig 1: Flow of a general Phishing Attack

From the above figure 1 in a phishing attack, bait often appears as a compelling email. Attackers go to great lengths to ensure that their emails appear as legitimate as possible. These emails most commonly direct target recipients to an attacker-controlled website that delivers malware or intercepts user credentials.

Example of a standard attack:
1. An attacker sends a mass email to employees posing as a member of the IT department.
2. The email is a notification for recipients to take the mandatory annual online IT security training module—however, the training module is attacker controlled.
3. During the course, the victim user is directed to enter their employee credentials which are then delivered directly to the attacker.

2. RELATED WORK

We have studied many papers so far and we have come across the different techniques and technologies which are used to develop the system and software to avoid or detect the phishing.

In [2] Zou Futai, Gang Yuxiang, Pei Bei, Pan Li, Li Linsen "Web Phishing Detection Based on Graph Mining," 2016 in IEEE; it uses network Phishing Detection database mining methodology. This will spot any possible phishing that cannot be identified by the review of the URLs. Uses the user-website browsing arrangement. To get the data collected from a Big ISP's real traffic. A special AD is allocated to the client customer, but a generic address is chosen from the ISP's own Address database. Therefore, we create a visit relationship graph with the AD and URL, called the AD-URL graph, and the Phishing website is identified by the graph's reciprocal actions.

In [3] Nick Williams, Shujun Li “Simulating Human detection of phishing websites: An investigation into the applicability of ACT-R cognitive behavior architecture model”, 2017 in IEEE; they suggested a framework that analyzes the concept of ACT-R cognitive-behavioral architecture. Simulate the neural mechanisms involved in assessing the authenticity of a specific website based mainly on the HTTPS padlock secure predictor features. ACT-R possesses good abilities that model the phishing use case well and that further research to more accurately reflect the spectrum of human security awareness and activities in an ACT-R system may lead to deeper insights into how best to integrate technology and human protection to reduce the probability of phishing attacks for users.

In [4] Xin Mei Choo, Kang Leng Chiew, Dayang Hanani Abang Ibrahim, Nadianatra Musa, San Nah Sze, Wei King Tiong, “Feature-Based Phishing Detection Technique”,2016 in JATIT; a tool for detecting Phishing website is focused on analyzing legitimate website log information. Whenever a person enters the phishing webpage, the phishing webpage may refer by demanding services to the legal webpage. There will now be a report registered by the official website server and from this log, you will identify a Phishing site.

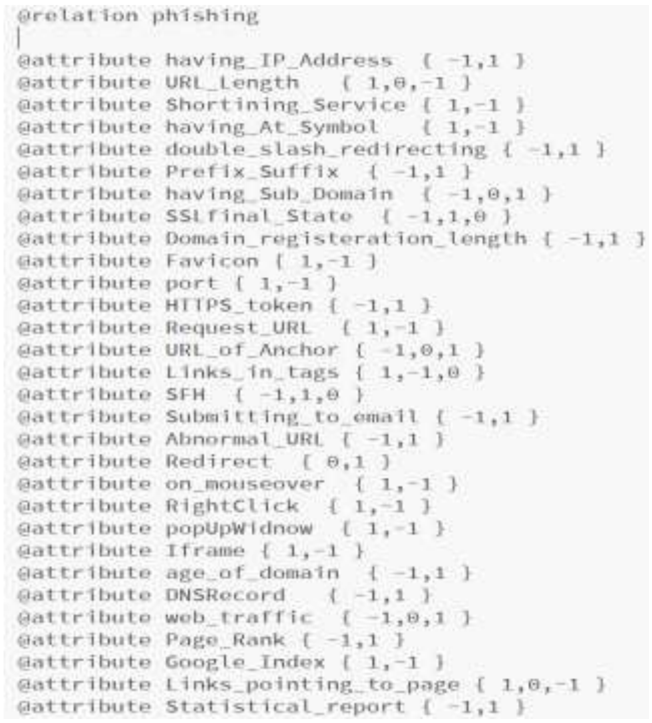
In [5] Giovanni Armano, Samuel Marchal and N. Asokan “Real-Time Client-Side Phishing Prevention Add-on”, 2016 in IEEE; they have suggested a phishing site identification method using Novel Algorithm. This identification algorithm will calculate the highest number of phishing URLs as it executes several checks, such as Blacklist check evaluation, Alexa ranking check, and various URL functionality analysis.

3. DATASET

For our proposed system we are using the dataset which has been referred from the UCI machine learning repository. The phishing websites attribute dataset consists of around eleven thousand URLs (examples) which consists of around six thousand phishing cases and some five thousand real incidents. All these examples have around twenty-five features and every feature is connected to the set of decisions and follows according to the set of rules.

The below figure 2 shows the set of attributes and features are segregated into different groups:

i) Features based on address bar
ii) Features based on abnormality
iii) Features based on JavaScript and HTML
iv) Features based on Domain systems



```
@relation phishing
|
@attribute having_IP_Address  { -1,1 }
@attribute URL_Length   { 1,0,-1 }
@attribute Shortining_Service { 1,-1 }
@attribute having_At_Symbol   { 1,-1 }
@attribute double_slash_redirecting { -1,1 }
@attribute Prefix_Suffix  { -1,1 }
@attribute having_Sub_Domain  { -1,0,1 }
@attribute SSLfinal_State  { -1,1,0 }
@attribute Domain_registeration_length { -1,1 }
@attribute Favicon { 1,-1 }
@attribute port { 1,-1 }
@attribute HTTPS_token { -1,1 }
@attribute Request_URL  { 1,-1 }
@attribute URL_of_Anchor { -1,0,1 }
@attribute Links_in_tags { 1,-1,0 }
@attribute SFH  { -1,1,0 }
@attribute Submitting_to_email { -1,1 }
@attribute Abnormal_URL { -1,1 }
@attribute Redirect  { 0,1 }
@attribute on_mouseover  { 1,-1 }
@attribute RightClick  { 1,-1 }
@attribute popUpWidnow  { 1,-1 }
@attribute Iframe { 1,-1 }
@attribute age_of_domain  { -1,1 }
@attribute DNSRecord   { -1,1 }
@attribute web_traffic  { -1,0,1 }
@attribute Page_Rank { -1,1 }
@attribute Google_Index { 1,-1 }
@attribute Links_pointing_to_page { 1,0,-1 }
@attribute Statistical_report { -1,1 }
```

Fig 2: Attributes in dataset

4. IMPLEMENTATION DETAILS

The dataset was placed in proportion 7:3 to plan and check. The consequences of our research will be provided in the area of tests.

A. Random Forest Classifier (Algorithm)

Random forests are optimization algorithms that incorporate several tree predictors, based on the values of the random variable sampled independently.

In fact, the same distribution occurs for all forest trees. The Random forest algorithm can give the right prediction of a class for most of the data. But there are few mistakes which are also made by the trees in some places. So for the model to predict more precisely, we decide to observe the class on the poll result by conducting the vote for each observation. As a result, the class decides the result more accurately.

In our proposed system we use this algorithm to predict if the web URLs are phishing or not. We use this algorithm for the classification of features on the data to predict the class from it. It consists of multiple instances with multiple outcomes. On that basis, the different decision trees are formed according to different features and outcomes and with the same features with multiple outcomes. The final result is based on the majority pole of the multiple decision trees altogether and the class prediction is based on it. In addition, an internal unbiased calculation of the generalization error is produced during the forest construction process. In fact, incomplete data can be well calculated. The loss of reproducibility is a big downside to wild forests, as the method of creating the forest is arbitrary. Furthermore, it is difficult to understand the final model and the resulting effects, since it includes several different variables decisions trees.

The suggested solution aims to develop a plugin extension powered by state-of - the-art machine learning technologies for phishing detection concerning chrome browser as its working platform. The below figure 3 depicts the same flow of our system.

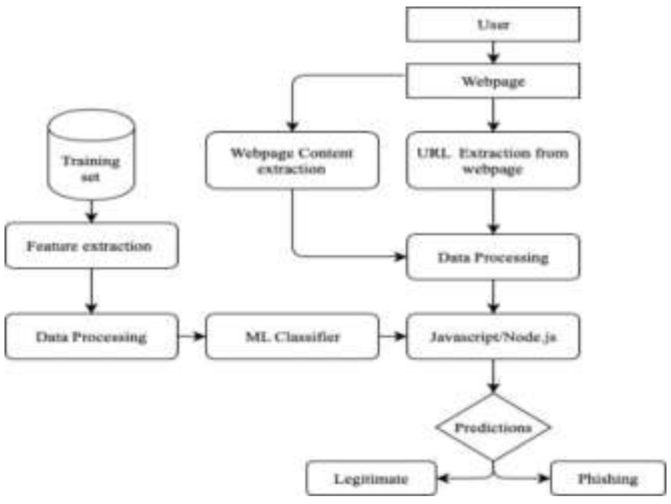
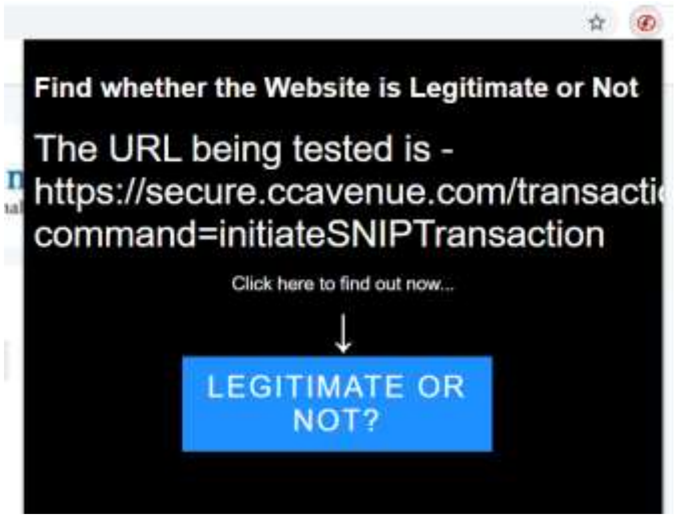


Fig 3: Flow chart of our Phishing Detection system

B. Results

UI Design: (Functional Result)

A simple and easy to use User Interface has been designed for the plugin using HTML and CSS and JavaScript and Chrome Manifest. The below figure 4 shows our User Interface of Chrome Plugin.


Find whether the Website is Legitimate or Not

The URL being tested is - https://secure.ccavenue.com/transacti command=initiateSNIPTransaction

Fig 4: User Interface of the Chrome Plugin

Pre–Processing on Dataset (Non-functional Result):

The phishing dataset which is referred from the repository is loaded by sci-kit learn library into the array. The phishing dataset has around 25 instances that need to be extracted through the classifier. This feature extraction plays an important role in classifying the class because each feature can have multiple outcomes. Preparing more features for classification helps in predicting the class more accurately. By using the k-fold validation technique the referred phishing dataset using the k-fold validation technique the referred phishing dataset is split into training and testing set by 7:3 ratio respectively.

The below figure 5 shows the output of Preprocessing we have done on our dataset.

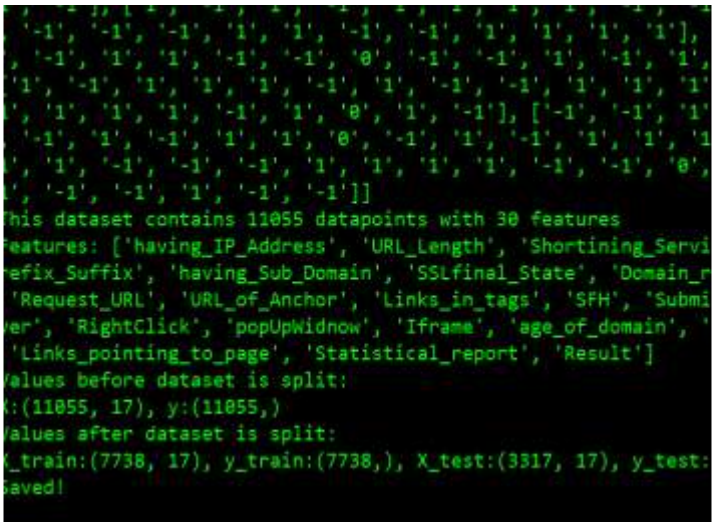
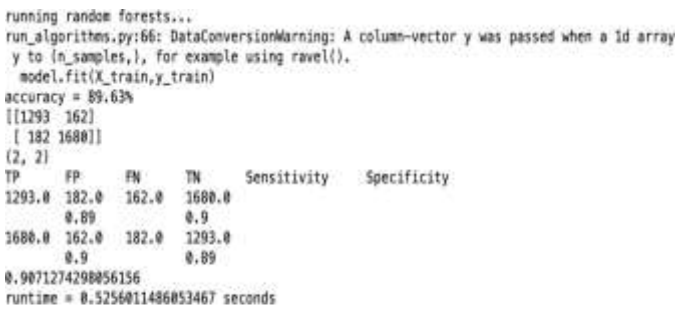

```
 '-1', '-1', '-1', '1', '1', '-1', '-1', '1', '1', '1', '1'],
, '-1', '1', '1', '-1', '-1', '0', '-1', '-1', '1', '-1', '1',
'1', '-1', '1', '1', '1', '-1', '1', '-1', '-1', '1', '1', '1'
', '1', '1', '1', '-1', '1', '0', '1', '-1'], ['-1', '-1', '1'
 '-1', '1', '-1', '1', '1', '0', '-1', '1', '-1', '1', '1', '1
', '1', '-1', '-1', '-1', '1', '1', '1', '1', '-1', '-1', '0',
', '-1', '-1', '1', '-1', '-1']]
his dataset contains 11055 datapoints with 30 features
eatures: ['having_IP_Address', 'URL_Length', 'Shortining_Servi
efix_Suffix', 'having_Sub_Domain', 'SSLfinal_State', 'Domain_r
 'Request_URL', 'URL_of_Anchor', 'Links_in_tags', 'SFH', 'Submi
er', 'RightClick', 'popUpWidnow', 'Iframe', 'age_of_domain', '
 'Links_pointing_to_page', 'Statistical_report', 'Result']
alues before dataset is split:
:(11055, 17), y:(11055,)
alues after dataset is split:
_train:(7738, 17), y_train:(7738,), X_test:(3317, 17), y_test:
aved!
```

Fig 5: Pre Processing of Dataset

The below figure 6 shows analysis of Random Forest algorithm by using Confusion Matrix. The result shows us the accuracy and the runtime. We analyzed Random Forest, Support Vector Machine and Neural Network; after observing all the three algorithms, we selected Random Forest because it gave us precise results and it also takes less time to run.



```
running random forests...
run_algorithms.py:66: DataConversionWarning: A column-vector y was passed when a 1d array
 y to (n_samples,), for example using ravel().
  model.fit(X_train,y_train)
accuracy = 89.63%
[[1293  162]
 [ 182 1680]]
(2, 2)
TP      FP      FN      TN      Sensitivity    Specificity
1293.0  182.0   162.0   1680.0
        0.89            0.9
1680.0  162.0   182.0   1293.0
        0.9             0.89
0.9071274298056156
runtime = 0.5256011486053467 seconds
```

Fig 6: Algorithm Analysis of Random Forest

Precision: It is defined as the fraction of retrieved objects that are relevant. In our case it is the fraction of URLs that are correctly classified as phished which are actually phished.

$$\textbf{\textit{Precision}} = \frac{TP}{TP + FP}$$

Recall can be defined as the ratio of the total number of correctly classified positive examples divide to the total number of positive examples. High Recall indicates the class is correctly recognized (a small number of FN).

$$\textbf{\textit{Recall}} = \frac{TP}{TP + FN}$$

F Measure: The harmonic mean of recall and precision both followed by the formula below:

$$\textbf{\textit{F - measure}} = \frac{2*\text{Recall}*\text{Precision}}{\text{Recall} + \text{Precision}}$$

True Positive Rate: The percentage of phished URLs in the dataset that are correctly classified as phished.

True Negative Rate: The percentage of ham URLs in the dataset correctly classified as ham.

False Positive Rate: The percentage of ham mails incorrectly classified by the model as phished.

False Negative Rate: The percentage of phished URLs that were incorrectly classified by the model as ham.

Training:

After the preprocessing of the phishing dataset now it will be used for training the model. By analyzing and on the basis of the confusion matrix, accuracy, turnaround time and efficiency the classifier is selected. In our case, it is random forest classifier and by using the k-fold method the phishing dataset will go for training under the appropriate split ratio so that the trained model does not face the problem of under-fitting and over-fitting. After the X and Y split of the dataset, the file will be saved separately for training process.

Plugin Feature Extraction:

We are using 16 features which needs to be extracted and encoded for each webpage in real-time while the page is being loaded.

A content script is used so that it can access the DOM of the webpage. The content script is automatically injected into each page while it loads. The content script is responsible to collect the features and then send them to the plugin. The main objective of this work is not to use any external web service and the features need to be independent of network latency and the extraction should be rapid. All these are made sure while developing techniques for the extraction of features. Once a feature is extracted it is encoded into values {-1, 0, 1} based on the following notation.

-1 - Legitimate
 0 - Suspicious
 1 - Phishing

Classification:

The random forest algorithm is used for the classification of the cached vector, where the JavaScript will load the cached file from the storage which is classified by the random forest classifier.
Below are the details used to identify phishing portals:

•isIPInURL(): Identify presence of IP address in the URL
•isLongURL(): Validate if length of the URL is beyond 75 characters
•isTinyURL(): Identify URLs smaller than 20 characters
•isAlphaNumericURL(): Check for alphanumeric '@' in URL
•isRedirectingURL(): Verify if '//' existing within the URL more than once
•isHypenURL(): Check for presence of '-' adjacent to domain name in URL
•isMultiDomainURL(): Domain name should be confined to top-level domain, country-code and second-level domain.
•isFaviconDomainUnidentical(): Verify if links on given webpage are loaded from other domains
•isIllegalHttpsURL(): Identify presence of multiple 'https' in the URL string

•isImgFromDifferentDomain(): Validate if images on given web-page are loaded from other domains
•isAnchorFromDifferentDomain(): Detect if links on given web-page are loaded from other domains
•isScLnkFromDifferentDomain(): Identify if scripts on given web-page are loaded from other domains
•isFormActionInvalid(): Detect invalid/blank form submissions
•isMailToAvailable(): Check for anchor tag incorporating mailto
•isStatusBarTampered(): Checks if manipulated the status bar display or not on MouseOverEvent()
•isIframePresent(): Identify sites, which exhibit iframes in the DOM
The extracted features, further, passed through the Random Forest model to Identify hostile web-URLs.

C. Chrome Plugin API

The approach deals with the preparation of the model with the existing data collection, utilizing the Random Forest discriminative classifier, with certain important feature extraction on the phishing dataset and preparing the platform for the chrome-browser. We developed the plugin by using JavaScript. Additional support was provided by python and other libraries for the integration of JavaScript and JSON objects and vectors for the smooth processing of the entire functionality of identifying the URLs as legitimate of phished. It can be used as an automated extension by the user or any individual of any organization. During the initial phase of the project, we evaluated different methods and, by analyzing the pros, cons and availability of resources, we concluded by the above-implemented technique and achieved the desired results.

5. CONCLUSION

Phishing, which is a social engineering attack has a huge impact on the global scale which destroys the financial and economic value of organizations, government sectors, and individuals by exploiting the data by different phishing techniques. We have overcome different traditional approaches of phishing detection due to exhaustive search and maintaining and updating a blacklist database with the new links which is not a suitable and accurate way of identifying phishing attacks. So here we have proposed a chrome plugin that automates the job of detecting and overcoming traditional methods by using a machine learning algorithm. We have used random forest as our main algorithm after analyzing various other algorithms and selected it based on the confusion matrix and execution rate using the k-fold validation technique. The random forest algorithm gives an accuracy and runtime of 89.60% and 0.59 seconds respectively. Hence the developed chrome-based plugin makes it easier for a user to detect and identify phishing websites.

For potential upgrades and future implementation, we can also report new phishing websites detected by our trained model to the Google Website Blacklist database and page ranking system so that they can be banned or processed.

REFERENCES

[1]https://www.transunion.com/blog/identity-protection/7-facts-about-cyber-security-and-phishing

[2]Zou Futai, Gang Yuxiang, Pei Bei, Pan Li, Li Linsen "Web Phishing Detection Based on Graph Mining", 2016.

[3]Nick Williams, Shujun Li "Simulating Human detection of phishing websites: An investigation into the applicability of ACT-R cognitive behavior architecture model", 2017.

[4]Xin Mei Choo, Kang Leng Chiew, Dayang Hanani Abang Ibrahim, Nadianatra Musa, San Nah Sze, Wei King Tiong, "Feature-Based Phishing Detection Technique",2016.

[5]Giovanni Armano, Samuel Marchal and N. Asokan "Real-Time Client-Side Phishing Prevention Add-on", 2016.

[6]Internal Revenue Service, IRS E-mail Schemes. Available at:https://www.irs.gov/uac/newsroom/consumers-warned-of-new-surge-in-irs-email-schemes-during-2016-tax-season-tax-industry-also-targeted

[7]https://en.wikipedia.org/wiki/Phishing#Technical_approaches

[8]E., B., K., T. (2015). Phishing URL Detection: A Machine Learning and Web Mining-based Approach. International Journal of Computer Applications, 123(13), 46-50. doi:10.5120/ijca2015905665

[9]Wang Wei-Hong, L V Yin-Jun, CHEN Hui-Bing, FANG Zhao-Lin., A Static Malicious JavaScript Detection Using Random Forest, In Proceedings of the 2nd International Conference on Computer Science and Electrical Engineering(ICCSEE 2013)

[10]Ningxia Zhang, Yongqing Yuan, Phishing Detection Using Neural Net- work, In Proceedings of International Conference on Neural Information Processing, pp. 714–719. Springer, Heidelberg (2004)

[11]Ram Basnet, Srinivas Mukkamala et al, Detection of Phishing Attacks: A Machine Learning Approach, In Proceedings of the International World Wide Web Conference (WWW), 2003

[12]Scikit-learn, library of ANN

[13]Scikit-learn, library of Random forests