International Journal of Scientific Research in Science and Technology
Print ISSN: 2395-6011 | Online ISSN: 2395-602X (www.ijsrst.com)
doi : https://doi.org/10.32628/IJSRST523103124

Phishing Website Detection Based on Machine Learning Algorithm

Prof. A. V. Mote, Hitesh Y. Patil, Prachi R. Patil, Omkar S. Ombase
Department of Computer Engineering, Zeal College of Engineering and Research Pune, Maharashtra, India

ARTICLEINFO

Article History:
Accepted: 05 May 2023
Published: 26 May 2023

Publication Issue
Volume 10, Issue 3
May-June-2023

Page Number
562-570

ABSTRACT

Phishing websites utilise a variety of techniques to imitate the URL address and page content of a legitimate website in order to steal users' personal information. In this study, we analyse the structural properties of the phishing website URL, extract 12 different types of data, and train four machine learning algorithms. Then, in order to identify unknown URLs, utilise the method that performed the best as our model. The recommendation of the original regular web page of the phishing web page is implemented after a snapshot of the web page is extracted and compared with the regular web page snapshot.

Keywords—Phishing Website Detection, Machine Learning, Webpage Similarity Comparation

I. INTRODUCTION

"Phishing" is a type of cyber fraud in which criminals spoof a legitimate website's URL address and page content using various methods, or they use flaws in the server programme of a legitimate website to introduce harmful HTML code into specific pages of the site. Use this to steal users' passwords, bank or credit card account numbers, and other personal information. Phishing prevents the further growth of the Internet and costs internet users money. In the present context, phishing prevention is essential. Even if the user's caution and experience are crucial, they cannot entirely guard against the user falling for a phishing scam [1]. Because attackers take into account the psychological traits of end users to boost the success rate of phishing assaults, particularly when tricking reasonably seasoned people [2]. End-user-focused cyberattacks have caused serious loss of sensitive and private information as well as financial losses to individuals totaling $1 billion US in only one year [3]. According to Fig. 1, which uses the Bank of China example, fraudsters trick consumers by replacing the ".com" in the URL with ".cOm," which is the most common technique employed by phishing websites. Phishing aims to obtain sensitive and private information from the victim by utilising this kind of bogus URL, including usernames, passwords, financial information, and other details [4]. In their article, Tanuja K. Sarode and Bhagyashree E. Sananse gathered phishing and non-phishing websites, created a database, used this information to train using random forest algorithms, and identified unfamiliar URLs. [5]. Eight different machine learning algorithms are used for detection in the article by Routhu Srinivasa Rao and Alwyn Roshan Pais, with random forest demonstrating the best results [6]. Seven different machine learning and natural language processing methods were utilised in the publications by Ozgur Koray Sahingoz, Ebubekir Buber, Onder Demir, and Banu Diri, and the outcomes were enhanced and compared. The outcomes remain the same best performance of random forests [7]. This system implements a simple phishing URL filtering system. The basic functions are:

- Identification and detection of phishing websites
- Implement the recommendation of regular websites corresponding to phishing websites.

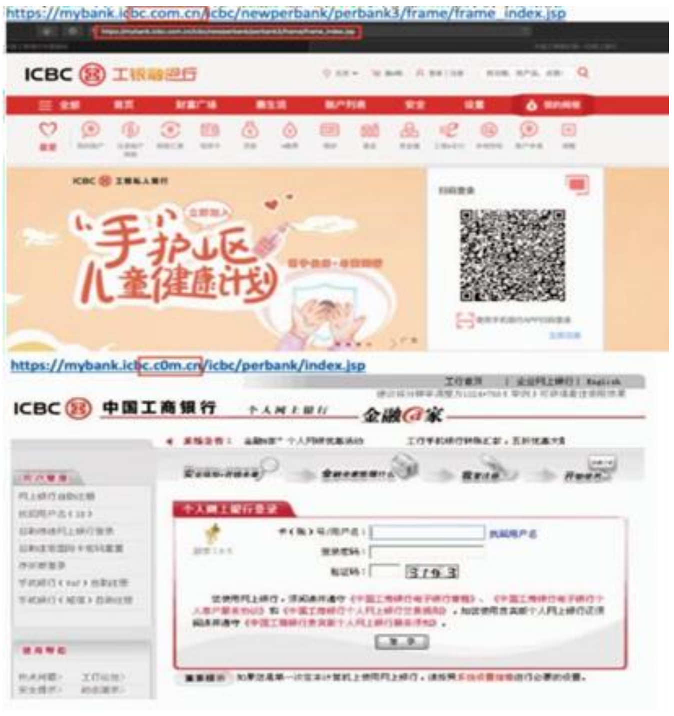

https://mybank.icbc.com.cn/icbc/newperbank/perbank3/frame/frame_index.jsp
ICBC 工银融e行
https://mybank.icbc.c0m.cn/icbc/perbank/index.jsp
ICBC 中国工商银行
金融@家

Fig. 1. An example of Phishing website

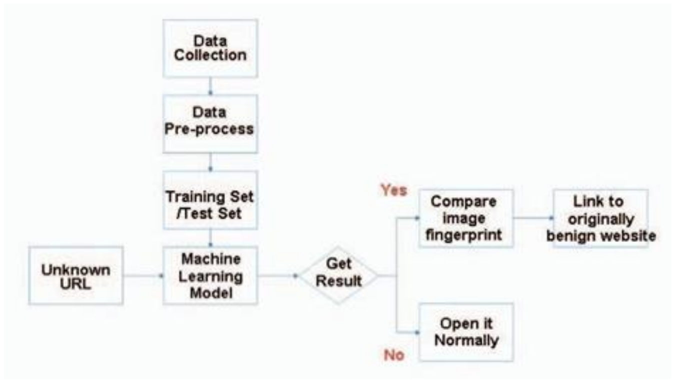


Fig. 2. The overall process of phishing website filtering and regular website recommendation system

Since we are monitoring online phishing websites [8], we still need to consider time consumption, so we compared four different machine learning algorithms and selected the one with the highest time efficiency and accuracy as the algorithm we need. As shown in Fig. 2, it is the flow of the entire system.

II. DATA COLLECTION

Start by gathering web pages using web collection. The objective of collecting is to efficiently gather as many usable interfaces as possible while also obtaining the link structure that connects these sites. This function is accomplished via a web collector. Set up the collected seed collection by first selecting one or more URLs. Select a URL from the seed collection to collect after that. Examine the pages that were collected and extract the content and links from them. The extracted URL is added to the URL pool to be collected, while the extracted text is fed into the text indexer. All of the URLs for the web pages that need to be collected are always present in the URL pool. The seed collection will initially be added to the URL pool. After being gathered, a URL will be removed from the pool. The complete collecting procedure, as depicted in Fig. 3, can be thought of as the Web graph's traversal procedure.

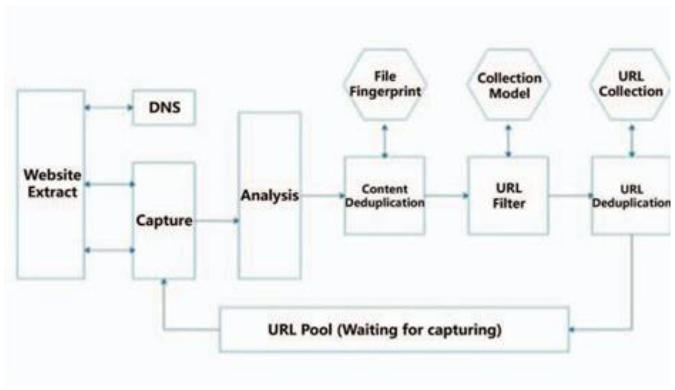


Fig. 3. web collector architecture

- The URL pool holds the most recent URL to be gathered.
- When crawling a webpage at a URL, the DNS resolution module is utilised to identify the IP address of the relevant Web server.
- Capture module: use the http protocol to retrieve the website linked to a particular URL.
- Analyse the collected web pages to extract content and links.
- The URL deduplication module checks extracted links to see if they have recently been crawled or have been in the URL pool.

A single URL is collected through a variety of steps, such as URL crawling, various results detection and filtering, etc., before being added back to the URL pool. A collecting thread first chooses a URL from the URL pool, after which it crawls the website that the URL points to.

The text and links were then removed from the webpage once it had been examined. Each retrieved link's information is also put though a battery of tests to see if it should be added to the URL pool. The URL filter then employs a number of checks to decide whether or not to add an extracted URL to the URL pool. For instance, the gathering procedure may disallow some domains (such all URLs ending in.com). In this case, we only need to filter out the URLs for every.com domain. The URL must then be checked again; if it is already in the URL pool or has been collected, there is no need to add it. A URL is given a priority when it is added to the URL pool, and this priority determines when the URL is finally dequeued for actual collection. We already mentioned that in a distributed system environment, the threads in the collector can run in several processes on various nodes. A hash function or other more specialised targeted methods can be used to distribute the host to be collected to each node. For instance, European collectors mostly concentrate on the collection. Of course, this method is not entirely trustworthy. Among the causes is the fact that the Internet's data packet transmission path does not always correspond to a location's vicinity. How do these many nodes exchange URLs and communicate with one another? The plan is to duplicate the flow on each collecting node that is depicted in the above diagram. After URL filtering, we must distribute the URLs found by filtering to various collection nodes using a host divider. In other words, the host objects that need to be collected will be dispersed among several nodes. Fig. 4 depicts the updated acquisition flow chart. The duplicate URL detection module of each collection will receive the output result of the host divider node of the distributed system.

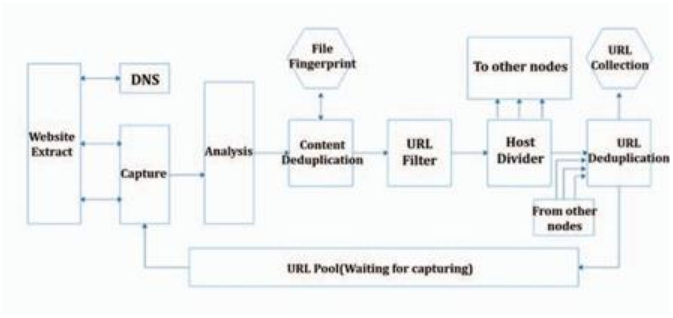


Fig. 4. Distributed web collector architecture

III.MACHINE LEARNING MODEL CONSTRUCTION

A. Data source selection

We chose harmful websites from the phishtank directory (3547) and good websites from the dmoz directory (3511).

A blacklist of phishing URLs made up of manually validated URLs submitted by users may be found on the Phishtank website (https://www.phishtank.com), which counts phishing websites by manually evaluating and reporting URLs. Some of the website's APIs can provide you with a.json file containing harmful URLs and relevant data, and you can use regular expressions to extract these URLs.

A worldwide open classified directory is the Dmoz directory. People from all over the world who are volunteers work together to build and maintain it. It includes all the trustworthy websites that made it through the review. It is ideal for serving as a safe website data source.

B. Feature selection

We select twelve kinds of features as shown below:

1) URL length [9]

The URL may be quite long since phishing websites often use large URLs to conceal suspicious content in the address bar.

2) The number of symbols such as <.> <;> <?> In the URL.

3) Whether the domain name contains an IP address [10].

If you use the IP address as a substitute for the domain name in the URL, such as <http://125.98.3.123/fake.html>, it means that someone may be trying to steal their personal information. Sometimes, the IP address will even be converted to hexadecimal code, as shown in the following link<http://Ox58.OxCC.OxCA.Ox62/2/paypal.ca/index.html>.

4) The number of <@> in the URL.

Using the <@> symbol in the URL will cause the browser to ignore everything before the <@> symbol, and the actual address usually follows the <@> symbol. Criminals may use this method to visually confuse URLs.

5) Whether there is a symbol <//>.

The presence in the path <//> means that the user will be redirected to another website. An example of such a URL is: <http://www.legitimate.com//http://www.phishing.com>. Will check if it exists <//>.

6) The number of </> in the URL.

7) The number of subdomains.

8) The length of the domain name.

9) Path length.

10) Whether it is HTTPS protocol.

Through observation, we can think that ordinary malicious URLs do not encrypt the transmission of their own web content, and most benign websites will use the HTTPS protocol.

11) URL word segmentation features

12) Host information characteristics

The fundamental justification for using WHOIS hosting information is that the malicious website might be accessible through a registrar with a bad reputation, or it might be hosted on a device that isn't often used for web hosting. Additionally, the update cycle on harmful websites is typically quicker. They must develop new domain names and submit applications after being blocked. Therefore, compared to innocuous web pages, their registration dates are typically shorter. In some ways, WHOIS data can be thought of as the location, identity, and management style of a malicious website. In this experiment, the WHOIS information about the URL's registration date, update date, and validity period are extracted package of Python.

C. Feature evaluation

The parameter k can be supplied into the SelectKBest () method in the Sklearn module, and the top k features with the best performance can be output by chi-square detection. These characteristics are most independently of one another at the same time, and the discrimination is good. The top six features among them are (text features following word segmentation are not taken into consideration):

1) WHOIS information

The initial step is to group the WHOIS data that has performed the best. Evidently, normally good websites have been registered for a while, and bad websites frequently need to be reregistered after being removed by security experts, for example. The distinction is better and the categorization effect is improved because the registration date is frequently the closest. has a motivating impact;

2) URL length

Next is the length of the URL that ranks second in performance. Fig. 5 shows the distribution of the length of 7O3O URLs in the training sample according to the labels they are labeled; blue represents benign URLs, and red represents malicious URLs. From the figure, it can be seen that the length of all URLs are basically concentrated in O ~ 2OO characters, of which benign URLs are basically distributed in the range of 25 ~ 3O, while malicious URLs are basically distributed in the range of 4O ~ 75. Although there is a partial intersection between the two, the distinguishing effect is still obvious, you can see the length of the malicious URL Relatively speaking, it will be longer than a benign URL, so in terms of discrimination, URL length is a relatively good feature.

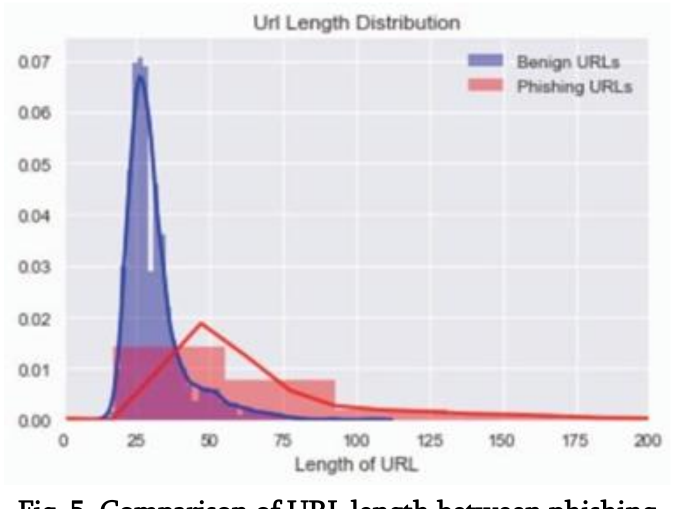


Fig. 5. Comparison of URL length between phishing and non-phishing websites

3) Path length;

4) Number of symbols <//>;

5) Number of symbols <->,

6) The length of the domain name;

The remaining five feature classification effects are poor.

D. Classifier model construction

1) Logistic regression classifier

As seen in Fig. (a), the accuracy is achieved without taking into account the text features by dividing the True Positive, True Negative, False Positive, and False Negative of the logistic regression classifier into 4: 1 for training and testing of the data set. About O.85O64O113798 is the rate. The logistic regression classifier performs reasonably well when there are few features, as shown in the picture below. Data preparation is necessary since the training data set has a higher influence on the logistic regression classifier's ability to identify dangerous URLs. Reduce as much noise as you can in the data collection, and make sure the characteristics are as independent as you can.

The classifier's training time is obviously extended when text features of approximately 3,OO dimensions are included, and the accuracy rate likewise increases to O.951258741254. As can be shown, the classification effect of the logistic regression classifier is greatly improved by the high-dimensional text characteristics.

2) SVM classifier

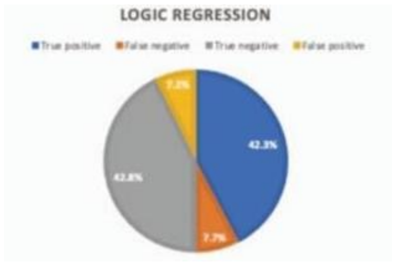
The support vector machine classifier's True Positive, True Negative, False Positive, and False Negative values are displayed in Fig. (b), and the segmentation, which is produced without taking into account the text features, is tested and trained using the data set's 4: 1. The precision rate is around O.889758179232. K-means is used to cluster the data, and the iterative process divides the data into multiple clusters that are independent of one another and have the effect of reducing the dimension. The accuracy of the SVM classifier achieved O.951578521436 after dimensionality reduction.

3) Naive Bayes classifier

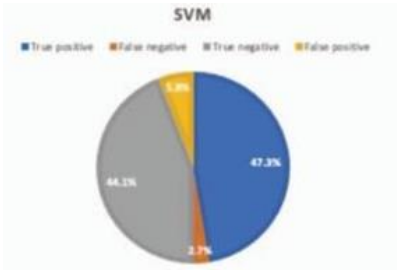
Without taking into account the text features, the Naive Bayes classifier's True Positive, True Negative, False Positive, and False Negative probabilities are displayed in Fig. (c), and a 4:1 ratio is utilised to train and test the data set. The obtained accuracy is around O.7OO56899OO43.

4) Decision tree classifier

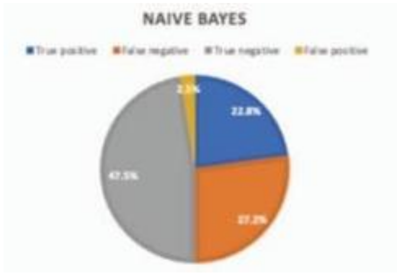
Figure (d) depicts the decision tree classifier's True Positive, True Negative, False Positive, and False Negative values. The data set's 4: 1 ratio is used for segmentation training and testing. About O.9OO426742532 is the rate.



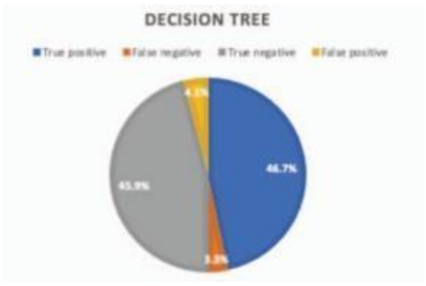
(A) The result of Logic Regression



(B) The result of SVM



(C) The result of Naïve Bayes



(D) The result of decision tree

Fig. 6. Comparison of the results of four machine learning algorithms

The L1 logistic regression classifier with a faster classification speed is chosen as the classifier in the detection module because the plug-in must return the detection result of the URL as soon as feasible while the user is browsing the web. The link results supplied by each search engine must be obtained by the detecting system. Through study and comparison of the webpage's source code, it is discovered that all URL results are included since these URLs are all located in the DOM tree of the original webpage and follow specific rules. A page can be found in the href> element of each child node of the DOM tree's "div id = "b_content"> has a total of ten search results, use the <find> function inJquery and store it in the array urls. Five URL statics, including WHOIS information, URL length, path length, number of symbols //>, and number of symbols -> are chosen instead of the TF-IDF text features with very high space costs. Features ensure that the feature vector set establishment and classifier decision processes move quickly. According to measurements, the plug-in requires 2–3 seconds to identify the URL and print the result, which has no impact on the user's regular online browsing. A search result page typically returns 1O URLs. The entire procedure is displayed in Fig. 7.

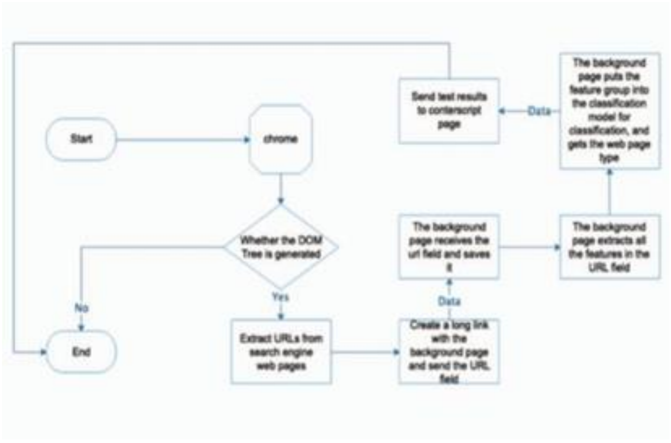

Start
chrome
Whether the DOM Tree is generated
No
Yes
End
Extract URLs from search engine web pages
Create a long link with the background page and send the URL field
Data
The background page receives the url field and saves it
The background page extracts all the features in the URL field
The background page puts the feature group into the classification model for classification, and gets the web page type
Data
Send test results to contenscript page

Fig. 7. Online detection process of phishing websites

IV.REGULAR WEBSITE RECOMMENDATION FOR PHISHING WEBSITES

A. Web snapshot acquisition

The process is shown below in Fig. 8,

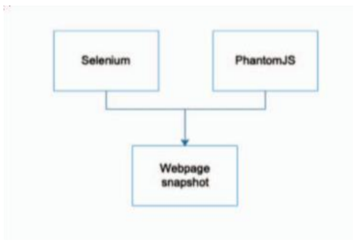


Fig. 8. Web snapshot acquisition

Selenium is a tool for web application testing, running directly in the browser;

PhantomJS is a JavaScript API based on webkit, using QtWebKit as the core browser function, equivalent to an invisible browser.

B. Web snapshot feature calculation

- Pretreatment

Grayscale the color image, and then use bicubic interpolation to normalize the size of the image to unify it into an image Ik×k of size k× k. In this system, k takes 256.

- SIFT feature extraction

Using the SIFT algorithm, extract the locally stable feature points of the image Ik×kto obtain the feature vector set of the web page snapshot, as shown in formula (1):

$F = \{F1, F2, \cdot\ \cdot\ \cdot , Fn\}, F \in Z128$ (1

Where Fi represents a 128-dimensional SIFT feature vector extracted, and the set F is an n × 128-dimensional feature vector, which represents a set of all locally stable feature points extracted.

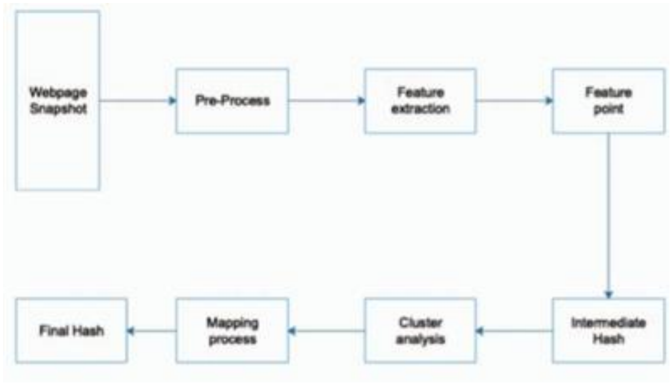


Fig. 9. Web snapshot feature calculation

- Feature point compression

The rows of the feature vector F are summed to obtain an intermediate hash G of 1 × 128, which realizes the purpose of compressing the feature matrix data, as shown in formula(2).

The value range of Sam is [0,1], and the similarity between webpage snapshots is determined by judging the size relationship between sam and threshold. The benchmark webpage snapshot and the snapshot of the webpage to be tested are compared when sam is higher than the predetermined threshold; if they are not similar, they are not compared and are not connected back to the benchmark webpage. When the threshold value is 0.9, according to experimental verification, it has a greater accuracy rate for determining the similarity of webpage snapshots.

n

G(i) = $\sum F$i,j

j=1 (2)

- Cluster analysis

First calculate the distribution centroid C of the intermediate hash G:

$\sum n\ Fn$

C = i=1

n

(3)

Among them, the centroid point is the mean value of the vector. Here, the cluster type is determined according to the centroid C, those smaller than C are classified as $C1$ clusters, and those larger than C are classified as $C2$ clusters.

- Mapping processing

According to the clustering result, the intermediate hash is mapped to 0, 1 string:

0 G(i) $\in$ $C1$

g(i) = {1 G(i) $\in$ C (4)

C. Comparison of webpage snapshot similarity

The process is shown below in Fig. 10.

After obtaining the 128-bit image hash hT of the webpage snapshot to be tested, calculate the Hamming distance from the image hash $h0$ of the reference webpage snapshot;

Calculate the similarity between the webpage snapshot to be tested and the benchmark webpage snapshot:

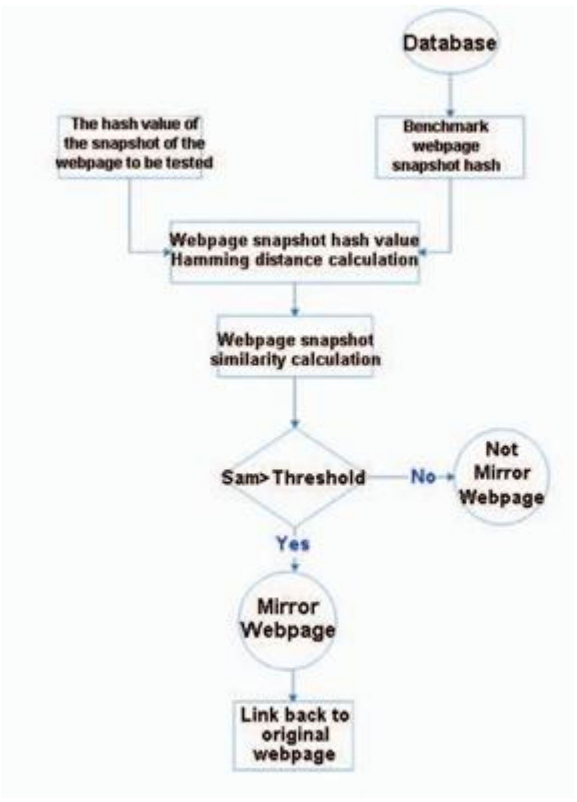

Database
The hash value of the snapshot of the webpage to be tested
Benchmark webpage snapshot hash
Webpage snapshot hash value Hamming distance calculation
Webpage snapshot similarity calculation
Sam>Threshold
No
Not Mirror Webpage
Yes
Mirror Webpage
Link back to original webpage

Fig. 10. Comparison of webpage snapshot similarity

V. CONCLUSION

12 effective features are chosen as the features of our machine learning model for phishing website identification in this research through inquiry and data analysis. And we choose the best option, L1 logistic regression classifier, by Among them, $C1 < C2$, the 128-bit final hash value is obtained.
comparing the True Positive, True Negative, False Positive, and False Negative of the four machine learning models. The most significant advancement of this article is the ability to apply recommendations for legitimate websites that correlate to phishing websites, as there are various ways to identify phishing websites. This article extracts the webpage snapshot first, uses the SIFT method to calculate the webpage snapshot's features, compares the webpage snapshots' similarity to produce a similarity recommendation for the regular website related, and ultimately to the phishing website.

VI. REFERENCES

[1]. Greene, K., Steves, M., & Theofanos, M. (2018). No phishing beyond this point. Com- puter, 51(6), 86–89.

[2]. Curtis, S. R., Rajivan, P., Jones, D. N., & Gonzalez, C. (2018). Phishing attempts among the dark triad: Patterns of attack and vulnerability. Computers in Human Behav- ior, 87, 174–182.

[3]. Shaikh, A. N., Shabut, A. M., & Hossain, M. A. (2016). A literature review on phishing crime, prevention review and investigation of gaps. In 10th international con- ference on software, knowledge, information management & applications (SKIMA) (pp. 9–15). 2016.

[4]. Gupta, B. B., Arachchilage, N. A. G., & Psannis, K. E. (2018). Defending against phishing attacks: Taxonomy of methods, current issues and future directions. Telecommunication Systems, 67(2), 247–267.

[5]. Bhagyashree E. Sananse, Tanuja K. Sarode, Phishing URL Detection: A Machine Learning and Web Mining-based Approach

[6]. Routhu Srinivasa Rao, Alwyn Roshan Pais. Detection of phishing websites using an efficient feature-based machine learning framework.

[7]. Ozgur Koray Sahingoz , Ebubekir Buber , Onder Demir , Banu Diri, Machine learning based phishing detection from URLs

[8]. Aaron Blum, Brad Wardman, Thamar Solorio, Gary Warner, Lexical Feature Based Phishing URL Detection Using Online Learning.

[9]. Doyen Sahoo, Chenghao Liu, Steven C.H. HOI, Malicious URL Detection using Machine Learning: A Survey.

[10]. Sujata Garera, Niels Provos, Monica Chew, Aviel D. Rubin, A Framework for Detection and Measurement of Phishing Attacks.

Cite this Article

Prof. A. V. Mote, Hitesh Y. Patil, Prachi R. Patil, Omkar S. Ombase, "Phishing Website Detection Based on Machine Learning Algorithm", International Journal of Scientific Research in Science and Technology (IJSRST), Online ISSN : 2395-602X, Print ISSN : 2395-6011, Volume 10 Issue 3, pp. 562-570, May-June 2023. Available at doi : https://doi.org/10.32628/IJSRST523103124
Journal URL : https://ijsrst.com/IJSRST523103124