WEB PHISHING DETECTION

A PROJECT REPORT

Submitted by

DEEPIKA S **(130719205013)**

JAYANTHINI M **(130719205021)**

LAKSHAYA S **(130719205026)**

SALATH ROBINSON **(1307192050)**

in partial fulfilment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

JERUSALEM COLLEGE OF ENGINEERING

(An Autonomous Institution, Affiliated to Anna University, Chennai)

ANNA UNIVERSITY, CHENNAI 600 025

NOVEMBER 2022

INTRODUCTION

Project overview:

Phishing emails are a routine occurrence for anyone with email in the Internet age. Masking themselves as reputable companies such as using devious means such as the use of company logos and standards, malicious actors, again and again, attemptto lure in individuals from a wide range of technical shades. Phishing ranges extensively in sophistication: from mass-produced misspelt requests for overly specific details to sophisticated spear phishing attacks focused on the details of the individual. The ability of phishing attacks to innocuously harvest your private credentials can leave you mercilessly exposed in our data-intensive world. All it takes is for a user to make the critical mistake of clicking on a single malicious link. Through a simple mistake, a user exposes themselves and their data from anything from drive-by downloads, cross-site scripting attacks to the harvesting of their details in an innocuous web form. Phishing has two main delivery vehicles: emails and websites. Emails being the foremost of these, are most classically associated with phishing. These often include malicious URLs to direct users towards maliciously crafted content. By obfuscating the real destination of a URL through a few simple manipulations, users can quickly find themselves on unknown and insecure ground. Therefore it is vital to tackle this massive worldwide problem. In the United Kingdom (UK) alone, phishing is expected to cost the UK economy as much as £280 million per year. This is encouraging companies such as Google to look into the future of Uniform Resource Locators (URLs) themselves. To tackle the problem of phishing, my project has been focused on tackling the malicious URLs included in them as ‒more than 75% of phishing mails include malicious URLs to phishing sites‖. Existing techniques to handle URLs involve automated phishing detecting (mainly employing machine learning techniques), user training (the best results of which are gained fromembedded training) and automated security indicators (providing information to help the users decide). I aim to create a system which incorporates aspects of these techniques, to inform and protect users from malicious.

Purpose:

To solve this problem, I have been building a user-focused tool which seeks to catch problematic URLs before they reach their full malicious potential. The tool that I have developed is a Phishing Learning and Detection tool, built for the Chrome platform in the form of an extension called Catch-Phish. It classifies URLsinto one of three safety states using a combination of natural language processing and knowledge acquisition, before providing users with the necessary information to understand how this classification was derived. It helps to train them in URL safety by presenting this information at critical points of intervention. The primary motivation behind this project is the lack of tools which purposefully prevent users from visiting malicious URLs and more crucially inform them of why they have been prevented. This is important due to the significant average availability time ofphishing links, which according to Canova et al. [11], was 32 hours and 32 minutesin the first half of 2014. Machine learning techniques are very successful in matching established patterns of URLs but not as successful at identifying new URL variations, which means malicious URLs can go sometime before being detected. The lack of user knowledge of phishing also encourages users to ignore warnings when presented to them. In the UK for example, only 72% of technology users had heard of phishing as a term despite 95% of organizations saying that theytrain end users [60]. For these reasons, it is important to train users to detected phishing themselves. As the first year of my Mini project, the focus this year was working on both designing the tool and implementing and evaluating the means forhow the user would interact with the tool. The project has involved a welcome and generous amount of advice from a PhD student who is an expert in phishing, computer security and URLs and is working on a similar project. This student has produced a lot of the research referenced in this report. However, I make a clear distinction between the work done by myself and this student throughout the report.

LITERATURE SURVEY

Existing Problem:

Phishing detection schemes which detect phishing on the server side are better than phishing prevention strategies and user training systems. These systems can be used either via a web browser on the client or through specific host-site software presents the classification of Phishing detection approaches. Heuristic and ML based approach is based on supervised and unsupervised learning techniques. It requires features or labels for learning an environment to make a prediction. Proactive phishing URL detection is similar to ML approach. However, URLs are processed and support a system to predict a URL as a legitimate or malicious. Blacklist and Whitelist approaches are the traditional methods to identify the phishing sites. The exponential growth of web domains reduces the performance of the traditional method.

References:

1. Anti-Phishing Working Group (APWG), https://docs.apwg.org//reports/apwg_trends_report_q4_2019.pdf

2. .Jain A.K., Gupta B.B. ―PHISH-SAFE: URL Features-Based Phishing Detection System Using Machine Learning‖, Cyber Security. Advances in Intelligent Systems and Computing, vol. 729, 2018,View Article, Google Scholar

3. Purbay M., Kumar D, ―Split Behavior of Supervised Machine Learning Algorithms for Phishing URL Detection‖, Lecture Notes in Electrical Engineering, vol. 683, 2021,View Article, Google Scholar

4. Gandotra E., Gupta D, ―An Efficient Approach for Phishing Detection using Machine Learning‖, Algorithms for Intelligent Systems, Springer, Singapore, 2021, https://doi.org/10.1007/978-981-15-8711-5_12.

5. Hung Le, Quang Pham, Doyen Sahoo, and Steven C.H. Hoi, ―URLNet: Learning a URL Representation with Deep Learning for Malicious URL Detection‖, Conference’17, Washington, DC, USA, arXiv:1802.03162, July 2017.

6. Hong J., Kim T., Liu J., Park N., Kim SW, ―Phishing URL Detection with Lexical Features and Blacklisted Domains‖, Autonomous Secure Cyber Systems. Springer, https://doi.org/10.1007/978-3-030-33432-1_12.

7. Kumar, A. Santhanavijayan, B. Janet, B. Rajendran and B. S. Bindhumadhava,―Phishing Website Classification and Detection Using MachineLearning,‖ 2020 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 2020, pp. 1–6, 10.1109/ICCCI48352.2020.9104161.

8. Hassan Y.A. and Abdelfettah B, ―Using case- based reasoning for phishing detection", Procedia Computer Science, vol. 109, 2017, pp. 281–288., View Article ,Google Scholar

9. Rao RS, Pais AR. Jail-Phish: An improved search engine based phishing detection system. Computers & Security. 2019 Jun 1;83:246–67.

10. Aljofey A, Jiang Q, Qu Q, Huang M, Niyigena JP. An effective phishing detection model based on character level convolutional neural network from URL. Electronics. 2020 Sep;9(9):1514.

Problem Statement Definition

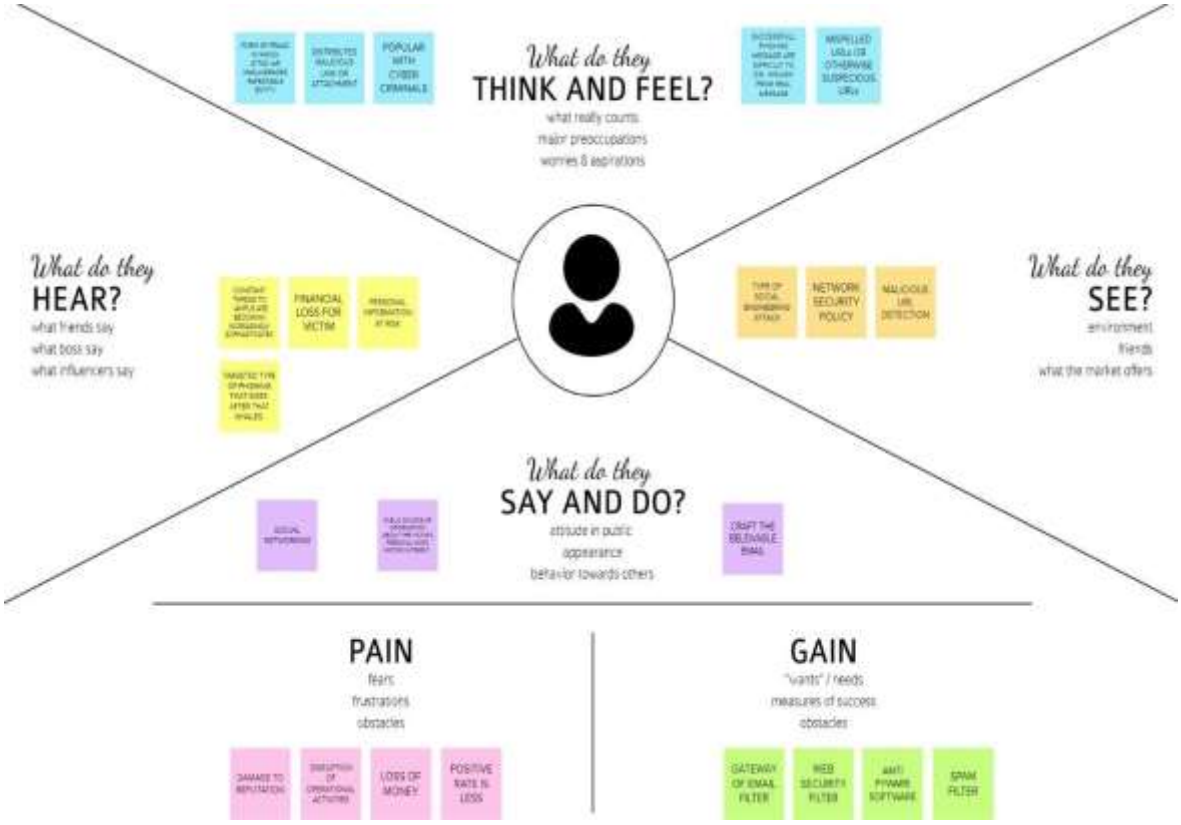
Phishing is a major problem, which uses both social engineering and technical deception to get users' important information such as financial data, emails, and other private information. Phishing exploits human vulnerabilities; therefore, most protection protocols cannot prevent the whole phishing attacks.

Phishing sites are malicious websites that imitate as legitimate websites or web pages and aim to steal user’s personal credentials like user id, password, and financial information. Spotting these phishing websites is typically a challenging task because phishing is mainly a semantics-based attack, that mainly focus on human vulnerabilities, not the network or software vulnerabilities. Phishing can be elaborated as the process of charming users in order to gain their personal credentials like user-id’s and passwords. In this paper, we come up with an intelligent system that can spot the phishing sites. This intelligent system is based on a machine learning model. Our aim through this paper is to stalk a better performance classifier by examining the features of the phishing site and choose appropriate combination of systems for the training of the classifier.

IDEATION & PROPOSED SOLUTION

Empathy Map Canvas

An empathy map is a collaborative visualization used to articulate what we know about a particular type of user. It externalizes knowledge about users in order to 1) create a shared understanding of user needs, and 2) aid in decision making. Traditional empathy maps are split into 4 quadrants (Says, Thinks, Does, and Feels), with the user or persona in the middle. Empathy maps provide a glance into who a user is as a whole and are not chronological or sequential. Empathy maps should be used throughout any UX process to establish common ground among team members and to understand and prioritize user needs. In user-centered design, empathy maps are best used from the very beginning of the design process.



Ideation & Brainstorming

Ideation is often closely related to the practice of brainstorming, a specific technique that is utilized to generate new ideas. A principal difference between ideation and brainstorming is that ideation is commonly more thought of as being an individual pursuit, while brainstorming is almost always a group activity. Brainstorming is usually conducted by getting a group of people together to come up with either general new ideas or ideas for solving a specific problem or dealing with a specific situation. For example, a major corporation that recently learned it is the object of a major lawsuit may want to gather together top executives for a brainstorming session on how to publicly respond to the lawsuit being filed.

Proposed Solution

Having hooked your audience into the problem, now you want to paint a picture of what the world will be like when you solve the problem. Your proposed solution should relate the current situation to a desired result and describe the benefits that will accrue when the desired result is achieved. So, begin your proposed solution by briefly describing this desired result.

Note: now that you have described the final result, think of your iterative enhancement plan as showing your audience the steps by which you will lead them to that result.

Problem Solution Fit

The Problem-Solution Fit simply means that you have found a problem with your customer and that the solution you have realized for it actually solves the customer's problem.

REQUIREMENT ANALYSIS

Functional Requirement

Functional requirements are the desired operations of a program, or system as defined in software development and systems engineering. The systems in systems engineering can be either software electronic hardware or combination software- driven electronic

- Address Bar based Features.
- Abnormal Based Features
- HTML and JavaScript Based Features.
- Domain Based Features.

Non-Functional requirements

Non-functional Requirements (NFRs) define system attributes such as security, reliability, performance, maintainability, scalability, and usability. They serve as constraints or restrictions on the design of the system across the different backlogs.

Non-functional Requirements:

Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	This is very user friendly any people with less knowledge also can easily understanded that they are using the fake websites through out alert message
NFR-2	**Security**	This website is secured as one cant hack our detection website , our website is easily trusted ones and they will be saved user financial information loss
NFR-3	**Reliability**	It has good consistency and performs as it actively detect the fake website and protect the confidential information and financial loss of the user
NFR-4	**Performance**	Web phishing detection performs high. It is very efficient, very easy to understand and it has a high security and scalability

. PROJECT DESIGN

Data Flow Diagrams

A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination. Data flowcharts can range from simple, even hand-drawn process overviews, to in-depth, multi-level DFDs that dig progressively deeper into how the data is handled. They can be used to analyze an existing system or model a new one. Like all the best diagrams and charts, a DFD can often visually ―say‖ things that would be hard to explain in words, and they work for both technical and nontechnical audiences, from developer to CEO. That's why DFDs remain so popular after all these years.

Solution & Technical Architecture

This project will be approached based on decision tree, random forest and SVM algorithms.

- The data is collected from the dataset and it is pre-processed.
- Then it is applied to the machine learning algorithmic model and analyzed.
- The model is trained and tested according to the problem specified.
- Python Flask and IBM cloud facility is used in developing this project.

User Stories

A user story is the smallest unit of work in an agile framework. It's an end goal, not a feature, expressed from the software user's perspective .A user story is an informal, general explanation of a software feature written from the perspective of the end user or customer. The purpose of a user story is to articulate how a piece of work will deliver a particular value back to the customer. Note that "customers" don't have to be external end users in the traditional sense, they can also be internal customers or colleagues within your organization who depend on your team.

RPROJECT PLANNING & SCHEDULING

Sprint Planning & Estimation

During sprint planning, we break the stories down into tasks, estimate those tasks, and compare the task estimates against our capacity. It's that, not points, that keep us from overcommitting in this sprint. No need to change the estimate.

Estimation is an essential management operation used to determine an approximate time frame required to start and finish any process in a controlled environment. It's crucial to any project planning to not go past the time limits, set budgets, and available resources.

For super-fast Agile estimation, the items to be estimated are simply placed by the group in one of three categories: big, uncertain and small. The group starts by discussing a few together, and then, like the Bucket System, uses divide-and- conquer to go through the rest of the items.

Sprint Delivery Schedule

sprint is a set period of time during which specific work has to be completed and made ready for review. Each sprint begins with a planning meeting. During the meeting, the product owner (the person requesting the work) and the development team agree upon exactly what work will be accomplished during the sprint. The development team has the final say when it comes to determining how much work can realistically be accomplished during the sprint, and the product owner has the final say on what criteria need to be met for the work to be approved and accepted.

The deliverables of a sprint aren't as predictable as they are for other projects. Sprint participants have produced sketches and drawings, writing, photographs, comic strips, videos and fully coded working prototypes. The answer is whatever's right to answer the problem. The Scrum Developer is the professional responsible for creating the project deliverables, together with the rest of the Scrum team.

Report from JIRA

When JIRA sends either standard notifications or user invitations to a mail server, they are listed as phishing attempts rather than legitimate emails

The mail server is receiving a constant stream of concurrent multiple email notifications from the same sender which, in turn, triggers security measures on the server which handle these messages as phishing attempts.

Resolution

- Check the base url of JIRA to see if it is set as a direct ip address with port number.
- Example: http://10.10.10.10:8080
- Some email servers (such as Microsoft Outlook) will consider messages from non-DNS urls as phishing attempts. You can correct this behaviour by setting JIRA's base url to a url address such as http://my-jira.com.
- Sometimes when certain mail servers receive multiple emails from the same sender security measures are triggered that will then list those emails as phishing messages. For this, it is best to check with the local mail server administrator for further assistance and confirmation.

CODING & SOLUTIONING

Feature 1

- Phishing URL detection

Phishing is a form of fraud in which the attacker tries to learn sensitive information such as login credentials of person in email or other communication channels.

Feature 2

- Accuracy of an URL

we present a way to detect such malicious URL addresses with almost 100% accuracy using convolutional neural network.

Work Flow

We propose a phishing detection approach, which extracts efficient features from the URL and HTML of the given webpage without relying on third-party services. Thus, it can be adaptable at the client side and specify better privacy. An efficient solution for phishing detection that extracts the features from website's URL and HTML source code proposed.

TESTING

Test Cases

For the URL verifier module in the ISOT phishing detection system, phishing detection is done using 16 different heuristic rules. In the system, 11 main classes were defined, and 1 class was defined with 5 sub-classes. This covers all 16 heuristic rules. To test the system, 15 test cases were designed using assertion methods. Ten test cases were designed to test the 10 main classes and 5 test cases were designed to test the class with five sub-classes. The getter-setter method was used to test the class with five sub-classes. The getter method is used to obtain or retrieve a variable value from the class, and the setter method is used to store the variables.

User Acceptance Testing

User Acceptance Testing (UAT) is a type of testing performed by the end user or the client to verify/accept the software system before moving the software application to the production environment. UAT is done in the final phase of testing after functional, integration and system testing is done

The main **Purpose of UAT** is to validate end to end business flow. It does not focus on cosmetic errors, spelling mistakes or system testing. User Acceptance Testing is carried out in a separate testing environment with production-like data setup. It is kind of black box testing where two or more end-users will be involved.

UAT is performed by –

- Client
- End users

RESULT

Performance Metrics

In a classification problem, the category or classes of data is identified based on training data. The model learns from the given dataset and then classifies the new data into classes or groups based on the training. It predicts class labels as the output, such as Yes or No, 0 or 1, Spam or Not Spam, etc. To evaluate the performance of a classification model, different metrics are used, and some of them are as follows:

- Accuracy
- Confusion Matrix
- Precision
- Recall
- F-Score
- AUC(Area Under the Curve)-ROC

ADVANTAGES & DISADVANTAGES

Advantages

Blacklists

- Requiring low resources on host machine
- Effective when minimal FP rates are required Heuristics and visual similarity

Heuristics and visual similarity

- Mitigate zero hour attacks. Machine Learning

Machine learning

- Mitigate zero-hour attacks.
- Construct own classification models.

Disadvantages

Blacklists

- Mitigation of zero-hour phishing attacks.
- Can result in excessive queries with heavily loaded server.

Heuristics and visual similarity

- Higher FP rate than blacklists.
- High computational cost.

Machine Learning

- Time consuming
- Costly
- Huge number of rules
- loss of money
- loss of intellectual property
- damage to reputation
- disruption of operational activities
- Phishing scams involve sending out emails or texts disguised as legitimate sources

CONCLUSION

Phishing websites are being designed to trick people into submitting credentials to access private information and assets by making the sites look like legitimate websites. Phishing attacks through URLs embedded in emails or SMS messages are one of the major issues faced by Internet users because of the huge number of online transactions performed daily. Phishing attacks cause losses to organizations, customers and Internet users. In this project, testing utilities were developed to support test automation for different aspects of the ISOT phishing detection system. In particular, two kind of test utilities were developed. 1. A mechanism to populate live test email accounts from different datasets, including spam, phishing, and legitimate emails, allowing online testing of the ISOT phishing detection system. 2. Automated test cases were used to unit test one of the modules of the ISOT phishing detection system.

FUTURE SCOPE

In the future, optimization can be done in the test units and these units can be made fully automated using Robot-Framework. This is important if more heuristics rules are included in the detection system. If the URL length is very long i.e. more than a million characters, then the system may crash. To prevent this situation, a timeout feature can be added when determining the URL length.

In future if we get structured dataset of phishing we can perform phishing detection much more faster than any other technique. In future we can use a combination of any other two or more classifier to get maximum accuracy. We also plan to explore various phishing techniques that uses Lexical features, Network based features, Content based features, Webpage based features and HTML and JavaScript features of web pages which can improve the performance of the system. In particular, we extract features from URLs and pass it through the various classifiers

APPENDIX

Source Code

App.py

#importing required libraries

From flask import Flask, request, render _template import requests

import numpy as np import warnings import pickle

warnings.filterwarnings('ignore')

from feature import Feature Extraction

file = open("model.pkl","rb")

gbc = pickle.load(file) file.close()

```
API_KEY = "UWEsUaH1i-FABXxbCpQ9lcPk5E0jIaivG8i-veVF9zJj"

token_response =requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey":
API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})

mltoken = token_response.json()["access_token"]

    header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}


app=Flask(_name_)

@app.route("/",
methods=["GET","POST"])

def index():

        if request.method == "POST":


url =request.form["url"]

 obj = FeatureExtraction(url)

x = np.array(obj.getFeaturesList()).reshape(1,30)


#1 is safe

#-1 is unsafe

y_pro_phishing =

gbc.predict_proba(x)[0,0]

y_pro_non_phishing =

gbc.predict_proba(x)[0:,1]print(

y_pro_phishing,y_pro_non_phishing)
```

```
# if(y_pred ==1 ):

pred = "It is {0:.2f} % safe to go ".format(y_pro_phishing*100)

payload_scoring={"input_data":          [{"field":
[["UsingIP","LongURL","ShortURL","Symbol@","Redirecting//","PrefixSuffix-
```

","SubDomains","HTTPS","DomainRegLen","Favicon","NonStdPort","HTTPSDomain URL","RequestURL","AnchorURL","LinksInScriptTags","ServerFormHandler","Info Email","AbnormalURL","WebsiteForwarding","StatusBarCust","DisableRightClick"," UsingPopupWindow","IframeRedirection","AgeofDomain","DNSRecording","WebsiteT raffic","PageRank","GoogleIndex","LinksPointingToPage","StatsReport"

]], "values":obj}]}

response_scoring=requests.post('https://us-south.ml.cloud.ibm.com/ml/v4/deployments/phishing_1/predictions?version=2022-11-11', json=payload_scoring,headers={'Authorization': 'Bearer ' + mltoken})

print("Scoring response") predictions=response_scoring.json() print(predictions)

pred=print(predictions['predictions'][0]['values'][0][0]) if(pred != 1):

print("The Website is secure.. Continue") else:

print("The Website is not Legitimate... BEWARE!!")

return render_template('index.html',xx =round(y_pro_non_phishing,2),url=url) return render_template("index.html", xx =-1)

if _name_ == "_main_":

app.run(debug=True)

Project Demo Link

GitHub Link

https://github.com/IBM-EPBL/IBM-Project-24473-1659943268