MALICIOUS WEB CONTENT DETECTION

A PROJECT REPORT

Submitted to

Visvesvaraya Technological University

BELAGAVI - 590 018

by

POOJARI BRIJESH R	**USN: 4SU17CS059**
MAHAMMAD SHAMEER	**USN: 4SU17CS038**
NISHANTH B S	**USN: 4SU17CS052**
NIPUN HEGDE	**USN: 4SU17CS049**

Under the guidance of

Mr.CHINTESH R

Assistant Professor

in partial fulfillment of the requirements for the award of the degree of

Bachelor of Engineering

Department of Computer Science & Engineering

SDM INSTITUTE OF TECHNOLOGY

UJIRE - 574 240

2020-2021

SDM INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi)

UJIRE – 574 240

Department of Computer Science & Engineering

CERTIFICATE

Certified that the Project Work titled '**Malicious Web Content Detection**' is carried out by **Mr.POOJARI BRIJESH R**, **USN: 4SU17CS059, Mr.MAHAMMED SHAMEER USN:4SU17CS038, Mr.NISHANTH B S USN: 4SU17CS052 , Mr.NIPUN HEGDE USN :4SU17CS049** a bona-fide student of SDM Institute of Technology, Ujire, in partial fulfillment for the award of the degree of **Bachelor of Engineering** in Computer Science and Engineering of Visvesvaraya Technological University, Belagavi during the year 2018-2019. It is certified that all the corrections/ suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The report has been approved as it satisfies the academic requirements in respect of project work prescribed for the said Degree.

Mr. Chintesh R	**Dr. Thyagaraju G S**	**Dr. Ashok Kumar T**
Asst. Professor and Guide	Professor and Head	Principal

External Viva

Name of the Examiners: **Signature with Date**

1.

2.

Acknowledgement

We express our deepest gratitude to our guide Mr. Chintesh R, Asst. Professor, Department of Computer Science & Engineering, for his valuable guidance and encouragement while doing this project work.

We are indebted to Dr. Thyagaraju G S, Head of the Department, and Dr. Ashok Kumar T, Principal, for their advice and suggestions at various stages of the work. We also extend our heartfelt gratitude to Prof. ABC for his assistance.

We thank the Karnataka State Council for Science and Technology (KSCST), Indian Institute of Science, Bangalore for sponsoring the project. We also extend our thanks to the management of SDM Institute of Technology, Ujire, for providing an excellent study environment, reference materials and laboratories facilities. We remain grateful to the co-operation and help rendered by the teaching and non-teaching staff of the department.

POOJARI BRIJESH R 4SU17CS059

MAHAMMED SHAMEEER 4SU17CS038

NISHANTH BS 4SU17CS052

NIPUN HEGDE 4SU17CS049

Abstract

Malicious URL, a.k.a. malicious website, is a common and serious threat to cybersecurity. Malicious host unsolicited content (spam, phishing, drive-by downloads, etc.) and lure unsuspecting users to become victims of scams (monetary loss, theft of private information, and malware installation), and cause losses of billions of dollars every year. It is imperative to detect and act on such threats in a timely manner. Traditionally, this detection is done mostly through the usage of blacklists. However, blacklists cannot be exhaustive, and lack the ability to detect newly generated malicious content. To improve the generality of malicious web detectors, machine learning techniques have been explored with increasing attention in recent years. This article aims to provide a comprehensive survey and a structural understanding of Malicious web Detection techniques using machine learning. We present the formal formulation of Malicious web Detection as a machine learning task, and categorize and review the contributions of literature studies that addresses different dimensions of this problem (feature representation, algorithm design, etc.)

Table of Contents

List of Figures

Chapter 1

Introduction

1.1 Project Introduction

The web attacks are the challenging issues of the web community. When the user visits the malicious web site the attack is initiated through various features (lexical, domain, path, web content and hyperlink etc). To prevent the user against accessing the malicious websites, several automated analysis and detection methods have been proposed .The attackers lure the visitor to access malicious web sites and they steal crucial information from the client machine or install the spyware for further exploits. Dynamic HTML gives attackers a new and powerful technique to compromise the security of computer systems. Amalicious dynamic HTML code is usually embedded in a normal webpage. The malicious webpage infects the victim when a user browses it. Furthermore, such DHTML code can disguise itself easily through transformation, which makes the detection even harder.

1.2 Problem Description

Detecting and preventing the user from these attacks are significant task. Ahuge number of attacks have been observed in last few years. Malicious attack detection and prevention system plays an immense role against these attacks by protecting the system's critical information. The internet security softwares and fire walls are not enough to provide full protection to the system. Hence efficient detection systems are essential for web security.

Chapter 2

Literature Review

2.1 General Introduction

Literature Survey is an important activity, which involves gathering information about a particular topic. It will help to get required information or ideas to do work. The following paragraphs discuss the related work and issues in the area of Machine Learning based smart Malicious Web Content Detection System.

2.2 Literature Survey

Anand Desai, Janvi Jatakia, Rohit Naik and Natasha Raul "Malicious Web Pages Detection Technique using Identifying Vulnerable Websites by Analysis of Common Strings in Phishing URLs[2]. 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. The results show that machine learning techniques areThe better author for finds detection17 features[4]. which can be extracted from the URL based on which a URL can be declared as phishing or no [5]. M. Aydin and N. Baykal used the feature extraction technique to form a feature matrix using which they classified the URL.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", AbuNimeh 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. 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]. 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. 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.

2.3 Summary

In the proposed project we are training the Classification algorithms to attain the best accuracy. It will be trained with a URL dataset with different expressions and then tested with the real time test cases to find the accuracy of the algorithm. We are aiming to get maximum accuracy by training algorithm with different URL datasets and test results. The dynamic and hybrid approaches opt for the present scenario. The dynamic and hybrid approaches consumes more time for detection. The limitations in classification techniques and issues in data sources are explained. The issues of various features and feature collection methods are reported.

Chapter 3

Problem Formulation

3.1 General

Before attempting to solve a problem, we need to first formulate or define the problem. It is important to precisely define the problem you intend to solve. Problem formulation is the act of a problem, determining the cause of the problem and, identifying the solution.

3.2 Problem Statement

All kinds of malicious web pages seriously threat the users'computer security. To avoid attacks from malicious web pages, it is required an efficient malicious webpage detection system to detect a webpage before user browses it. It is required an efficient malicious webpage detection system to detect a webpage before user browses it, and stop opening malicious webpage using machine learning.

3.3 Objectives of the Present Study

The goal for malicious detection is to maximize the predictive accuracy and to help users not to be tricked into giving away their credentials or downloading malicious data.Certain algorithms will be used after obtaining data set such K nearest neighbour and many more.Later adding an extension to chrome to suggest if website is genuine or not. Main purpose behind choosing this project and requirement of this project in are as follows:

1.Protect user's sensitive information.

2.Helps user to identify the malware files.

3.Helps in avoiding social engineering attacks which requires interaction with humans.

4.Provide protection to Computer-based engineering attacks that uses computer software that attempts to retrieve the desired information.

3.4 Summary

From all these observations, following objectives are formed

1. Development of a detection System using certain algorithms.

2. Developing an chrome extension which will help to detect malicious sites.

3.It can be used to avoid different types of attacks to protect data and avoid installing malware

Chapter 4

Requirements and Methodology

4.1 Requirements

The proposed project consists of following modules:

4.1.1 Hardware requirements

4.1.2 Software requirements

4.1.1 Hardware Requirements

The hardware requirements for the proposed project are depicted in the table below:

Table 4.1: Hardware requirements

Sl.No	Hardware/Equipment	Specification
1	Graphic card	Intel 621 graphic card or above
2	RAM	4GB and above

4.1.2 Software Requirements

The software requirements for the proposed project are depicted in the table below:

Table 4.2: Software requirements

Sl.No	Software	Specification
1	Anaconda	Anaconda 64 bit
2	Python	Python 3 and above

4.2 Methodology Used

- The Malicious Web Content Detection consists of four main steps: Obtaining Dataset, Feature Selection, Choosing Classification Algorithm and Google Chrome Extension.
- In general, the system works as follows: The dataset was obtained from the UCI - Machine Learning Repository.
- Next, From the dataset, out of the 30 features present, it was infeasible to extract all the features.Some of them URL Length, Google Index.
- After that, For classifying the URL entered, as either safe or malicious, we considered the following three algorithms: K-Nearest Neighbours (KNN), Support Vector Machine (SVM) Random Forest.
- Finally Chrome Extensions are add-ons to the browser which help in adding more features and making browser usage easier for the user.

Chapter 5

System Design

5.1 System Design

System design is a one important phase in software or system development. System design can be defined as method of defining different modules required for software or system to fulfil all requirements.

5.1.1 Architecture of proposed system

A Classifier is used to distinguish malicious pages from beginning ones. We collected web pages from the Internet to be the training data for the Classifier. The data are processed through a feature extraction engine to get the features for the Classifier. The framework is shown in Fig. 5.1

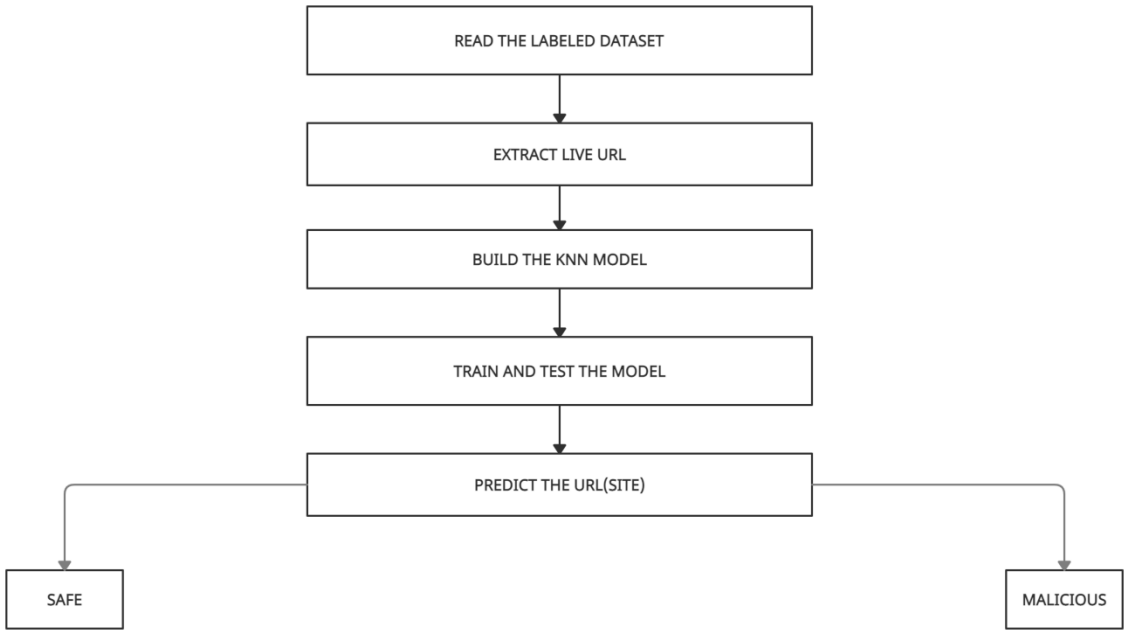

Figure 5.1. A general processing framework for malicious URL Detection Using ML

5.1.1.1 Use Case Diagram

The use case diagram of the proposed system is depicted in figure 5.2 shown below:

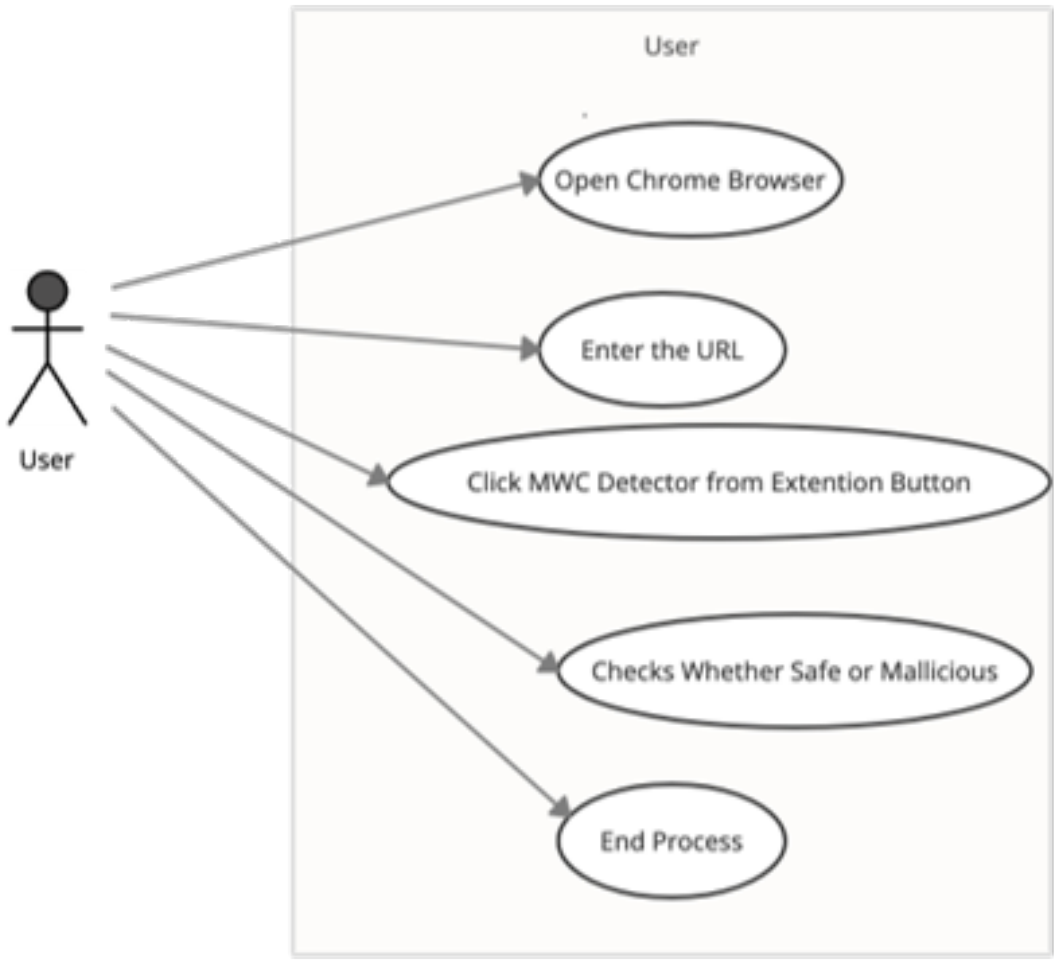

Figure 5.2 : Use Case Diagram of the proposed system

5.1.1.2 Sequence Diagram

The Sequence diagram of the proposed system is depicted in figure 5.3 shown below:

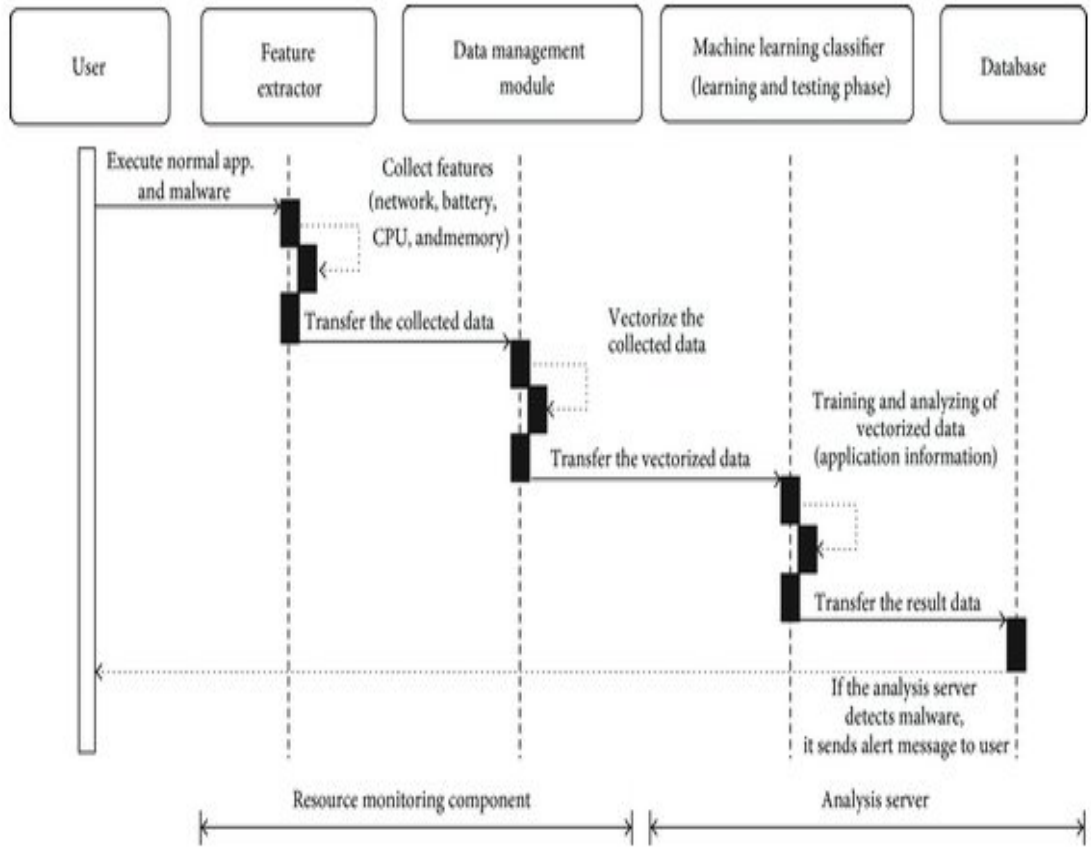

Figure 5.3: Sequence Diagram of the proposed system

5.1.1.3 System Flowchart

The flowchart of the proposed system is depicted in figure 5.4 shown below:

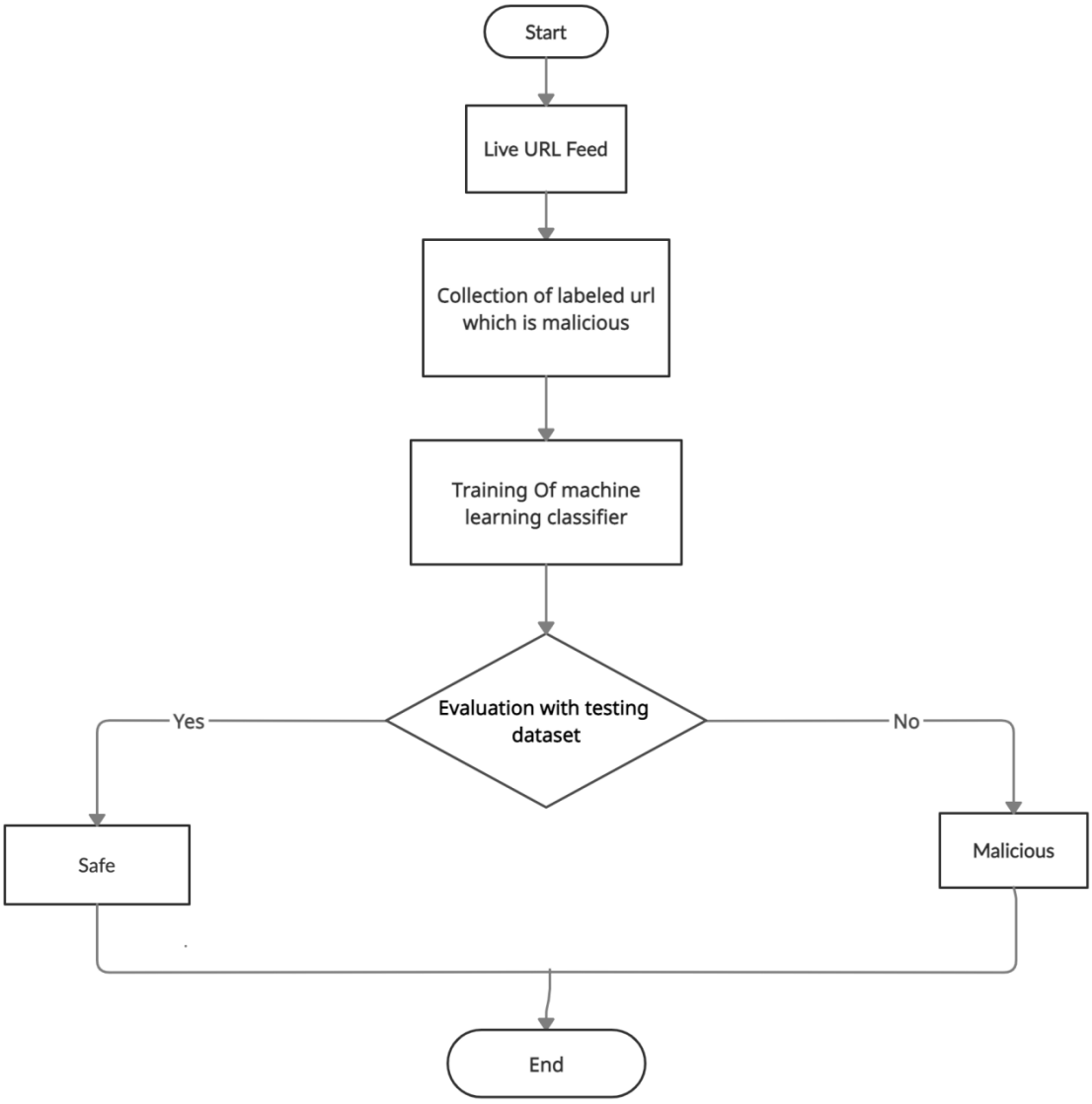

Figure 5.4: Flowchart of the proposed system

Chapter 6

Implementation

6.1 Pseudocode

Pseudocode is a step-by-step written outline of the code that can gradually transcribed into the programming language.

//Malicious Web Content Detection

1. Read The Dataset from Kaggle
2. Input: Extraction of URL,URL Length , Google Index Using Extention
3. Train the model using:
   i. Random Forest Classification
   ii. Support Vector Machine
4. Load Trained Model
5. Test The Model By User Input URL
6. If Prediction == 1:
   - Result = Malicious
7. Else If Prediction == 0:
   - Result = Safe
8. Show Result to User Through Extention

6.2 Implementation Code

```
import time

def calculate_metrics(y_test,Y_predicted):

                    from sklearn import metrics

                    from sklearn.metrics import
classification_report,confusion_matrix

                    accuracy = metrics.accuracy_score(y_test,Y_predicted)

                    print "accuracy = "+str(round(accuracy * 100,2))+"%"
```

```
                    confusion_mat = confusion_matrix(y_test,Y_predicted)
                    print confusion_mat
                    print confusion_mat.shape
                    print “TP\tFP\tFN\tTN\tSensitivity\tSpecificity"
                    for i in range(confusion_mat.shape[0]):
                    # i means which class to choose to do one-vs-the-rest calculation
                    # rows are actual obs whereas columns are predictions
                    TP = round(float(confusion_mat[i,i]),2)  # correctly labeled as i
                    FP = round(float(confusion_mat[:,i].sum()),2) - TP  # incorrectly
labeled as i
                    FN = round(float(confusion_mat[i,:].sum()),2) - TP  # incorrectly
labeled as non-i
                     TN = round(float(confusion_mat.sum().sum()),2) - TP - FP - FN
                    print str(TP)+"\t"+str(FP)+"\t"+str(FN)+"\t"+str(TN),
                    sensitivity = round(TP / (TP + FN),2)
                    specificity = round(TN / (TN + FP),2)
                    print "\t"+str(sensitivity)+"\t\t"+str(specificity)+"\t\t"
                    f_score = metrics.f1_score(y_test,Y_predicted)
                    print f_score
def neural_network(dataset,class_labels,test_size):
                    import numpy as np
                    import pandas as pd
                    from sklearn.cross_validation import train_test_split
                    from sklearn.neural_network import MLPClassifier
                    X = pd.read_csv(dataset)
                    Y = pd.read_csv(class_labels)
                    X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=
test_size, random_state=42)
```

```
                model = MLPClassifier(hidden_layer_sizes=(100),
activation='logistic',random_state = 42)
                model.fit(X_train,y_train)
                Y_predicted = model.predict(X_test)
                return y_test,Y_predicted
def random_forests(dataset,class_labels,test_size):
                import numpy as np
                import pandas as pd
                from sklearn.cross_validation import train_test_split
                from sklearn.ensemble import RandomForestClassifier
                from sklearn import metrics
                X = pd.read_csv(dataset)
                Y = pd.read_csv(class_labels)
                X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=
test_size, random_state=42)
                model = RandomForestClassifier(n_estimators = 5, criterion =
'entropy',random_state = 42)
                model.fit(X_train,y_train)
                Y_predicted = model.predict(X_test)
                return y_test,Y_predicted
def support_vector_machines(dataset,class_labels,test_size):
                import numpy as np
                from sklearn import svm
                import pandas as pd
                from sklearn.cross_validation import train_test_split
                X = pd.read_csv(dataset)
                Y = pd.read_csv(class_labels)
```

```
                X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=
test_size, random_state=42)

                # 'rbf' value is the gaussian kernel, 'C' is the coefficient used for
regularization during training

                model = svm.SVC(kernel='rbf',C=2.0)

                model.fit(X_train,y_train)

                Y_predicted = model.predict(X_test)

                return y_test,Y_predicted

def main():

                dataset = "Dataset.csv"

                class_labels = "Target_Labels.csv"

                test_size = 0.3

                print "\nrunning neural networks..."

                start_time = time.time()

                y_test,Y_predicted =
neural_network(dataset,class_labels,test_size)

                calculate_metrics(y_test,Y_predicted)

                end_time = time.time()

                print "runtime = "+str(end_time - start_time)+" seconds"

                print "\nrunning random forests..."

                start_time = time.time()

                y_test,Y_predicted =
random_forests(dataset,class_labels,test_size)

                calculate_metrics(y_test,Y_predicted)

                end_time = time.time()

                print "runtime = "+str(end_time - start_time)+" seconds"

                print "\nrunning support vector machines..."

                start_time = time.time()
```

```
                    y_test,Y_predicted =
support_vector_machines(dataset,class_labels,test_size)

                    calculate_metrics(y_test,Y_predicted)

                    end_time = time.time()

                    print "runtime = "+str(end_time - start_time)+" seconds"

if __name__ == '__main__':

                    start_time = time.time()

                    main()

                    end_time = time.time()

                    print "runtime = "+str(end_time - start_time)+" seconds"
```

Chapter 7

System Testing, Results and Discussion

7.1 System Testing

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner.

Software testing is the process of checking whether the developed system is working according to the original objectives and requirements. Software testing process commences once the program is created and the documentation and related data structures are designed. Software testing is essential for correcting errors. Otherwise the project is not aid to be complete.

7.2 Result Analysis

- 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.

7.3 Summary

There are a a wide variety of machine learning algorithms in literature that can be directly used in the context of Malicious URL Detection. Due to potentially a tremendous size of training data (millions of instances and features), there was a need for scalable algorithms, and that is why Online Learning methods have found a lot of success in this domain. Efforts have also been made to automatically learn the features, and to perform feature selection. Lastly, there have been efforts in modifying the problem from a typical binary classification algorithm to address class imbalance and multi-class problems. We categorize the representative references according to the machine learning methods applied. Using these technologies to build live systems is another challenging task. In the following we discuss real systems to demonstrate how Malicious URL Detection can be used as a service.

Chapter 8

Conclusion and Scope for future work

8.1 Conclusion

This project proposes the development of a Chrome Extension for identifying malicious 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.In this paper we use online learning algorithms to detect malicious webpages. To make feature value can reflect importance of features; each feature is assigned a weight. The weight of features is decided by the difference of feature frequency in malicious and safe samples. The project 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 project 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.

8.2 Scope for future work

The results of this project can be applied and implemented in information security technologies in information security systems. The results have been used to build a free tool to detect malicious URLs on web browsers.This project proposes the development of a chrome Extension for identifying phishing websites.The future scope of this idea is very broad. Some websites only have a few components , one can block this malicious part and display the complete safe webpage to the users. This can be implemented in a similar as to that of AdBlock Extension which blocks a particular part of the webpage and displays the rest.

References

• Anand Desai, Janvi Jatakia, Rohit Naik and Natasha Raul “Malicious Web Pages Detection Technique using Identifying Vulnerable Websites by Analysis of Common Strings in Phishing URLs

• B. Wardman, G. Shukla and G. Warner, ;”Identifying vulnerable web_x0002_sites by analysis of common strings in phishing URLs,”; 2009 eCrime Researchers Summit, Tacoma, WA, 2009, pp. 1-13.

• A. B. Sayambar, A. M. Dixit, “On URL Classification, International Journal of Computer Trends and Technology” 2014.

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

• Hou et al., “Malicious Web Content Detection by Machine learning, Expert Systems with Applications”, International Journal 2010.

• 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.

• Khamis et al, “Characterizing A Malicious Web Page”, Australian Journal of Basic and Applied Sciences 2014.

• Curtsinger et al., “Zozzle: Fast and Precise In-Browser JavaScript Mal_x0002_ware Detection”, SEC’11 Proceedings of the 20th USENIX conference on Security 2011.

Personal Profile

Mr. Chintesh Asst. Prof.

Project Guide

Prof. Chintesh received the B.E. degree in Computer Science Engineering from ABC Engg. College in the year 1980, and M.E. in Data Communication form XYZ in 2001.

His subjects of interest include Image Processing, Communication Engineering and Data Mining.

Prof. Kumar is a life member of Indian Society of ISTE and member of IEEE.

Name:Nishanth B S

USN:4SU17CS052

Branch: CSE

Address:Chandragiri Nilaya Shirlalu post Belthangady, DK-574217

E-mail ID:nishanthbangera107@gmail.com

Contact Phone No:9741963630

Name:Nipun hegde

USN:4su17cs049

Branch: CSE

Address:#33 vishwashawarya layout sidhedalli -79

E-mail ID:nipunhegde@gmail.com

Contact Phone No:8762499013

Name:Poojari Brijesh R

USN:4SU17CS059

Branch: CSE

Address:Gurukripa Nivas Padangady Post and Village Belthangady,DK-574217

E-mail ID:brjshpoojari@gmail.com

Contact Phone No:7499850136

Name: Mahammad Shameer

USN:4SU17CS038

Branch: CSE

Address: Bedrabettu House,Bangady Post,Indabettu Village,Belthangady,574214.

E-mail ID:mohdshamishaaz167@gmail.com

Contact Phone No:9663640613