**EVENT-DRIVEN MALICIOUS URL EXTRACTOR**

2021-085

Project Proposal Report

B.Sc. (Hons) Degree in Information Technology Specializing in Cyber Security

Department of Information Technology

Sri Lanka Institute of Information Technology

Sri Lanka

March 2021

**EVENT-DRIVEN MALICIOUS URL EXTRACTOR**

2021-085

Project Proposal Report

Jonathan S. W. S.

IT18071412

Amila Senarathne

Kanishka Yapa

B.Sc. (Hons) Degree in Information Technology Specializing in Cyber Security

Department of Information Technology

Sri Lanka Institute of Information Technology

Sri Lanka

March 2021

# Declaration

We declare that this is our own work and this proposal does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other university or Institute of higher learning and to the best of our knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

|  |  |  |  |
| --- | --- | --- | --- |
| Student Number | Student Name | Student Email | Signature |
| IT18071412 | S. W. S. Jonathan | It18071412@my.sliit.lk | Shape  Description automatically generated with medium confidence |

The above candidates are carrying out research for the undergraduate Dissertation under my supervision.

Signature of the supervisor Date

...................................

# Abstract

The Internet has seen a tremendous boom in the past decade. Along with it, attackers have come up with various ways to infiltrate networks. Successful attacks do not necessarily require 0-days or complex attack vector, but a mere manipulation of the human factor. Social Engineering is an effective way to trick users to perform a malicious actor’s bid. A prominent social engineering method is using malicious URLs to conduct phishing, drive by downloads and spam. With the increased variation of cyberattacks, URL classifiers models must also adapt to the requirements. The prevention method evolves by not only using blocklists, but also by leveraging Machine Learning/ Deep Learning Capabilities. The current research space has provided various outputs of having classifiers in both the Deep Learning and Machine Learning Domain to increase accuracy. Our research identifies the current gaps in detecting these malicious URLs by proposing an ensemble model and using feature reduction to build a accurate and robust model. In addition, we propose an event-driven system to classify malicious URLs and provide lists to researchers/organizations that can be used in their existing blacklists. This system encompasses a modular approach since various models will be used. Working with modules, we can also work on improving the scalability of the system. At its core, the system would depend on the event provided by the user to classify URLs accordingly. Scalability should be integrated to each of the modules depending on the event.

**Keywords – Ensemble Models, Malicious URLs, Machine Learning, Deep Learning, Event-driven**

Table of Contents

[Declaration iii](#_Toc67518909)

[Abstract iv](#_Toc67518910)

[List of Figures vi](#_Toc67518911)

[List of Tables vii](#_Toc67518912)

[List of Abbreviations viii](#_Toc67518913)

[1 Introduction 1](#_Toc67518914)

[1.1 Background & Literature Survey 1](#_Toc67518915)

[1.2 Research Gap 3](#_Toc67518916)

[1.3 Research Problem 5](#_Toc67518917)

[2 Objectives 6](#_Toc67518918)

[2.1 Main Objectives 6](#_Toc67518919)

[2.2 Specific Objectives 6](#_Toc67518920)

[3 Methodology 7](#_Toc67518921)

[3.1 System Architecture 10](#_Toc67518922)

[3.2 Commercialization 12](#_Toc67518923)

[4 Project Requirements 13](#_Toc67518924)

[4.1 Functional Requirements 13](#_Toc67518925)

[4.2 User Requirements 13](#_Toc67518926)

[4.3 System Requirements 14](#_Toc67518927)

[4.4 Non-Functional Requirements 14](#_Toc67518928)

[4.5 Use Case 15](#_Toc67518929)

[4.6 Wireframes 16](#_Toc67518930)

[5 Work Breakdown Structure (WBS) 18](#_Toc67518931)

[5.1 Gantt Chart 19](#_Toc67518932)

[6 Budget and Budget Justification 20](#_Toc67518933)

[Reference List 21](#_Toc67518934)

[Appendices 22](#_Toc67518935)

[Appendix A – Plagiarism Report 22](#_Toc67518936)

# List of Figures

[Figure 1.1: Phishing Activity Trend [3] 1](https://stdntpartners-my.sharepoint.com/personal/sinhalawelandage_sajethjonathan_studentambassadors_com/Documents/RP/IT18071412_new.docx#_Toc67517317)

[Figure 3.1: Ensemble Modeling 7](#_Toc67517318)

[Figure 3.2: Deep Learning for Phishing Attack Detection [10] 8](#_Toc67517319)

[Figure 3.3: Activity diagram for the Web Interface - Extractor 9](#_Toc67517320)

[Figure 3.4: Overall System Architecture 10](#_Toc67517321)

[Figure 3.5: Ensemble Model + Web Interface Architecture 11](#_Toc67517322)

[Figure 4.1: Homage - Extractor 16](#_Toc67517323)

[Figure 4.2: Results Page - Extractor 16](#_Toc67517324)

[Figure 4.3: Visuals Page – Extractor 17](#_Toc67517325)

[Figure 5.1: WBS 18](#_Toc67517326)

[Figure 5.2: Gantt Chart 19](#_Toc67517327)

# List of Tables

[Table 1.1: Comparisons with previous works 4](#_Toc67517489)

[Table 3.1: Required Technologies, Techniques and Algorithms 7](#_Toc67517490)

[Table 3.2: Pricing Models 12](#_Toc67517491)

[Table 4.1: Use Case Diagram 15](#_Toc67517492)

[Table 6.1: Expenses for the proposed system 20](#_Toc67517493)

# List of Abbreviations

|  |  |
| --- | --- |
| Abbreviation | Description |
| CNN | Convolutional Neural Network |
| DBN | Deep Belief Network |
| DL | Deep Learning |
| kNN | k-nearest Neighbor |
| LSTM | Long-Short Term Memory |
| ML | Machine Learning |
| SVM | Support Vector Machines |
| URL | Uniform Resource Locator |

# Introduction

## **Background & Literature Survey**

Badadhe et al. [1] suggest that a phishing attack consists of 4 phases. First, the attacker prepares a phishing webpage. Then, delivers it to the masses generally using emails or other messaging platforms. Thirdly, the victim is tempted to click the link and visit the malicious crafted page. Finally, the victim will be manipulated in disclosing their credentials or sensitive information. Malicious URLs are not only limited to phishing attacks as stated by Sadique et al. [2]. Other malicious URLs include drive by downloads URL and spam URLs.

Chart, line chart

Description automatically generatedThe Anti-Phishing Working Group’s (AWPG) report [3] for the last quarter of 2020 shows that the rise of phishing activity has been drastic [Figure 1].

Figure 1.1: Phishing Activity Trend [3]

Authors in Benavides et al. [4] conducted a literature review on current Deep Learning (DL) solutions and concluded that most regularly utilized are Deep Neural Networks (DNNs) and Convolutional Neural Networks (CNNs). Yi et al. [5] present a detection model based on Deep Belief Networks (DBNs) and achieve approximately 90% true positive rate and 0.6% false positive rate.

CNN networks deal better in URL classification more than a Long-Short Term Model (LSTM), also consuming lower amount of time for training [6]. The training of the CNN network lasted for three minutes while two milliseconds was used for the URL Checking. Authors in Yazhmozhi et al. [7] proposed an ensemble model using two DL networks, Recurrent Neural Networks (RNNs) and CNNs providing a precision of 97%. It is note-worthy to point out their model achieved this precision without using a feature extraction process thus saving time.

Unlike the Deep Leaning domain, the Machine Learning space has seen a lot of research done to detect malicious URLs. Authors in Isphany et al. [8] experimented with different ML classifiers such as Support Vector Machines (SVMs), kNNs, Naïve Bayes, Logistic regression and AdaBoostM1. They found that kNNs produced the highest accuracy (99.2%) and the lowest false positive rate (2%). Following that SVMs proved to have the second-highest results.

In another comparison done by A. Makkar et al. [9] , ensemble models such as Monotone Multi-Layer Perceptron Neural Network, Multi-Layer Perceptron, Neural Networks with Feature Extraction worked best in terms of accuracy. The results were improved by introducing Cross-Validation to the ensemble models

## **Research Gap**

A lot of research has been done to detect malicious URLs using Deep-learning and Machine Learning models to increase the accuracy [4, 5, 8]. However, the scope for incorporating ensemble models is yet to be explored thoroughly [10]. Ensemble models are techniques to create multiple simpler models and combine them to produce better and accurate results.

The need for an accurate model is justified with the advancements made in the artificial intelligence domain. The domain adopts new approaches every year and previous URL classifiers becomes less efficient as time goes on. The latest advancements can be utilized to build a modern and efficient model. On the other hand, reducing the false positive rate is a crucial feature of a robust model. False positive results can cause overhead in an organization since it would require extra steps of processes to whitelist the false positive URL. In a previous work for an Intrusion Detection System, ensemble model proves to have the highest classification accuracy and the lowest false positive rates [11]. Other works in the URL detection space, also have utilized some form of ensemble modeling aiming for the above-mentioned features [12] [7] [9].

A comprehensive literature survey strengthened our gap for the need of the ensemble model [10]. The same authors proposed an ensemble model using three machine-learning classifiers: Artificial Neural Network (ANN), K-Nearest Neighbors (KNN), and Decision Tree (C4.5) to use in an ensemble method with Random Forest Classifier (RFC) [12].

This work could be extended by comparing with other models such as Bidirectional LSTM and CNNs, as mentioned by Yazhmozhi et al [7]. The models does not have to be limited to certain models, but we can also use various other combination of models and compare them with metrics to settle on a robust and suitable model for this event-driven system.

While many researchers have proposed novel ideas for malicious URL detection, even going for the extent of creating automated frameworks [2]. We had yet to come across a system designed to correlate malicious URLs with events. This system can be a practical approach for organizations and researchers to create malicious URLs lists with regards to any event. For example, the COVID-19 pandemic caused unprecedented disorder resulting in a drastic increase in phishing URLs [3]. They ranged from sites posing as World Health Organization to Video conferencing domains [13] . Attackers took advantage of the sudden remote working/learning culture. This also included tricking users who were looking for employment.

Such an event-driven system to output Malicious URL List can be used by organizations/researchers to build blocklist to stop a phishing attack in the early stage of its lifecycle.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Solution | Considers accuracy of the model | Considers  Efficiency of the model | Categories URLs based on events | Building a scalable system |
| Research A | ´ |  | ´ | ´ |
| Research B |  | ´ | ´ | ´ |
| Research C |  | ´ | ´ | ´ |
| Proposed system |  |  |  |  |

Table 1.1: Comparisons with previous works

## **Research Problem**

A single event, COVID-19 Pandemic created a variety of pathways for attacker [8]. This gives some insight where a single event, can give arise to various related events in which attackers can leverage to craft malicious domains. We have to keep in mind that cyberattacks are also coming up with novel ideas to social engineer unwitting users.

Similar to this, there are lot of events happening in a global or local scale at any given time [14] . Even though, they might not have a higher entropy as the COVID-19 pandemic, they still do contribute to a portion of malicious URLs. Collecting, labelling and categorizing these URLs by co-relating to an event can provide new insight for researchers and targeted protection for organizations. Our proposed system uses an event-based approach to label malicious URLs and categorize them. The system consists of a web-application for the user input events and outputs the labelled malicious URLs. The labelling is done by correlating the URL with keywords. The user interface has to be simple, clean and intuitive to make it welcoming for the user.

Combining the relevant research done in the DL and ML space to build a robust ensemble model can increase the accuracy of the system while having a better performance. In addition, ensemble models tend to showcase high accuracy along with a lower false positive rate. This model can fit into the event-driven system as a module. Whenever, a better ensemble model is introduced, this system can adapt to it. We have to determine if the suitable ensemble model’s accuracy is adequate enough without trading too much in performance.

# Objectives

## **Main Objectives**

This study focuses on building an ensemble model to increase accuracy of the classification of malicious URLs and reduce the false positive rates. This ensemble model along with the rest of the components will form a system to extract event-based malicious URLs and provide endpoint protection from malicious URLs.

For improved usability, the web interface of this system has to build in a usable manner. Phishing activity has been booming for the past year (Figure 1) with the digital transformation of various industries. Better classification methods with a usable user-interface can equip organizations to better defend themselves.

## **Specific Objectives**

The main objectives can be achieved, by conducting the below specific objectives.

1. Experiment with different ML and DL models to achieve an acceptable ensemble model. Top priority will be given to experiment with a Bidirectional LSTM + CNN model.
2. Building a web user interface using Python Flask Framework which will allow users to export malicious URLs list in CSV, JSON and XML format
3. Collect Training and Testing data from publicly available feeds and label the URLs with keywords. Data can be collected from these sources,
   1. OpenPhish
   2. PhishTank
   3. Common Crawler
   4. WhoDNS
   5. Yandex Search API

# Methodology

As stated in the research gap, the goal of this research is to introduce a solution to list malicious URLs to the user by correlating with the keywords provided. To achieve this an ensemble model is built.

|  |  |
| --- | --- |
| Technologies | Azure Machine Learning Studio  SciKit Learn  TensorFlow  Keras  Flask |
| Techniques | Ensemble Modeling using ML and DL |
| Algorithms | LSTM CNN |

Table 3.1: Required Technologies, Techniques and Algorithms

OpenPhish’s feed provides malicious URLs and tags them with the brand they targeted. But other feeds, only provide the malicious URL. Therefore, we have to label the collected URL dataset with the targeted keyword. This dataset will be used to train the keyword matcher module built by another team member

An ensemble model is formed by combining various simpler models (Figure 2). As stated in the research problem, this model will showcase a higher accuracy and false positive rate that solely using ML and DL models individually.

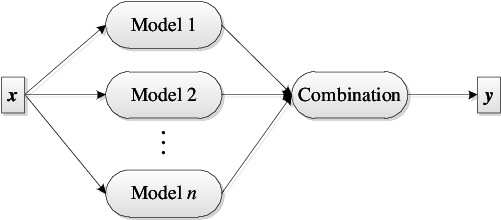


Figure 3.1: Ensemble Modeling

Different ML and DL models will be examined considering their effective combination to prepare ensemble models. These ensemble models will be then compared with each other to determine the suitable model for this system. The conditions for benchmarking would be as follows.

1. The rate of False Negative (FN)
2. The rate of False Positive (FP)
3. The rate of True Positive (TP)
4. The rate of True Negative (TN)
5. The time taken for the model to classify the URLs

These five conditions can be used to determine metrics such as.

1. Precision
2. Recall
3. F-Measure
4. Accuracy

Text

Description automatically generated

Text

Description automatically generated with medium confidence

Text

Description automatically generated

Text

Description automatically generated with medium confidence

By gathering these metrics, we can compare the robustness of various ensemble models.

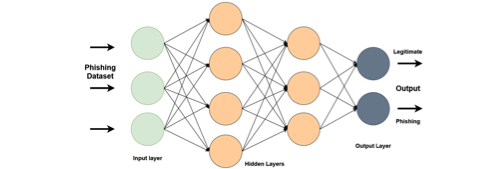


Figure 3.2: Deep Learning for Phishing Attack Detection [10]

Bidirectional LSTM and CNNs will be given priority to build the ensemble model since it was suggested as a potential model by a previous literature [7]. LSTMs and CNNs are part of the Deep Learning Domain (Figure 3).

Figure 4.1 depicts the activity diagram for a user using the web interface to collect Malicious URL List.

Diagram

Description automatically generated

Figure 3.3: Activity diagram for the Web Interface - Extractor

## **System Architecture**

Graphical user interface, application

Description automatically generated

Figure 3.4: Overall System Architecture

Graphical user interface, application

Description automatically generated

Figure 3.5: Ensemble Model + Web Interface Architecture

## **Commercialization**

Since a part of the userbase for this system would be small-medium enterprises (SME), this system can be commercialized with the note of providing basic security. Even though, at this stage the system cannot compete with state-of-the-art endpoint management system, this will be suitable for SMEs due to the lower cost and higher usability.

Two versions of this system can be implemented.

* A free version that SMEs and Researchers can use to collect URL lists based on events with limited export capability.
* A paid version that will provide seamless export capability in addition to the basic endpoint protector using the browser plugin.

|  |  |
| --- | --- |
| Free Version | Rate Limit on event-based malicious URL list and restricted export capabilities. |
| Paid Version | * Browser plugin to protect users from malicious pages with basic reporting to the administrator * No limit on export and event-based malicious URLs |

Table 3.2: Pricing Models

The low-cost barrier should be a key aspect to attract SMEs. Researchers/ Investigators investigating certain events can make use of the data to correlate with the incident.

# Project Requirements

## **Functional Requirements**

1. The system should be capable of produce adequate amounts of malicious URLs for keywords

The foremost purpose of this research is to build an event-driven system to categorize malicious URLs. To achieve this, the system should generate an adequate list so that the end user can benefit from it. If the list of URLs are not enough, then the system should expand the keyword search.

1. Provide visual data for the user to make sense of the URL list produced

For the user to get the maximum use out of the system, we can generate necessary charts from the event-based URL list. Trivial charts such as number of related keywords found for the event, and how the URLs are distributed among the keywords can help the user understand the event clearly.

1. Build a robust ensemble model with high accuracy and low false positive rate

The research gap of exploring the capabilities of ensemble modeling is necessary to be addressed since the system will heavily rely on the capabilities of the URL Classifier. Experimenting with different combinations of DL and ML models and comparing them using necessary metrics is crucial.

## **User Requirements**

This system will consist of two sets of users.

1. Small/Medium Enterprises – They can use this system to generate event-based blocklists to stop a phishing attack at the early stages of the lifecycle. In addition, with the pre-trained plugin installed in their employee’s browsers will assist in preventing them from visiting malicious pages and help the organization receive alerts.

1. Security Researchers/Investigators – They can utilize this system to investigate URLs related to any events in a short span of time. This tool will assist them in determining the gravity of the event and help them decide to follow the investigation path or not.

## **System Requirements**

This research component would require the following system requirements,

1. Desktop OS Platform with a Python Flask Framework Development Environment

The final outcome of this study would require a web application to be maintained. Therefore, any Desktop OS Platform with a supported browser can be utilized

1. API – Tensorflow Python API

Tensorflow is currently the most widely used framework for Deep Learning and Machine Learning. To build the ensemble model, Tensorflow along with Keras can be utilized

1. Azure Machine Learning Studio

This will be used to deploy the model on the cloud. With the increase of decentralized office workspaces and the use of cloud computing, it is necessary for this system to adapt to it. When the system is tested and ready to deploy, deploying it to the cloud using Azure Machine Learning Studio will be viable option.

## **Non-Functional Requirements**

1. Performance

A phishing attack can be stopped at the early stage of its lifecycle if it is identified rapidly. To fulfil that requirement, the system will be expected to classify URLs in a timely manner.

1. Scalability

The system should be able to meet demands of the user. If there is an overhead of employees browsing the internet, there will be a lot of real time URL collected to be scanned by the Ensemble URL Classifier. To ensure that the user has a smooth experience, this system should be able to scale vertically.

1. Accessibility

The web interface should maintain a clean and accessible look so that the user can use it without any hassle. The visualization of malicious URLS list needs to be presented a readable way.

## **Use Case**

|  |  |
| --- | --- |
| Use case name | User providing an event expecting malicious URLs related to it |
| ID | S01 |
| Description | The user will provide a keyword of an global/local event and the system should populate related keywords and correlate it with the Phishing Database |
| Actors | Organization Admin |
| Triggers | Users log in to the system using the login page of the web application |
| Pre-condition | The user must be a part of a valid organization; otherwise, the user must be registered user |
| Main Course | 1. User logs in to the system 2. Enters a keyword of an event 3. The system will out a list of Malicious URLs correlating with that event |
| Exceptions | 1a. If the user is not a registered user, the user can register individually or as an organization  1b. The user is required to login to the system after registration  2a. If the keyword does not yield any results, the user must try another keyword |

Table 4.1: Use Case Diagram

## **Wireframes**

Graphical user interface

Description automatically generated

Figure 4.1: Homage - Extractor

Graphical user interface, table

Description automatically generated

Figure 4.2: Results Page - Extractor

Graphical user interface, application

Description automatically generated

Figure 4.3: Visuals Page – Extractor

# Work Breakdown Structure (WBS)

Diagram

Description automatically generated

Figure 5.1: WBS

## **Gantt Chart**

Table

Description automatically generated with medium confidence

Figure 5.2: Gantt Chart

# Budget and Budget Justification

|  |  |  |
| --- | --- | --- |
| Task | Cost($) | Cost(Rs.) |
| Azure Machine Learning Studio (Implementation and Cloud Deployment) | 100 (Monthly Fee) | **19,900** |
| Web Application Hosting | 72 | **14300** |
| Database – Mongo DB | 57 | **11400** |
| Plugin upload to Chrome Web store | 5 (One-time fee) | **995** |
|  | Total Cost | **46,595** |

Table 6.1: Expenses for the proposed system

To cover these costs, we can use the pricing models mentioned in the commercialization section.

# Reference List

|  |  |
| --- | --- |
| [1] | N.Badadhe, S.More and N. Puri, "An Efficient Approach To Detecting Phishing A Web Using K-Means And Naïve-Bayes Algorithms With Results.," *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET,* vol. 3, no. 5, p. 1, 2014. |
| [2] | F. Sadique, R. Kaul, S. Badsha and S. Sengupta, "An Automated Framework for Real-time Phishing URL Detection," in *10th Annual Computing and Communication Workshop and Conference (CCWC)*, Las Vegas, NV, USA, 2020. |
| [3] | "APWG Trend Report," 2020. [Online]. Available: https://docs.apwg.org/reports/apwg\_trends\_report\_q4\_2020.pdf. [Accessed 25 February 2020]. |
| [4] | E. Benavides, W. Fueters, S. Sanchez-Gordon and M. Sanchz, "Classification of Phishing Attack Solutions by Employing Deep Learning Techniques: A Systematic Literature Review," Research Gate, 2020. |
| [5] | P. Yi, Y.Guan, F. Zou, Y. Yao, W. Wang and T. Zhu, "Web Phishing Detection Using a Deep Learning Framework," *Wireless Communications and Mobile Computing,* vol. 2018, 2018. |
| [6] | W. Wei, Q. Ke, J. Nowak, M. Korytkowski, R. Schererd and M. Woźniake, "Accurate and fast URL phishing detector: A convolutional neural network approach," *Computer Networks,* vol. 178, 2020. |
| [7] | V. Yazhmozhi, B. Janet and S. Reddy, "Anti-phishing System using LSTM and CNN," in *IEEE International Conference for Innovation in Technology (INOCON)*, Bengaluru, 2020. |
| [8] | J. Ispahany and R. Islam, "Detecting Malicious Urls of COVID-19 Pandemic Using ML Techniques," 2020. |
| [9] | A. Makkar, N. Kumar, L. Sama, S. Mishra and Y. Samdani, "An Intelligent Phishing Detection Scheme Using Machine Learning," in *Sixth International Conference on Mathematics and Computing, Advances in Intelligent Systems and Computing*, Singapore, 2021. |
| [10] | A.Basit, M. Zafar, X. Liu, A. R. Javed, Z. Jalil and K. Kifayat, "A comprehensive survey of AI-enabled phishing attacks detection techniques," *Springer,* 2020. |
| [11] | D. Gaikwad and R. C.Thool, "Intrusion Detection System Using Bagging Ensemble Method of Machine Learning," in *International Conference on Computing Communication Control and Automation*, 2015. |
| [12] | A. Basit, M. Zafar, A. Javed and Z. Jalil, "A Novel Ensemble Machine Learning Method to Detect Phishing Attack," in *IEEE 23rd International Multitopic Conference (INMIC)*, 2020. |
| [13] | T. Kelly, "Security Magazine," [Online]. Available: https://www.securitymagazine.com/articles/92666-how-hackers-are-using-covid-19-to-find-new-phishing-victims. [Accessed 25 02 2021]. |
| [14] | L. O'Reilly, "Security Boulevard," 30 10 2020. [Online]. Available: https://securityboulevard.com/2020/10/four-days-to-the-election-dont-fall-for-these-phishing-scams/. [Accessed 25 02 2021]. |
| [15] | A. Petrakova, M. Affenzeller and G. Merkurjeva, "Heterogeneous versus Homogeneous Machine Learning Ensembles," *Information Technology and Management Science,* 2015. |

# Appendices

## **Appendix A – Plagiarism Report**

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, chat or text message

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated