Detection of Visual Similarity Snooping Attacks in Emails using an Extended Client Based Technique

George Mwangi Muhindi, Geoffrey Mariga Wambugu, Aaron Mogeni Oirere

Abstract*: This paper provides an Extended Client Based Technique (ECBT) that performs classification on emails using the Bayessian classifier that attain in-depth defense by performing textual analysis on email messages and attachment extensions to detect and flag snooping emails. The technique was implemented using python 3.6 in a jupyter notebook. An experimental research method on a personal computer was used to validate the developed technique using different metrics. The validation results produced a high acceptable percentage rate based on the four calculated validation metrics indicating that the technique was valid. The cosine of similarity showed a high percentage rate of similarity between the validation labels indicating that there is a high rate of similarity between the known and output message labels. The direction for further study on this paper is to conduct a replica experiments, which enhances the classification and flagging of the snooped emails using an advanced classification method.

Keywords: Client-based, Naïve Bayes, Snooping Attacks, Technique, Visual Similarity, Security, Emails, Technique, Bayers Theorem.

I. INTRODUCTION

Threats to email security continue to be among the greatest threats to organizations in the world [1]. Snooping is a form of attack that is socially and technically engineered to steal confidential data like login credentials, passwords and credit card information. Snooping attacks occur when an attacker masquerades as a legitimate and trusted entity and then dupes the unknowing user to open the email. The recipient then clicks on malicious malware that infiltrate the technique revealing confidential and sensitive information. It is important for organizations to prevent financial and ethical loses through snooping attacks and understand how the snooping attacks occurs [2]. Visual similarity snooping attacks are the types of snooping attacks that mimic emails and snoops the organization staffs. The legitimately looking emails are sent by the snooping attacker to gather sensitive and confidential information.

Snooping attacks problem has attracted huge attention over the years, as it has contributed to massive losses among institutions such as banks whose systems contain crucial personal information such as ID numbers, usernames and passwords that need to be protected from hackers [2].

Manuscript received on March 02, 2021.
Revised Manuscript received on March 16, 2021.
Manuscript published on April 30, 2021.
* Correspondence Author
George Mwangi Muhindi*, Master's Student, Murang'a University of Technology, Kenya.
Geoffrey Mariga Wambugu, CoD of Information Technology (IT) Department at Murang'a University of Technology, Kenya.
Aaron Mogeni Oirere, Lecturer and CoD of Computer Science Department at Murang'a University, Kenya.

Hackers use different approaches to illegally access systems, and this includes luring target users to a mimic website by sending them e-mails that are fake with redirected links and malwares that are in form of executable files in the email attachments [3].In response to this, numerous snooping techniques have been developed to help in detecting and preventing snooping attacks. These techniques can be broadly classified as client based and server based techniques [4]. Plenty of research has been conducted in server based cases with techniques such as search engine, visual similarity, DNS and Proactive phishing URL detection based, appearing among the research community. On the other hand, client based techniques usually rely on user feedback or third party reporting to create a database of blacklisted email features against which incoming email features are compared to detect snooping activities. This is limited since the techniques do not check on the content of the received emails. This paper therefore exists to fill this gap by developing a bayesian classifier and preprocessing technique that can be used to expand the scope of existing client based techniques.

II. RELATED WORK

Snooping is a significant security challenge. Although this is not an entirely new concept, users continue to be tricked into giving out their personal information on illegitimate web pages. A number of anti-snooping techniques have been provided in literature for detecting snooping attacks. In this sub-section, this paper will provide an overview of the snooping detection techniques. Overall, snooping detecting techniques may be classified as either software or education based. Software-based approaches may be further categorized as heuristic based, list-based, or visual similarity based techniques [5]. As cited by, Dhamija and Tygar came up with a solution that uses a dynamic security skin on the potential victim's web browser, requiring the individual to actively countercheck the identity of the server [6]. However, this approach had two problems: one, users who are victimized by snooping attacks are not sophisticated, which means they barely pay adequate attention to available signs. Second, Dhamija and Tygar revealed that 20% of victims do not pay attention to visual cues. Additionally, attacks based on visual deception may trick even the most sophisticated technique users [6]. The common and broadly-used anti-snooping techniques are founded on blacklists which maintain a record of a group of snooping domains that the browser prevents potential victims from visiting. Recently, Microsoft adopted an anti-snooping solution based on blacklist into its Internet Explorer browser.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

Similar tools include the NetCraft tool bar or the Google Safe Browsing tool.List-based anti-snooping techniques use white-list, black-list, or both. In black-list technique, a black-list with suspicious IP addresses and domain names is stored. Notwithstanding, a majority of approaches relying on

black-lists are ineffective with regard to how they deal with zero-hour snooping attacks. A research carried out by [7]

found out that 47-83% of snooping domains update in the black-list after twelve hours. Some black-list related techniques include predictive black-listing, DNS-based black-list, and Google Safe Browsing. However, black-list maintains needs a lot of resources to effectively countercheck illegitimate websites. As numerous snooping pages are formulated each day, updating each snooping webpage in the black-list is quite a demanding task.

Google offers a safe browsing service that enables applications to countercheck a URL with the help of a list of suspicious domains that Google regularly updates. It may be an experimental API; however, it has been integrated in Mozilla Firefox and Google Chrome. Moreover, using it is quite easy. The Safe Browsing Lookup API enables users to relay the illegitimate URL to a safe browsing service that will identify if the received URL is real or malicious. The user API relays the URL using POST or GET requests, which are counterchecked with the phishing and malware lists that Google provides. The Safe Browsing Lookup API has shortcomings such as: there is boundary lookup server's response time, and that also no hashing is carried out before the URL is sent [8].

As cited by [9], PhishNet investigates the black-listed URLs and apply specific heuristics to come up with new URL variations. PhishNet substituted top-level domains with 3209 different top-level domains which resulted into child URLs that required examination. To come up with different URLs, host equivalence classes with similar IP addresses are retained, and all integrations of the path and hostnames are applied when creating fresh URLs. The URLs with similar directories are put in one group, and the formulated URLs are arrived at by exchanging the names of files within the same group. Suppose two URLs share a similar directory structure with varying query sections, the query section may be swapped to come up with different URLs [21].

The automated individual white-list proposed by [10] keeps track of the actual webpage Login User Interfaces (LUIs). Every time a user keys in his/her credentials to the LUI, the white-list will be checked for it and suppose it is not found in the list, the user is warned. AIWL has two primary elements: the legitimate LUIs white-list, and the automated white-list maintainer. The former is used for checking if URLs are suspicious or familiar to determine if or not a warning will be given. For white-lists, every LUI is kept as a vector that consists of the webpage feature, URL address, and the DNS-IP mapping. The latter element is the automated white-list maintainer which is a Naïve Bayes classifier that decides if it should an LUI should be kept in the white-list. The white-list maintainer will check how many logins have been done in a particular LUI. If it is more than the threshold, the LUI will treated as a white-list. This paper shall borrow the Naïve Bayes classifier concept.

Liu, Deng, Huang, & Fu (2006) analyse and compare genuine and illegitimate webpages to establish the metrics that may be used for detecting snooping attacks. According to them, a web page is illegitimate when the visual similarity value of the page is not within a specific threshold. In their technique , webpages are first broken into salient blocks in line with visual cues. After which, the visual similarity between the genuine and ungenuine page is evaluated. A webpage is illegitimate if the similarity to the genuine page is more than the given threshold [7].Liu, Deng, Huang, & Fu (2006) came up with an anti-snooping technique that uses visual features. Their technique evaluates the visual similarity between the stored real website and the current website. Their technique took different visual features for evaluation. To identify a snooping attack, the technique has two modules. The first one operates on the local server to identify suspicious keywords and URLs from the email. The second one will compare the visual similarity of the stored real webpage and the suspicious one. [11] also presented a technique that can identify zero-hour snooping attacks. The technique directly extracts the related webpage as it indirectly extracts the relevant webpage. Those that are directly associated will be accessed with the help of hyperlinks available in the webpage source code. A majority of commonly-occurring keywords, which are identified with the help of the TF-IDF algorithm, together with title word are sought for using a dependable search engine to extract indirectly related pages. After the direct and indirect extraction of related webpages, the technique will compare the associated webpage and the suspicious one using ranking relation, link relation, layout and text similarity relations.In a real-time context, detecting snooping attacks ought to be very fast and effective. Techniques depending on blacklists are quite fast. However, they fail to identify zero-hour snooping attacks. Visual similarity-based techniques are overall consume a lot of time, need huge memory space, and may not identify zero-hour attacks [9]. Heuristic approaches may identify zero-hour attacks, but their performance is hugely reliant on the feature set, classifier, and training dat [14]a. Thus, different authors have worked develop approaches that consumes less time, memory, can identify zero-hour attacks, and address other limitations of conventional visual similarity approaches.Visual similarity snooping attacks involves sending of large amounts of emails that are spoofed asking the targeted users to click the links or open attachments that are embedded in the emails [6]. The hyperlinks in the emails just at a mere glance are normally difficult to suspect and this makes it easy for the victim, to click on them without their knowledge [7].The following diagram shows the snooping mechanism used by snooping attacks.

The genuine website is targeted by a clone website that has some input fields such as the text box. When a user enters his or her individual information, this information is then transferred to the user [8]. Subsequently, the user can receive a malicious link that they click on which then automatically collects private information and then sends it back to the cloning site where the hackers steals the confidential information through the following [8]:

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

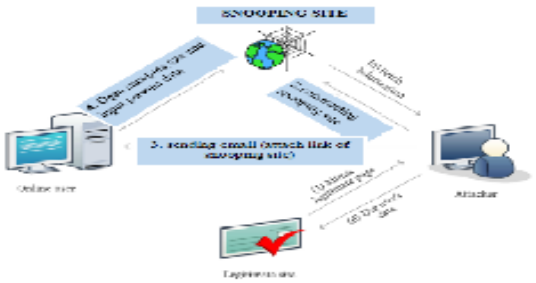

Figure 2. 1: Diagram showing the mechanism of snooping attack

Source: [8]

Classification and the categorization process of the text happens in two phase that is: the training phase and the prediction stage. In the training stage, an extractor converts each input that is the email of the content into a feature set. The features that result are the foundation for the normal information with regards to the inputs, whereby each can be used for classification. They can be inclusive of the words that are extracted from the snooping emails like the Account, Click on this and Send, and these are the references that are used in classifying the emails. The groups of feature sets along with the labels like the keywords are then input into the algorithm of machine learning to come up with a technique [9].

Then later comes the phase of prediction. This involves using the same extractor in transforming the inputs that are unobserved to feature groups that are input to produce the labels that are predicted.

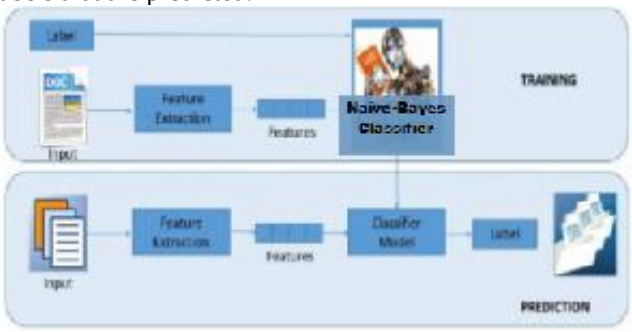

Figure 2. 2: The process of text classification

Source: [9]

Upon a detailed literature review, the researcher concluded that there is need for a linguistic technique that captures the commonly used terminologies in snooping emails such that the technique can integrate the evidence of snooping key phrases from classification of text and embedded links to detect snooping attacks via email. In this line, this paper sought to develop a client-based technique to detect visual similarity snooping attacks with the aforementioned limitations in mind.

III. METHODOLOGY

a. Research Design

The research design enables the research process to be efficient as possible in order to offer adequate information. The research design enabled the researcher to collect information that was relevant and essential within a short period of time, less money and effort [13]. This study employed experimental research design in validating the developed technique.

The research presented by this work entailed identifying the primary features in creating a snooping classifier technique centred around the workings of the Naive Bayes method on a textual datasets. The study created a technique that can classify plain text and also validate a given dataset against the resultant classification technique. For technique creation, the researcher employed state of the art foolproof libraries on the teetering edge of data science and data wrangling. After creation of the technique, a Command Line Argument Software Implementation was created by the researcher for the purpose of on-demand classification of plain text messages alongside a feature to run against a document with the plain text messages organised in a pre-ordained format.

b. Experimental Design

This study used the experimental research design to validate the developed technique.This type of research enables the researcher to manipulate the conditions in order to find out their effects and behaviours (Berger, Maurer, & Celli, 2018). This design elaborates the process that the researcher will carry out in order to determine the aim of adjusting the variables and manipulating other factors. This strategy is normally randomized to make sure that there is no exaggeration or any errors that can compromise the results of this research study

Experimentation was done in a virtual environment created through the anaconda data science software suite on a personal computer with python 3.6 installed on it. The jupyter environment was deployed on a browser and libraries downloaded and installed using the pip3 package manager. Some of the libraries included are: nltk, matplotlib, pandas and sklearn.The code was written through an Integrated Development Environment and updated via GitHub.A full version of the code is available for cloning or contribution in the section 4.5.2 of this paper. The technique development started with the importation of the required libraries then reading in the dataset into dataframes, which was done by the pandas library, specifically the read_csv() function, this function was selected since the source data was in a csv format.

c. Research Procedure

This sub-section provides an overview of the phases that the researcher used to implement and validated the developed technique. The main phases that were used in this research are; (1)data gathering- in this step the dataset to used to train the developed technique was collected . It involved collecting/ mining the data from secondary sources. It is also possible to source data from primary sources. (2) Data preparation- this included a series of actions performed on the data in preparation for use. (3) Data wrangling- this is manipulating the data and reducing noise so as to allow for proper input used in the actual technique.

(4) Technique building- this is where the actual implementation code for the technique was written. (5) Training the Technique- this is where the technique was trained against a particular dataset.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

(6) Testing and Validation of the technique- this is where the researcher checked the validity and soundness of the technique, by use of four validation metrics which are Accuracy, Precision, recall and F1 Score metrics. In validation, the known email labels were compared to the email labels produced by the technique after individually running all the messages in the validation dataset. Cosine of similarity was used to check the measure of similarity between the known email labels to the technique output labels.

i. Data Sourcing

For the researcher to be able to create a viable technique, a viable data source was required, this was indeed aided by the existence of a central repository of open source real life data that is available for scrutiny and download at kaggle.com. Upon which a dataset with both ham and snooping emails was found to be available for the research at hand. The secondary data was collected through published journals, other internet publications and other materials linked to snooping attacks. Secondary data through the study of journals and other publications was also used to examine the current technique used for classify snooping emails with a primary goal of extending it.

ii. Data Preparation

Once the training and testing dataset was availed, the next step was to make the necessary actions to ensure the data is in the right format, size, location among many other considerations before being termed as ready for use. In this case, the dataset was in one large csv document, which would have caused significant performance deterioration when run. Therefore, a script to cut down the file to manageable sizes was written. The script is a command line native python application that takes in as a parameter the source of the data with the .csv extension explicitly included at the end of the input string. The script uses the csv library to read the previously allotted file name, after the file is read and loaded onto a variable.The script cycles through a thousand rows and subsequently creates a new csv with a sequential naming method that names the resultant csv's in the order that they appear within the file. The secondary data collected classified and scrutinize for literature analysis based on snooping attacks and available anti-snooping techniques

iii. Data Wrangling

Data wrangling, alternatively known as data munging, entails the transformation and mapping of data from one raw data form into another with the aim to make it more valuable and appropriate for different downstream purposes, for example, analytics.

iv. Data Frame Loading

Data frame loading describe how the technique training and testing datasets are loaded into the developed technique (19). After data wrangling, the researcher loaded the fractionated datasets into data types known as data frames where the actual manipulation of the data was conducted. The researcher needed the use of the jupyter interface where he used a popular and common library known as pandas to load data in masses into a single dataframe using the read_csv() function from pandas by simply providing the file path of the dataset to be examined as a parameter to this function.

Since the researcher had no issues loading the data into the data frame, the next step was to remove the columns that are not needed for this research, the function used was df.drop(); df represents the data frame being acted upon. The researcher, by this action, rids the data frame of the less useful columns. The researcher then mapped the dataset to the functions of the research, which were snooping text, which were tagged with the binary representation of 1 and the corresponding ham to 0.

v. Train -Test Data Split

To ensure the data representation was not biased, as this would negatively affect the outcome of the technique, the best approach was to use random number generators to ensure an unbiased distribution of the dataset. For this reason the researcher used numpy's np.random.uniform() function that principally generates random numbers using the uniform data distribution provided that you give the function a range to operate on. The generated numbers were then used to pick out a specific row using the index and a new order is then picked out from the resultant process. The researcher then broke the data into two sections mainly the training data set and the testing data set. Traditionally, the pareto principle is used for this method, where 80% of the data is used for training and the remaining 20% for testing.

vi. Word Processing

This involves prepocessing of the texts in the training and testing dataset to remove redundancy (Deshmukh, Popat, & Student, 2017). An essential part of this process that the researcher needed to be keen on was the quality of the words used in the study as this greatly affects the quality of the technique. The first step was to make all the text lowercase because upper case words would be treated as different words in lexicographical value. The researcher then tokenized the words, by removing the punctuation marks and breaking down the sentences into words. The researcher then stemms the words, The words like 'go', 'goes', 'going' show similar activity. All of these words may be replaced with one word 'go.' This is called stemming. For this purpose, the researcher used a function known as PorterStemmer(). The researcher then moved on to get rid of the stop words. Stop words include those occurring frequently in the text. They include 'the', 'a', 'an', 'is', 'to' etc.

vii. Technique Building

A python environment was set up in order to conduct the experiments, write the required code, and wrangle the data for classification. The tool used for wrangling was Jupyter Notebook and native python for the Command Line Tools.

The technique was developed using python 3.6 on jupyter notepad. During the development, the mathematical expression of Bayesian classifier expressed in section 4.5.1 was implemented for snooped emails classification. The developed technique classify against input provided by a user and generate results of the classification as explained in part 4.5.2 of this paper.

The script was availed in a GitHub blob screenshotted and posted at section 4.5.2 and the appendix of this paper . The code blob has been availed at the appendix of this paper.

Retrieval Number: 100.1/ijeat.D22960410421
DOI:10.35940/ijeat.D2296.0410421

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

viii. Technique Training and Testing

This entailed the provision of inputs dataset into the technique that is used for processing, technique training and technique testing (Gavankar & Sawarkar, 2016). The researcher ran the application function against the training dataset. The classifier used was the Naïve Bayes classifier which is good in predictive modelling for determining the snooped emails. The training dataset was the input into the technique so as to find out the presence of snooped emails that are sent based on the parameters of the sent emails (these include the email attachments, the commonly used phishing terms and malicious links). The dataset was then be divided into training and testing datasets. There were several iterations that was carried out in the process of training; each process was aimed at reducing the error rates as well as modifying the inputs. The dataset collected was split into the training dataset (Composed of 80% of the entire population) and the test dataset (composed of the 20% of the entire population).

ix. Technique validation

Validation involves running the developed technique to check the compliance level and to evaluate and determine whether the developed technique is accurately operating as intended (Mishra, 2018). To validate the technique, the researcher used four metrics (Accuracy, Precision, Recall and F1-Score), where the metrics input were obtained after running the mined email messages, collected from kaggle, through the developed technique. The collected emails had a verified emails labels i.e whether ham or spam. These messages were run to identify the technique labels to the expert labelled emails. After running the messages via the developed technique, the, the result of this comparison was used to attain the true positive, true negative, false positive and false negative inorder to calculate the given four validation metrics. Four metrics measurements namely: precision, recall, f-score and accuracy were used to validate the developed technique as described in section 4.6 of this paper.

x. Ethical issues

Since this work entailed developing a client-based technique that can effectively check against snooping attacks, the primary ethical issue related to this paper is accuracy. In technique design, implementation, and validation, accuracy is a wide topic with so many related ethical issues. Technique inputs, internal processing, and output all have an impact on accuracy, and at each of these levels, there numerous significant ethical problems. The ability of technique analysts to identify and predict all states (particularly error states) is low for sophisticated systems (Granik & Mesyura, 2017). This contribute to software accuracy-related issues. These issues are best handled by documenting any assumptions, developing appropriate test conditions and carrying out a thorough technique verification and validation. It may appear that technique developers are ethically bound to rectify all the technique errors. However, handling errors raises ethical dilemmas, and it is projected that 15-20% of efforts to do away with program errors result in additional errors. For technique s with numerous lines of codes, like this work, the opportunity of coming up with a severe error when rectifying an initial error is huge that it could be a decent decision to retain and work around the initial error instead of trying to correct the error. This is one of the ethical issues related to the design of the proposed technique.

IV. RESULTS AND DISCUSSION

xi. Description of the Current Client Based Technique

A client based technique is a software component implemented at the user's computer in a browser plug-in or the users inbox to monitor incoming emails so as to identify the ones that can be classified as snoopers [15]

Current client based techniques uses user feedback or third party reporting to blacklist or whitelist an email. A filter is created that compares the extracted email features with a database of backlisted features thereby identifying snooping activities in emails [15]. However, there are several other factors such as spelling errors, abnormal DNS record and corrupted URL addresses that can be used to distinguish legitimate emails from snooped or fake emails. This paper uses text based classifier to analyse the other factors thereby extending the current client based technique.

This conceptual diagrams shows how the current client based technique classifies snooped emails and flag them. According to the concept, the current technique classifies extracted emails features based on the blacklist/whitelist approach. This conceptual was generated from was generated by the author based on the publication by [15].

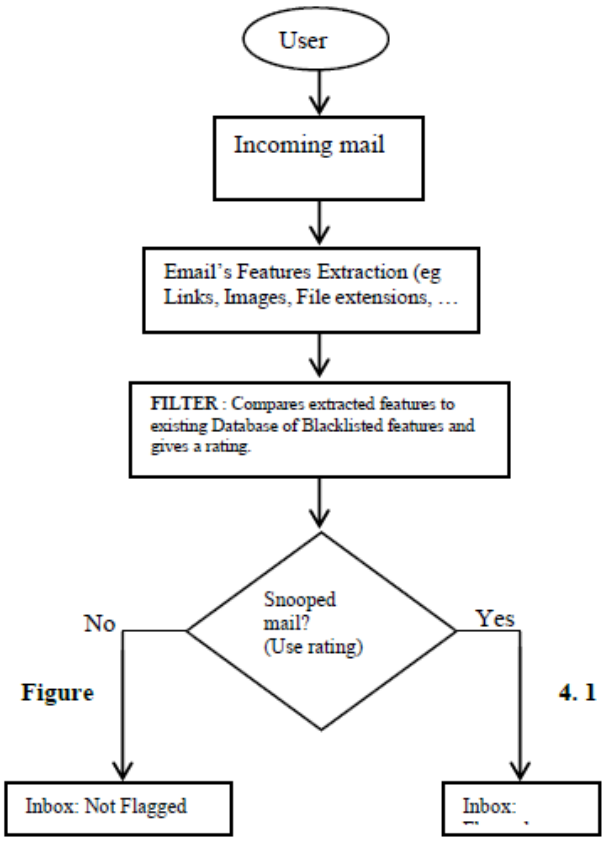

Figure 4. 1

Figure 4. 1 Description of the Current Client bases Technique [12]

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

d. Description of the Extended Client Based Technique

This paper expands the current client-based technique by introducing a Naïve Bayes classifier to analyse email content for identification of suspicious words. The rating for this classifier and the rating from the existing client based technique are combined to produce an improved classification threshold labelled combined ratings on Figure 4.2.

Figure 4.2 shows an illustration of the extended client based technique. A description of each component is given.

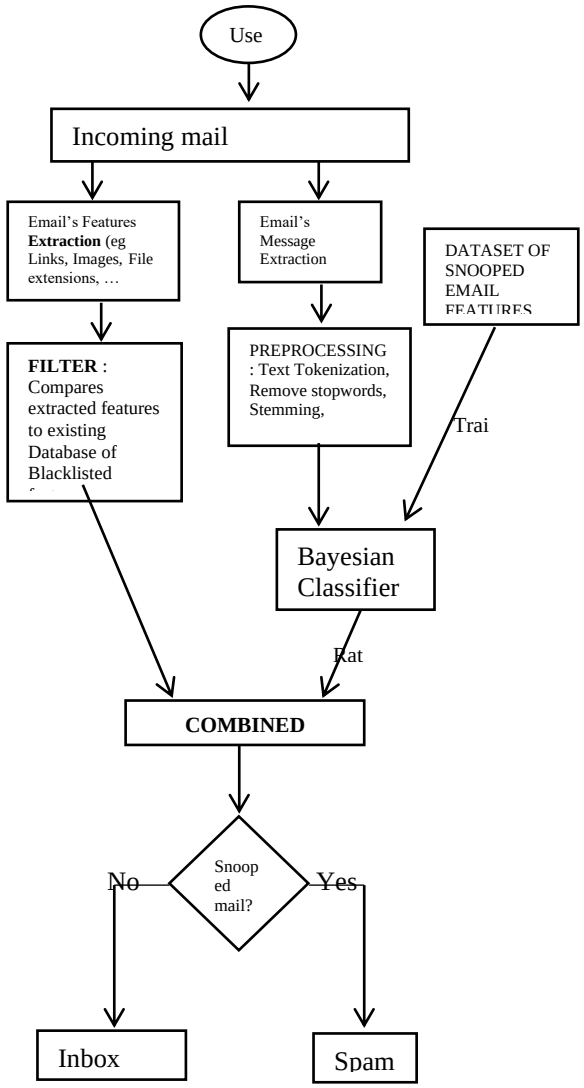

Figure 4. 2 The Extended Client Based Technique

The extended client based technique examines the user's email inbox to check whether there are new emails after successful login by the user. It then extracts the features of the new emails to identify their features. Using frequent expressions, links and other attachments are separated from the written text/message. The technique then fetches the written text, tokenizes it to do away with all the undesirable symbols so as to be left with words only. It also disposes all stop words from this category of words and passes the rest of the terms to the Bayesian Classifier to calculate the probability of each and every word being related to snooping emails. The technique calculates the log of the likelihood of each word being snooped and sums them so as to get a snoop score. Any word related to snooped emails is assigned 0 and any other not related to snooped emails is assigned integer 1. When the total number of zeros exceeds the ones, the message is considered as snooped.

The technique goes forward to analyse each and every link acquired in the email body. Analysis of the links on the approach of the current client-based technique. The technique then looks at the email attachments. At this point, it only selects the extension from the file name and compares it with mime types and a list of malicious file extensions provided in the current technique database. If the attachment is malicious, it is added to the positive status score of the email. The technique does not take out any attachment; it only acquires the name of the file of the enclosure, which exists in the Content-Disposition section of every email body according to the RFC822 format.

For all the analyses done, the reputation mark is cumulatively calculated with consideration to the number of checks that the report was performed. When an email attains a positive score from the class category by the Naïve Bayes classifier, the technique acknowledges that it is a snooping email and awards it a reputation score of positive. However, a reputation score of positive is also given to malicious links as well as that of the attachments so as to raise the correctness of the algorithm. A snooping email should have either a snooping link or a malicious attachment for it to be referred to as an attack. All emails with similar content that are sent to different clients will obtain similar scores and are moved to the spam folder. The technique frequently searches for new emails; the same process is repeated of analysis and identification of snooping emails. In case the attacker wants to send a malicious email again, the technique quickly identifies it since it is already hashed in the database. It is, therefore, flagged even without necessarily performing the complete analysis. This case only applies when the content of the email body remains the same thus, the email account user continues to read and receive clean emails regularly.

e. Mathematical Description of the Bayesian Classifier

Bayes theorem can be expressed mathematically as:

$$P(A \mid B) = \frac{P(B \mid A)\, P(A)}{P(B)}$$

Where A and B are events and P(B) #0.

P (A) and P (B) are the probabilities of observing A and B without regard to each other.

P (A| B), a conditional probability, is the probability of observing event A given that B is true.

P(B| A) is the probability of observing event B given that A is true

In the context of this paper, the above mathematical expression is a powerful theorem that can be used to build a spam classifier program in python that will tell if or not a message is a spam. It uses the probability theory known as the Baye's Theorem.

The following message m = (w1, w2..., wn), where (w1, w2…, wn) is a group of unique terms contained in the message. In this case, w1 stands for word 1, w2 for word 2, w3 for word 3,

and so forth. It will find the following:

$$P(\text{spam} \mid w1 \cap w2 \cap \ldots \cap wn) = \frac{P(w1 \cap w2 \cap \ldots \cap wn \mid \text{spam}).\, P(\text{spam})}{P(w1 \cap w2 \cap \ldots \cap wn)}$$

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

The Bayes theorem adopted for this paper suggests that the occurrence of a word is independent of all the other words. Thus, the above expression can be simplified to;

$$\frac{P(w1|\,spam).\,P(w2|\,spam)\ldots P(wn|\,spam).\,P(spam)}{P(w1).\,P(w2)\ldots P(wn)}$$

For us to classify the phrase, we need to determine which is greater;

$$P(spam \mid w1 \cap w2 \cap \ldots \cap wn) \text{ versus } P(\sim ham \mid w1 \cap w2 \cap \ldots \cap wn)$$

The technique executed two methods: Inverse Document Frequency (IDF) and Bag of words. It classified them one at a time. The technique started with the Bag of words.
In Bag of words technique, the 'term frequency is discovered,' this calculates how frequent a word occurs in a given dataset. For example, the number of events of each word in the data set. In this manner for word w,

$$P(w) = \frac{\text{Total number of occurrences of w in dataset}}{\text{Total number of words in dataset}}$$

And

$$P(w|\,spam) = \frac{\text{Total number of occurrences of w in spam messages}}{\text{Total number of words in spam messages}}$$

TF-IDF is used to represent the Term Frequency-Inverse Document Frequency (TF-IDF). To add on Term Frequency, calculate Inverse Document Frequency.

$$IDF(w) = \log \frac{\text{Total number of messages}}{\text{Total number of messages containing w}}$$

For example, there are two messages in the data set, 'hello too bar' and 'hello world.' Term Frequency ('hello') is 2. Inverse Document Frequency ('hello') is log (2/2). If a word frequently occurs, this means that the word gives fewer details.

In this technique, every word that is used has a score, which is Term Frequency (w)* Inverse Document Frequency (w). The probability of all the words is accounted for as:

$$P(w) = \frac{TF(w) * IDF(w)}{\sum \forall \text{ words } x \in \text{train dataset } TF(x) * IDF(x)}$$

$$P(w \mid spam) = \frac{TF(w|\,spam) * IDF(w)}{\sum \forall \text{ words } x \in \text{train dataset } TF(x|\,spam) * IDF(x)}$$

In addition, imagining a scenario where the technique experienced a word in the test data set, which is not part of the train data set. In such a case, P(w) will be 0; as a result, the P(spam|w) undefined (because there is need to divide by P(w), which is 0. Recall the formula?). To handle this issue, additive smoothing is added. In additive smoothing, number alpha is included to the numerator and include alpha times number of classes over which the likelihood is in the denominator.

$$P(w \mid spam) = \frac{TF(w|\,spam) + \alpha}{\sum \forall \text{ words } x \in \text{train dataset } TF(x) + \alpha \sum \forall \text{ words } x \in \text{spam in train dataset}}$$

When utilizing TF-IDF;

$$P(w \mid spam) = \frac{TF(w|\,spam) * IDF(w) + \alpha}{\sum \forall \text{ words } x \in \text{train dataset } TF(x) * IDF(x) + \alpha \sum \forall \text{ words } x \in \text{spam in train dataset}}$$

This is done with the goal that minimal likelihood of any word currently should be a finite number. To make additions in the denominator is to make the resultant sum of the considerable amount of probabilities of words in the spam messages as 1.

When alpha = 1, it is known as Laplace smoothing.
The above formulas based on Bayesian filter calculation were implemented in python to identify the probability of a word being either a spam or a ham;

The coding started by loading all the libraries by which the technique depended on. To load the dependencies the researcher used the code demonstrated below, whereby the researcher imported all the required python together with integrating data stores for stopwords. Stop words are the words that happen incredibly as often as possible in any content. For instance, words like 'the', 'an', 'a', 'to', 'is' and so forth. These words don not provide any data about the substance of the text. Therefore, it will not make any difference even if these words are removed from the text.

The technique utilized n-grams to enhance accuracy. When two words are used together, the meaning of the concept changes. For instance, 'great' and 'not great' are inverse insignificance. Assuming a book contains 'not great,' it is smarter to consider 'not great' as one token as opposed to 'not' and 'great.' Consequently, in some cases, accuracy is improved when we split the content into tokens of (at least two) words instead of one word.The NLTK (Natural Language Toolkit) was used to process the messages while Matplotlib and WordCloud was used for Visualization. pandas was used to input information that is reading the datasets into dataframes. NumPy to generate random probabilities for the train-test split to ensure the data representation was not biased as this would negatively affect the outcome of the technique, the best approach was to use random number generators to ensure an unbiased distribution of the dataset. sklearn was used for random shuffle. <import re> was used to import regular expressions in python.

```
#we are importing all the libraries we would like to use.
#including data stores for stopwords and such
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
import matplotlib.pyplot as plt
from wordcloud import WordCloud
from math import log, sqrt
import pandas as pd
import numpy as np
import re
from sklearn.utils import shuffle
%matplotlib inline
```

At this stage, the technique was directed to read the data sets from the location prescribed by the program

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

mails = pd.read_csv('datasets/ham_spam.csv', encoding = 'latin-1'). This code direct the technique to read a csv file "ham_spam" as a training and testing dataset.

The next step was to show the count of the train-test dataset so as to identify whether the dataset contained enough messages for the splitting of the dataset into train and test data.

The researcher then created another column named " label" where ham messages was assigned 0 and spam messages were assigned 1 . The indications of whether the email in the dataframe is ham or spam is indicated by binary narrations of either 1 or 0, that needed to be changed by mapping them to being either ham or spam. The code below maps the ham and spam texts to identities that can be easily used in processing

To train and test the technique, the dataset was split into two, that is, train data set and test dataset. The train dataset was used to train the technique and afterwards the test dataset was used to test the technique. 80% of the data set was used as a training dataset and the others as test dataset. The selection of 80% of the dataset is consistently random as is shown by the code below. The method generates random numbers using the uniform distribution of numbers technique, this is to ensure randomness of the data. To split the data into training and testing sets, and initialize the empty lists to hold training and testing data respectively. Also dataset had to be restructured that is the restructuring the structure of the columns in question, so as to allow for uniform processing for the testing data set.

From the training and the testing data splitted, the visualization of the keywords mostly used for snooping was done. The technique determined the most repeated terms in the spam messages. To achieve this purpose, WordCloud was used as a library.

After the technique training, the technique tokenizes the test data for classification. Before classification, the technique needs to pre-process the message. It first makes all the character into lower cases. This is due to the fact that 'FREE' and 'free' have the same meaning, and they should not be regarded as two different words. Tokenization is the act of separating a text into pieces and discarding the punctuation characters.

The words such as 'go,' 'goes,' 'going' indicates a similar movement. The technique can replace every one of these words by one word 'go.' This is known as stemming. Porter Stemmer was used for stemming, which is a well-known stemming algorithm.

The words such as 'go,' 'goes,' 'going' indicates a similar movement. The technique can replace every one of these words by one word 'go.' This is known as stemming. Porter Stemmer was used for stemming, which is a well-known stemming algorithm.

After the message pre-processing, the pre-processed words are introduced into the Bayesian classifier for the calculation of the probability of the message being a spam or ham. The classifying function returns true if the threshold for it being spam has been reached. otherwise, it is false, meaning ham. In the classifier, initialization was firstly done to initialize the training data as data to be used for processing. The technique was then directed to refer to the training which it had received earlier for self-method whereby it will differentiate the spam from ham based on the "label' column created earlier.

The calculation of TF-IDF of words in the message is then called under the function def calc_TF_IDF(self). This finds the frequency of occurrence of a word found in a message in all the snooped messages in the training datasets. which are the expressed calculations formulae in section above where the probability of a word being spam or ham is done.

For classification, in every word w in the processed messaged, the technique discovers a product of P(w|spam). If w does not exist in the training data set, the technique uses TF(w) as 0 and discover P(w|spam) using the above equation. It then multiplies this item with P(spam) The resultant thing is the P(spam|message). Essentially, it discovers P(ham|message). Whichever likelihood among these two is more noteworthy, the relating tag (spam or ham) is appointed to the dataset message. Note that it is not dividing by P(w), as shown in the equation. This is because that will distribute both the numbers and not influence the correlation between the two.

The code below shows the technique classifying a message.

The first message is "I cant pick the phone right now. Pls send a message" which is classified is ham meaning it's authentic

```
pm = process_message('I cant pick the phone right now. Pls send a message')
if(sc_tf_idf.classify(pm)):
    print("Message is spam")
else:
    print("Message is Ham")
```

```
Message is Ham
```

After the text classification, the technique classify the links and other attachments based on the blacklist used by the current client based technique as shown below.

```
In [25]: M import re
         def extract_url(myString):
             links = re.findall(r'(https?://[^\s]+)', myString)
             links = tuple(links)
             return links

In [26]: M from spam_lists import SPAMHAUS_DBL

         a_message = "No, this isn't a reference for the website which you used. http://dbltest.com Your reference implies that you ar
         available_links = extract_url(a_message)
         if(len(available_links)>0):
             if(SPAMHAUS_DBL.any_match(available_links)):
                 result = SPAMHAUS_DBL.filter_matching(available_links)
                 print("these are the spam links: ")
                 print(list(result))
             else:
                 print("No spam links found")
         else:
             print("no links found")

         these are the spam links:
         ['http://dbltest.com']

         a true result indicates the presence of a spam link, in this case, "http://dbltest.com"
```

f. Validation of the Extended Client Based Technique

For validation, 5583 messages were used for the developed technique validation purposes. The messages were sequentially run through the developed technique to generate the predicted technique label which was compared with the known technique label using Accuracy, Recall, Precision and F1 Score metrics for validation. Cosine of similarity was also used to generate the measure of similarity between the two data labels.

Retrieval Number: 100.1/ijeat.D22960410421
DOI:10.35940/ijeat.D2296.0410421

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

In the validation process of this study, the known email labels were compared against the technique output labels. The technique output labels were acquired by separately running each message in the dataset through the developed technique. The classification flag of a message being either spam or ham was termed as the technique output label. This comparison results assisted in calculation of true positive, true negative, false positive and false negative inorder to compute the four metrics used for technique validation measurement.. After validating the technique through the four metrics, the cosine similarity was used to evaluate a measure of similarity between the known message label to the technique message label.

i. Dataset description

Validation dataset was harvested from kaggle.com website, SMS Spam Collection dataset. The dataset was a set of SMS tagged messages that have been collected for SMS Spam research. It contains a set of 5,583 SMS messages in English, tagged according being ham (legitimate) or spam. This data was preferred since it was already classified and labelled and therefore was used as a baseline for comparative analysis. The files have one message in each of their lines.

Every line is made up of two columns: v1 is made up of the label (ham or spam) while v2 has the raw text. This corpus was gathered from free or free for research sources at the internet: A group of 425 spam SMS messages was obtained manually from the Grumbletext Web site which is a United Kingdom forum whereby mobile phone users report claims of SMS spam messages, majority of them without making any reports of the spam messages that are received. Identifying the text of spam messages from the claims is a difficult task and it entails meticulously scanning many web pages ranging into hundreds. A subnet of 3375 random SMS of ham messages that are randomly chosen of NU SMS Corpus (NSC), this is a dataset of close to 10,000 legitimate messages gathered at the Computer Science at the National University of Singapore for research purposes. The messages mainly come from Singapore citizens and the university students. The messages were gathered from volunteers that were aware of their contributions being made public. 450 ham messages were gathered from Caroline Tag's PhD Dissertation. Lastly, SMS Spam Corpus v.0.1 Big has been incorporated. It contains 1002 SMS ham and 322 spam messages and is publicly the other messages were mined from the researcher personal email.

ii. Description of the Metrics

The researcher used four metrics for validation. The metrics are; Accuracy, Precision, Recall and F1 score metrics. To calculate these metrics, the identification of the True negative, True Positive, False Positive and False negative based on the known message labels and the technic predicted labels was needed.

True Positives (TP) are the positive values estimated accurately. It is an outcome where the model correctly predicts the positive class. In this research, true positive occurs if the known data label shows that a particular message is spam and the predicated technique label suggests the same message is spam. True Negatives (TN) are the accurately estimated negative values. In this research true negative occurs when the known data label is spam and the predicted data label by the developed technique is ham. False Positives (FP) occurs when the test makes a positive prediction, and the subject has a negative result. For example, in this research, false positive occurs when the known data label is ham and the predicted technique label is ham. .False Negatives (FN)occur when a test result that incorrectly indicates that the condition being tested for is not present when, in fact, the condition is actually present.. For example, in this research, the false negative occurs when the known data label is ham and the predicted technique label is spam. Both the false negatives and false positives occur when there is a contradiction between the predicted class and the actual class. After the computation of these elements, the metrics were evaluated.

Accuracy metric is the ratio of accurately estimated observations to the entire observations, and it is the most intuitive performance measure. One might think that the best technique is gotten from the highest accurate technique. Although accuracy is a considerable measure, its accuracy is attained when one has symmetric data sets where the values of false negatives and false positives are almost the same. As a result, one should evaluate other factors before accessing the performance of a technique. The permitted level 0 accuracy is more than 50%. It is calculated as follows; Accuracy = (True Positives + True Negatives)/ (True Positives + False Positives + False Negatives + True Negatives)

Precision metric is the ratio of accurately estimated positive observations to the sum of the expected positive views. The maximum precision rate is related to the low False Positive rate. The permitted precision level is more than 0.5 precision. It is calculated as;Precision = True Positive /(False Positive +True Positive) Precision is a good measure to determine, when the costs of False Positive is high. For instance, in email spam detection, a false positive means that an email that is non-spam (actual negative) has been identified as spam (predicted spam). The email user might lose important emails if the precision is not high for the spam detection model.

Recall (Sensitivity) metric is the ratio of the accurately estimated positive observations to the overall observations in a real class – yes. The question answered by the recall is among the total number of passengers that survived. How many were labeled? The allowed level of recall is more than 0.5. It is calculated as; Recall = True Positive/(False Negative +True Positive). Recall calculates how many of the Actual Positives our model capture through labeling it as Positive (True Positive).

F1 Score metric is the calculated average of Recall and Precision. Thus, this Score includes both false negatives and false positives.

Intuitively F1 Score is complex to understand as compared to accuracy, although the F1 Score is more useful as compared to accuracy, particularly if the distributions one has been uneven. Precision performs best if both the false negatives and false positives have similar prices. It is advisable to look at both Recall and Precision if there is a difference between the rates of false negatives and false positives. The allowed level of F1 Score is more than 50%. It is calculated as;

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

F1 Score = (Precision*Recall)/ (Precision + Recall). F1 Score is needed when you want to seek a balance between Precision and Recall.

4.6.3 Validation results and interpretation

During the developed technique validation, 5583 emails were run through the extended client based technique to find the predicted technic label so that the researcher can be able to compare them with the true classification of the emails. According to this classification, this conditions were used:

- True Positive (TP)-occurs when the known message labels is spam and the technique prediction output is spam.
- True Negative (TN) occurs when the known message labels is ham and the technique prediction output is ham.
- False Positive (FP) occurs when the known message labels ham and the technique predicted output is spam.
- False Negative (FN) occurs when the known message labels is spam and the technique output ham.
- All messages have known (true classification) labels

After running all the messages while recording each message predicted label by the developed technique, table 4.6 indicates the binary classification output of the recorded data.

TABLE I

Binary Classification	Total numbers
TP	645
TN	4678
FP	131
FN	129
TOTAL	5583

The above binary classification results (total numbers) were used to calculate the validation metrics as indicated below;

Accuracy metric = (TP + TN)/ (TP + FP + FN + TN)
=(645+4678)/645+131+129+4678) = 0.95=95%

Precision metric = TP/FP+TP
=645/ (131+645) =0.83=83%

Recall Metric= TP/FN+TP
=645/ (129+645) = 0.83=83%

F1 Score=2*((Precision*Recall)/(Precision+ Recall)).
= 2*((0.83*0.83)/ (0.83+0.83)) = 0.83=83%

The above-calculated metrics are summarized in the table 4.7.

TABLE II: METRICS OUTPUT CALCULATED VALUES

Metric	Values in %
Accuracy	95
Precision	83
Recall	83
F1 Score	83

The technique output had 95% accuracy. This means that for all the 5583 messages tested, 5528 results were correct. Which is also a translation that for every 100 messages predicted, 95 messages were correctly predicted. The technique prediction output also had 83% precision. This means that of all the true positive results predicted, 83% were correct. Therefore, (83% of 697= 622) out of the 697 true positive results predicted, 622 were correct. The technique prediction result also had 83% recall. This means that of all the correct true positives (622) predicted, 83% was correct. Therefore, (83% of 622= 516) out of the 622 correct true positives prediction, 516 were correct. The Final metric was the F1 score whereby the technique output attained an F1 score of 83%. This means that the weighted average between precision and recall was at 83%. In conclusion, because the minimum accepted value for the four metrics is 50% and the higher the value the better, having attained minimum accepted value/threshold with such a higher output value for the four metrics, the researcher hereby concluded that the developed technique was valid

iii. Cosine Similarity

In this sub-section, the cosine of similarity was used to measure the similarity between the known message labels to the message labels produced by the developed technique.

A cosine similarity is a metric utilized to measure the similarity of documents irrespective of their size. Mathematically cosine similarity is used to calculate the cosine of the angle between two vectors, projected in a multidimensional space. In cosine similarity, the result obtained from computing the similarity of product A and product B is similar to calculating the similarity of product B to product A. Therefore, the score of each pair of nodes can be computed once.

$$similarity(A,B) = \frac{A \cdot B}{\|A\| \times \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}}$$

The cosine similarity formula was converted/put into a python 3.6 code and run in jupyter notebook. Both the known message labels and the technique output labels were put in a linear numbering and fed to the coded cosine similarity measurement in jupyter notebook. Figure below shows a screenshot of the cosine similarity code in python measuring the similarity between the known message labels and the technique labels.

```
In [10]: import math
         def cosine_similarity(KnownLabel,TechnLabel):
             "compute cosine similarity of KnownLabel to TechniqueLabel: (KnownLabel dot TechniqueLabel)/(||KnownLabel||*||TechniqueLabel
             sumxx, sumxy, sumyy = 0, 0, 0
             for i in range(len(KnownLabel)):
                 x = KnownLabel[i]; y = TechnLabel[i]
                 sumxx += x*x
                 sumyy += y*y
                 sumxy += x*y
             return sumxy/math.sqrt(sumxx*sumyy)

         KnownLabel = [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, (
         TechnLabel = [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, (
         print(cosine_similarity(KnownLabel,TechnLabel))

         0.7803834824827595
```

Based on the cosine similarity output of 0.78 produced in figure 4.31, it can be concluded that the validation labels compared for validation, attained an accepted similarity threshold therefore it was valid.

g. Conclusion

This chapter outlined the procedure for designing, implementing and validating the extended client based technique developed by the study.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

The chapter has described the different types of UML diagrams used in technique implementation all of which helped to connect shapes representing an object or class with other shapes to provide an understanding of the relationships and information flow within and outside the developed technique. The chapter has also covered the Bayesian theorem, which was the classifier used for classification in the developed technique. The classifier combined the aspect of Bag of Words and Inverser Document Frequency to arrive at its classification results. In this chapter sub-section, the overall coding of the developed technique has also been described with supported screenshots of the code lines. The chapter also covered the validation methods of the developed technique. In the validation, Accuracy metrics, Precision Metrics, Recall Metrics and F1 Score metrics were used to validate the developed technique. The metrics were calculated from the binary classification values, which are true positive, true negative, false positive and false negative, which were attained after comparing the known email labels to the labels produced by the developed technique after running the emails through it. After the calculations of the four metrics, the results attained the minimum accepted threshold of each metric calculated with; Accuracy metric at 95%, precision metric at 83%, recall metric at 83% and F1 score metrical 83%. Therefore, this proved that the developed technique was valid since the accepted minimum threshold of the four metrics is 50%. Cosine similarity was also used to measure the similarity between the known message labels and the developed technique output label. The cosine of similarity results for the validation data was 0.78, which concluded that; the validation data attained the accepted minimum threshold of 0.5 hence, the data was valid.

V. SUMMARY, CONCLUSION AND RECOMMENDATIONS

h. Summary of findings

Snooping attacks are an appalling threat in the web security domain. Because of the rapidly changing technology, snooping techniques have advanced in process, and crucial entities such as banks continue to lose millions of cash every year via email snooping. Visual similarity snooping attacks happen when large amounts of emails are spoofed requesting the targeted users to click and open the emails that are embedded. Visual similarity snooping attacks are the types of snooping attacks that mimic emails and snoops the banking services that are offered [17]. The legitimately looking emails are sent by the snooping attacker to gather sensitive and confidential information. This has become a norm over the years despite institutions investing hugely in monitoring, filtering tools and enhancing awareness by training the staff members on how to easily detect snooping emails.

Several techniques have been developed to detect and prevent snooping attacks. They may be categorized as either software or education based anti-snooping techniques. Software-based techniques may be further categorized as heuristic based, list-based, or visual similarity based techniques. However, most of these approaches still have limitations like accuracy, failing to detect embedded objects, amongst other limitations. Moreover, snooping attacks reveal a hideous characteristic, which means that the above techniques are not able to adequately identify them in their hidden forms. After analyses of the current client based technique proposed by Marchal et al. 2016 with a goal of extending, the researcher found that there is the need to improve their proposed approaches by adopting additional features alongside text classification and moving of snooped emails from inbox to spam box (12). Removing other features of their approach may also increase the system's running time complexity which reduces the efficiency of their technique. Moreover, specific features like domain age also need third party assistance which may not be dependable. Based on a review of the existing literature on the existing mechanisms of detecting snooping attacks, the researcher identified a research gap. The gap was that there is no linguistic technique that has been used to capture the commonly used terminologies in snooping emails such that the technique can integrate the evidence of snooping key phrases from classification of text and embedded links to detect snooping attacks via email (18). This called for a need to develop and implement a technique that is able to collectively check against snooping attacks by checking for malicious attachments and malicious links. This paper thus, presented an extended client-based technique that visually compares snooping emails from a legitimate mail. The paper classified snooping emails, using Naïve Bayes classifier, in order to attain in-depth defence by developing a technique that can detect and flag snooping emails before they reach the target users. The paper identified several algorithms that may be used in classifying text in order to come up with an appropriate classifier for classification. They entail the Random forest algorithm, SVM (Support Vector Machine), Naives Bayes, K-Nearest Neighbour, Decision Tree, and the Bayesian Network Algorithm. Each of these techniques was covered, together with their applications and limitations [20]. The main reason why the proposed technique used Bayes classifier is that it only needs a small amount of training data for estimating the parameters (which includes variances and means of the variables) needed for classification. The Naïve Bayes classifier is also a probabilistic technique , which means the algorithm can be easily coded and the probabilities made in real-time, making it highly scalable and is conventionally the algorithm of choice for real-world applications that need to respond to the requests of use in real-time. The developed technique is the extended client based whereby it classifies the emails from the application network layer. The technique training and testing datasets were mined from different programming community websites such as kaggle, buzzfeed and fivethirtyeight. The technique was able to analyze the snooping links, evaluate email attachments that are malicious in an email and flag them. The technique was validated using four metrics; that is accuracy metric which was at 95%, precision metric which was at 83%, Recall metric which was at 83% and F1 score metric which was at 83% while real and known data were checked for similarity using the cosine similarity.

The cosine similarity showed that there is high positive similarity between the technique output labels and the known output labels.

This is significant to this study as it showed that the technique generated results and known data collected and analyzed results were matching.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

The explained relation is also high but not perfect implying that the information can actually be explained at a higher margin. This study validation emphasized more use of the four metrics for validation and according to the calculated metrics, it was concluded that the metrics results were high; implying that the recommended technique for detecting Visual Similarity Snooping attacks in emails is valid.

i. Conclusion

This paper presented an effective and novel approach for checking against snooping attacks by checking for malicious attachments and malicious links using an extended client-based technique that visually compares snooping emails from legitimate mails. In a real-time environment, detecting snooping attacks ought to be very fast and effective. Approaches such as the black-list based techniques are quite fast. However, they fail to detect zero-hour snooping attacks. Visual similarity-based approaches are overall time consuming, need huge memory, and may fail to identify zero-hour attacks. Heuristic approaches, on the other hand, may detect zero-hour attacks, but their performance is hugely reliant on the feature set, classifier, and training data. The extended client-based technique proposed in this paper, sought to take care of this limitations by adopting the Naïve Bayes classifier algorithm. Naïve Bayes is itself a simple and powerful technique that can be easily understood. It gives good results and makes a technique fast to build and make predictions due to its need for a smaller amount of training datasets. The technique was able to analyze the snooping links, evaluate email attachments that are malicious in an email and flag them. Upon a validation process to assess its effectiveness in checking against snooping attacks, the results from the metrics output were satisfactory, which means that the proposed technique is valid.

j. Recommendations

This paper discussed different classification techniques together with their possibilities and weakness in knowledge extraction from data. At this stage, it is important to be aware of the problems related to different text classification techniques, so that judging different classifier would be less challenging. The results of this research were a success in terms of detecting snooping attacks by checking for malicious links and attachments. The proposed technique used the Naïve Bayes text classifier algorithm, and upon testing, it gave satisfactory results. However, as mentioned previously in the literature review section, a detailed comparison with other text classification algorithms revealed that Bayes classification is outperformed by more recent classification approaches such as artificial neural network, random forest and boosted trees, and is significantly gaining significance in text classification due to the efficiency. Moreover, it is impractical to prescribe a specific classifier for the snooping attack problem. In this line, simpler yet powerful algorithms like random trees ought to be considered by future research, with a particular emphasis on how such text classifier algorithms may be used to detect snooping attacks.

Moreover, this research feels that there is need to improve on the proposed technique. In this line, the following recommendations have been made:

I. Because of the huge number of emails that specific users get, there is need to increase the speed of snooping detection while ensuring accuracy. The only way the same can be realized is by directing additional focus on adaptive machine learning.
II. Most malicious links and attachments are sent through email. Thus, email platforms should put in place third factor and Single Sign-On authentication like in Gmail. The Single Sign-On option will enable access to different independent software components that are related on one login instance. This will guarantee the security of users' private information by enabling login to email accounts through applications like algorithm access while ensuring verification through third party authentication. Here, users allow the algorithm to access their accounts, upon which they are notified of any other login attempt that have been permitted.
III. Accuracy was a primary ethical issue related to this work. Although a properly trained Naïve Bayes text classification algorithm is relatively accurate and faster to train compared to other classifier builders, there is room to further improve the accuracy of the classifier used in the proposed technique. This includes tuning the classifier by adjusting the tuneable parameters of the classifier, integrating classifier combination techniques such as ensembling and ego-boosting, looking at the data keyed into the classifier- and either adding extra data, improving basic parsing, or improving the selected features from the data. Improving the accuracy of the algorithm also means extra focus on feature selection and pre-processing. Feature selection is a process of manually or automatically selecting features that will contribute significantly to the prediction variable of the output of interest. This feature will filter features and do away with the irrelevant features that could limit the technique accuracy.

On the other hand, pre-processing will transform a series of raw text strings into structured vectors via processes such as synonym finding, stemming, and neutral words. The Fisher Method may also be integrated into the proposed technique to optimize the Naïve Bayes Classifier. The classifier used in the proposed technique makes use of feature probabilities to come up with an entire document probability. In addition, the classifier evaluates the probability of every category for every feature before combining the probabilities of these features to compare it against the probability of other random features.

k. Recommendations for future work.

This study majorly focused on extended client-based technique as a technique that can be used to detect Visual Similarity Snooping attacks in emails. Although the experimental results of the proposed technique were encouraging, they are relatively preliminary, and the variety of related techniques gives room for more comparative studies. For instance, it should be determined to which degree the techniques for inducing the Naïve Bayes classifier and the probabilistic concept offer benefits beyond text classification which has been the main focus of this study. More studies should also be carried out to determine the effect of the design decisions made when implementing the Naïve Bayes classifier. The usefulness of other text classification algorithms with regard to detecting snooping attacks should be further investigated.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

More directions for further research on this study include conducting replica experiments, which enhance the classification and flagging of the snooped emails at the network server to create a new classification method for industrial use. Based on the Naïve Bayes classifier, future work should also focus on optimizing term frequency estimation as this would contribute to a greater percentage of accuracy of the algorithm in specific classification cases. Additionally, future work should focus on developing simpler algorithms, with regard to both coding and implementation, improved techniques for knowledge distillation, multilingual text classification, domain knowledge integration, and subjective detection, all of which are areas that researchers can explore in this regard.

REFERENCES

1. M. A. Adebowale, K. T. Lwin, E. Sanchez and M. A. Hossain, "Intelligent web-phishing detection and protection scheme using integrated features of Images, frames and text," International Journals of Expert Systems with Applications, pp. Vol 115, Pg 300-313, 2018.
2. K. Jain and B. B. Gupta, "Phishing Detection: Analysis of Visual Similarity based approaches," Journal of Security and Communication Networks, pp. 1-21, 2017.
3. J. Jansen and R. Leukfeldt, "Phishing and malware attacks on online banking customers in the Netherlands," A qualitative analysis of factors leading to victimization. International Journal of Cyber Criminology, vol. 10, no. 1, pp. 79-91, 2016.
4. Y. Zhang, J. Hong and L. Cranor, "CANTINA: a content-based approach to detecting phishing websites," in In Proceedings of the 16th international conference on World Wide Web, Pittsburgh, 2007.
5. M. Moghimi and A. Varjani, "New rule-based phishing detection method.," Expert systems with applications, vol. 53, pp. 231-242, 2016.
6. R. Dhamija, J. D. Tygar and M. Hearst, "Why phishing works," In Proceedings of the SIGCHI conference on Human Factors in computing systems, pp. 1-10, 2006.
7. W. Liu, X. Deng, G. Huang and A. Fu, "An antiphishing strategy based on visual similarity assessment," in IEEE Internet Computing, 2006.
8. K. Jain and B. B. Gupta, "Comparative analysis of features based machine learning approaches for phishing detection.," in In 2016 3rd International Conference on Computing for Sustainable Global Development, New Delhi, India, 2016.
9. W. Richert and L. P. Coelho, Building machine learning systems with Python, Birmingham, UK: Packt Publishing Ltd, 2013.
10. S. Judith and R. B. Johnson , "How to Construct a Mixed Methods Research Design," KZfSS Kölner Zeitschrift für Soziologie und Sozialpsychologie, vol. 69, no. 2, p. 107–131, 2017.
11. P. D. Berger, R. E. Maurer and G. B. Celli, Experimental Design with Applications in Management, Engineering and the Sciences, New York City, USA: Springer International Publishing, 2018.
12. S. Marchal and N. Asokan, "On Designing and Evaluating Phishing Webpage Detection Techniques for the Real World," In 11th {USENIX} Workshop on Cyber Security Experimentation and Test ({CSET}, pp. 1-8, 2018.
13. M. Granik and V. Mesyura, "Fake news detection using naive Bayes classifier.," in First Ukraine Conference on Electrical and Computer Engineering (UKRCON) (pp. 900-903), Kiev, Ukraine, 2017.
14. P. Malathi and P. Vivekanandan, "Client Side Script Phishing Attacks Detection Method using Active Content Popularity Monitoring," International Journal on Future Revolution in Computer Science & Communication Engineering, vol. 4, no. 4, pp. 579-584, 2018.
15. Mishra, "Metrics to Evaluate your Machine Learning Algorithm," [Online]. Available: https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234. [Accessed 20 january 2020].
16. J. Patel, "Design and Implementation of Heuristic based Phishing detection technique.," DSpace Library, Online, 2018.
17. McCarthy, K. (2017). Polish banks hit by malware sent through hacked financial regulator. Retrieved March 2, 2019, from The Register: https://www.theregister.co.uk/2017/02/06/polish_banks_hit_by_malware_sent_through_hacked_financial_regulator/
18. Samuel, M., Kalle, S., Nidhi, S., & Asokan, N. (2016). Know your phish: Novel techniques for detecting phishing sites and their targets. In 2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS) (pp. 323-333). IEEE. Retrieved from https://arxiv.org/pdf/1510.06501.pdf
19. Sharma, A. (2019). Importing Data into Pandas. Retrieved 03 20, 2020, from Data Camp: https://www.datacamp.com/community/tutorials/importing-data-into-pandas
20. Saini, I., Singh , D., & Khosla, A. (2013). QRS detection using K-Nearest Neighbor Algorithm (KNN) and evaluation on standard ECG databases. Journal of advanced research, 4(4), 331-344.
21. Patel, J. (2018). Design and Implementation of Heuristic based Phishing detection technique. Online: DSpace Library. Retrieved from https://dspace.library.uvic.ca/bitstream/handle/1828/9880/Patel_Jaynish_MEng_2018.pdf?sequence=1&isAllowed=y

AUTHORS PROFILE

Mr. George Mwangi Muhindi is a Master's Student at Murang'a University of Technology pursuing a Master's Degree in Information Technology. He holds a Bachelor of Business Information Technology from the Jomo Kenyatta University of Agriculture and Technology (JKUAT). His interests include Cyber Security, Machine Learning, and Software Development.

Dr. Geoffrey Mariga Wambugu is a Lecturer and Ag. CoD of Information Technology (IT) Department at Murang'a University of Technology. He obtained his BSc Degree in Mathematics and Computer Science from Jomo Kenyatta University of Agriculture and Technology in 2000, and his MSc Degree in Information Systems from the University of Nairobi in 2012.
He holds a Doctor of Philosophy in Information Technology degree from JKUAT. His interests include Machine Learning and Text Analytics. Dr. Mariga has been involved in the design, development and implementation of IT/ICT and Computer Science Curricula in different Universities and Colleges in Kenya.

Dr. Aaron Mogeni Oirere is a Lecturer and CoD of Computer Science Department at Murang'a University. He obtained his BSc. Degree in Computer Science from Periyar University in 2007, and his MSc. Degree in Computer Science from Bharathiar University in 2010. He holds a Ph.D. in Computer Science from Dr. Babasaheb Ambedkar Marathwada University. His interests include: Database Management Systems, Hardware & Networking, Human Computer Interface, Information Systems, Data Analytics and Automatic Speech Recognition. He has presented papers in scientific conferences and has many publications in refereed journals.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication