A Behavioral Based Detection Approach for Business Email Compromises

by

Nasim Maleki

Bachelor of Software Engineering, Shahid Beheshti University, 2014

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Computer Science

In the Graduate Academic Unit of Faculty of Computer Science

Supervisor(s):	Ali A. Ghorbani, Ph.D, Faculty of Computer Science
Examining Board:	Suprio Ray, Ph.D, Faculty of Computer Science, Chair
	Rongxing Lu, Ph.D, Faculty of Computer Science
	Donglei Du, Ph.D, Faculty of Management, UNB

This thesis is accepted by the
Dean of Graduate Studies

THE UNIVERSITY OF NEW BRUNSWICK

December, 2019

Abstract

The most recent infectious vector in email attacks is Business Email Compromise (BEC), which is an entry point for attackers to get access to an enterprise network and obtain valuable company data. According to the Symantec Internet Threat Security Report (ISTR), around 7,710 organizations are hit by a Business Email Compromise attack every month. A BEC is a type of phishing attack that criminals impersonate a person of authority in an organization (CEO) through spoofing or take-over accounts. Since spoofing techniques are detectable using SPF, DMARC, and DKIM, we proposed and implemented a behavioral-based framework for the detection of BEC when accounts or machines are compromised. This framework stops malicious emails on the sender-side because the lack of enough email of the sender on the receiver-side cannot result in a representative user-profile. Moreover, a compromised account or machine turns into a devastating weapon targeting many people. Hence it ought to be stopped from the sender-side, and the real owner should be notified of this disaster. Our framework in the experiment on Enron Dataset for all users has reached a total average of 92% and 93% for Accuracy and F1 score, respectively.

Dedication

In memory of my Dad, who always believed in my ability and gave me the courage to run from what's comfort. You are gone, but your belief in me has made this journey possible.

To my Mom and my Siblings with Love and Eternal Appreciation.

Acknowledgements

I would like to extend my deepest gratitude to my supervisor, Dr. Ali Ghorbani, for his unwavering guidance, knowledge, and unparalleled support throughout this thesis.

I also had the great pleasure of working with Dr. Arash Habibi Lashkari as my project manager in industry-side projects for his support and profound belief in my work.

I cannot leave UNB without the help and kindness of my friends and colleagues at the Canadian Institute for Cyber Security, who always helped me keep my spirits high during hard times.

Table of Contents

List of Tables

List of Figures

List of Symbols, Nomenclature or Abbreviations

SPF	The Sender Policy Framework is an email-authentication technique
DMARC	Domain-based Message Authentication, Reporting and Conformance
DKIM	DomainKeys Identified Mail is an email authentication method
MTA	A Message Transfer Agent or Mail Transfer Agent or Mail Relay
MUA	Mail User Agent is a computer application to send and retrieve email
BEC	Bussiness Email Compromise
WSS	Within-cluster Sum of Square in k-means clustering
PGP	Pretty Good Privacy is an encryption program
MIME	Multipurpose Internet Mail Extensions extends the format of emails
CBOW	Common Bag Of Words Model
Word2Vec	Word To Vector
t-SNE	t-Distributed Stochastic Neighbor Embedding
PCA	Principal Component Analysis

Chapter 1

Introduction

1.1 What is Bussiness Email Compromise (BEC)?

Email is the first step of different types of attacks in the cyber world. It is often used to spread malware, spam, and phishing attacks. Phishers use deceptive content to lure people and gain their sensitive information through malicious URL, Drive-by Downloads, and Malicious Attachments.

Email is also a common entry point for attackers looking to get access to an enterprise network and obtain valuable company data. This type of attack is called a Business Email Compromise (BEC), which is increasing every day. Around 7,710 organizations are hit by a Business Email Compromise attack every month, and spear-phishing is the most widely used infection vector, according to the Symantec Internet Threat Security Report (ISTR) [12].

[30] and [31] have tried to understand what happens after a webmail credential is leaked by applying honeypot methods. They monitored abused Gmail accounts for seven months and came up with a taxonomy. 1- Curious people 2- Golden diggers who try to find something valuable through that account. 3- Spammers use that account to send spam. 4- Hijackers.

According to this study, if an account that belongs to a person of authority gets pwned by a spammer, he can impersonate that high-rank person and targets all employees of the company. A BEC is a type of phishing attack that criminals impersonate a person of authority in an organization (CEO), and tries to get employees' sensitive information to the phisher. Topics [2] of BEC usually are CEO fraud, Bogus invoice scam, Attorney impersonation, Account compromise, and Data theft. Unlike traditional phishing attacks, which target a large number of individuals, BEC attacks are highly targeted. Cybercriminals will scrape compromised email inboxes, study recent company news, and research employees on social media sites in order to make these email attacks look as convincing as possible. This high level of targeting helps these email scams to bypass filters and evade email whitelisting campaigns. Moreover, it makes it harder for employees to recognize that the email is not legitimate and coming from an untrusted source.

1.1.1 Attack Target Perspective

In terms of **attack target** there are different kinds of phishing attacks:

- **Regular Phishing Attacks**, This type of phishing emails would be sent toward victims en masse.
- **Spear Phishing Attacks**, While in traditional phishing, mass emails are sent to as many people as possible, spear phishing is a targeted attack in which the hacker knows what specific individual or organization. They research the target to make the attack more personalized and increase the likelihood of the target falling into their trap.
- **Whaling** is a type of spearphishing which is targeting high-profile targets like C-level executives.

- **Bussiness Email Compromise (BEC)** as a man-in-the-middle email attack is an exploit in which the attackers get access to a corporate email account and spoof the owner's identity to defraud the company or its employees, customers, or partners of money.

1.1.2 BEC with Attack Vector Perspective

In terms of **attack vector** there are different kinds of BEC attacks using Technical subterfuge:

1. **Creating Similar Account**: An attacker creates an account with an email address that is very similar to ones in the corporate network.

2. **Envelope From Spoofing**: One technique to spoof email addresses is through the invisible field in the email, which is Envelope From.

3. **Header From Spoofing**: The other technique for running email spoofing is through spoofing the Header From.

4. **Display Name Spoofing Attack**: Impersonating through spoofing the name in the Header.

5. **Compromised Account (mailbox) and Machine Attack**: Impersonating through the compromised account.

BEC Detection refers to the fraudulent email-sender detection in Business Email Compromise or Email Account Compromise or spear phishing in enterprises. In the next chapter, we will go over the BEC attack techniques and the solutions to them. We propose a detection framework to stop BEC attacks on the sender-side before it is received by the victims.

1.2 Summary of Contributions

The main purpose of this thesis is to develop a new framework for detecting BEC attacks through compromised accounts or machines on the sender-side. The contributions of this thesis can be summarized as follows:

- Proposed and implemented a behavioral-based framework for the detection of BEC on the sender-side that enhances the previous frameworks by utilizing user and group profiling.
- Implementing a parser to get the primary features from the text of emails.
- Implementing different modules for different categories of features.
- The feature extractor can be extended with new modules for getting features in other categories.
- Supporting timely analysis of streaming features such as network-based and social-based features.
- Stopping BEC from the sender side before sending.
- Improving detection of BEC emails through a two-stage firewall made by behavioral profiling.
- Our experimental evaluation to detect BEC attacks performs reasonably. The average accuracy is 95% for people who have sent more than 500 emails. The total average of Accuracy and F1 score for all users has reached more than 92% and 93%, respectively.

1.3 Thesis Organization

The rest of this thesis is organized as follows:

Chapter 2 reviews different types of email attacks and different detection approaches to stop them. Email attacks include phishing attacks, spear phishing emails, and BEC attacks. Machine learning methods and Natural Language Processing can help to detect phishing and spear emails. Afterward, by analyzing the email structure, we found out different kinds of impersonation techniques in BEC, such as name spoofing and email address spoofing. Finally, we found that one of the methods that still is challenging in BEC detection is when it happens through a take-over account or compromised machine.

Chapter 3 provides a general overview of the framework, followed by a description of each module. It explains how our two-step alerting system detects malicious emails sending from the compromised accounts. First, emails are analyzed through the parser. Then, a two-level feature extractor extracts different categories of features from the parsed emails. Finally, the framework will make user profiles for each user. It then inegrates user profiles to form the group profile. The framework detects BEC attacks using a two-level alerting system according to the user and group profiles.

The design and implementation criteria of the proposed framework are described in Chapter 4. This chapter introduces the main components and classes implemented in our framework and lists their primary functions. Chapter 5 reports the evaluation dataset, metrics, and comparative experimental results of the proposed framework with one existing spearphishing detector on the sender side. Finally, Chapter 6 concludes the thesis by discussing its contributions, limitations, and possible improvements to the work that has been done in this thesis.

Chapter 2

Literature Review

2.1 Email Security

Email is an entry point and the first step of different types of attacks in the cyber world. Email is often used to spread malware, spam, and phishing attacks. Phishers use deceptive content to lure people and gain their sensitive information by persuading them to click on the malicious hyperlinks embedded in the email. In addition to the malicious URL, we face other attack scenarios, likely including Malicious Attachments, and Drive-by Downloads, which is installing malware on the victim's device. Email is also a common entry point for attackers looking to get access to an enterprise network and obtain valuable company data. This type of attack is called a Business Email Compromise (BEC), which is increasing every day. Around 7,710 organizations are hit by a Business Email Compromise attack every month, and spear-phishing is the most widely used infection vector, according to the Symantec Internet Threat Security Report (ISTR) [12]. Hence, Email security includes various technologies to keep sensitive information safe in email communication and make accounts secure once unauthorized access, loss or compromise happen [10].

We have studied different types of email attacks, as well as several proposed solutions to prevent and detect them. Our purpose of this study is to find an unresolved and the most problematic email attack these days. Finally, after going over the current solutions, we will propose a detection framework which is also covering the weaknesses of existing solutions.

2.1.1 Phishing Detection

A phishing attack adopts technical subterfuge and social engineering techniques to get access to people's credentials or their sensitive information such as electronic bank account or email account. Attackers target people en masse. Typically, phishers benefited from people's emotional weaknesses and put them in an urgent or stressful situation to persuade them to reveal their credentials. Researchers and industry have come up with detection approaches using Machine Learning methods to prevent phishing attacks.

2.1.1.1 Machine Learning Approach

To detect phishing emails, several filtering systems [46][32][29] have applied machine learning algorithms to train classifiers with previous phishing incidents. These methods extract features from hyperlinks, header, and body of emails. PhishCatch[46] is an old framework to detect phishing emails based on some rules and heuristics. It has three modules, a module for fetching emails, Filtering and classifier module, and Alerter module. The detection is based on the "recvdFrom" and "From" features in the header, and the features from hyperlinks such as special characters, IP found, number_of_folder, number_of_subdomain, length of phishing link.

Two papers [32] and [29] have applied SVM classifier to detect phishing emails. In this method[32] first, it has a test module to detect obfuscated URLs based on 1-DNS Test 2-IPAddress Test 3-URL Encode Test 4-Shorten URL Test 5-White List Test 6-Black List Test 7-Pattern Matching Test.

Then it will train SVM classification with features from hyperlinks. They[29] used a new hybrid classifier which is combining cuckoo search(CS) and SVM due to avoid the effect of kernel parameter on accuracy. They extracted 23 features from Header, Body, and URL.

The following items are some of the most applicable features utilized by phishing filtering frameworks:

- **Header Features** Sending time of email is not in normal work time, Presence of dark copy, Content Type of email (HTML Mail, Textmail,Multi Part Mail), Number of CC, BCC, and To, Whether a reply email, DKIM signature, SPF

- **Hyperlink Features** Having IP Address, URL Length, Shortening Service, Having @, -, Symbol, Double slash redirecting, Having Sub Domain, Abnormal URL from Who-Is Database, Age of domain, Page Rank, Links Pointing to page, Number of dots in domain name, BlackListURL, Disparities between href attribute and LINK text, Presence of sensitive words in the LINK text, Active duration of domain names, ALEXA rank, Registration duration, Number of http/https in an URL,

- **Body and Subject Features** Exsiting some specific verbs, You, Your in a sentence segment, Presence of order, payment,and RE- in the title, Presence of javascript (Onclick,Popup, ...), HTMLform, Number of Attachments, Size of Document, Body has words (Dear, Suspension, verify, Function), Number of Chars in subject and body, Number of Words in subject and body, Number of Unique Words, Subject has special words (Bank, Debit, Fwd, Reply, Verify)

Applying Artificial Neural Network and Reinforcement Learning Using the artificial neural network and reinforcement learning, it can adapt itself to produce a new phishing email detection system that reflects the new behaviors of phishing attacks [40]. Also Nguyen et. al [28] have applied deep learning and hierarchical long short-term memory networks to model the body of phishing emails(at the word and sentence level) which is improving anti-phishing detection tools.

Applying Natural Language Processing Phishing emails generally have a specific context to access people's property. For instance, if phishers attempt to get access to a victim's account, it is probable to see "your account" in the email content. To extract such contextual features from the body of emails, Natural Language Processing(NLP) plays a vital role. [33][44][43] took advantage of NLP modules such as n-gram, stemming, removing stop word, wordnet, lexical analysis, part of speech tagging, etc. PhishNetNLP [44] has provided a framework utilizing NLP techniques and all information in the content of the email(link, header, text body). There are three classifiers and three types of feature groups: Text Analysis, Link Analysis, Header Analysis. For combining the result of these three classifiers, "Majority Voting" has been applied. Text Analysis has two features. The first is Text Score based on "link indicator words and verbs", "Number of hyperlinks", "urgency words", "place indicators", and their subsets. The other feature is the Context Score calculating TF-IDF for vectors of words, then it computes the similarity of the new receiving email with the previous email and assigns a value between zero and one to the context score. Then it combines these two scores as a final text score.

For the Link feature, it extracts all domains of URLs. If the size of the vector is 0, it would be considered as legitimate. Otherwise, if it is more than ten, It will search the top four of them using their TF-IDFs. If they appear in the first 30 results, it will be marked as legitimate otherwise not.

The header analysis classifier of this paper is more advanced than PhishCatch[46]. It extracts features from "FROM", "Delivered-To", "Received From" and their DKIM signature, SPF, and NSLookup.

In this paper [43], they have proposed a semantic feature selection based on a statistical t-test. They extensively employed different methods of NLP, such as lexical analysis, part of speech tagging, stemming, stop-word removal, and wordnet. Two modules named "action detector" and "nonsensical detection phase" help to form the property set to see if the attacker is looking for victims' properties. In the next classifiers, part-of-speech tagging, sense and preventing ambiguity in the meaning of polysemous, and WordNet has been added to improve the accuracy. In comparison to similar phishing detector PhishNet_NLP [44] and ML-based methods like PILFER (accuracy is 92%), it achieved a higher accuracy 95.02% only by applying content features.

[33] performed a semantic analysis over the body text of the email. A sentence is considered to be malicious if it inquires sensitive information or commands a performance of an action that might expose personal information. If the topic of the sentence is in the topic blacklist, it is phishing. This framework has another module based on the Naive Bayes machine learning method to extract all malicious topics. The result of the whole framework regarding recognizing phishing email on Enron dataset is 95%.

2.1.2 Spear phishing Detection

Spear phishing is a targeted phishing that is attacking known people or people of authority in organizations to get access to sensitive information. Since the spear phishing impersonates trusted sources, it gets believable for recipients. The impersonation can be either through name spoofing, and address spoofing or through unseen attacker, and lateral attacker [22].

The problematic issue in spear phishing detection is a significant number of false positives that the goal is to lower this. Another challenge is the unbalanced dataset in which the number of spear phishing is so small among the millions record of emails. On the other hand, even with 99.9% of detection rate, there are still 327000 phishing emails can not be detected.

In [22], they have implemented a practical detection method for enterprises' security teams to overcome the mentioned issues. The first step of every spear phishing attack is to lure the victim. The next step is to exploit on behalf of the attacker, which is revealing the credentials. According to these two steps, they have proposed practical features in Figure 2.1. The key point of their detection method is that in

Data Source	**Fields/Information per Entry**
SMTP logs	Timestamp `From` (sender, as displayed to recipient) `RCPT TO` (all recipients; from the SMTP dialog)
NIDS logs	URL visited SMTP log id for the earliest email with this URL Earliest time this URL was visited in HTTP traffic # prior HTTP visits to this URL # prior HTTP visits to any URL with this hostname Clicked hostname (fully qualified domain of this URL) Earliest time any URL with this hostname was visited
LDAP logs	Employee's email address Time of current login Time of subsequent login, if any # total logins by this employee # employees who have logged in from current login's city # prior logins by this employee from current login's city

Figure 2.1: Features and Data sources for [22]

addition to employing the features of emails (SMTP logs), they are applying other features in NIDS and LDAP logs, which help them to detect more precisely.

The name of classification is DAS(Directed Anomaly Scoring), which is simple and can overcome different issues such as the problem of choosing rules manually like tuning thresholds in supervised learning, the problem with the base rate in unsupervised learning, and requiring hyperparameter tuning. Besides, it is not direction agnostic, and if one dimension is remarkably changed, it will not alert.

To limit false positive, each security team has a limited time with a B alert budget. For each event, it sets a suspiciousness score and then ranks them. Finally, it will output the B most suspicious events for the security team. They could detect 17 out of 19 spearphishing (89%), and in comparison to other classical methods, they could only discover 4/19, which is an outstanding result. They control the cost of FP as well as the time which is taking less than 15 minutes for an analyst to investigate alerts during each month.

2.1.3 BEC Detection

A business email compromise (BEC) as a man-in-the-middle email attack is an exploit in which the attackers get access to a corporate email account and spoofs the owner's identity to defraud the company or its employees, customers, or partners of money. In some cases, an attacker creates an account with an email address that is very similar to one of the corporate network. BEC Detection refers to the fraudulent email-sender detection in Business Email Compromise (BEC) or Email Account Compromise (EAC) or spear phishing in enterprises.

2.1.3.1 Email Structure

The email was invented in the 1960s, and the standard RFC 822 was written in 1982. At that time, they did not care about secure infrastructure and protocols for email communication. Email servers were explicitly configured as open relays. However, this was changed years later when spammers and criminals began exploiting open relays. Also, some necessary checks are not considered during email communications. Attackers can intercept the contents of emails or forge the header fields for spoofing. Email as we know it today consists of three major sections:

- The envelope
- The message header
- The message body

The box in red above highlights the email's envelope, which is not visible for neither the sender nor the recipients. The fields in the blue part are the header and body. Each email's envelope has "Mail From" which is invisible. Also, there is another "From" showing the sender's name and email to the recipients. In most cases, when the author composes an email in the email clients, the envelope information is the same as the information that comes into the header. However, when spoofing an

Figure 2.2: Email Structure published by Proof Point Co. [11]

email, a threat actor can put whatever they want into the following fields [11]:
Mail from, From, Date, To, Reply-To
There are four types of attack scenarios in BEC described in the following parts with their mitigation methods. Two mechanisms for spoofing are :

- Method 1-Email Address Spoofing: Alice's email address and her name are spoofed on an incoming email so that the sender appears to be: Alice alice@unb.ca.
- Method 2-Display Name Spoofing: Only Alice's name is spoofed, but not the email address: alice@gmail.com.

Envelope From Spoofing Attack One technique to spoof email addresses is through the invisible field in the email, which is Envelope From. SPF can handle this one by checking the DNS of the server from which email is coming.

Header From Spoofing Attack The other technique for running method 1 is through spoofing the Header From, but SPF is not enough to detect. However, IT staff, by using SPF and DMARC, or DKIM and DMARC, can recognize this kind of spoofing.

Display Name Spoofing Attack In this attack, which refers to method 2, the attacker finds the name of a person of authority and register an email with their name

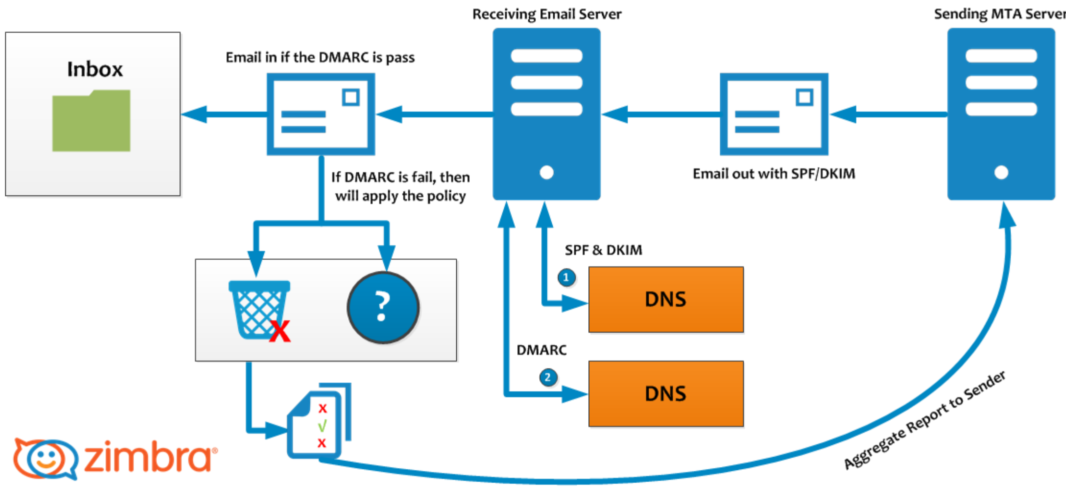

Figure 2.3: SPF, DKIM, and DMARC mitigation methods against spoofing

in one of the existing free web-based emails. Many email clients do not show the email address while recipients received emails, and only when the receivers click on the header part can see the full email address. Because display name spoofing cannot be handled by the real sender in many cases, the solution to detect it automatically is through applying machine learning methods like the current advanced threat filters on the receiver-side, detecting phishing and email frauds.

Compromised Account (mailbox) and Machine Attack In Business Email Compromise, in addition to the spoofing methods, email accounts, or even a machine which is including an email account, can be pwned by the attackers. If a high-rank person owns this email, the attacker uses the account for sending spear phishing with the content of wire transfer, sending back sensitive data, opening an attachment, and clicking on hyperlinks. In this type of attack, all information in the email is legitimate. Hence, the only way to detect a compromised account, and a malicious email is sent through it, is possible via monitoring the behavior of the user. If the behavior deviates from the normal, it gets suspicious. We had a study on several protection methods. Although most of them have detection on the receiver-side, not the sender-side, our platform is similar to a sender-side firewall recognizing malicious and benign email messages.

2.1.3.2 Profiling Approach for BEC Detection

The next two papers [16] and [42] have applied user behavior profiling to detect spear phishing and BEC attacks.

The goal of Email-Profiler [16] is to identify that received email originates from the author claimed in email's metadata. It is comparing incoming email with the behavior profile of each exciting senders' profiles.

Email Profiler [16] has two modes: Receiver-trained Profile and Generating profile for the sender in a trusted server.

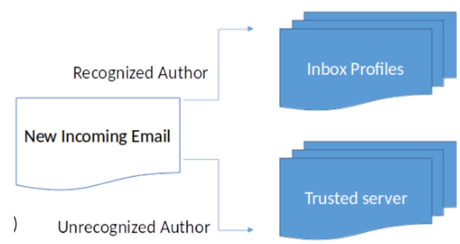

Figure 2.4: Features and Data sources for [16]

Totally 222 features have been extracted to train the classifier. Inbox Profiler aggre-

Feature	Type
Lexical features	Body Features
Syntactic features	Body Features
Structural features	Body Features
Email header features	Body Features
X-originating-hostname	Return-path Features
X-originating-IP	Return-path Features
X-spam flag	Return-path Features
Timestamp (morning,afternoon,night,midnight)	Return-path Features

Table 2.1: Features by [16]

gates all emails with the same sender, then it extracts the feature values and trains the SVM classifier (Positive label: corresponding author, Negative label: others) after that final weights for each feature are used as coefficients in decision functions. Sentbox Profiler makes a profile for cooperating users and stores feature vector and behavior profiles in a trusted server. They also have provided a very trustable protocol to transfer these profiles to the server.

For doing an evaluation, they used 20 contemporary volunteers' profiles, using SVM Classifier and 10-fold cross-validation for generating profiles of each recognized author that had 215 profiles created in total. For the identified user, the average accuracy was 93%, and for an unrecognized user the accuracy was between 80% and 100%.

This paper [42] is utilized user profiling to detect whether spear phishing has sent by a compromised sender/account or not before being sent. The Email Identifier makes profiles of people using their composition habits, writing habits, and interaction habits. The problem with this method is that it needs sender cooperation and only can be implemented in company settings for employees and is not practical for individuals. The difference between Email Identifier [42] and Email Profiler [16] is that the former detects on the sender-side, but the latter detects on the receiver-side. [42] has considered only senders' behavior while [16] has two kinds of profiles 1- Receiver-trained Profiles 2- Generating profiles which are the employees' behaviour. It makes a profile of all previously sent emails by a user. To this end, it compares all sender's emails with the sent emails by the others in the company to make a signature or profile for this specific user by these three classes Composition habits, Writing habits, and Interaction habits.

Maybe the attackers can mimic the user's behaviors such as the email address that is more frequently contacted, and at what time the user generally sends emails. However, because she cannot access all other emails sent by other users in the company, she will not know the specific and representative behaviors of this user.

A disadvantage of this method is its high rate of false-positive, which is annoying.

The other limitation is that it needs sender cooperation and is only applicable in company settings. The other job [20] for detecting spear phishing is learning profiles of a large set of senders and identifying spoofed emails based on the characteristic of sender profiles using Behavior Features, Composition Features, and Transport Features.

[21] is detecting spear phishing with email profiling. Four categories of email profiling features have been applied: origin features, text features, attachment features, and recipient features. The detection method was attributed to graph propagation-based semi-supervised learning.

[24] analyzes that there are three threats in BEC. First one is **impersonation**, the second one is **name spoofing**, and the last one is **account takeover**. So they proposed multi-phase filtering. It decides the incoming message is malicious, benign, or neutralized based on the score. For each one, it has a specific action. The first filter is checking the three risks mentioned above. The second phase is according to the user experience.

2.1.3.3 Industry-Side Methods

Also, in the industry side, Trend Micro Company [6] has focused on developing a framework to prevent BEC. They categorized BEC characteristics that make it hard to detect into three classes. 1)The sense of urgency, a request for action, or a financial implication. 2)The fake email is made to appear legitimate (impersonating CEO) 3) When a user's account or mailbox is compromised. They proposed three modules. The first one is an expert system to track attackers' behavior, the second one is for detecting spoofing, and the third one is using writing styles to detect compromised accounts.

2.2 Concluding Remarks

Email fraud is challenging and yet unresolved in email security. More complicated techniques come to play every day to bypass the controls by which attackers can reach their goals with a higher success rate. We started our study from the oldest email fraud, which is the phishing to the newest one, which is BEC.

Researchers and industry have proposed several methods applying machine learning algorithms, natural language processing, and user profiling to stop phishing and spear phishing attacks. The newest attack in email fraud, which is growing every day, is Business Email Compromise. In BEC, attackers impersonate a person of authority in an organization. They may exploit name/address spoofing or use a compromised machine or takeover account. Applying DMARC, SPF, and DKIM can detect spoofing attacks, however detecting BEC is still challenging when it happens through an account takeover attack or a compromised machine.

Chapter 3

Proposed Framework

3.1 Overview

We studied different types of email frauds in the previous chapter and some of their mitigation methods. Around 7,710 organizations are hit by a Business Email Compromise attack every month, according to the Symantec Internet Threat Security Report (ISTR) [12]. A business email compromise (BEC) as a man-in-the email attack is an exploit in which the attackers get access to a corporate email account and spoofs the owner's identity to defraud the company or its employees, customers, or partners of money. In some cases, an attacker creates an account with an email address that is very similar to one of the corporate network.

We had a study on some BEC protection methods. However, most of them determine the malicious emails on the receiver-side, not the sender-side. In phishing and spear phishing, which is a targeted attack, the sender may not be known at all. Thus the detection methods have been proposed on the receiver-side. However, in business email compromises, the attacker impersonates a person in a company who is a trusted source for the attack targets who are employees, customers, or partners of money. Hence, impersonation is a necessary step of this kind of attack. There exist five impersonation techniques like the following list which help attackers to run a BEC attack.

3.1.1 Attack Scenarios [9]

- *Envelope From Spoofing:* SPF can handle this one by checking the DNS of the server which email coming from
- *Header From Spoofing:* SPF+DMARC or DKIM + DMARC or three of them can prevent email spoofing
- *Display Name Spoofing:* It merely is spoofing the name with another domain, so it can be detectable easily by noticing the unknown domain name. In a more advanced way, threat filters can automate detection for the recipients. Since the attacker only spoofs the name in another domain, the impersonated person would not be involved with this attack. Hence, the detection process only can carry out on the receiver-side.
- *Compromised Account (mailbox)*
- *Compromised Machine*

The first three attacks on the list have mitigation ways of BEC on the receiver-side. However, the receiver-side should deploy them. Without deploying SPF, DKIM, and DMARC, the attacks are not detectable.

Some profiling approaches detect BEC on the receiver-side, as mentioned in the previous chapter although they have some issues. First, making the profile of a sender on the recipient side with a limited number of their emails is not practical, especially when the recipients are not company's employees, such as customers or partners of money. Second, it is impossible to force all clients to install such tools in their email clients, especially recipients out of the company's network. Third, when an account or machine has been compromised, it turns to a devastating weapon to target many people. So it ought to be stopped from the sender-side, and the real owner should be notified of this disaster.

As a result, there is a gap for detecting malicious emails coming from a legitimate email address that happened because of the compromised email address or compromised machine. We intend to fill this gap by proposing the following framework. Our platform is a multi-layer firewall recognizing suspicious, malicious, and benign emails. We have designed a two-stage identifier. The former is based on the user profiler, and the latter is employing group profiling.

3.2 Background

3.2.1 Architecture of Email System

Before describing the proposed framework, it is required to go over the electronic mail system, its modules, and how it works. There are two sorts of software functionality involved in the transfer of E-mail.

3.2.1.1 Agents

- **Mail Transfer Agents (MTAs):** MTAs are running programs on hosts named mail servers permanently. An MTA listens for an incoming e-mail from both MUAs and remote MTAs. It either saves it locally for retrieval by the MUA or identifies a remote MTA and transfers the e-mail to that. Several MTAs may be involved in the transfer of e-mail from originator to destination, the intermediate MTAs are known as mail relays.

- **Mail User Agents (MUAs):** An MUA is a program that runs on a user's machine to send or receive E-mail. It provides a user interface for composition, local storage of mail messages, and also has facilities for communicating with MTAs. There are several MUAs for desktop-like Outlook or Web-based MUAs.

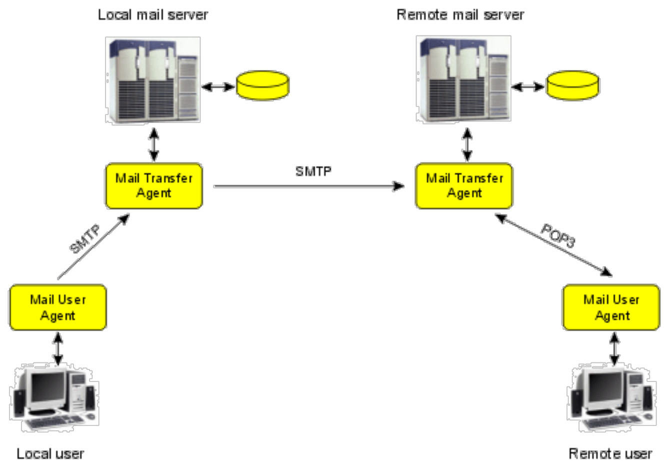

Figure 3.1: Email system by [1]

3.2.1.2 Mechanism

The transfer of an E-mail from a local user to a remote user will typically involve the following steps:

1. *Composing:* The local user starts composing their email in the MUA program which is configured with the Internet address of a host running an SMTP server MTA. Mail server administrators normally configure MTAs, so they will only accept requests from or for users in the same domain as the server.

2. *Sending E-mail from MUA:* After composing the email, MUA sends the email to MTA using the SMTP protocol. The MUA automatically includes the required headers in the email message based on information provided by the user either as part of the MUA configuration or entered interactively during mail composition.

3. *E-mail transmission:* The MTA analyses the headers of the incoming message to identify the destination mail server. The MTA then attempts to communicate with the destination mail server and transfers the message using SMTP.

4. *E-mail receipt:* The destination mail server examines the incoming message to identify the local part of the address from the headers. The server will store the incoming E-mail in the mail spool.

5. *MUA receives E-mail:* The remote user runs the MUA program, which is configured with the Internet address of a host running a POP3 server MTA. After establishing the identity to the POP3 server by username and password, the server searches into the mail spool and notifies the recipient with the number of incoming E-mail messages. The user can request to transfer or copy the E-mail messages to their machine.

3.2.2 User Profile Approach

User profiling is personalization, in which information retrieval is done to personalize a scenario making a user profile for the individual user. All relevant information and attributes of the users are extracted from the raw data to make user-profiles. Then through integrating the extracted information, a user profile will be formed for individuals. The goal of group profiling is learning the collective behavior of user

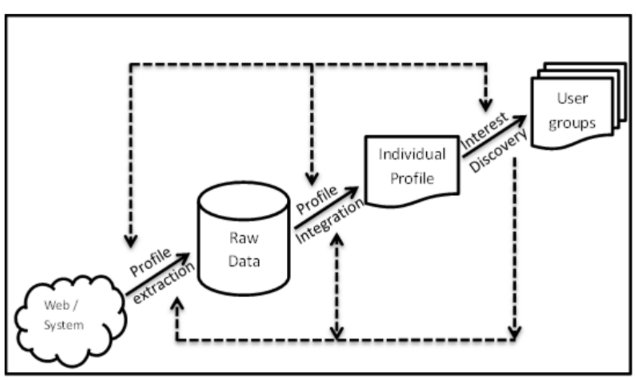

Figure 3.2: User Profiling [25]

groups who are belonging to the same community. An approach to performing user profiling at a group level is a two-stage framework, including community discovery problems and aggregation of members' profiles [19].

3.2.3 K-means Clustering with Elbow method

K-means is a partitioning-based clustering approach [27], which has a simple algorithm and divides the data into K clusters with their centroids. Described in Sects. 3.3.1 and 3.3.2, we applied the K-means algorithm to find the centroids and the within-cluster sum of the square (WSS) of each cluster to determine the user/group profiles. However, K-means clustering inherently cannot recognize the suitable number of clusters. Thus, using the ELBOW Method helps to find the K or the number of the clusters. K-means is an iterative clustering that has two steps [4]: **1-Cluster**

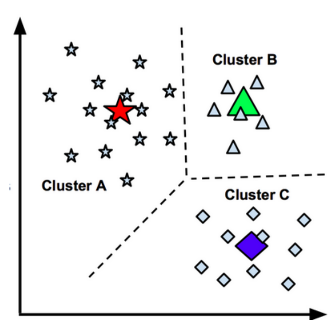

Figure 3.3: K-means Clustering

Assignment 2-Move Centroids.

It initializes K random centroids at first. Then assigns each point to the closest centroid, and finally moves the centroid to the average of points in each cluster. It repeats this process until no change happens.

To find the optimal K using the elbow method, first, a maximum number of K is selected. After computing the total within-cluster sum of the square (WSS) of each clustering, the below figure is formed. The point has the most distance from the assumed line, which is connecting the first point to the endpoint has the optimal value for K.

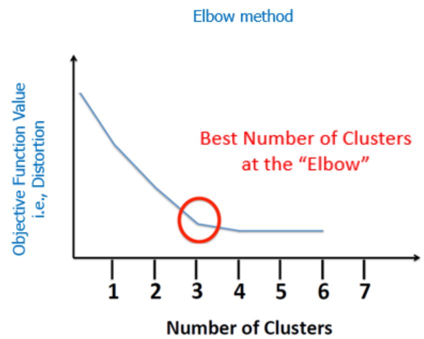

Figure 3.4: ELBOW Method

3.3 BEC Framework

Electronic mail works as described above and shown in Figure 3.1. Each email would be sent from MUA toward MTA. MTA is responsible for receiving the message and relaying it on the network toward the receiver's MTA. Here as shown in Figure 3.5, we have designed a two-level protection system. This protection system is such a plugin control all incoming emails from MUAs before entering to MTA. This system has two training modules, the first one is based on user profiling, and the other is based on group profiling. The result of user profiling and group profiling is two alerting systems that recognize malicious emails coming from a compromised machine or account.

3.3.1 Stage One Protection

3.3.1.1 User Profiler Module

We make user-profiles from the user's behaviors applying K-means clustering with the elbow method. It extracts the features that specify the user's behavior from their sending emails. All feature vectors are sent to the Profiler Module. It makes a user profile for all users in the company, each of which only contains the centroids

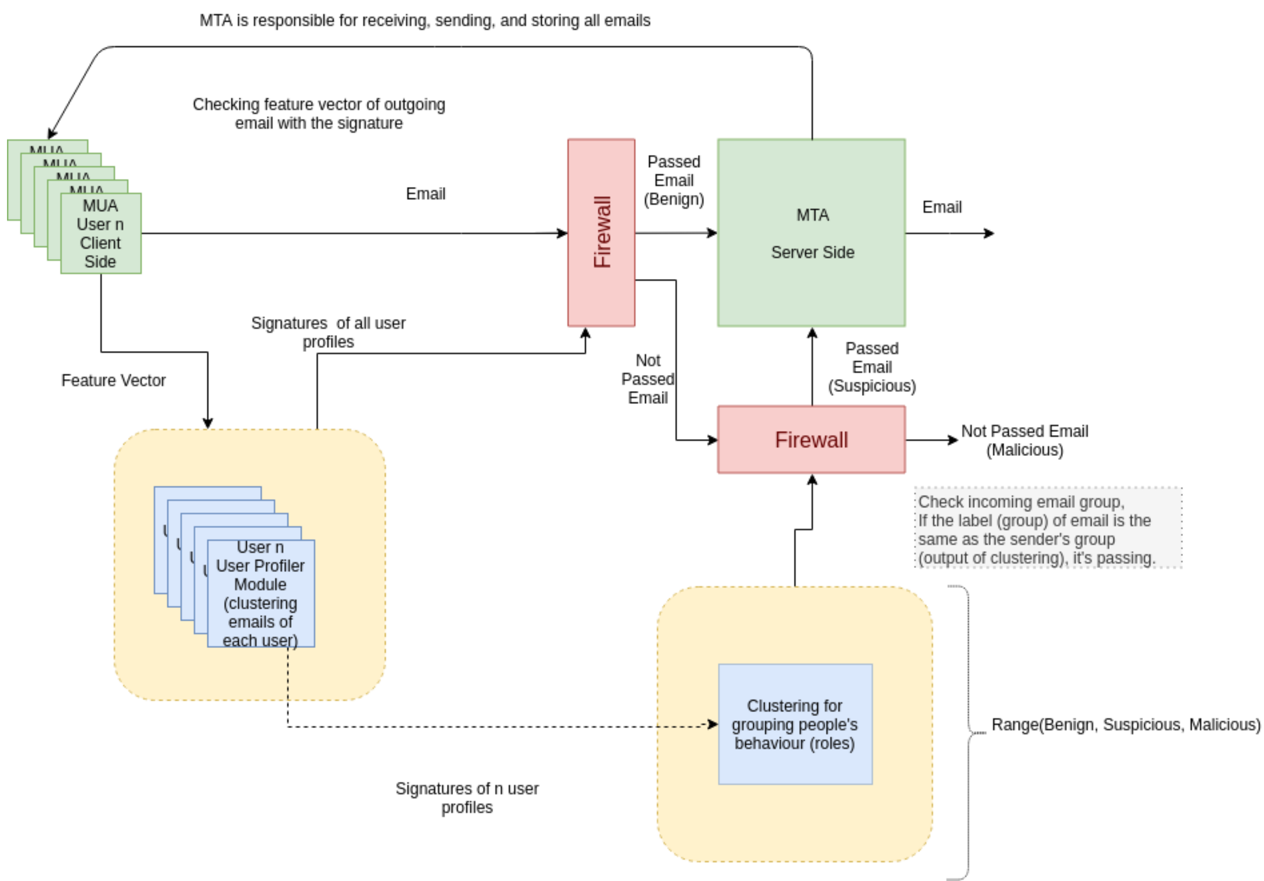

Figure 3.5: System Architecture

and radius. The radius of each cluster is the average distance of the points from the corresponding centroid. The user-profiles after a while need to be retrained again because the behavior of the real users is not stable and changes during the time.

3.3.1.2 User-profile Firewall (Alerting System 1)

After making a profile of the real email owner's behavior, the result of this profile would be a few signatures (centroids of the clusters) used on the first firewall to detect whether the outgoing email belongs to a real user or it is compromised. As a result, if an email account in MUA is compromised, this firewall should not pass it. Otherwise, the passed email would be entered into the second firewall as a suspicious email.

For detection to which cluster it belongs, the following formula has been proposed.

Algorithm 1 Alerting System 1

Require: $centroids$, $radiuses$, $EmailFeatureVector$
Ensure: $cluster$ OR 0

$closest = -1$
for $index$, $centroid$ in $centroids$ **do**
    $distance = 2 * euclidean(centroid, EmailFeatureVector) - radiuses[index]$
    if $closest == -1$ OR $distance < closest$ **then**
        $closest = distance$
        $distTocent = euclidean(centroid, EmailFeatureVector)$
        $cluster = index$
    end if
end for
if $distTocent > C * radiuses[cluster]$ **then**
    return 0
else
    return $cluster$
end if

The regular clustering is formed by computing the distance of the point to the centroids and select the closest ones. But, our formula not only considers mentioned

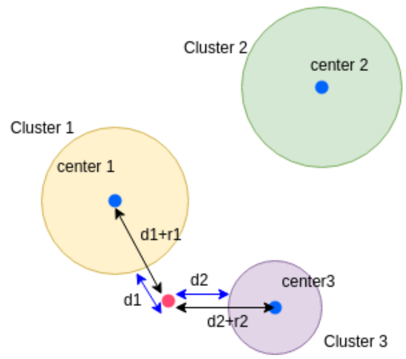

Figure 3.6: Clusters-Alerting System 1

distance but also calculates the distance of the point to the border of each cluster.

$$distance = (2 \times euclidean(centroid, point)) - radius \tag{3.1}$$

$$euclidean\_distance = \sqrt{(centroid_1 - point_1)^2 + ... + (centroid_n - point_n)^2} \tag{3.2}$$

If $d_1 + r_1$ is equal to $d_2 + r_2$, so we consider the other measurement which is the distance to the border. Here d_1 and d_2 determine to which cluster the red point belongs.

3.3.2 Stage Two Protection

3.3.2.1 Group Profiler Module

After Stage one, user profiling results in the profiles of users' behavior. Stage two clusters user-profiles of all people with similar behavior and puts them in the same group. To this ends, we again employed K-means clustering with elbow method . For instance, it makes a group of students, prof, bosses, employees, etc. In Group Profiler, we have two modules here.

The behavior of each user is, in fact, the centroids of user-profiles from the previous stage. Because each user can belong to more than one group, the group profiler also computes the degree of membership. The degree of membership shows that the user does belong to which group more.

3.3.2.2 Group-profile Firewall (Alerting System 2)

As shown in the system architecture in Figure 3.5, if the email could pass the first firewall successfully, it is benign. Otherwise, it is suspicious and would enter the second firewall (group-profile firewall). Each outgoing email's features would be checked with the firewall's conditions.

Here for each group of people based on their behavior, we have a set of signatures that are centroids of the group-profiles and people's degree of membership. Thus, for example, if an outgoing email has been written by a student it would be checked with the student group's signatures. Some people may belong to more than one group, so the degree of membership is considered as a coefficient variable. The distance in

Algorithm 2 Alerting System 2

Require: $membershipDegree$,$person$,$centroids$, $radiuses$, $EmailFeatureVector$
Ensure: $cluster$ OR 0

$closest = -1$
for $index$, $centroid$ in $centroids$ **do**
  $memDegree = membershipDegree[index]$
  $distance = (2 * euclidean(centroids[index, :], EmailFeatureVector) - radiuses[index])/memDegree$
  if $closest == -1$ OR $distance < closest$ **then**
    $closest = distance$
    $distTocent = euclidean(centroid, EmailFeatureVector)$
    $cluster = index$
  end if
end for
if $distTocent > C * radiuses[cluster]$ **then**
  return 0
else
  return $cluster$
end if

group profiling is the same as the above one, but the difference is considering the degree of membership. If a user has more similar behavior to one cluster more, its email more likely belongs to that group.

$$distance = \frac{(2 \times euclidean(centroid, point) - radius)}{(degree\_of\_membership)} \tag{3.3}$$

Algorithm 3 Degree of Membership for each user

Require: $user\_centroids$,$result\_of\_group\_profiling$
Ensure: $list\_of\_membership\_degrees$

$user\_cent\_num = length(user\_centroids)$
$list\_of\_membership\_degrees = 0:0, 1:0, 2:0, ..., n:0$ ▷ keys are clusters intialized with zero
for $user\_centroid$ in $user\_centroids$ **do**
    $cluster = result\_of\_group\_profiling(user\_centroid)$
    $list\_of\_membership\_degrees[cluster] += 1$
end for
$list\_of\_membership\_degrees = \frac{list\_of\_membership\_degrees}{user\_cent\_num}$
return $list\_of\_membership\_degrees$

3.3.2.3 A fusion of two alerting systems

A combination of group-profiles and user-profiles help to detect malicious emails and pass the legitimate ones. If an email passes the first firewall, it is benign. If it does not, the feature vector of the outgoing email is checked with the second firewall. If it passes the second firewall, it is suspicious. Otherwise, it is malicious.

3.4 Parser

We have implemented a parser to extract all the following features from emails. The parser first gets the raw email, including the header and body. Then it separates all parts and fields of emails such as body, fields in the header with their values. Finally, different modules are implemented to extract the features in each category, as described in Table 3.1.

3.5 Implemented Features

We analyzed all the different categories of features that can be extracted from emails and chose the ones which can detect outgoing emails from compromised accounts. Figure 3.7 listed all categories and their features gathered from the following works, which concentrated on detecting BEC and spear phishing [41][20][15][21][37][38][16][34][17]. The purpose of our system is to detect Bussiness Email Compromise once machines or accounts are compromised. Thus, we have to select the features which distinguish outgoing malicious emails with real ones.

In an email spoofing attack, the attacker makes a structure of an email based on the mail user agent (email client) that converts the provided addresses, text, and attachments into a suitable format for delivery. All composition features of an email's header are made with the email client and can be different in email spoofing. However, when an account or machine is compromised, the attacker has access to the email client or uses the same email client as the owner does. So features such as PGP and S/MIME fields would be the same, no matter the activity is malicious or benign.

Some features also refer to the behavior of the owner as a receiver, not the sender, which we ignore such as "indegree centrality".

There are features that refer to the owner's behavior during a time window and are not extracted from only a single email, such as social-based, interaction-based, and netword-based features in Table 3.1.

We also have added two powerful features into our feature set, which help to specify the user's behavior more precisely, word2vec, and sentiment analysis. Refer to Appendix 6.2 for the description of other features in Table 3.1.

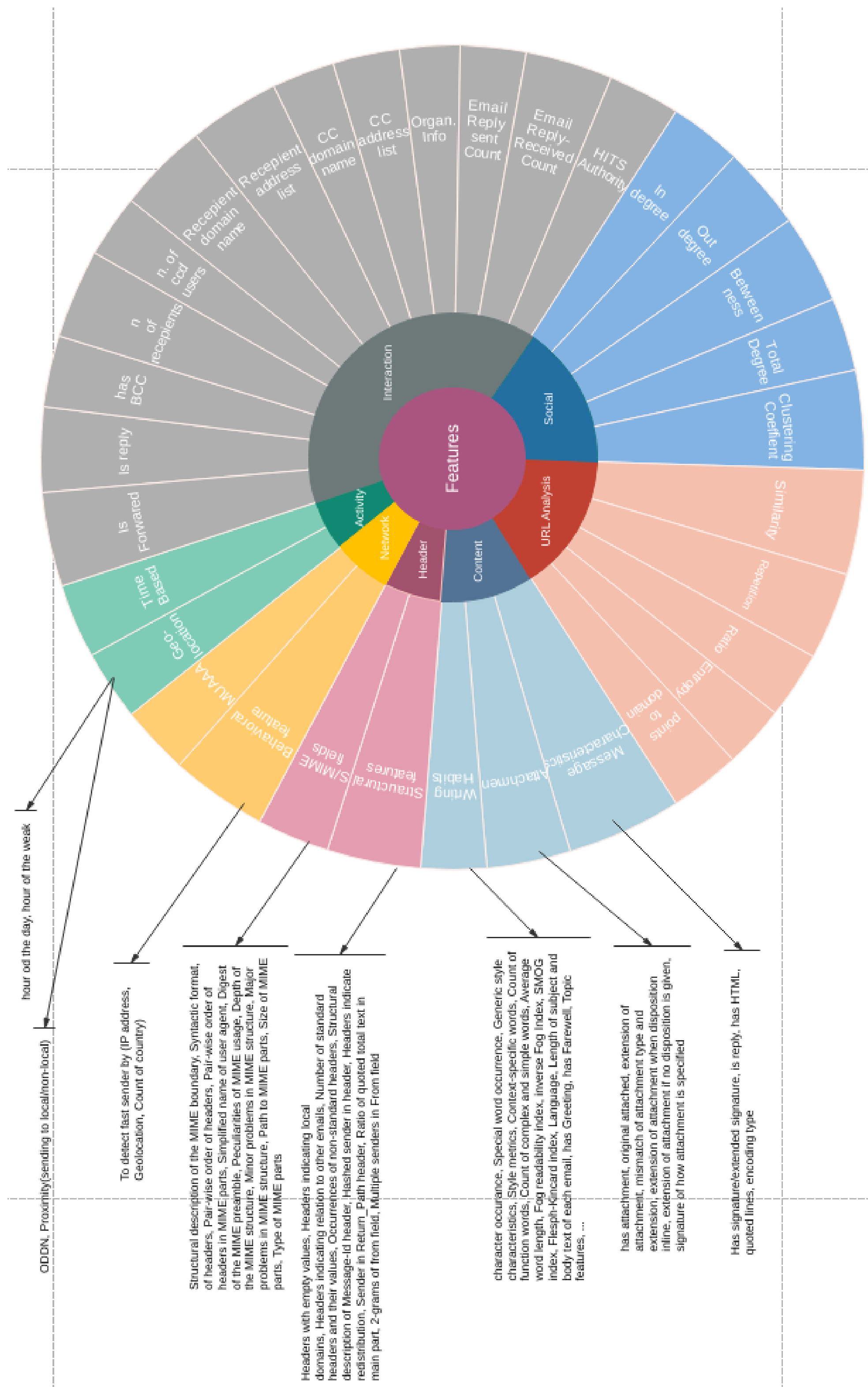

Figure 3.7: Features from [41][20][15][21][37][38][16][34][17]

Activity-Based Features	
Time	Business Time, Day, Month, Weekday
Geolocation	Longitude, Latitude, UTC offset
Interaction-Based Features	
Is Reply, Is Forwarded, Recipient Number, CC Number, BCC Number, Visited To Address, Visited CC Address	
Network-Based Features	
Sending Rate	
URL-based Features	
Has URL	
Social-Based Features	
Out Degree Centrality	
Content-Based Features	
Writing Style - Message Characteristics	Has html, Has Attachment, Attachment Type
Writing Style - Stylometry	Number of quoted sentences, Ratio of quoted total text, Number of caps sentence start, Number of caps sentence start, Number of small sentence start, Number of sentence end space, Number of sentence end without space, Number of Indented lines, Number of long lines, Number of short lines, Average of sentences per paragraph, Number of paragraphs, Average word length, Length of body text of each email, Number of words, Number of unique words, Legomenon, Dislegomenon
Writing Style - Generic	Emotions, Enumerated, Bulletized, Number of caps, Sentence start, Has comma in large digits, Oxford comma, Space Punctuation
Writing Style - Readability	GunningFOG index, Ari index, Flesph-Kincard Grade, Rix, Lix, Smog
Writing Style - Syntactic	Number of Adverb, Number of Adjective, Number of coordinating conjunctions, Tense Feature, Number of Complex words, Number of passive, Sentimental Analysis, Word embedding
Writing Style - Semantic	Sentimental Analysis, Word embedding
Writing Style - Subject	Number of Letters in subject, Number of words in subject, Ratio of letters in words, Number of Caps in Subject

Table 3.1: Our Selected Features

3.5.1 Word Embedding

The vector representations of words learned by word2vec [7] models have been shown to carry semantic meanings and are useful in various NLP tasks. Consider the following similar sentences: "Have a good day" and "Have a great day"[8]. Then a set of the vocabularies is made, V = Have, a, good, great, day. Then each word is represented with a one-hot vector [45]. The results of encoding is as follows:
Have = [1,0,0,0,0]; a=[0,1,0,0,0]; good=[0,0,1,0,0] ; great=[0,0,0,1,0]; day=[0,0,0,0,1]
The problem is that the meanings of "good" and "great" are as different as "day" and "have". However, word embedding is a type of mapping that allows words with similar meanings to have the same representation.

Using the words in a context, we can encode the target words with a hidden layer of Neural Network. Word2Vec has two types, Skip-gram and Continuous Bag of Words (CBOW).

For skip-gram, the input is the target word, while the outputs are the words surrounding the target words. A softmax activation function is applied at the end of the output layer, describing the probability of that word to appear in the context. However, in Continuous Bag of Words (CBOW), the idea is that by giving a context,

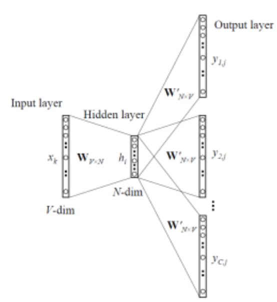

Figure 3.8: Skip-gram-[36]

it results in a word that is more probable to appear in that context.

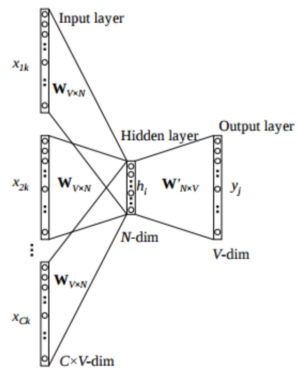

Figure 3.9: CBOW-[36]

3.5.2 Sentiment Analysis

Sentiment analysis or opinion mining is a field of Natural Language Processing and Data Science. At the most basic level, the task is to analyze a text and automatically score it for the opinions[13]. Sentiment Analysis is applied in movie reviews, products, elections, predictions, etc.

- "I had the most **wonderful** day."(positive)
- "I'm really **disappointed** with place I have rented."(negative)

Sentiment analysis is the detection of attitudes. Each attitude has a holder, target, and type. Types of attitude can be a common weighted polarity such as Positive, Negative, and Neutral. The most straightforward task in sentimental analysis can be positive or negative. The more complex one is ranking the attitude from 0 to 5[5].

3.5.3 Generating Malicious Data

There are several emails in each user's sent box in Enron dataset. However, they are all benign and legitimate, which have been sent by the real owner. To evaluate our system, having a negative sample is not enough since it can only assess true negative or false positive.

To determine whether the alerting system can detect fake or malicious emails being sent by legitimate senders, it is required to generate or simulate some malicious emails. To this end, malicious emails are generated regarding some attack scenarios.

3.5.3.1 Attack Scenarios

- Attacker sends phishing emails containing a malicious attachment, links, or drive-by download using a compromised machine or account to unknown people massively.

- Attacker sends phishing emails containing a malicious attachment, links, or drive-by download using a compromised computer or account to known people(exist in the contact list) such as employees, customers, and money partners.

- Attacker sends phishing emails by mimicking the writing style of the real sender via a compromised account to known people(exist in the contact list) such as employees, customers, and money partners.

- Attacker sends phishing emails by mimicking the writing style of the real sender via a compromised account to unknown people.

Malicious emails can affect the typical feature values that the users usually have. For instance, the attacker can send phishing emails through a real account to many people who do not exist in the user's recent contact list. So, in this case, interaction-based features probably get different values from the usual ones. What we intend to do is generating some rows of data with different values from the common ones. In anomaly detection, the number of anomalies should not be high. The power of alerting systems depends on their detection rate, especially when anomalies happen rarely.

There are two parts in emails that we can change the values and make the fake data: header features, and body features. For body features, we will use previous datasets for phishing emails and extract the body features from them.

For header features, we need to design an algorithm to generate valid fake data based on the real attack scenarios. Unfortunately, in the Enron dataset and the header

Geolocation Features	min	max	feature values	discrete?	Association
UTC- offset	-		-	-	longitude and latitude
Longitude	-180	+180	(-180)-(+180)	No	UTC-offset
Latitude	-90	+90	(-90)-(+90)	No	UTC-offset

Table 3.2: Geolocation Features

fields of emails, there is no geolocation information or any IP addresses, so there is no value for longitude, latitude, and UTC-offset. For generating fake value for

Interaction-based Features	min	max	feature values	discrete
To-num	1	infinite	1-infinite	yes
CC-num	0	infinite	0-infinite	yes
BCC-num	0	infinite	0-infinite	yes
is-reply	0	1	0 or 1	yes
is-forward	0	1	0-1	yes
visited-xcc-add	0	infinite	0-infinite	yes
visited-to-add	0	infinite	0-infinite	yes

Table 3.3: Interaction-based Features

bcc_num, to_num, cc_num which are discrete and infinite. We can use the following formula. C in the following formulas is a constant value as a coefficient.

$$new\_value = mean + C \times standard\_deviation \tag{3.4}$$

For generating fake value for visited_xcc_num and visited_to_num, we consider once phishers accessing an account might send emails to people whom the real user does not interact with recently or send emails to unknown accounts. As a result, the values can be decreased using the following formula.

$$new\_value = int(mean - C \times standard\_deviation), new\_value > 0 \tag{3.5}$$

To generate is-reply and is-foward, which are boolean features, we negate the value, which is in the majority among the rows.

To generate the fake value for sending rate, first, the place of inserting the fake rows would be selected randomly. To know how many emails should be appended, we apply the following formula.

Network-based Features	min	max	feature values	discrete
Sending rate	0	infinite	0-infinite	yes

Table 3.4: Netword-based Feature

$$number\_of\_emails = round(mean - C \times standard\_deviation) \tag{3.6}$$

We do not need to generate the values for these fields, because emails are sorted based on their dates. Only some places in the rows would be selected. Then the values of these fields are copied from below or above rows.

Time-based Features	min	max	feature values	discrete
Business Time	0	1	0-1	yes
Month	1	12	1-12	yes
week-day	1	7	1-7	yes

Table 3.5: Time-based Features

To generate Business Time, which is a boolean feature, we negate the value, which is in the majority among the rows.

Using the following formula, out-degree centrality gets a value. The condition is

Social-based Features	min	max	feature values	discrete
Outdegree Centrality	0	1	0-1	No

Table 3.6: Social-based Features

that the value should be between 0 and 1.

$$new\_value = mean - C * standarddeviation \tag{3.7}$$

3.6 Concluding Remarks

In this chapter, we proposed a behavioral-based framework for the detection of BEC on the sender-side that enhances the previous frameworks by utilizing user and group profiling. The proposed framework has a parser that allows the appending of different modules to extract features in different categories.

The central part of the framework is a two-stage behavioral detector, which is in charge of detecting malicious emails before they were sent from the compromised account or machine. In general, the proposed framework has the following features and advantages:

- Implementing a parser to get the primary features from the text of emails
- Implementing different modules for different categories of features
- The feature extractor can be extended with new modules for getting features in other categories
- Supporting timely analysis of streaming features such as network-based and social-based features
- Stopping BEC from the sender-side before sending
- Improving detection of BEC emails through a two-stage firewall made by behavioral profiling.

Chapter 4

Implementation

4.1 System Overview

This chapter explains the current implementation of the proposed framework for the detection of business email compromises. All modules have been developed in Python version 3. The current implementation of the framework is illustrated in the UML diagram shown in Figure 4.1, followed by the description of the critical functions of each module.

4.2 Parser

4.2.1 Email Parser

parser(input_email): It receives the email in its standard format. Using regular expression and pattern matching, it separates the contents of the body from the header. Then it separates different fields with their values in the header. Each email can have some non-standard features. However, the parser extracts those features existing in the list named *keys_to_extract*. In the body part, first, it removes the Forwarded part. Thus, it only considers the part written by the sender. Then it will

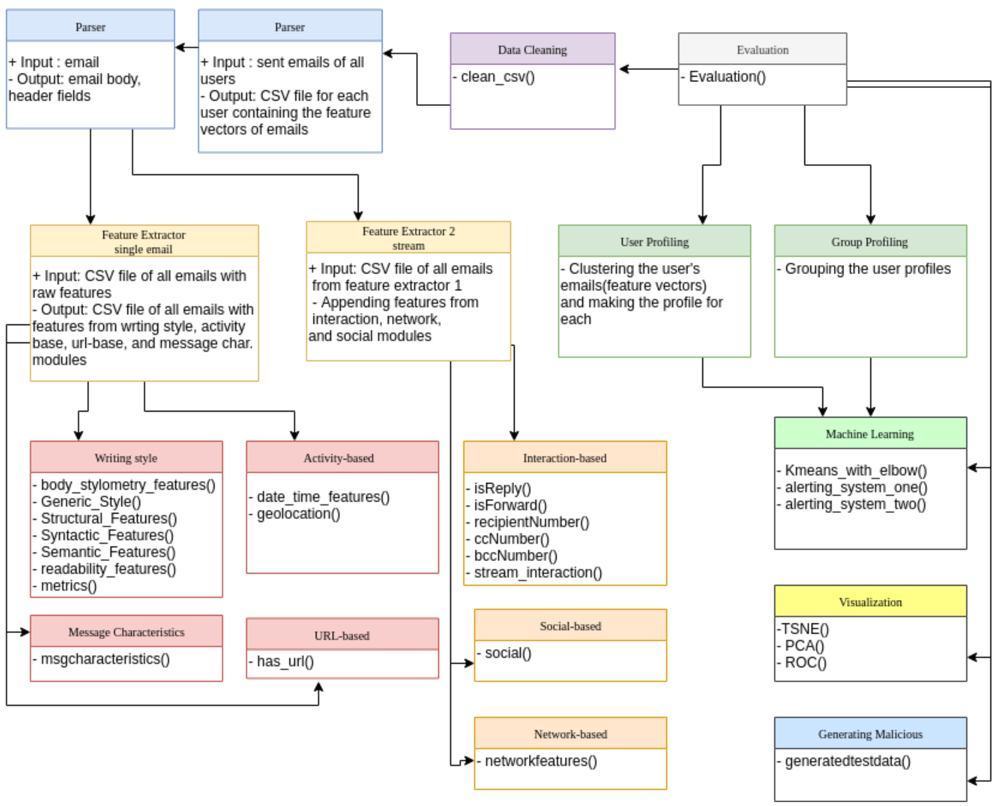

Figure 4.1: UML Diagram of BEC Detection System

send required fields extracted from the email to level-one feature extractors.

```
re.findall("[-]+.Original.\s.Message", line)
or re.findall("[-]+.Forwarded", line)
```

4.2.2 Enron Parser

*enron_ds():*The applied dataset is an Enron dataset containing 150 users. However, 148 of them have sent box folder. For each user, the emails in the sent_items, sent, sentmail folders are sent to the Email Parser Module. Email Parser Module will respond to this function call with a key-value list, which is a feature vector of email. Finally, all emails of one person are stored in a CSV file. For each CSV file, Feature Extractor Two is called to compute the features that require time-window.

4.3 Feature Extractors

4.3.1 Feature Extractor Level-One

There are some features extracted from a single email without requiring previous emails. The design of feature extractors is modular. Thus, it is possible to add new features or remove existing ones.

4.3.1.1 Writing Style

The input of the Writing Style Module is the body of the email. We applied different libraries in NLP, such as NLTK, SPACY, and StanfordCoreNLP to extract the corresponding features. Before extracting the features, we do the below preprocessing over the input text.

1. Removing URLs, E-mail addresses

2. Sentence Tokenizing

3. Word Tokenizing
4. Recognizing Proper Nouns
5. Removing Stop Words
6. Word Stemming

body_stylometry_features(body): It gets the body of the emails and returns a key, value array containing the name of features and their values. All stylometry features have been listed in Appendix 6.2, Table A.8 with their descriptions.

Generic_Style(body_w_lines): This function gets the body of an email and computes the number of emotions using Emojies' Unicode, bulletized and enumerated lists, comma_large_digits shows if large digits have a comma, oxford_comma, and space_punctuation.

Structural_Features(body, sentences): This function checks the body of an email to see whether it has a signature or not. To this end, we applied regular expression, which is an efficient approach to detect signatures in emails.

```
for c in string.punctuation:# p is a list of all punctuations
        p += c
re.findall("[\s]*[-]*\n*[A-Za-z\d" + p + "]+$", body)
```

Syntactic_Features(words, proper_nouns, body): It computes the number of adverbs, adjectives, coordination conjunctions in the body of an email by using part of speech library in NLTK.

Applying parts of speech tags of the verbs, it detects the tense of the email. It also recognizes that the language of the email is passive or active.

The number of complex words is another syntactic feature in which the word has more than three syllables, and should not be a proper noun.

readability_Features(body): ReadabilityTool is module computing the following readability measurements ari, fleschGrade, gunningFog, smog, coleman, lix, rix. Refer

Appendix 6.2 for definition of them.

metrics(words): It gets the words of emails and returns the value of hapax legomenon and dislegomenon.

semantic_Features(words, body) This module has two functions as follows.

sentimental_analysis(body): Stanford's CoreNLP[1][26] makes a text data analysis module, which is powerful and efficient. Stanford University has developed it in Java, although it has interfaces to connect to other languages such as Python. We call CoreNLP for each sentence and get a degree from 0 to 4. The average degree of all sentences in the body is the sentiment degree of the whole email.

word_embedding_features(words): As described word embedding in the previous chapter, each vector represents a word and some hidden features such as semantic and analogies. FastText is a library for the learning of word representations. It builds on modern Mac OS and Linux distributions. Since it uses C++11 features, it requires a compiler with C++11 support. These include :(gcc-4.6.3 or newer) or (clang-3.3 or newer)[14]

For training the neural network model and numerical vectors, we need a large corpus. Here we trained the system with English Wikipedia (enwiki9). The configuration of the training system is Skipgram, and the vector dimension is 300. When the vector size is larger, it can capture more hidden information. However, the training gets slower. The popular size of the vector is between 100-300.

When the model is ready, in function word_embedding_features(words), the model is called for each word of the email through bash codes written in Python. The final result is two vectors. One is the average of all vectors, and the other is the standard deviation of them.

[1]https://stanfordnlp.github.io/CoreNLP/

4.3.1.2 Message Characteristic

msgcharacteristics(content_types, mime_version, content_dispositions): This function gets the list of content_types, content_dispositions, and mime version of an email. The outputs has_html, has_attachment, and attachment_type. If text\html exists in the content_types list, has_html gets 1. If content_disposition has value, it means that the email has the attachment. To rank the types of the content an email has, we use the following ranking list based on the possibility of hazards.

```
{"application": 2, "audio": 1, "font": 1, "image": 1,
"message": 0, "model": 2, "multipart": 0, "text": 0, "video": 1}
```

4.3.1.3 Activity-Based Module

It includes two modules: Geolocation and Time-based Modules.

date_time_features(date): Based on the format of timestamp, it extracts the weekday, month, day, and time. It also checks if the email has been sent during the work or not and sets the *business_time* feature.

```
pattern = re.compile("[a-zA-Z][a-zA-Z][a-zA-Z],
[0-9]+ [a-zA-Z][a-zA-Z][a-zA-Z] [0-9]+ [0-9]+")
```

geolocation(ip): Plenty of online APIs and services give geolocation information using the IP address in the email. We applied ipgeolocation API [2], which is returning a JSON containing 'longitude', 'latitude', and 'UTC_offset'.

4.3.1.4 Interaction-Based Module

The interaction-based Module has two parts. The first one gets features that can be extracted from a single email.

isReply(subject): Using regular expression, it recognizes the email is a reply or not.

[2]https://api.ipgeolocation.io/

It returns is_reply, which is a binary variable.

isForward(body): Using regular expression, it recognizes the email is forwarded or not. It returns is_forward, which is a binary variable.

recipientNumber(to): By counting @ sign in to field, it returns the number of recipients, which is to_num.

ccNumber(cc): By counting commas in cc field, it returns the number of Carbon Copy recipients, which is cc_num.

bccNumber(bcc): By counting commas in bcc field, it returns the number of Blind Carbon Copy recipients, which is bcc_num.

4.3.1.5 URL-Based Module

has_url(body): The input of this function is the body of the email. Its functionality is to find the URL in the body using the regular expression. The output is a binary value showing that it has a URL or not.

```
re.findall("https?://(?:[-\w.]|(?:\%[\da-fA-F]{2}))+", body)
```

4.3.2 Feature Extractor Level-Two

4.3.2.1 Social-based Features

social(address): The input is the address of a CSV file in which all the feature vectors of the user's emails exist. The file is sorted based on [‘year’, ‘month’, ‘day’, ‘time’]. Here we used a sliding window to compute *outdegree_centrality*. For each record, the value of *outdegree_centrality* is the number of unique receivers over the number of people have been contacted during the time window.

4.3.2.2 Interaction-based Features

interaction(address): The input is the address of a CSV file in which the feature vectors of the user's emails exist. The file is sorted based on [‘year’, ‘month’, 'day', 'time']. Here we used a sliding window to compute *visited_to_address* and *visited_xcc_address*. In each window, the value of *visited_to_address* and *visited_xcc_address* is the number of receivers in cc and to who have been contacted during the window.

4.3.2.3 Network-based Features

networkfeatures(address): The input is the address of a CSV file in which the feature vectors of the user's emails exist. The file is sorted based on ['year', 'month', 'day', 'time']. Here we used a sliding window to compute *sending-rate*. In each formed window, the *sending-rate* of the email is the number of emails sent on one day.

4.4 Machine Learning Modules

4.4.1 Kmeans with Elbow Method

This function is the Python implementation of Kmeans clustering with the elbow method described in detail in the previous chapter. The first input of the function is X, which is the dataset containing the rows, and each row is a feature vector. The other parameter is MaxK, which is the maximum number of K in the elbow method. The outputs are the final Kmeans model and inertia. Inertia is a list that includes average distances of the points in each cluster to the corresponding centroid.

```
def kmeans_with_elbow(X, MaxK):
    wcss = []
    kmeans = []
    number_of_clusters = 1
    for i in range(1, MaxK):
            if len(X)>i:
                kmeans.append(KMeans(n_clusters = i, init = 'k-means++',
                            random_state = 42).fit(X))
                wcss.append(kmeans[i-1].inertia_)
                number_of_clusters = i
            else:
```

```
            number_of_clusters = i-1
            break
distances = []
p1 = Point(1,wcss[0])
p2 = Point(number_of_clusters,wcss[number_of_clusters-1])
l = Line(p1, p2)
dist = 0
maxpoint = 0
for i in range(0,number_of_clusters):
    p = Point(i+1,wcss[i])
    distances.append(l.distance(p).evalf())
    if l.distance(p).evalf() > dist:
        dist = l.distance(p).evalf()
        maxpoint = i
labels = kmeans[maxpoint].labels_
centers = kmeans[maxpoint].cluster_centers_
# Returning Inertia
inertia = {}
counts = {}
for i, label in np.ndenumerate(labels):
    if label in inertia.keys():
        inertia[label] += euclidean(X[i], centers[label])
        counts[label] += 1
    else:
        inertia[label] = euclidean(X[i], centers[label])
        counts[label] = 1
for key,value in inertia.items():
         inertia[key] = value/counts[key]
return kmeans[maxpoint], labels, inertia
```

4.4.2 Alerting System One

This function is the Python implementation of Alerting System One described in detail in the previous chapter. The inputs are centroids and inertia of a userprofile and test, which is the feature vector of an outgoing email that should be checked with the firewalls. The last parameter is C, which is constant in the algorithm and can be tuned to get higher accuracy. The output of the function is either -1 when the point does not belong to any cluster or the number of the cluster it belongs to.

```
def alerting_system_one(centroids,inertia,test, C):
        closest = -1
        cluster_ = -1
        test=[float(i) for i in test]
        for i, c in enumerate(centroids):
            dist = 2*euclidean(c, test) - inertia[i]
            if closest == -1 or dist < closest:
                closest = dist
                dist_to_cent = euclidean(c, test)
                cluster_ = i
        if dist_to_cent > C*inertia[cluster_]:
            return -1, (C*inertia[cluster_] - dist_to_cent)
        else:
            return int(cluster_)
```

4.4.3 Alerting System Two

This function is the Python implementation of Alerting System Two. The inputs are centroids and inertia of a group-profile, test is the feature vector of an outgoing email, membership degree is a 2-d array in which each row is a pair containing the cluster number and the user's membership degree for that cluster. The last parameter is C, which is constant in the algorithm and can be tuned to get higher accuracy. The output of the function is either -1 when the point does not belong to any cluster or the number of the cluster it belongs to.

```
def alerting_system_two(membership_degree, centroids, inertia, test, C):
```

```
closest = -1
cluster_final = -1
test=[float(i) for i in test]
for index, row in enumerate(membership_degree):
    cluster = row[1]
    mem_degree = row[3]
    dist = ((2*euclidean(centroids[cluster,:], test) -
    inertia[cluster])/mem_degree)
    if closest == -1 or dist < closest:
        closest = dist
        dist_to_cent = euclidean(centroids[cluster,:], test)
        cluster_final = cluster
if dist_to_cent > C*inertia[cluster_final]:
    return -1, (C*inertia[cluster_final]- dist_to_cent)
else:
    return int(cluster_final)
```

4.4.4 User Profiling

userprofile(source_add, dest_add): This function gets the address of the source, which includes the CSV files. Each CSV file contains the feature vectors of emails sent by a user. In userprofile function first, the data is preprocessed, such as standard scaling. Then they are fed into the clustering function. The output of each clustering is a user-profile, including the centroids and inertia. The user-profiles will be stored in the *dest_add*.

4.4.5 Group Profiling

groupprofile(source_add, dest_add): This function gets the address of the source, which includes a CSV file. The CSV file contains the centroids of the user-profiles. In group-profile function first, the data is preprocessed, such as standard scaling.

Then they are fed into the clustering function. The output of clustering is a group-profile, including the centroids and inertia. In group profiling, we also compute the degree of membership for each user. The degree of membership shows that the user belongs to which group more. Finally, the centroids and inertia of the group-profile and degree of memberships will be stored in the dest address.

4.5 Visualization

tsne(X, Y) , *pca(X, Y)*: t-SNE and PCA are two methods for dimensionality reduction, thus we can show the results of clustering in two dimensions. After dimensionality reduction, the results will be shown through scatter plot using Python.matplotlib. The inputs are feature vector and the label of each.

4.6 Conclusion

This chapter describes the implementation of our proposed model for detecting BEC Attacks. The system has been implemented by Python 3 language. It has three main parts, Parser, Feature Extractors, and Machine Learning Models. After parsing each email, the results will be sent to feature extractors. There are two levels of feature extraction. Some features can be extracted from a single email. After extracting the level one feature of all emails of a user, the level two features or streaming features will be extracted using the sliding window. The third part of implementation is user profiling and group profiling, to which the data will be sent. For the detection of BEC, A two-layer alerting system has been implemented.

Chapter 5

Experiments and Results

To demonstrate the capability of the proposed framework in accurately detecting BEC emails on the sender side, we compare our framework with the only similar framework proposed previously.

This chapter starts by describing the reference dataset used for the experiments, then explaining the evaluation method and system setting, which is to tune the parameters for the maximum performance of the framework. Eventually, our two-step clustering will be validated by comparing it with one-step clustering using membership degrees.

5.1 Dataset and Labeling

There is a free public dataset we have employed as an email dataset because of having roles. Enron[3] is an old dataset. However, it is still employed in this area for proving the ideas, and also a comparison with previous papers applied this dataset. The Enron data was originally collected at Enron Corporation headquarters in Houston in May 2002 and prepared by the CALO Project [1]. The corpus contains a total of about 0.5M messages. It includes data of 150 users organized into folders.

Its data was initially made public and published into the web by the Federal Energy

[1] http://www.ai.sri.com/project/CALO

Regulatory Commission [2] during its investigation. MIT later purchased the dataset and turned out to have several integrity problems that have been fixed. The dataset does not include attachments, and some messages have been deleted because of privacy issues. Invalid email addresses were converted a form such as user@enron.com. Enron dataset is the only publicly available mass collection of real emails.

Enron dataset fits with our needs in this proposed framework. It has 150 users working in a company. Thus we can make their user profiles, afterward, make a group profile and assign the people with similar behavior to the same groups. To evaluate whether this framework and corresponding alerting systems are working precisely, we ought to test it with both malicious and benign data. However, only benign data are available in this dataset.

To test the framework with malicious data, we have proposed an automatic approach which generates fake rows of malicious data among the preprocessed benign emails of the Enron dataset. The approach has been elaborated in Sect. 3.5.3 of Chapter 3.

To generate data related to header fields, we porposed some formula. However, for generating the fake data of email contents. One assumtion in attack scenarios mentioned in Sect. 3.5.3.1 of Chapter 3 was that the attacker tries to mimic the real owner's writing style. The other assumtion is that the attacker sends phishing emails using the compromised account or machine. Hence for the second assumption, we need a phishing corpus, so used CLAIR collection [35] of "Nigerian" fraud emails which is an e-mail corpus containing criminally deceptive information, usually with the goal of convincing the recipient to give the sender a large amount of money. It is the best known type of fraudulent e-mails according to the Nigerian Letter or 419Fraud [3].

[2]http://www.ferc.gov/

[3]https://www.fbi.gov/scams-and-safety/common-fraud-schemes/nigerian-letter-or-419-fraud

5.2 Evaluation Method

We evaluate the proposed framework, applying ten times 10-fold cross-validation. As shown in Figure 5.1, one fold is considered for the test data. Test Generator Module gets the benign test data, and generates fake malicious data as described in 3.5.3 of Chapter 3.

The system was trained with nine folds and finally was tested with Test Set, including benign and malicious data. This procedure is repeated ten times then the results can be averaged to produce a single estimation.

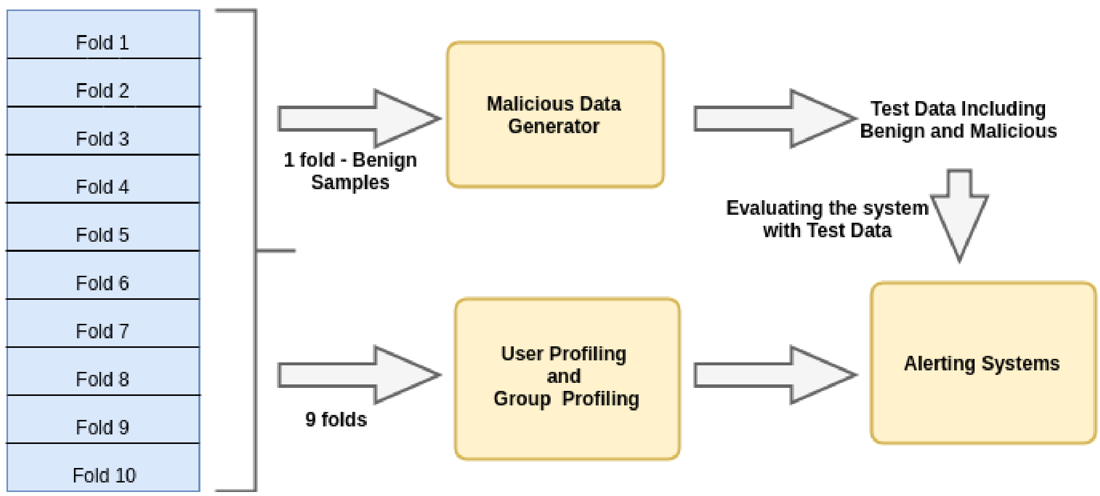

Figure 5.1: Evaluation Method

5.2.1 System Setting

- **Kmeans-intial:** In ScikitLearn, "kmeans++" smartly selects initial cluster centers to speed up convergence. The k-means++ initialization finds a solution that is O(log k) competitive to the optimal k-means solution, especially when the approximation found can be arbitrarily bad concerning the objective function. We just applied this function as a blackbox.

- **Kmeans-random state:** *random_state* is an integer random number. It determines random number generation for centroid initialization. K-means algorithm is not optimal. Thus, it does not find the best solution necessarily and can be stuck into a local minimum. Good practice to find a good minimum is to rerun the algorithm several times with several initializations and keep the best result.

- **K-means elbow method-Maximum number of K:** In the Elbow method as described in Chapter 3, k-means was run with 1-k times for K, and the best k was selected. In our proposed framework, we considered 60 for the maximum value of K, although it made the training time slower.

- **Alerting System One and Two-Constant Threshold:** As described the alerting system one in Chapter 3, there is a constant value as a threshold in the if statement $distTocent > C * radiuses[cluster]$. Its value affects the choice of whether a point belongs to the cluster or should be discarded. In alerting system one, the value of C is 1.5, and in alerting system two, it has been assigned 1. This value should be chosen in a way to keep the balance between true positive rate and true negative rate. If it is so big, malicious data are selected wrongly as legitimate, so true positive lessens. If it is so small, it will consider legitimate data as malicious and true negative rate comes down.

5.2.2 Visualization and Results

The Enron dataset has 150 users that 148 of them have sent an email which stored in the sent folder. 129 people out of 148 have sent more than 50 emails. In our evaluation, we consider these 129 people since the system would not be able to detect with reasonable accuracy when such a small number of emails is in the training set. Figure 5.2 shows the accuracy of the framework for all 129 users. The users are sorted based on the number of emails they have sent. Y-axis shows the average accuracy of ten times 10-fold cross-validation for each user. As shown in the figure, except for some points, the accuracy mostly is between 0.9 to 1. To make the results more

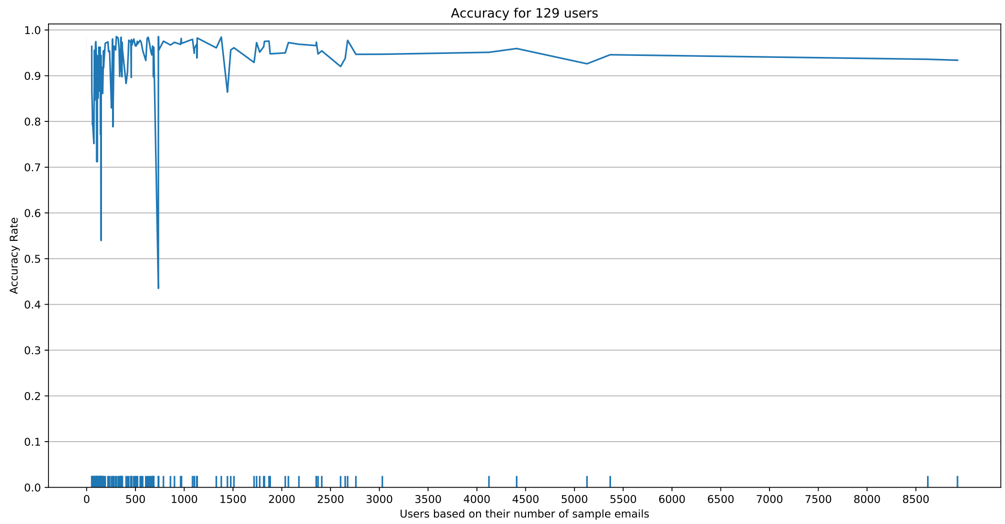

Figure 5.2: Accuracy Figure for all users - Users orderd based on the number of emails they sent

insightful, we made some batches of users based on their number of sent-emails. As appeared in Figure 5.3, we considered four measures, which demonstrate that the suggested framework is detecting precisely. All these four measures have been computed for all batches as well as the total value, which is the average of them.

Accuracy is one of the metrics that help to choose the best model. However, in some cases, such as anomaly detection, it cannot show the system incapabilities. Since the number of anomalies is by far less than the benign data, for instance, out of 1000 benign data 1 malicious can happen. If the system can detect legitimate data even with 100% rate, but cannot detect the malicious data happened once, the accuracy would be roughly 99% while the system is not able to detect the malicious one[39].

$$Accuracy = \frac{(TruePositive + TrueNegative)}{Positive + Negative} \tag{5.1}$$

Precision and *Recall* are two other metrics that indicate how accurate or precise our model concerns true positive. The cost of false positive can be high, especially in fraud detection. Thus *Precision* regarding the number of false-positive in the denominator of its formula, it considers to keep this cost less.
Recall metric indicates the cost of false negatives. For instance, in fraud detection, if a fraudulent activity gets discarded, it can have unacceptable consequences.

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive} = \frac{TruePositive}{TotalPredictedPositive} \tag{5.2}$$

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative} = \frac{TruePositive}{TotalActualPositive} \tag{5.3}$$

F1 Score might be a better measure to use if it is required to seek a balance between Precision and Recall once there is an uneven class distribution such as a large number of Actual Negatives.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{5.4}$$

According to Table 5.3 and Figure 5.4, our model in total has achieved a reasonable percentage regarding four metrics, which is above 91%. Not only did the accuracy reach a good percentage, but also the F1 score did, which means that the model does not have a high rate of a false negative or false positive. As it is evident in

Batches	Accuracy	Precesion	Recall	F1 Score	Frequency
50-500	0.906569	0.919278	0.920078	0.899361	67
500-1000	0.943827	0.938707	0.950126	0.930096	26
>1000	0.953749	0.925066	0.959616	0.937291	36
Total (> 500)	0.949588	0.930786	0.955636	0.934274	62
Total	0.927245	0.924809	0.937168	0.916141	129

Figure 5.3: Results

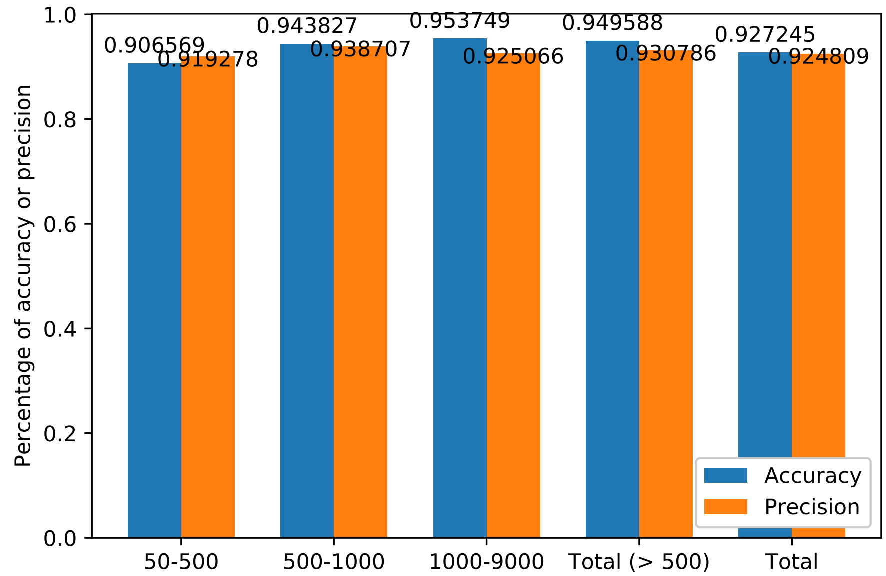

Figure 5.4: Average Accuracy for 129 users- each user belongs to a group based on the number of emails they sent

the above figures, having more sent emails for each user help to make better user profiles, which are more representative of the user's behavior. Thus the accuracy and F1 score would be higher when there are sufficient samples. However, there is another criterion affecting the detection rate, which is the consistency of users' behavior.

For instance, if a user uses their business email for both work and personal use, their behavior will be variant, and the diversity of feature values increases. Thus distinction of the anomalous data from legitimate ones gets more complicated.

As explained previously, different types of attack scenarios produce changes in the value of content and header features. In our experiments, as well as getting the results of total true positives, we captured the true positive rate for each attack type. In Figures 5.5 and 5.6, blue and green lines related to the header and both attack types have mostly detection rates more than 90% to 100%. However, the orange line, which is related to content attack types, comes down as the number of emails gets higher. Although if the number of samples gets higher, the accuracy should be higher too, here it is not as accurate as it should be.

As mentioned, the other criterion is behavior consistency. To check on this issue, we selected a user with reasonable accuracy in a content type(Allen-p with 1500 emails) and a user with an unfortunate result(Mann-k with 8926 emails). As appeared in Figure 5.7, the green line has a significant standard deviation from the mean and has a broad normal distribution curve, but the blue curve is a way tighter. It shows that the great variety of data in Mann-k results in a poor detection rate despite its vast number of training data.

Since the user's behavior, especially writing style changes continuously, the solution to this problem is to make the existing model to a streaming framework. To prove our idea, we split the data into four batches. We then trained and tested each one. In Figure 5.7, the yellow, blue, and purple curves are tighter than the original green curve, which means the data diversity is less in each batch. We predict that the true positive rate should be improved in comparison to the original data as shown in Figure 5.8.

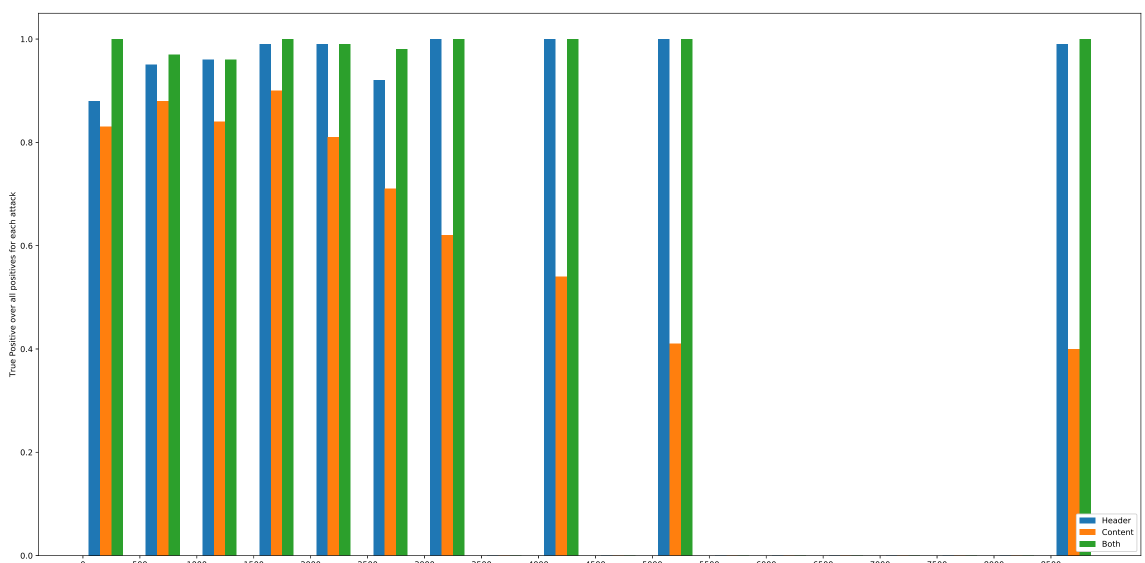

Figure 5.5: True positive over positives of different types of attacks

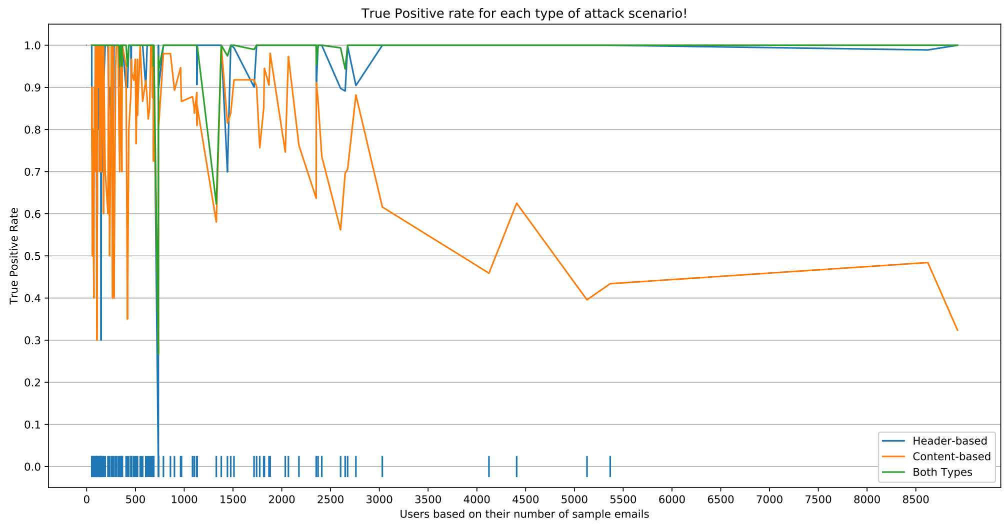

Figure 5.6: Rate of detection of different types of malicious emails - Users orderd based on the number of emails they sent

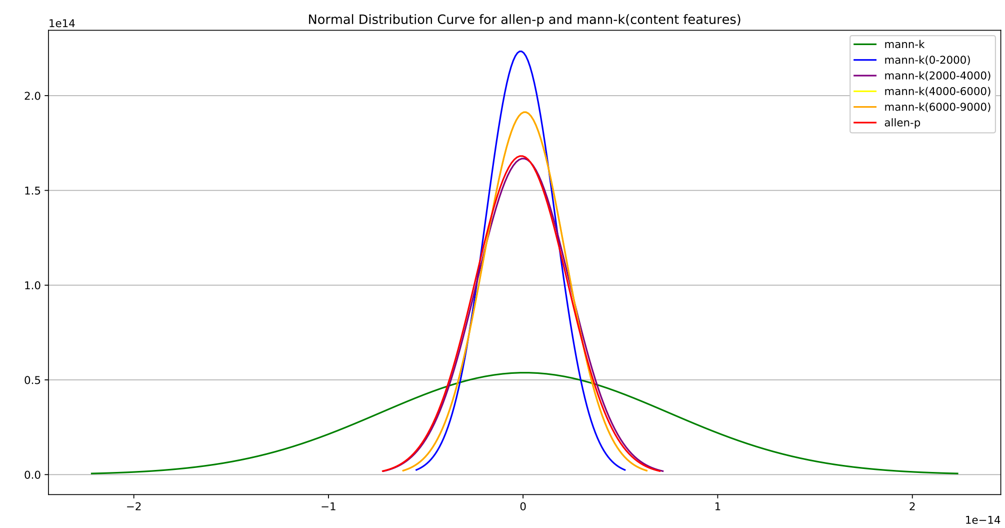

Figure 5.7: Normal Distribution Curve for allen-p and mann-k(content features)

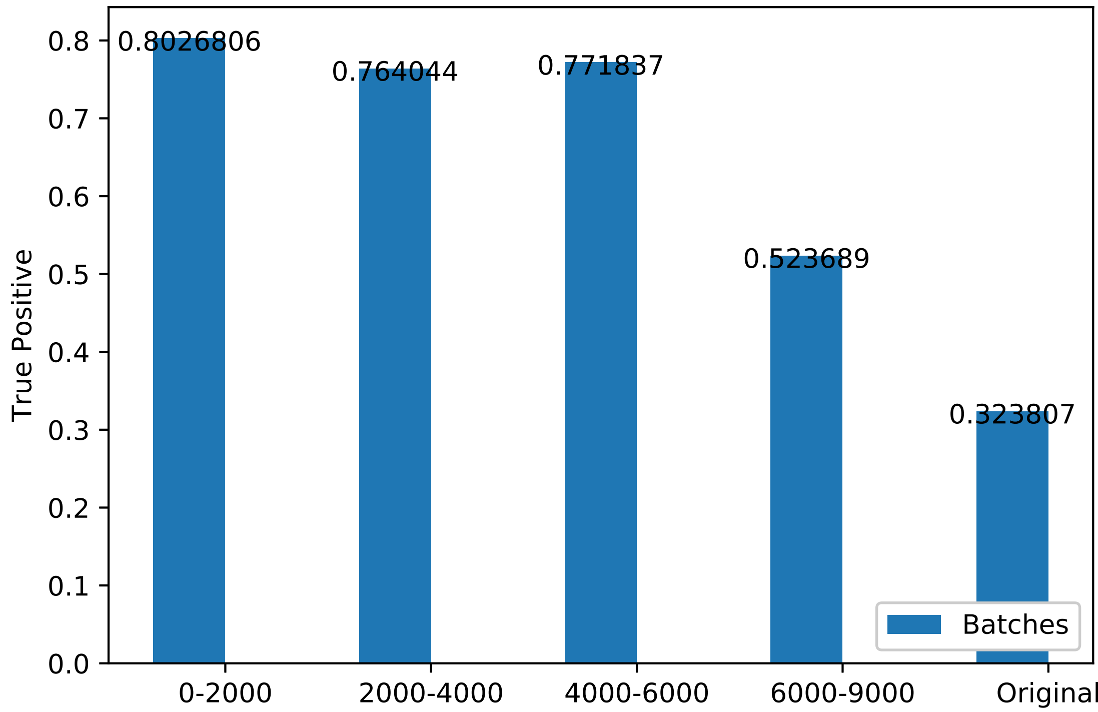

Figure 5.8: True Positives in batches of data in Mann-k

5.3 Comparative Evaluation

Our framework is supposed to detect BEC attacks in compromised accounts or machines on the sender side. IdentityMailer[42] is the first and only framework published in 2015 tried to stop malicious emails before they are sent. Hence this is the most related research to our idea, which has been done so far. Although we have got inspiration from this paper, our idea and framework try improving the previous solution in different aspects.

1. Our framework, compared to the IdentityMailer, is lighter. They applied classification methods to make the user profiles. Then they stored a model for each user. However, we applied k-means clustering for making user profiles. Each user profile is just some centroids and inertia of each cluster. Thus it would not have any overload for the mail servers and is very cost-effective.

2. Making each user profile is only based on user's behavior while IdentityMaler makes each user profile uses other people's emails too. Thus in comparison to IdentityMailer, our framework is more straightforward and practical.

3. They claimed that if they classify emails of each user with emails of other people in the company, they can get the specific behavior of this user. However, if the other users' behaviors are similar, this idea will not work correctly. Since people are working in the same company, they likely have the same behavior, especially people with the same positions. Unlike their idea, we applied other people's user profiles in group profiling to group people with similar behaviors.

4. We have tested our framework with generated malicious emails and reported the average results of all users in four metrics (accuracy, precision, recall, and F1 score). IdentityMailer has trained its system with Enron Dataset and tested its framework with three different datasets of malicious emails. However, they did not mention that how they handled the features of the header fields of these datasets because they are not consistent with the Enron dataset. Moreover, it only has reported the recall, true positive rate, and true negative rate of one user with 8000 emails. They have reported that as the number of emails that a user sends increases, the accuracy and recall gets better. IdentityMailer can detect malicious emails with a 90% rate when the user has sent more than 1000 emails. Also, when the number of emails is 8000, the recall gets 96% while our framework works more precisely (more than 90%) even when less number of sent emails(< 1000), as shown in Figure 5.3. Also, It should be noted that our results are according to the average of all tested users(in batches or total) while they have reported the results of only one user, so we have no clue how their system works for other users.

5. IdentityMailer also tested their framework with some simulated malicious emails

regarding some evasion methods such as sending to coworkers, Time, Top contact, Mimic 10 words, Mimic 20 words, and combinations of them, which are close to what we did in testing our system by defining some attack scenarios. They reached to 82% detection rate when the number of emails is 1000. However, our system detection rate is more than 90%, even for fewer emails. We should add that our simulation has more intensive attack scenarios. For instance, we copied all content features of a benign email and only changed some features in the header. This is a replay attack taking an email that was written by a user in the past, which can evade the detection in IdentityMailer easily as they confessed.

6. We have applied two important content features, which are sentiment analysis and word embedding, in our framework. By using word embedding, the semantic of words and the context are converted to a numerical feature. However, IdentityMailer only has used some features with limited and static values for functional words, particular words, and context words which can be changed during the time or can be evaded by the attacker easily.

5.4 Validation of two-step clustering

Although user profiling and group profiling are common approaches that have been proposed or applied vastly before, we show the validity of it in our idea using the degree of users' memberships. In Figure 5.11, the first Figure 5.9 is t-SNE of the two-step clustering, and the second one 5.10 is the result of one-step clustering. To do this, we compare each cluster of each method with the clusters of the other one. Indeed, through computing the intersection and union of the differences of the clusters in each method, we show how much two clusters in each approach have overlapped. For instance, the intersection of both clusters zero(0-0) in two approaches is 126, and

Cluster no. from method one(i) Cluster no. from method two(j)	Number of Users in each cluster(Method 1)	Number of Users in each cluster(Method 2)	Intersection between each cluster of method one and method two $C_i1 \cap C_j2$	Number of samples do not belong to the intersection $(C_i1 - C_j2) \cup (C_j2 - C_i1)$	\—**Sum**($membership\_degree\_C_i1$)-**Sum**($membership\_degree\_C_j2$)\—
0 - 0	132	127	126	7	31.9208
0 - 1	132	147	132	15	42.0459
0 - 2	132	97	84	61	18.8924
1 - 0	147	127	127	20	49.3021
1 - 1	147	147	147	0	46.1883
1 - 2	147	97	97	50	40.8051
2 - 0	134	127	117	27	29.4787
2 - 1	134	147	134	13	42.1736
2 - 2	134	97	96	38	15.3240

Table 5.1: Validation Using Membership

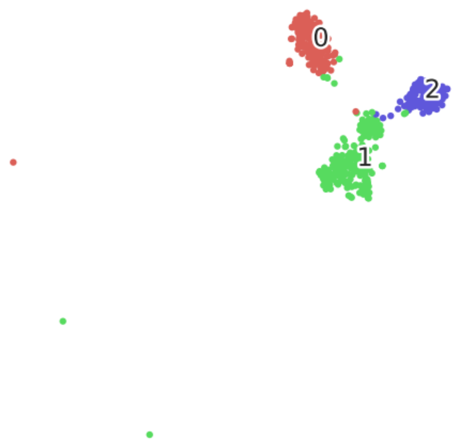

Figure 5.9: Two-step Clustering

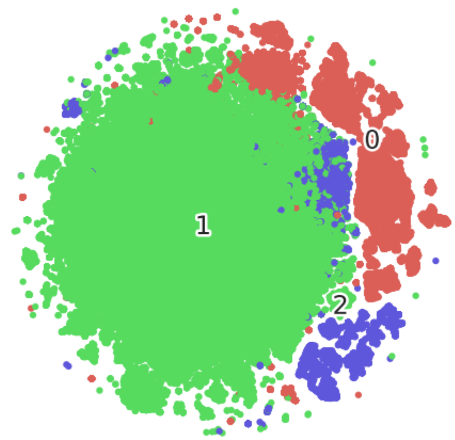

Figure 5.10: One-step Clustering

Figure 5.11: t-SNE Figure of clustering users with two ways

only seven users are not in the intersection part. Also, the summation of membership degrees of users in the intersection set is another criterion that should be similar for every two clusters. So when the difference between them is smaller, it means that two clusters are more similar. In 5.1, the gray rows show the best matches. First, we choose the clusters 1-1 that completely have overlapped. Then we select clusters 0-0, and finally, we choose 2-2 as the best match. This selection is based on the rows with minimum values of the summation of column-6 and column-7. So we can conclude that two-step clustering results in the same with one-step clustering in grouping the users.The advantage of using two-step clustering in group profiling is that it is lighter and has less overload for the server.

5.5 Concluding Remarks

In this chapter, we presented the results of our framework on a real email dataset named Enron. Our framework in the experiments to detect BEC attacks performs reasonably. The results range from 0.9 to 1.0 for people who have sent at least 1000 emails. The total average of Accuracy and F1 score for all users has reached more than 92% and 93%, respectively. We compared our framework with a similar research IdentityMailer. Our framework outperforms in comparison with IdentityMailer even when the number of emails is less than 1000. Also, in testing our framework with designed attack scenarios in compare to what IdentityMailer has done to evade their detection system, our attack scenarios not only are more intensive, but also the detection rate is higher.

We found out that the number of samples can result in a stronger detection, however more diverse data make the detection harder.

Finally, we have shown that clustering the users with two-step clustering, and one-step clustering brings about almost similar results.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

The newest attack in email fraud, which is growing every day, is Business Email Compromise. In BEC, attackers impersonate a person of authority in an organization. It has to be noted that this type of email fraud is inherently different from phishing attacks. They may exploit name/address spoofing or use a compromised machine or takeover account. Applying DMARC, SPF, and DKIM can detect spoofing attacks, however detecting BEC is still challenging when it happens through an account takeover attack or a compromised machine.

We have proposed a behavioral-based framework for the detection of BEC on the sender side that enhances the previous frameworks by utilizing user and group profiling. The main part of the framework is a two-stage behavioral detector, which is in charge of detecting malicious emails before they were sent from the compromised account or machine. Our framework in the experiments on a real email dataset named Enron to detect BEC attacks performs reasonably. The total average of Accuracy and F1 score for all users has reached more than 92% and 93%, respectively. We compared our framework with a similar research IdentityMailer. Our framework outperforms IdentityMailer when the number of emails is at least 1000. We finally found out that the number of samples can result in a stronger detection, however, more diverse data makes the detection harder.

6.2 Future Work

- Experiment on another real dataset: The Enron email dataset is a real dataset that we evaluated our framework with. However, it is an old dataset. In the future, if we access an available real dataset and test our framework, we can find the flaws and weaknesses better, especially a dataset containing happened BEC attacks.

- Experiment with real streams of malicious emails: Simulating malicious emails may help us to evaluate our system in the experiments. However, it is not sufficient since the real-world attacks always are harsher and more novel to evade the detections.

- Experiment on streaming type of our framework: As we found out in the experiments, the number of training data and their degree of diversity can affect the accuracy of the detection. Because the behavior of users changes continuously during the time, the diversity of training data increases and have an adverse effect on the detection rate. The best solution to this issue is to convert the existing framework into a streaming one. Thus the detection will focus on the newer behavior which a user has, and the data are less diverse.

- Applying Dynamic Feature Selection for Clustering High Dimensional Data Streams: In our clustering phase, we did not apply any feature selection. Some more essential features could have had some weights when computing similarity distances, although changing the users' behavior and the attack scenarios can change the weights too. So it is required to employ a dynamic feature selection. This approach of dynamic feature selection in clustering has recently in 2019 proposed by [18] that can contribute our idea in the specific field of BEC detection.

Bibliography

[1] *Electronic mail*, http://telescript.denayer.wenk.be/ hcr/cn/idoceo/tcp$_s$*mtp.html*, 2002.

[2] *Bec*, https://blog.barkly.com/what-is-a-business-email-compromise-bec-attack-and-how-can-i-stop-it, 2016.

[3] *Enron dataset*, https://github.com/brianray/data.world-scripts/blob/master/enron_email_script.ipynb, 2017.

[4] *kmeanselbow*, https://medium.com/@dilekamadushan/introduction-to-k-means-clustering-7c0ebc997e00, 2017.

[5] *sentimental analysis*, https://web.stanford.edu/class/cs124/lec/sentiment2017.pdf, 2017.

[6] *trend*, https://www.trendmicro.com/vinfo/ph/security/news/cybercrime-and-digital-threats/curbing-the-bec-problem-using-ai-and-machine-learning, 2018.

[7] *word embedding*, https://towardsdatascience.com/word-embedding-with-word2vec-and-fasttext-a209c1d3e12c, 2018.

[8] *word to vec*, https://towardsdatascience.com/introduction-to-word-embedding-and-word2vec-652d0c2060fa, 2018.

[9] *attacks*, https://www.reddit.com/r/sysadmin/comments/aph6ee/lets_talk_about_email_spoofing_a
2019.

[10] *Email security*, https://www.proofpoint.com/us/glossary/email-security, 2019.

[11] *email spoofing*, https://www.proofpoint.com/us/corporate-blog/post/how-does-email-spoofing-work-and-why-it-so-easy, 2019.

[12] *Istr*, https://www.symantec.com/security-center/threat-report, 2019.

[13] *sentimental analysis*, https://www.gavagai.io/text-analytics/sentiment-analysis-opinion-mining, 2019.

[14] *Word representations*, https://fasttext.cc/docs/en/unsupervised-tutorial.html, 2019.

[15] Abtin Zohrabi Aliabadi, Fatemeh Razzaghi, Seyed Pooria Madani Kochak, and Ali Akbar Ghorbani, *Classifying organizational roles using email social networks*, Canadian Conference on Artificial Intelligence, Springer, 2013, pp. 301–307.

[16] Sevtap Duman, Kubra Kalkan-Cakmakci, Manuel Egele, William Robertson, and Engin Kirda, *Emailprofiler: Spearphishing filtering with header and stylometric features of emails*, Computer Software and Applications Conference (COMPSAC), 2016 IEEE 40th Annual, vol. 1, IEEE, 2016, pp. 408–416.

[17] Manuel Egele, Gianluca Stringhini, Christopher Kruegel, and Giovanni Vigna, *Towards detecting compromised accounts on social networks*, IEEE Transactions on Dependable and Secure Computing (2017), no. 1, 1–1.

[18] Conor Fahy and Shengxiang Yang, *Dynamic feature selection for clustering high dimensional data streams*, IEEE Access **7** (2019), 127128–127140.

[19] Aleksandr Farseev, Mohammad Akbari, Ivan Samborskii, and Tat-Seng Chua, *360 user profiling: Past, future, and applications by aleksandr farseev, mohammad akbari, ivan samborskii and tat-seng chua with martin vesely as coordinator*, ACM SIGWEB Newsletter (2016), no. Summer, 4.

[20] Hugo Gascon, Steffen Ullrich, Benjamin Stritter, and Konrad Rieck, *Reading between the lines: Content-agnostic detection of spear-phishing emails*, International Symposium on Research in Attacks, Intrusions, and Defenses, Springer, 2018, pp. 69–91.

[21] YuFei Han and Yun Shen, *Accurate spear phishing campaign attribution and early detection*, Proceedings of the 31st Annual ACM Symposium on Applied Computing, ACM, 2016, pp. 2079–2086.

[22] Grant Ho, Aashish Sharma Mobin Javed, Vern Paxson, and David Wagner, *Detecting credential spearphishing attacks in enterprise settings*, Proceedings of the 26rd USENIX Security Symposium (USENIX Security'17), 2017, pp. 469–485.

[23] Thomas Jakobsen and Thomas Skardal, *Readability index*, (2007).

[24] Markus Jakobsson, *Short paper: Addressing sophisticated email attacks*, International Conference on Financial Cryptography and Data Security, Springer, 2017, pp. 310–317.

[25] Sumitkumar Kanoje, Sheetal Girase, and Debajyoti Mukhopadhyay, *User profiling trends, techniques and applications*, arXiv preprint arXiv:1503.07474 (2015).

[26] Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky, *The stanford corenlp natural language processing toolkit*, Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations, 2014, pp. 55–60.

[27] Hai-Long Nguyen, Yew-Kwong Woon, and Wee-Keong Ng, *A survey on data stream clustering and classification*, Knowledge and information systems **45** (2015), no. 3, 535–569.

[28] Minh Nguyen, Toan Nguyen, and Thien Huu Nguyen, *A deep learning model with hierarchical lstms and supervised attention for anti-phishing*, arXiv preprint arXiv:1805.01554 (2018).

[29] Weina Niu, Xiaosong Zhang, Guowu Yang, Zhiyuan Ma, and Zhongliu Zhuo, *Phishing emails detection using cs-svm*, 2017 IEEE International Symposium on Parallel and Distributed Processing with Applications and 2017 IEEE International Conference on Ubiquitous Computing and Communications (ISPA/IUCC), IEEE, 2017, pp. 1054–1059.

[30] Jeremiah Onaolapo, Enrico Mariconti, and Gianluca Stringhini, *Harvesting wild honey from webmail beehives*, (2016).

[31] ———, *What happens after you are pwnd: Understanding the use of leaked webmail credentials in the wild*, Proceedings of the 2016 Internet Measurement Conference, ACM, 2016, pp. 65–79.

[32] Prajakta Patil, Rashmi Rane, and Madhuri Bhalekar, *Detecting spam and phishing mails using svm and obfuscation url detection algorithm*, Inventive Systems and Control (ICISC), 2017 International Conference on, IEEE, 2017, pp. 1–4.

[33] Tianrui Peng, Ian Harris, and Yuki Sawa, *Detecting phishing attacks using natural language processing and machine learning*, Semantic Computing (ICSC), 2018 IEEE 12th International Conference on, IEEE, 2018, pp. 300–301.

[34] Tien Phan Jr, *A machine learning based language model to identify compromised users*, (2018).

[35] D. Radev, *Clair collection of fraud email, acl data and code repository, adcr2008t001*, http://aclweb.org/aclwiki, 2008.

[36] Xin Rong, *word2vec parameter learning explained*, arXiv preprint arXiv:1411.2738 (2014).

[37] Carlo Schäfer, *Detection of compromised email accounts used for spamming in correlation with mail user agent access activities extracted from metadata*, Computational

Intelligence for Security and Defense Applications (CISDA), 2015 IEEE Symposium on, IEEE, 2015, pp. 1–6.

[38] ———, *Detection of compromised email accounts used for spamming in correlation with origin-destination delivery notification extracted from metadata*, Digital Forensic and Security (ISDFS), 2017 5th International Symposium on, IEEE, 2017, pp. 1–6.

[39] Koo Ping Shung, *Accuracy, precision, recall or f1?*, https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9, 2018.

[40] Sami Smadi, Nauman Aslam, and Li Zhang, *Detection of online phishing email using dynamic evolving neural network based on reinforcement learning*, Decision Support Systems **107** (2018), 88–102.

[41] Gianluca Stringhini and Olivier Thonnard, *That ain't you: Blocking spearphishing emails before they are sent*.

[42] ———, *That ain't you: Blocking spearphishing through behavioral modelling*, International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment, Springer, 2015, pp. 78–97.

[43] Rakesh Verma and Nabil Hossain, *Semantic feature selection for text with application to phishing email detection*, International Conference on Information Security and Cryptology, Springer, 2013, pp. 455–468.

[44] Rakesh Verma, Narasimha Shashidhar, and Nabil Hossain, *Detecting phishing emails the natural language way*, European Symposium on Research in Computer Security, Springer, 2012, pp. 824–841.

[45] wikipedia, *One-hot vector*, https://en.wikipedia.org/wiki/One-hot, 2019.

[46] Weider D Yu, Shruti Nargundkar, and Nagapriya Tiruthani, *Phishcatch-a phishing detection tool*, Proceedings of the 2009 33rd Annual IEEE International Computer Software and Applications Conference-Volume 02, IEEE Computer Society, 2009, pp. 451–456.

Appendix A

Selected Features

A.1 Activity Based Features

Name	**Type**	**Machine or Account Compromise**	**Window-based**	**Description**
Business Time	Binary	Both	No	Is it sending during the work time or not?
Week day	1-7	Both	No	-
Month	1-12	Both	No	-

Table A.1: Activity-based features (Time)

Name Type	**Type**	**Machine or Account Compromise**	**Window-based**	**Description**
Longitude	Float	Account Compromise	No	Based on the IP address and using ipgeolocation.io API
Latitude	Float	Account Compromise	No	Based on the IP address and using ipgeolocation.io API
UTC offset	Float	Account Compromise	No	Based on the IP address and using ipgeolocation.io API

Table A.2: Activity-based features (Geolocation)

A.2 Interaction-based features

Name Type	Type	Machine or Account Compromise	Window-based	Description
Is Reply	Binary	Both	No	-
Is Forwarded	Binary	Both	No	-
Recipient Number	Unsigned Integer	Both	No	Number of emails in To field
CC Number	Unsigned Integer	Both	No	
BCC Number	Unsigned Integer	Both	No	
Visited To Address	Unsigned Integer	Both	Sliding window	Counts the number of recipients had interaction before in 30(window size) emails
Visited CC Address	Unsigned Integer	Both	Sliding window	Counts the number of CC recipients had interaction before in 30(window size) emails

Table A.3: Interaction-based features

A.3 Network-based features

Name	**Type**	**Machine or Account Compromise**	**Window-based**	**Description**
Sending Rate	Unsigned Integer	Both	Sliding Window	Sending rate is showing the number of emails is sending by the user during the time window

Table A.4: Network-based features

A.4 URL based features

Name	**Type**	**Machine or Account Compromise**	**Window-based**	**Description**
Has URL	Binary	Both	No	Extracted from the body

Table A.5: URL-based features

A.5 Social-based features

Name	Type	Machine or Account Compromise	Window-based	Description
Out degree centrality	Unsigned integer	Both	Sliding Window	Computes the number of different receivers in the window over the number of contacts because here we care about the sender's behavior! It shows this sender has to announce behavior or not

Table A.6: Social-based features

A.6 Content-based features

Name	Type	Machine or Account Compromise	Window-based	Description
Has html	Binary	Both	No	It is extracted from content-types, mime-version, content-dispositions fields
Has attachment	Binary	Both	No	-
Attachment type	0-3	Both	No	Based on the type of the attachment , they have a risk degree.

Table A.7: Message Characteristics

Name	**Type**	**Machine or Account Compromise**	**Window-based**	**Description**
Number of quoted sentences	Unsigned integer	Both	No	-
Ratio of quoted total text	Unsigned float	Both	No	-
Number of caps sentence start	Unsigned integer	Both	No	-
Number of caps sentence start	Unsigned integer	Both	No	-
Number of small sentence start	Unsigned integer	Both	No	-
Number of sentence end space	Unsigned integer	Both	No	-
Number of sentence end without space	Unsigned integer	Both	No	-
Number of Indented lines	Unsigned integer	Both	No	Starts with tab or space
Number of long lines	Unsigned integer	Both	No	length(line) $>$ 70
Number of short lines	Unsigned integer	Both	No	length(line) $<$ 40
Average of sentences per paragraph	Unsigned float	Both	No	-
Number of paragraphs	Unsigned integer	Both	No	-
Average word length	Unsigned float	Both	No	-
Length of body text of each email	Unsigned integer	Both	No	-
Number of words	Unsigned integer	Both	No	-
Number of unique words	Unsigned integer	Both	No	-
Legomenon	Unsigned float	Both	No	-
Dislegomenon	Unsigned float	Both	No	-

Table A.8: Writing Style Features (Stylometry)

Name	**Type**	**Machine or Account Compromise**	**Window-based**	**Description**
Emotions	Unsigned integer	Both	No	-
Enumerated	Unsigned float	Both	No	-
Bulletized	Unsigned integer	Both	No	-
Number of caps sentence start	Unsigned integer	Both	No	-
Has comma in large digits	Unsigned integer	Both	No	Checks id large digits has comma or not?
Oxford comma	Unsigned integer	Both	No	-
Space Punctuation	Unsigned integer	Both	No	-

Table A.9: Writing Style features (Generic)

Name	**Type**	**Machine or Account Compromise**	**Window-based**	**Description**
GunningFOG index	Float	Both	No	$0.4 \times (\frac{Words}{Sentence} + 100 \times \frac{ComplexWords}{Words})$
Automated Readability Index (ARI)	Float	Both	No	Automated Readability Index) 4.71(chars/words)+ 0.5(words/sentences)- 21.43
Flesph-Kincard Grade	Float	Both	No	$Flesph - KincardGrade = 0.39 \times \frac{TotalWords}{TotalSentences} + 11.8 \times \frac{TotalSyllables}{TotalWords} - 15.9$
Coleman	Float	Both	No	CLI = 0.0588L-0.296S-15.8, L is the average number of letters per 100 words. S is the average number of sentences per 100 words.
RIX	Float	Both	No	$\frac{LongWords}{Sentences}$ RIX =(long words = words where number of characters >6)
LIX	Float	Both	No	LIX is a readability measure to calculate the difficulty of reading a foreign text. LIX = A/B + (C x 100)/A, A = Number of words, B = Number of periods (defined by period, colon or capital first letter), C = Number of long words (More than 6 letters)
Smog	Float	Both	No	$\sqrt{TotalComplexWords \times (\frac{30}{TotalSentences})} + 3$

Table A.10: Writing Style features (Readability) [23]

Name	**Type**	**Machine or Account Compromise**	**Window-based**	**Description**
Number of Adverb	Unsigned integer	Both	No	-
Number of Adjective	Unsigned integer	Both	No	-
Number of coordinating conjunctions	Unsigned integer	Both	No	-
Tense Feature	Binary	Both	No	If it is present 1 otherwise 0
Number of Complex words	Unsigned Integer	Both	No	Single & Number of words with more than 3 syllabus
Numebr of passive	Unsigned Integer	Both	No	Number of Passive sentences
Sentimental Analysis	0-4	Both	No	-
Word embedding	Float	Both	No	-

Table A.11: Writing style features (Syntactic)

Name	**Type**	**Machine or Account Compromise**	**Window-based**	**Description**
Number of Letters in subject	Unsigned integer	Both	No	-
Number of words in subject	Unsigned integer	Both	No	-
Number of words in subject	Unsigned integer	Both	No	-
char divide word	Float	Both	No	Ratio of letters in words
Number of Caps in Subject	Unsigned Integer	Both	No	Single & Number of words with more than 3 syllabus

Table A.12: Writing style features (Subject)

Vita

Candidate's full name: Nasim Maleki
University attended:

Master of Computer Science
University of New Brunswick
2017-2019

Bachelor of Software Engineering
Shahid Beheshti University
2010-2014

Publications:

Conference Papers:
Generating Phishing Emails Using Graph Database, Nasim Maleki and Ali A. Ghorbani, The 15th International Conference on Information Security Practice and Experience(ISPEC2019)

Probabilistic Classification for Industrial Cyber Risk Using Deep Neural Network, Xichen Zhang, Arash Habibi Lashkari, Nasim Maleki, Joe Mudge, and Ali A. Ghorbani, The 2019 International Conference on Security and Management (SAM'19)

Poster Presentation:
Simulating Phishing Email Through Graph Database Models, Nasim Maleki, and Ali Ghorbani, Poster Session of UNB Research Expo 2019, Fredericton, Ca

Simulating Phishing Email Through Graph Database Models, Nasim Maleki, and Ali Ghorbani, Poster Session of CAN-CWIC 2018, Halifax, Ca