High-Confidence Computing 2 (2022) 100086

Contents lists available at ScienceDirect

High-Confidence Computing

homepage: www.elsevier.com/locate/hcc

SubStop: An analysis on subscription email bombing attack and machine learning based mitigation

Aurobinda Laha [a,*], Md Tahmid Yasar [b], Yu Cheng [b]

[a] Big River Steel LLC, 2027 E. State Hwy 198, Osceola, 72370, Arkansas, USA
[b] Illinois Institute of Technology, 3301 S. Dearborn St., Chicago, 60616, Illinois, USA

ARTICLE INFO

Keywords:
Email bombing
Subscription
Reverse engineering
Support vector machine

ABSTRACT

Email Bombing, a kind of denial-of-service (DoS) attack is crippling internet users and is on the rise recently. A particularly notorious type is the Subscription Bombing attack, where a victim user's inbox is bombarded with a stream of subscription emails at a particular period. This kind of attack helps the perpetrator to hide their real motive in lieu of a barrage of legitimate-looking emails. The main challenge for detecting subscription bombing attacks is that most of the attacking email appears to be legitimate and benign and thus can bypass existing anti-spam filters. In order to shed some light on the direction of detecting the bombing attacks, in this paper we first conduct some reverse engineering study on the Gmail anti-spam mechanism (as the information is not publicly available) and in-depth feature analysis of real-life bombing attack emails. Leveraging the insights from our reverse engineering study and data analysis, we propose a novel layered detection architecture, termed as SubStop, to detect and mitigate subscription bombs. SubStop exploits the statistics of incoming volume, source domain distribution, the correlation among different features, and implements machine learning to achieve effective detection. In specific, we utilize the weighted support vector machine (WSVM) and properly tune the class weights to achieve high accuracy in detecting bombing attacks. Despite the scarcity of public email data sets, we conduct extensive experiments on a real-life subscription bomb attack and real-time attacks using our bombing simulation script (which is facilitated by our reverse engineering findings), on test email accounts. Detailed experimental results show that our proposed architecture is very robust and highly accurate in detecting and mitigating a subscription bombing attack.

1. Introduction

Email has been one of the greatest boon for mankind along with the advent of Internet. Since it was first sent by Raymond Tomlinson in 1971 [1], way before the mobile age boom, emails have made a permanent place in nearly every modern household and all of the research and industrial sectors. However, email went from being a sophisticated means of communication to a supremely loved and popular messenger and then to a nightmare of spams in just over half a century. The journey has been startling to say the least. Now email users dread from the over usage of emails to completely deriding it, courtesy to its ability of inviting spams.

Spams [2] are junks or trashes in the cyber community. Unsolicited messages sent by bad or unethical people or programs via email contribute to what is one of the digital modern day's biggest threat - Spams. Spams can be of various types, even commercial, in nature. One of the malicious kinds are the phishing emails - which are means to extract sensitive personal or professional information by fraudulent links or emails while posing as trusted sources or entities. Phishing emails are classic case of a social engineering attack where a user is duped utilizing techniques to exploit its digital flaws. Every year the number of these kind of attacks and costs bore by the affected parties are increasing at an alarming rate [3,4].

A serious form of spam emailing is the email bombing attack [5]. It is the phenomenon that a specific user email account receives a huge number of emails, most of which are junk or phishing emails but some of them are legitimate regular emails. These mail bombs cause inconvenience not only to the affected users by overflowing their inboxes but jam the mail servers as well, making it one of the most simplistic yet dreaded forms of a denial of service attack. The most vicious form of an email bomb attack is the subscription bombing attack. In this scenario, a bad guy or attacker signs up a victim email address to mass subscription services. The attacker uses some mechanisms to sign up an email address to several internet forums and newsletters. Each of these websites then usually send confirmation emails to the specified address and in the process floods the inbox. Although there are other forms of email bombing attacks namely zip bombing, mass mailing to take down servers etc. those are the not the focus of this work. A mass mailing at-

* Corresponding author.

https://doi.org/10.1016/j.hcc.2022.100086
Received 1 August 2022; Received in revised form 12 October 2022; Accepted 12 October 2022

tack which is able to take down email servers are handled by the email providers by employing tarpitting or other sophisticated methods they can afford. Zip bombing attacks are a bit easier to detect as you can filter by attachments. The primary focus of this research is to provide a simplistic yet robust and effective solution to a normal user from subscription bombing attacks.

A subscription bombing attack is often used as a distraction to bury important emails like legitimate transaction emails in heaps of subscription confirmation emails. Thus, the victim might lose out on some critical information and emails, if they try to perform mass deletion to get rid of the bombing attack. For example, if an attacker got hold of a victim's paypal account and committed a fraudulent transaction, the victim will usually get an email from paypal confirming the transaction details. The victim in this case can easily raise an alert with paypal and decline the transaction. However, if the attacker masks this transaction with a subscription bombing attack at the same time, the victim's email address will be flooded with a large number (normally hundreds of or even thousands) of unsolicited emails and the fraudulent transaction email will get buried in the heap of all others. This might go on for several days. There have been no existing mechanism or technique that can effectively detect or deal with subscription bombing attacks, to the best of our knowledge. Hence, normal people suffer in their daily lives to cope up with the mess created by this attack.

The main challenge for detecting subscription bombing attacks is that most of them appear to be legitimate and benign at first glance and thus bypasses existing spam filters. These attacks do not always make bold or declarative statements of their malicious intent in the flood of messages. Marshall McLuhan first coined the phrase, the medium is the message [6]. The most recent study [7], on barriers to stopping unsolicited emails provides very important insights regarding the frustration endured by common users. The paper particularly focused on the user experience and challenges faced in an experiment to unsubscribe from unwanted emails. The study in [8] presents a similar linked-list email bombing attack.

Currently, almost every email provider utilizes machine learning technology and artificial intelligence to detect all kind of spams [9–12]. However, subscription bombs still get through the existing anti-spam mechanisms, primarily because the bombing attack emails differ significantly than what traditional spam filtering techniques consider as spam. Almost all the subscription emails are benign in nature and are actually legitimate confirmation emails appearing due to the unfair usage of the user's email address. Some detailed analysis on manually dealing with bombing attacks has been done by [13,14]. The studies clearly acknowledge the inability of existing frameworks to block bombing attacks as it can block legitimate emails in the process as well. This served as a strong motivation for us to work on an under researched problem.

In order to shed some light on the direction of detecting the bombing attacks, in this paper we first conduct some reverse engineering study on the Gmail anti-spam mechanism (as the information is not publicly available) and in-depth feature analysis of real-case bombing attack emails[1] Such efforts indeed help us move the steps ahead to detect the bombing attacks. Our reverse engineering studies (to be presented in Section 3.1) tell that Gmail almost always allows an email from new email addresses to a inbox, unless it fits the bill of an outright spam. Our findings also show that Gmail tends to allow emails containing the user's name from any domain. The above insights from our reverse engineering study allow us to create a simulation script (with Python language and by the Simple Mail Transfer Protocol (SMTP)) to launch a subscription bombing attack to a target email address[2] Such an attack not only enhance our claim that existing spam filtering techniques and plug-ins do not work effectively in case of subscription bombs, but also allow use to emulate enough amount of attacking data to test the detection techniques to be developed in this paper. Note that there are only a couple of publicly available email data set [15] which pose as a serious challenge for researchers in limiting their contributions to this area. Existing third-party spam filters also do not share their techniques in public. Our analysis of a real-life bombing attack shows that subscription bomb emails are correlated and have similar features, while, regular and spam emails come in plenty of variety. Moreover, clusters of emails gets dumped in inbox during a bomb attack varies significantly compared to regular state. State of the art machine learning based spam filters use available spam email database for training. These filters are thus not suitable for subscription bomb attack. In this paper, we will leverage appropriate machine learning techniques facilitated with the features from our data analysis for effective detection of bombing attacks.

In this paper, we are to provide a systematic solution for the Internet community to shield themselves against the annoying subscription bombing attacks. Particularly, we propose a novel layered architecture - 'SubStop' and leverage the Support Vector Machine (SVM) learning algorithm (with some enhancement design). In specific, our proposed architecture first involves a 'Time Filter', which monitors the volume of incoming emails in a moving time window. If the monitored volume is much higher than a long-term average value, it indicates that the inbox may get bombarded. If there's a red flag, the emails then get passed on to an 'Address Locator' filter. We define 'address locator' as the second part of an email address after the @. Here our system creates two streams: emails with new address locators and known address locators. This is important to minimize the regular emails being flagged falsely.

We utilized the weighted support vector machine (WSVM) and adjusted the class weights to maximize the performance of the classifier during a subscription bomb attack. After parsing the emails from the 'Address Locator' filter, the emails gets filtered through proposed WSVM. This enhances the accuracy of the system.

We extensively analyzed a compromised inbox from subscription bomb attack and used the data to train proposed WSVM for classification. The test results substantiate our design model compared to traditional spam filters based on Naive Bayesian [16] and Support Vector Machine [17]. We also tested our model with recently published deep learning (DL) based spam classifiers [18,19]. The SubStop architecture was deployed to the test email address. We wrote a python script to launch bomb attack on any email address. We also utilized the website [20] to instigate a subscription bomb attack. The performance of the SubStop architecture was evaluated exhaustively and compared with other well-know machine learning based spam filters. Our system achieves 99.92%, 100% and 100% precision to classify subscription bomb emails on three cases we launched a bomb attack. SubStop also performs exceptionally well in correctly identifying legitimate emails. This ensures that victimized users are not losing out on their desirable emails.

We offer the choice between SubStop classifier trained by emails from the user's inbox or publicly available emails. Personal email provide higher reliability to filter out regular emails from subscription bomb emails. In case a user is reluctant to allow his/her data for training, our SubStop architecture will use publicly available email database for training. We evaluated the performance of both cases and presented the results to bolster the robustness of the proposed model.

The main contribution of this paper can be summarized as follows:

- Reverse Engineering study on Gmail anti-spam mechanism: The reverse engineering study reveals why the current Gmail anti-spam cannot effectively filter out bombing emails, and also facilitates us to create a simulation script.
- In depth email bombing attack analysis through a real-life case study: We provide a real-life case study on a recent subscription bombing

[1] The experiments and research conducted in this manuscript are solely those of the authors'. The opinions expressed in this article are the authors' own and do not reflect the view of Big River Steel LLC.

[2] The authors understand the ethical issues in creating such attack. Hence, we ensured that we created a test email account in gmail, only for research purposes. We can confirm that no user is affected, barring the test account, during our conducted experiments.

attack which affected a user for days. Based on the dossier, we tried to understand how a subscription bombing attack works and identify features that may be exploited to differentiate them from normal emails.
- A novel layered detection architecture (SubStop): We propose a layered system model in detecting and mitigating a subscription bomb attack. SubStop incorporates a 'Time Filter', an 'Address Locator Filter' and an enhanced SVM algorithm.
- Extensive experiments over real-life bombing attacks: The experiments demonstrate that our developed architecture is fairly accurate in flagging subscription emails in the attacks. We also conduct performance comparison with different machine learning algorithms. We showed the enhanced performance of personalized classifier from generic model.

The remainder of this paper is organized as follows. Section II describes how a subscription bombing attack works and gives a detailed description of traditional email bomb defending techniques. In Section III, the reverse engineering process to find the inability in defending against subscription bombing attack is discussed along with a real-life subscription bombing attack case study and dossier. Section IV presents our proposed framework and approaches in our goal to build a subscription bomb detector. Section V presents the experiment results and related performance analysis. Section VI is the concise summary of the existing related work. Finally, Section VII concludes the paper.

2. Subscription bombing attack: Basics and traditional defending mechanisms

Email bombs, can be extremely annoying and frustrating for a user or group of users who encounter it. This kind of attack basically pollutes the whole inbox and requires a long time to completely get rid of, or on many occasions just to reduce the intensity. Primarily email bombs can be created in the following ways [21]:

- Mass Mailing: This attack is of the simplest form and includes sending several copies of same emails over and over again to the victim user. Attackers generally use bots to create these kind of large attacks which makes it difficult for traditional spam filtering techniques to detect such emails in huge number.
- ZIP Bombing: This is essentially email bombing with zip attachments. Usually, every email server scans for any kind of attachments sent through emails. However, this attack places text files as zip attachments with millions and billions of characters and hence require the spamming filter to utilize a large amount of its processing power in detecting the spam emails.
- Link-list Email Bombs (Subscription Bombs): This kind of attack subscribes a user's email id to unwanted list of subscriptions. As a result, unwanted subscription confirmation emails always enter the inbox bypassing the filtering techniques of the email server. The victim will have to unsubscribe from the list of subscriptions or blacklist the sources to get rid of the regularly receiving spams. However, that is hugely time consuming.

In this paper, we particularly focus on a Subscription Bombing attack since it is difficult to differentiate between a regular email and an unwanted email, which makes the attack more interesting.

2.1. Launching an email bombing attack

A subscription bombing attack is easy to launch despite its notoriety. The real challenge caused by a bombing attack is that it uses SMTP and leverage user confirmation, so that it can be launched easily and can bypass the existing spam filters. Fig. 1 shows a real time overloaded inbox with email bombing attack.

To closely recreate a real life attack, it was necessary to create something which can bombard a target email address with hundreds of emails at a given time instance. We wrote a python script [22] which can launch a subscription bombing attack, a form of email bombing. We show how easy it is to launch a bombing attack and that too without any expenses. An attacker might easily leverage the simplicity and attack several users. The sample script has been listed in the appendix. There are websites like [20] which can be used for free to send bulk emails. Attackers can lever-

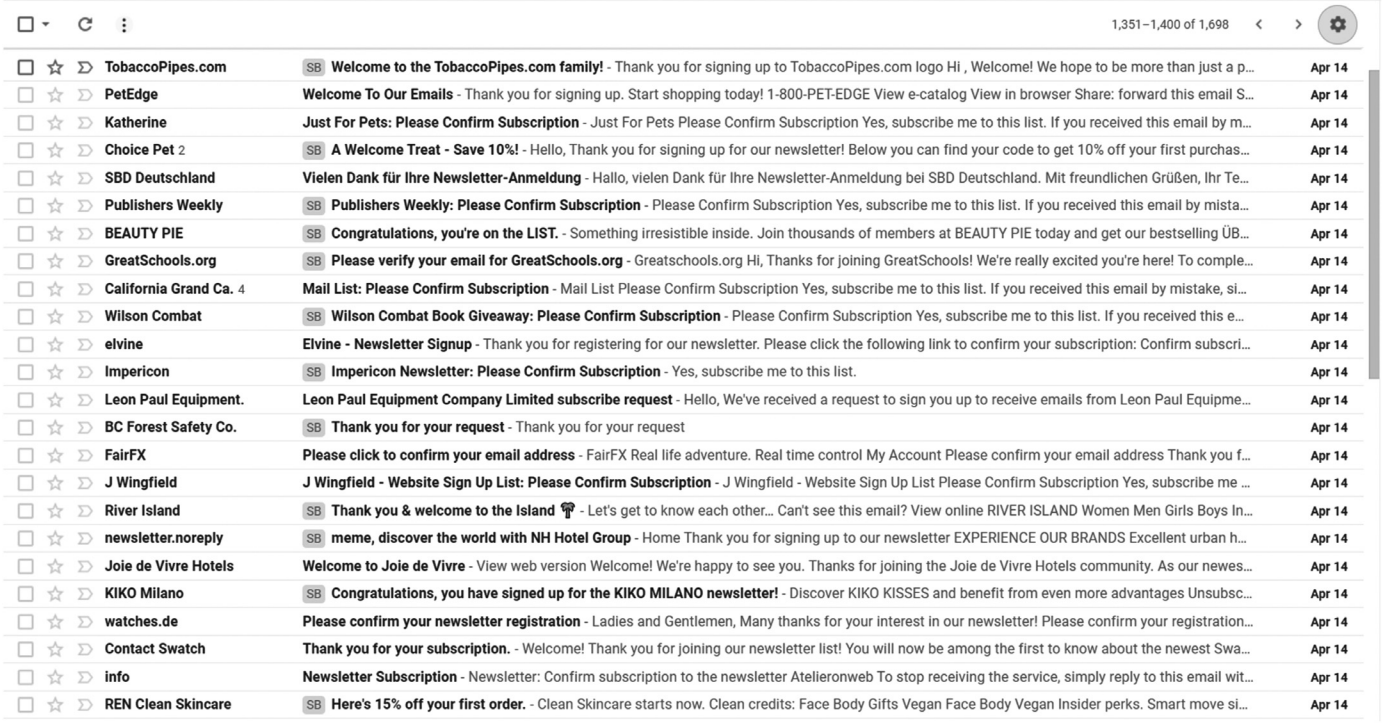

Fig. 1. Overloaded Inbox in a Real-life Bombing Attack.

Fig. 2. Victim Inbox under emulated Subscription Bombing Attack.

age these websites to launch a subscription bombing attack as shown in Fig. 2.

2.2. Existing defending techniques

Here, we concisely discuss the existing techniques that have certain effects in defending against bombing attacks.

2.2.1. Profiling for spam emails

Different types of spamming profile techniques, implemented in email servers, can be mainly categorized into the following methods as described in Table 1. We discussed in brief, some of the current techniques used to mitigate such email attacks based on these profiling schemes, in the related works section.

We confirm that blacklisting a domain or sender works in our launched subscription bombing attack as well. However, this is not always a feasible and timely solution, primarily because with the ease of launching the attack, an attacker might use several sender addresses to launch the attack and blacklisting each sender or domain after carefully going through the email bombs is a humongous and time consuming task.

Safe Sender Lists or whitelisting of email addresses are easy to implement, however, the user might lose out on several important emails coming from new email addresses making the false positive rates for this mechanism high.

Network based spam detection techniques while effective for bulk blocking domains are not cost effective for a normal user. In case of hierarchical domains, network based profiling might not be the best solution available. (For example: A user wants emails from booking.com but not from sg.booking.com). These problems motivate our work and facilitate the need for a robust system which can help affected users in mitigating these attacks.

2.2.2. Completely automated public turing test to tell computers and humans apart (CAPTCHA)

One of the primitive yet effective solutions in defending against email bombs is the CAPTCHA [23]. Captcha is a test to determine whether a user is human or a machine. RECAPTCHA was introduced later to significantly improve the already existing security measures. Some other variants include recognition of images or sounds to engage humans in verifying subscriptions but not bots.

In the context of subscription bombs, CAPTCHA is particularly useful on the server side. If an attacker uses a bot to sign up a victim email address and the website requires a CAPTCHA signature, the attack won't happen from that particular site. But sending confirmation emails and requiring a user to sign it using CAPTCHA has been the most used form of defense. However, this just piles on the misery of an email bomb victim. The victim has to detect the legitimate emails from a swamped inbox in order to confirm their subscriptions or unsubscribing from them. Typically this requires a certain level of awareness amongst the users.

2.2.3. Form-sub header

The Internet Engineering Task Force (IETF) proposed a new form of header, named as the Form-Sub Header [24], as recently as in 2017. The idea involves including the IP address of the subscriber in the header itself and masking a portion of it to protect sensitive information. The

Table 1
Spam profiling.

Profiling Techniques	Advantages	Issues
Blacklisting	Easy to implement	Cannot detect new spam emails
Safe Sender List	Easy to implement	False positive is extremely high
Network Based	Blocks bulk domains or IP addresses	Costly

header was introduced to help in identifying similar patterns of the emails' source IP addresses. But this will fail in case of highly sophisticated and co-ordinated email bombing attacks where the IP addresses are not identical in the emails.

2.2.4. Mass unsubscription services

There are websites like unroll.me [25] which lets the users to unsubscribe from unsolicited list of subscriptions at a single platform by a click. Users can peform the unsubcription feature in bulk by simply opting out from the email senders list, conglomerated and grouped by the website. The work from a user point of view is still cumbersome, since they have to go through the sender domains manually for unsubscribing them. However, one major drawback of these websites is that, if users want to opt out from the service, they might again get subscribed to many of the previously unsubscribed domains. Also, data sharing and privacy is always an issue. It is important to read in detail, how the websites manage and use the user data.

2.2.5. Machine learning based

Machine learning in the context of email spam filtering is not new. There have been several well researched studies [16], [17] in this domain. Google, Yahoo etc. themselves utilize sophisticated machine learning techniques to mitigate spam email issues for gmail, ymail etc. However, subscription bomb emails are different from ordinary spam emails. Regular spam filters cannot categorize them due to similarity of these emails with regular emails.

Our proposed framework: The layered system architecture we propose in this paper, does a sort of profiling initially based on arrival time of the emails and domain names. However, that is not identical to the traditional methods described above. We leverage a machine learning based algorithm and modify it to detect and flag the profiled emails. This layered approach ensures that our framework performs well in case of subscription bombing attacks.

3. Reverse engineering study and real-life attack analysis

In this section, we try to reverse engineer in finding how the current Gmail Antispam mechanism works in case of subscription bomb attacks. We also examine a real-life case study of a subscription bombing attack which involves one of the authors of this paper, as a victim of the attack. The number of emails suddenly spiked over a period of time. Most of the emails bombed the user's account at a specific time of the day for about a week. The emails were varied and extremely diverse based on the email's subjects and bodies, address locators etc. However, the emails can be categorized as three different types: regular emails (rmail), subscription emails (smails) and spams.

After carefully studying the compromised user account, initially, we used the unroll.me [25] feature to unsubscribe from the list of subscriptions but realized that it is hugely time consuming and newer subscription emails kept on coming every day. The need of the hour was an automated system which can do the bulk flagging of the unwanted emails directly with limited and necessary human intervention.

3.1. Reverse engineering of gmail anti-spam mechanism

"Gmail can differentiate between promotional and social media emails which is largely effective in a normal day-to-day scenario for a user. In most cases a user signs up with consent to receive such emails. However, during a subscription bombing attack the attacker uses automated bots to subscribe a victims email address to multiple lists per second, including forums and message boards, newsletters, retail mailing lists, and other everyday communications. These are extremely difficult to defend against because the Gmail classification will still consider them legitimate. An unauthorized transaction might find its way through in the promotion sections. Also, if a user is using smartphone apps to use Gmail, their app will crash during such an attack and valuable time might be lost in identifying unauthorized transactions. Due to limited reference works, it became extremely important to reverse engineer the current Gmail anti-spam mechanism and try to actually know the reason behind the probable inability that exists in detecting and mitigating these attacks. To achieve this, apart from an extensive study of the actual attack, we created a test Gmail account where we could mimic a subscription bomb attack with different scenarios.

3.1.1. Attack model

The attack model is fairly simple, however Gmail currently fails to defend against it.

- Send hundreds of subscription looking emails, in batches, to the test Gmail account.
- Send few legitimate regular emails to the test account from recognized or whitelisted email addresses, in the same time period.
- Send few mass spam emails as well, in the mix.

After carrying out the attack in various sizes and at several time periods, we found out that Gmail almost always allows emails from new email addresses or domains which do not fall in the their definition of traditional spams. This is particularly problematic, in our scenario, because all the subscription confirmation emails are deemed legitimate by Gmail's filtering technique.

We carried out several experimental scenarios to reach the reverse engineering conclusions, which bolstered our motivation in carrying out this research. We describe a couple of the experimental details to show how we reached our conclusions. Every time we bombed 100 emails to the victim email address at a particular time instant.

Victim email id: abcd@gmail.com (Full Name: Ab Cd) **Attacker email ids:** a@gmail.com, b@gmail.com

We first used the a@gmail.com attacker id to launch a subscription bombing attack to the victim email id. Once the attack was carried out, we marked the attacker id as spam. We repeated the same attack from a@gmail.com and all the emails went to 'Spam' as expected. We carried out the same attack from the b@gmail.com id (same subject, same body) to the victim id. All the emails went through to the inbox of abcd@gmail.com. This proves that the current Gmail anti-spam mechanism allows emails from new email addresses to a user's inbox. Next, we carried out the same attack again from a@gmail.com, however, this time we added the name of the victim user (Dear Ab Cd) in the email body while keeping the subjects and the rest of the email bodies the same. All the emails went to the inbox, inspite of previously marking the emails from a@gmail.com as spams. This helped us reach the conclusion that Gmail allows emails containing the user's name to the user's inbox.

All the experimental scenarios and results are presented in Table 2[3]

Specifically, the findings from our reverse engineering are described below:

- In majority of the cases, Gmail allows emails from new email addresses to a user account unless the domain of the sender address has a past spam history with that particular receiver account or in cases of outright spams.
- Emails containing the actual name of the user for the receiver email account almost always gets in to the inbox, in spite of the emails containing heavily used subscription or spam words.
- Marking an email from a sender as spam won't necessarily flag the sender address as a spam one. Gmail relies on keywords initially for anti-spam mechanism instead of email addresses.
- The process of blacklisting an email address goes through several steps. Even if an user is receiving bulk unsolicited emails from a particular email address, just marking "Report Spam" would not do

[3] Any details related to the author names have been consciously avoided due to the double-blind policy of the conference.

Table 2
Reverse engineering experiments.

Scenario	Subject	Body	Got into Victim Inbox (out of 100)	Notes
Subscription Emails	Random	Dear User, Please confirm your subscription by replying to this email.	97	Single attacker email id, all emails contained the same bodies but different randomized subjects. The 3 that went to the junk folder probably had subjects identified by Gmail as Spam.
Legitimate Emails	Resume for Internship	Dear Professor, PFA my resume with this email. Thanks	100	Same attacker id, still bombing with legitimate emails.
Spam Emails	Spam	Spam	100	We copied a spam email from a different user inbox and used the same subject and body to form our bombing emails. The attacker id is our created one and used to bombard the victim id. Gmail clearly failed to stop the bombing and let everything to the inbox because the sender email address is new and not obvious spam or spoofed email address.
Spam Emails	Spam	Spam	0	Same attacker (sender) id. After marking the id as 'spam' after the previous case. Interestingly, Gmail recommended to 'mute' instead of marking 'spam'.
Subscription Emails	Please confirm!	Confirm your subscription.	0	After reporting Spam
Legitimate Emails	Resume for Internship	Dear Professor, PFA my resume with this email. Thanks	0	After reporting Spam. After this step we went ahead and marked the sender id as 'not spam' again.
Legitimate Emails	Hello Professor	Hope you are doing good.	100	After marking the sender id as 'not spam' in the previous step.
Subscription Emails	Please confirm!	Confirm your subscription.	100	After marking the sender id as 'non spam'
Spam Emails	Spam	Spam	93	After marking the sender id as 'not spam'. 7 emails went to junk making an interesting case. Gmail labeled junk emails as "looks suspicious". Gmail still failed to stop the bombing but probably recognized something might be wrong. Next, we marked only the legitimate emails with the subject 'Hello Professor' as 'spam', not the sender id.
Subscription Emails	Please confirm!	Confirm your subscription.	100	After marking the specific legitimate emails as 'spam' in the previous step. Quite clearly, Gmail is now filtering for the subject or body in the bombing attack.
Spam Emails	Spam	Spam	100	After marking the specific legitimate emails as 'spam'.
Legitimate Emails	Hello Professor	Hope you are doing good.	0	After marking the specific legitimate emails as 'spam'.
Legitimate Emails	Hello Professor	Dear ****, Hope you are doing good.	100	Interestingly, everything remains same in this case as compared to the previous case with just the user name mentioned in the body of the email. Gmail allowed all the emails, considering the 'user name' being mentioned as the biggest factor. Probably, according to their filter this passed as a legitimate email. Next, we marked all these emails containing the user name as 'spam' and carried out the same experiment.
Legitimate Emails	Hello Professor	Dear ****, Hope you are doing good.	89	Even in this case, the user name provided the single most important factor in deciding the fate of the emails according to Gmail filtering systems. The emails that went to junk were labeled as 'Looks suspicious'. This probably happened due to the victim id and emails coming from it were marked as 'spam' on several occasions and hence Gmail based on that history got some success in blocking some.
Subscription Emails	Welcome to our Newsletter!	We heartily welcome to your new subscription	100	Gmail failed in blocking subscription bombings.

the needful. Gmail will ask the user to "Mute" the sender instead of directly marking them as spam or blacklisting them. Only after several times of marking emails from a sender as spam, Gmail will start considering them as "potential spam".

- Emails in the Gmail 'spam' both automatically detected or manually marked contain warning labels to explain why they are placed in the spambox.
- If a user marks emails from a rogue sender as spam, Gmail identifies the keywords and based on that further emails from that sender might be sent straight to the spam box with the label showing "You previously marked messages from xxx@xxx.xxx as spam". However, if an email is sent from a different previously whitelisted or new sender involving the same keywords or email content, Gmail allows it to your inbox.

Our reverse engineering analysis show that Gmail probably prefers the way of giving benefit of doubts until and unless there is a concrete proof of a sender or an email being malicious. This proves why current Gmail anti-spam mechanism is unable to deal with crippling subscription bombing attacks.

Google's stance on spam in general can be found on their website [26]. We provide a snippet in Fig. 3.

"Google Security Check-up" yield to no red flags during our test. Gmail clearly expects the user to report the issue, however, there is no concrete and full-proof way of doing that other than marking individual emails separately, which is a humongous task for an attack of this volume. Designing a system architecture which can inform the user and flag the emails is the need of the hour.

3.2. Real-life subscription bomb case study

We systematically analyze the data from the emails that were received during the course of the bombing attack. Our objective is to

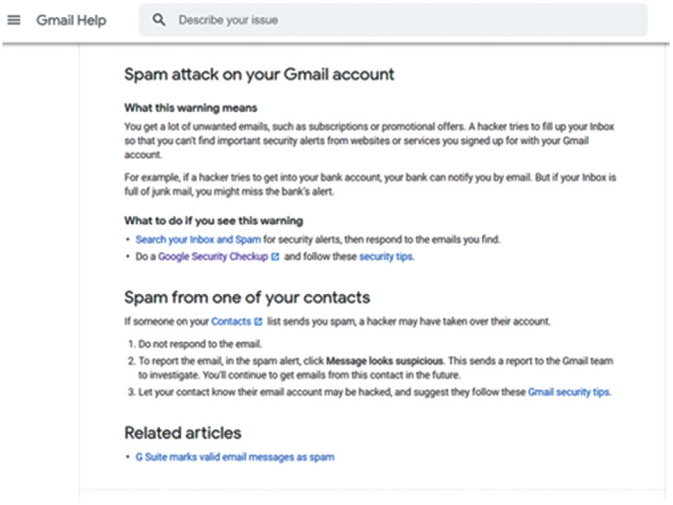

Fig. 3. Report Spam - Gmail.

study and generate features that can differentiate between regular emails, subscription bomb emails and traditional spams. These features will then be leveraged by our machine learning based filtering algorithm for correctly flagging the emails to their corresponding categories.

3.2.1. Time and volume

Over a week of sustained subscription bombing attack, we figured that the attack first started around 7:00 am everyday. The email volume increased abruptly within a time period. The highest recorded number of subscription emails is 486 on a single day. The tricky part was the regular legitimate emails that were delivered to the inbox during this time period. It is a humongous task for a user to go through all the emails to ultimately land upon the important ones. Fig. 4 shows the number of subscription emails and spams received in the 7 days of subscription bombing attacks over the 2 minute period in comparison to the number of regular emails and spams within the same period. It is interesting to note that over the period of a single day, when the subscription bombing onset for the first time, it flooded inbox within a couple of minutes. This went on for several rounds for a couple of hours, until it slows down just to reappear again the next day. The hourly email volume pattern on the worst day of the of the attack is shown in Table 3.

Table 3
Emails over different time periods.

Time	Number of Emails	Number of Bomb Emails
6:00 am	3	0
7:00 am	163	159
8:00 am	147	143
9:00 am	108	99
10:00 am	41	30
11:00 am	12	6
12:00 pm	2	0

The distribution of the volumes of the emails are significantly different than what the user encountered on normal days. From the above tabulated data, we can confirm that the rate of incoming emails to a user inbox gets significantly increased when a subscription bombing attack happens. This spike in the numbers on specific time periods and the deviation from regular behavior helped us in determining the threshold used in our 'Time Filter' later on.

3.2.2. Frequency of new address locators

Majority of the emails, that bombarded the user's inbox were from new domains or from different hierarchical levels of the same domain. For better readability, we defined the address locator term, which is the part of an email address after the @. In other words, almost all the subscription confirmation emails were from email addresses with address locators which were unknown or new to the user's inbox prior to the attack. Regular emails were mostly from email addresses with previously known address locators (user had emails from these address locators before). The legitimate transaction detail containing emails were from known email addresses, as expected. Fig. 5 shows the difference in numbers, over the period of 7 days.

3.2.3. Occurrences of keywords

Fig. 6 demonstrates the 50 most frequent words (case sensitive) that appeared in the compromised inbox, during the attack period. It is of note that the majority of these words are related to confirmation of subscription. The frequency of certain words like 'Welcome', 'Confirm', 'Subscription', 'Newsletter' etc. were staggering, an indicator of how the number of subscription emails during the bombing attack far outweighs the number of legitimate or ordinary spam emails. These high frequency words were particularly important in generating features for our machine learning algorithm in properly classifying the different types of emails. Here, 'vocab' is the word count in dictionary, 'words' is the word count in data set and 'hapax' is the word count of single used word.

The analysis in this section help us on how to design our solution. However, the design of the framework is independent of the specific contents of this single attack.

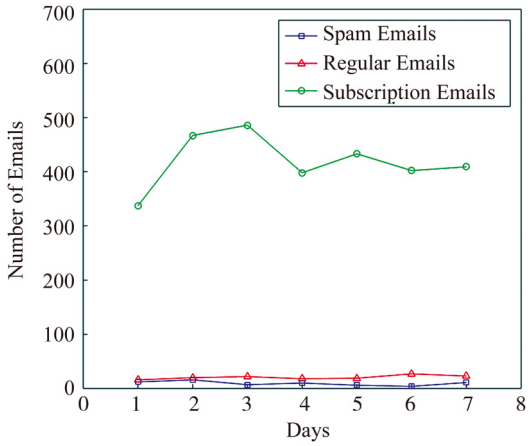

Fig. 4. Daily Emails over 7 Days.

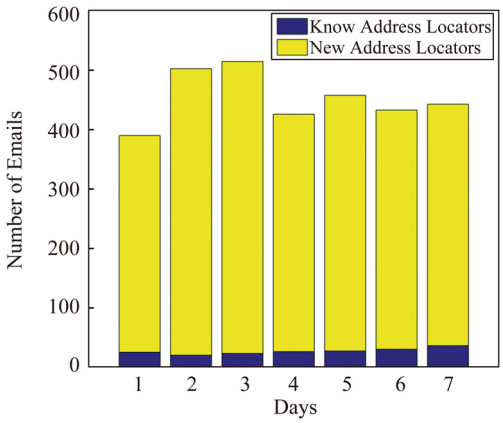

Fig. 5. Email classification based on Address Locators.

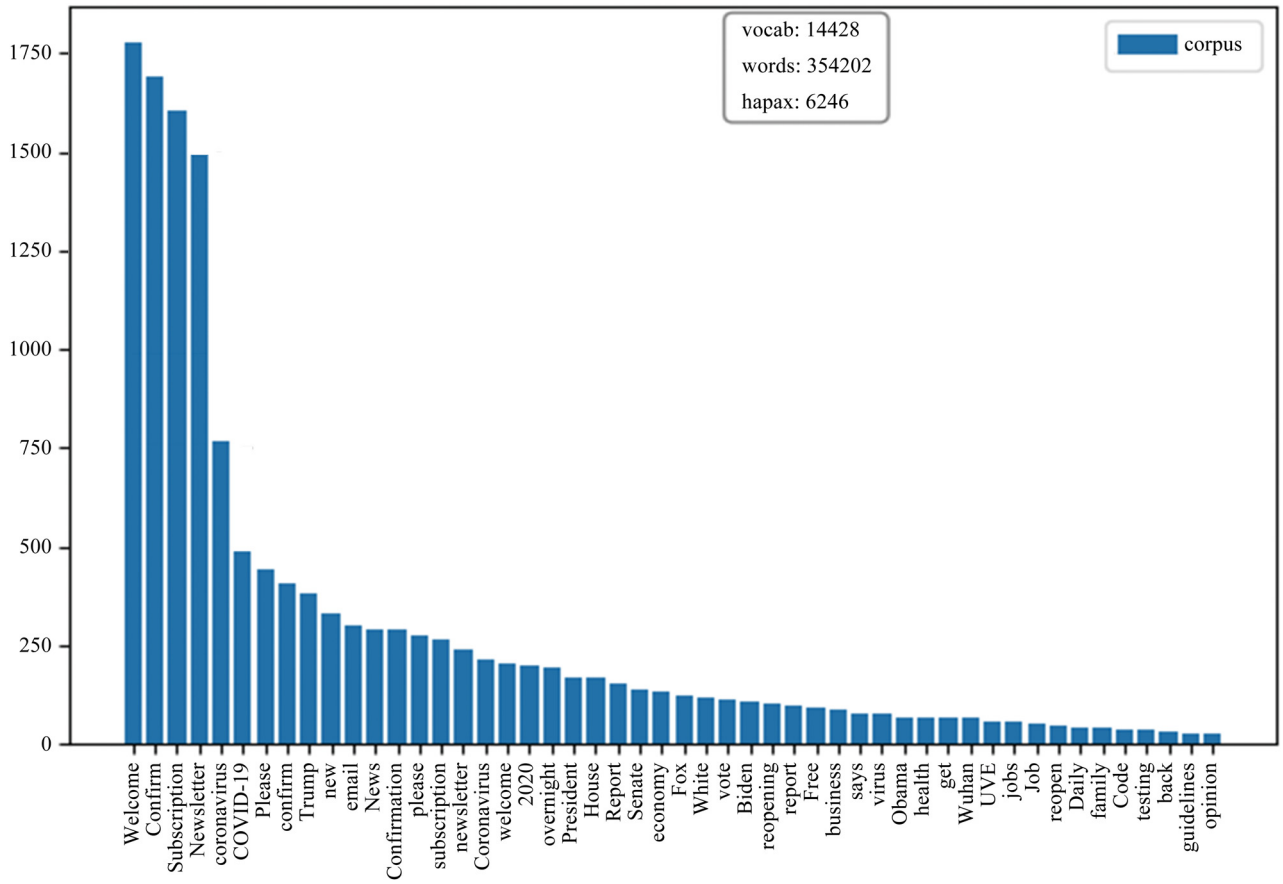

Fig. 6. Frequency of top 50 words from a Compromised Inbox.

4. Substop: Proposed system architecture

To deal with this kind of attack, we propose a novel layered system model which is shown in Fig. 7.

Based on our reverse engineering we find out that the emails that come to a user's Gmail inbox during a subscription bombing attack are of three kinds: regular emails (rmail), subscription emails (smail) and outright spams. Our proposed framework starts processing the emails after they enter the user's Gmail inbox.

First, the incoming email data gets processed through a time filter which does the primary job in detecting a possible subscription bombing attack, every hour. Once, a red flag is raised all the incoming email details including the sender address, address locators, subjects and bodies are extracted and stored in a database locally. Our proposed mechanism works on this created database, thereby ensuring that the original emails remain unaffected throughout the mitigation process and only the user takes necessary actions based on SubStop's evaluations results. Next, the extracted email data are parsed through the address locator filter. This is the place where emails from known address locators and completely new addresses get separated. This is needed to correctly distinguish the bulk subscription emails from the regular emails and spams. If no red flag is raised by the time filter, our proposed mitigation architecture will not proceed. This enables SubStop to only work for email bombing scenarios.

Since, majority of the subscription bomb emails come from new addresses or domains, the emails with new address locators get filtered through a Weighted Support Vector Machine (WSVM) which is specifically trained with features of subscription emails. If any regular legitimate email from a completely new domain enters the user's inbox, it will also be processed through this WSVM based filter and

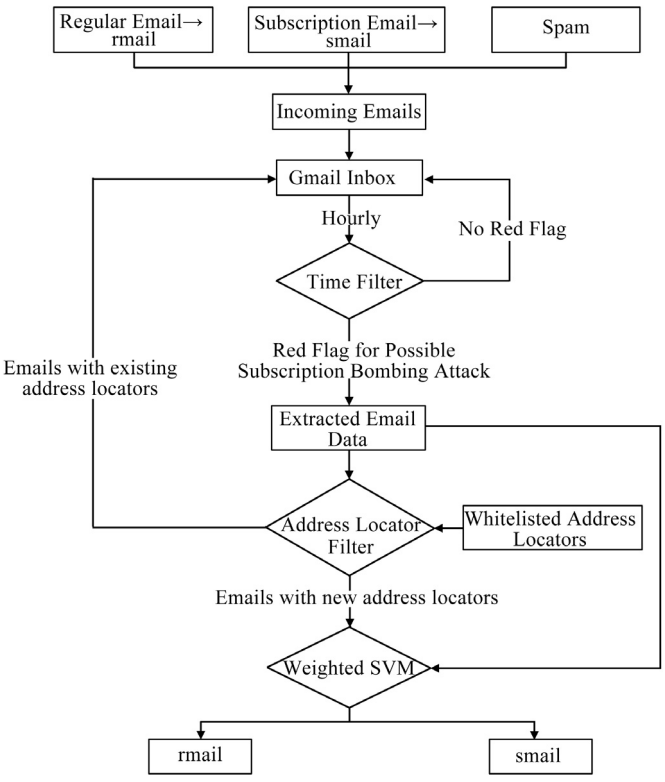

Fig. 7. Proposed SubStop Architecture to Detect Subscription Bomb Attack.

will be flagged appropriately, because of its difference with the bomb emails.

4.1. Time filter

Our case study reveals that there have been sudden huge spikes in volume of emails when a subscription bombing attack occurs as shown in Table 3. We can actually determine an abrupt change in the volume of emails of a user's inbox by calculating a rolling average of hourly email volume. A potential indication of a bombing attack can be a sudden spike in email volume.

We first calculate the average number of email a user's inbox receives over the past 60 days period. We choose the 60 day period because it will give a fair projection of the email activity in a user's inbox. For example, for a particular user, the average hourly rate of incoming emails is 10 over the past 60 days. Our moving window to detect an email bombing attack is of 10 minutes. This is because usually this kind of attack happens within a very short time.

The maximum number of emails received by the user in any one hour window during this period is 37. Based on that, we define a threshold on the number of emails $T(E)$ which will be deemed acceptable in one hour window frame. We specifically look for sudden deviations in email volume of the user in this particular time period. Let $E(t)$ be the number of emails a users inbox receives at a time t. Then the threshold is a moving average at time $t = 24$, after checking back for a span of 24 hours and is given by the following equation:

$$T(E) = \frac{1}{24} \cdot [E(1) + E(2) + \cdots + E(24)] \tag{1}$$

The average hourly rate of incoming emails during a bombing attack is highly spiked, as can be calculated based on the Fig. 4. However, our framework, allows the user to overwrite the observing window time and number of emails threshold based on their usage and understanding.

If at a time t, the number of emails that appeared in the inbox, in the prior one hour: $E(t) \geq T(E)$, a red flag is raised for a possible subscription bombing attack. To determine the exact duration of the attack, we compare the number of emails for the immediate window times of before and after t. Once, the email volume goes below the threshold, we consider the attack to be stopped. When we suspect an attack, the emails are then passed on to the Address Locator Filter after the extraction.

4.2. Address locator filter

An email address has two parts: the one before the @ locator is the username and we define the part after it as the 'address locator'. Our system, which uses a javascript code (asking for appropriate Google recommended authorizations) in the Google ScriptsApp extracts all the email addresses, the address locators, subjects and the bodies and form a detailed database of a user's inbox. This is particularly important because every user's inbox might vary a great deal and without proper representation of their personal email habits, the generalized procedure will be erroneous.

The Address Locator filter part creates two separate streams of emails for the machine learning processing to be done later. Usually, when a subscription bombing occurs, majority of the emails come from domains which the victim user's inbox is encountering for the first time. This is the reason existing machine learning algorithms do not perform well. As per our study, Gmail usually allows every new email address to a user's inbox unless it falls under their straight out spam category. So, our address filter locator clearly marks the emails with new address locators to the proposed Weighted Support Vector Machine (WSVM). It is to be noted that, we consider address locators instead of domain name because of the possibility hierarchical domain names (For eg: booking.com and sg.booking.com) where an user might be particularly interested in only one level of the domain.

This filter enables the user to separate the majority of the subscription bomb emails from the regular emails, and in the process enhances the accuracy of the system greatly in detecting these attacks.

4.3. Weighted support vector machine learning algorithm

Based on our analysis on subscription bomb attack, we utilized machine learning based classifiers to filter out regular emails. The WSVMs took input from the domain filters followed by the time filter. We used dataset from [15] to build machine learning based filter. Our dataset contains 528 subscription bomb emails from our compromised inbox and 4360 regular emails.

4.3.1. Pre-processing

Raw email data are not suitable to train. The emails are processed to make them ready for the machine learning training and testing procedures. This is done in the following ways:

- Lower-casing: Capitalizing is ignored by converting the whole email into lower-case.
- HTML Stripping: Any kind of HTML tags are removed from the emails. We remove HTML tags from emails which often come with HTML formatting, so that only the content remains.
- Remove Punctuation: Punctuation from the emails are removed.
- Remove Stop words: Prepositions, conjunctions and articles are usually considered to be stop words. Stop words appear frequently in any text and they are not useful in feature generation. Stop words are removed before feature extraction.

4.3.2. Generating features

We now want to create a dictionary based on the pre-processed email dataset. After we got the data prepared, we can start creating the dictionary where we are going to choose the features (words in this case) based on which the algorithm will later decide the type of the email.

First thing we need to do is to create the dictionary of words that will be used for our model. In our case, we have chosen the most frequent words counting all the emails (from the data sets). We count the number of times each and every word has occurred in the emails. The resultant data structure looks like the following matrix:

$$\begin{bmatrix} 1 & 17 & 3 \\ 1 & 22 & 2 \\ 1 & 9 & 4 \\ 1 & 12 & 2 \\ 1 & 5 & 3 \end{bmatrix}$$

In the above matrix, each row corresponds to:

- First Column - Sequence number of the document
- Second Column - The word's sequence number in the dictionary
- Third Column - Frequency of the word in a given email

4.3.3. Feature analysis

Data for subscription bomb emails are not prevalent unlike other email format. The imbalanced classification problem is difficult to handle than orthodox spam filter. However, most of the emails of a subscription bomb attack are highly correlated as shown in Fig. 6. The principle component analysis (PCA) of the extracted features demonstrates the separability and correlation of the desired class. Fig. 8 illustrates the first two principle component of regular email and subscription bomb email. The features of regular emails are sparser than the features of subscription bomb emails. Moreover, the features overlap in the PCA domain and cannot be separated linearly. Smaller dataset and tightly bound features reflect the higher priority of subscription bomb class over regular emails. From the feature analysis, WSVM seems most suitable for classifying emails from a subscription bomb attack.

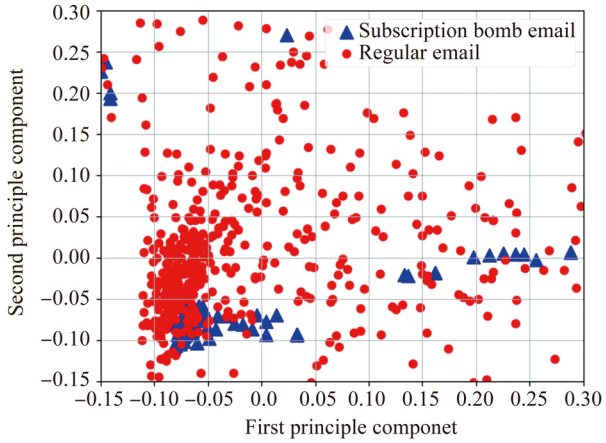

Fig. 8. Principle Component Analysis of Regular and Subscription Bomb Emails.

4.3.4. Weighted support vector machine

A classifier learns from training set to sort categorical data. Let, (x, y) represent the training data set containing N elements where, x is the feature vector of each data and y is the class label. Support vector machine determines the weights β and intercept β_0 to create a linear boundary among classes. For linearly non-separable data, a non-linear kernel $\Phi^T(x_i)$ is used that transforms the feature vectors into higher dimensional space. The Lagrangian primal function L_P of WSVM is given by,

$$L_P = \frac{1}{2}\|\beta\|^2 + \gamma C \sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \alpha_i [y_i(\Phi^T(x_i)\beta + \beta_0) - 1 + \xi_i] - \sum_{i=1}^{N} \mu_i \xi_i \tag{2}$$

Here, $\alpha_i, \xi_i, \mu_i \geq 0$ for all i. C is called the cost parameter and it determines the margin of the boundary of the classifier. γ is the class weight that can be tuned to adjust imbalanced data set. The Lagrangian dual objective function L_D of the above mentioned primal L_P is given by,

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \Phi^T(x_i)\Phi(x_j) \tag{3}$$

Here, L_D is maximized with subjected to the conditions $\sum_{i=1}^{N} \alpha_i y_i = 0$ and $0 \leq \alpha_i \leq \gamma C$. Solution of Lagrangian dual would yield optimum α_i^*, which leads to

$$\beta^* - \sum_{i=1}^{N} \alpha_i^* y_i \Phi(x_i) = 0$$
$$\alpha_i^* \left[y_i \left(\Phi^T(x_i)\beta^* + \beta_0^* \right) - 1 + \xi_i \right] = 0, \tag{4}$$

The decision function $G(x)$ of a new input x_k from the classifier is given by,

$$G(x_k) = sign[\Phi^T(x_k)\beta^* + \beta_0^*] \tag{5}$$

4.3.5. WSVM for new address locator emails

The weight of subscription bomb class is kept higher than regular email. Very few of the emails received during a subscription bomb attack are regular emails. The accuracy of the classifier depends largely on detecting most of the subscription bomb emails. Here, weight for regular class is, $\gamma_+ = 1$ and weight for subscription bomb class is, $\gamma_- > 1$. The weights, $\gamma_- > \gamma_+$ ensures that, most of subscription bomb emails are classified accurately and poses insignificant threat of misclassifying regular emails.

4.4. Training with dataset from user's inbox

The challenge in defending subscription bomb attack robustly for different user comes with variation of individual user's email preferences. Unlike spam emails, the format of bomb emails are similar with legitimate emails. An undesired bomb email may be deemed legitimate email for a different user. The attack described in Section III-B had multiple emails from news organizations. The user was not subscribed to those news organization and categorized those emails as bomb emails. A different user, who had subscribed to those news organization would receive the same emails irrespective of a potential bomb attack. The same emails are required to be categorized as legitimate emails for a different user. The only solution for this issue in a data-driven approach is to train the system using individual's email for the legitimate email class. We kept the concern for user's privacy and reluctance to share the private emails. We provide the system that is trained with publicly available database and ask permission to individual user to utilize the emails from inbox for better performance. Our system works offline and data privacy of each user would be ensured.

5. Experiments and results

We exhaustively analyzed performance of our proposed SubStop architecture with state of the art machine learning based spam filters. We demonstrated the quality indices and comparative analysis of our proposed WSVM on testing data. We carried out a real-time subscription attack and presented the significance of each stage of our methodology.

5.1. Performance metrics of classifier

We defined the subscription bomb class as positive and the regular email class as negative. The confusion matrix of email classifier is demonstrated in Table 4.

The performance metrics for classifier are defined using the confusion matrix in Table 4.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}$$

$$Precision = \frac{TP}{TP + FP} \tag{7}$$

Accuracy indicates the separability of the classifier. Low precision shows that, FP is high which infers significant legitimate emails are classified as subscription bomb emails.

During a subscription attack, substantially fewer regular emails are received compared to bomb emails as illustrated in Fig. 5. It is worth noting that different classes of emails involved in bombing attacks are imbalanced in volume. The number of regular email is typically very small. In order to avoid mis-interpretation of performance due to volume balance, we utilized the metric "balanced accuracy" (b_Acc) which is given by,

$$b\_Acc = \frac{1}{2}\frac{TP}{TP + FP} + \frac{1}{2}\frac{TN}{TN + FN} \tag{8}$$

In fact, b_Acc is an integrated evaluation of detection precision crossing both bombing attack emails and regular emails. In the following part

Table 4
Confusion matrix of email classifier.

Emails	Detected as Subscription bomb	Detected as Regular
Subscription bomb	True Positive (TP)	False Negative (FN)
Regular	False Positive (FP)	True Negative (TN)

Table 5
Performance of classifiers.

Quality Parameter	Accuracy	b_Acc	Precision
Naive Bayesian	88.54%	53.72%	100%
SVM	94.78%	78.92%	100%
DL	97.04%	82.63%	100%
Proposed WSVM	96.01%	83.88%	100%

of this section, extensive experiment results will demonstrate b_Acc is the key metric demonstrating that the proposed SubStop achieves robust performance in labeling both bombing attack and regular emails.

5.2. Comparative analysis of proposed WSVM

This part evaluates the performance of the key machine learning component of SubStop, i.e., the WSVM module for differentiating regular emails from the bombing attacks. The evaluation is conducted through comparison with traditional spam filters based on Naive Bayesian method and SVM. We used dataset from [15] containing 4360 regular emails. We also utilized 528 subscription bomb emails from the compromised inbox described in Section III-B. The ratio of testing set and training set (σ) used for our design is 20/80. The parameters used for classifiers are described below.

- Naive Bayesian [16]: Multinomial Naive Bayesian (NB) classifier is used with no prior knowledge.
- SVM [17]: Radial basis function (RBF) is used for non-linear kernel function. Equal weights are given to each classes.
- DL [19]: Deep learning based text classifier utilizing bidirectional encoder representations from transformers (BERT).
- Proposed WSVM: RBF is used for non-linear kernel function. The weight for subscription bomb class is set to $\gamma_- = 1.6$

The comparative performance of subscription bomb filter based on test data are demonstrated in Table 5. Subscription bomb emails are highly correlated and regular emails are rarely misclassified as subscription bomb emails. The effect of weight γ_- on accuracy and b_Acc are illustrated in Fig. 9. Accuracy and b_Acc get saturated at 96.01% and 83.88%, when $\gamma_- = 1.6$.

5.3. Real-time experiment and results

In this part, we evaluate the performance of the whole SubStop framework under a real-time emulated bombing attack to real-life email accounts (created only for experiments). In our experiment, we sent several hundreds of subscription emails every hour to a test email account

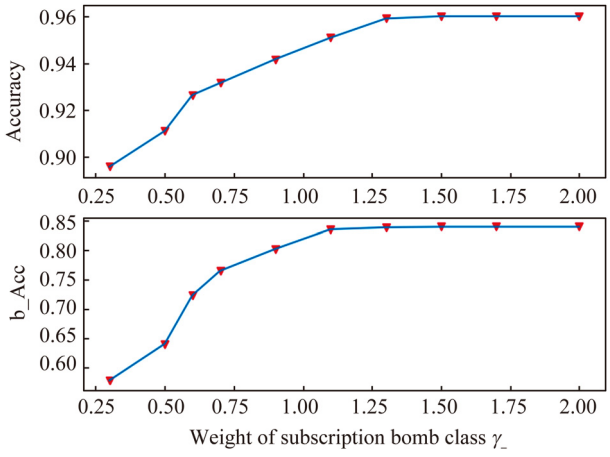

Fig. 9. Impact of weight (γ_-) on accuracy and b_Acc of the classifier.

Table 6
Email data distribution of real time attack.

		Regular email	
Attack scenario	Bomb email	White-listed	New domain
case-1	1311	2	3
case-2	1127	4	1
case-3	1293	2	2

by our bombing script and by utilizing the services from [20]. Contribution of the SubStop architecture is evaluated and analyzed on the compromised test Gmail inbox.

5.3.1. Experimental setup

Our proposed SubStop method was compared with Naive Bayesian and SVM based filter. These classifiers had only regular email and subscription bomb email class and no spam email class. Spam emails are rarely received during a bomb attack and has insignificant effect in system performance. We also address locator filter with NB and and SVM to compare with our architecture. Our robust system is designed to handle all possible threats. SubStop are evaluated and analyzed on compromised inbox.

We opened a Gmail account with default setting provided by Google. Only the Google domain is now white-listed. The threshold for time filter $T(E)$ is set to 15 emails per hour. We launched the subscription bomb attacks three times. All the attacks spanned in two days period. The details of the attacks are demonstrated in Table 6. The first two attacks were launched using the website [20]. The third attack was instigated using the script given in appendix. The Gmail account was linked to a credit card and a purchase was made from Amazon during the attack. After each attack, the regular emails received were marked as white-listed.

5.3.2. Experimental findings

After each launch, time filter marks all the emails mentioned in Table 6. In case-1, all the regular emails except from google domain came from new domain. The emails received from the credit card company, Amazon and Google are considered to be regular emails After address locator filter processing, WSVM processed all the emails. The first regular email came from the credit card company for confirmation. The email was flagged falsely by WSVM due to the similarity of the email with subscription bomb emails. The second email came from the amazon for the purchase. The third regular email came from the credit card company to alert the user for over-expenditure as illustrated in Fig. 10. The only legitimate email from the credit card company is circled. The description of the card user is masked for privacy. Discovering these emails without filtering is nearly impossible during a bomb attack. Both of these emails were flagged properly by the proposed Sub-stop architecture. Two more regular emails came from gmail domain.

There were 5 regular emails in case-2. 4 of those emails came from white-listed domain and they were white-listed. In case-3, 2 email came from white-listed domain and 2 emails were received from new domain. All bomb emails and regular emails were classified accurately by address locator and WSVM. Contribution of the WSVM on regular emails received from new domain is presented in Table 7.

Table 7
Performance of proposed WSVM on regular emails from new domain.

Attack scenario	Email received	Classified as regular
case-1	3	2
case-2	1	1
case-3	2	2

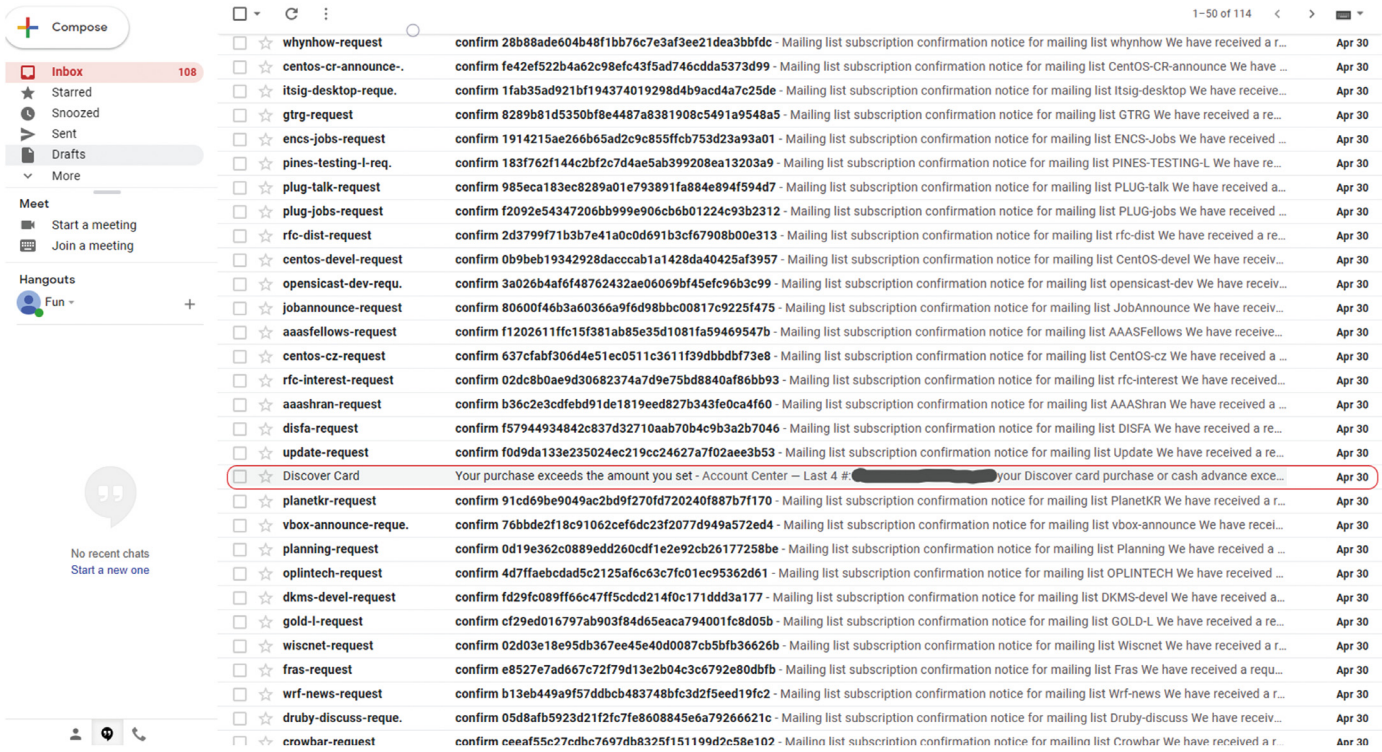

Fig. 10. Gmail inbox after subscription bomb attack.

5.3.3. *Results and discussion*

Performance of the attacks are demonstrated in Table 8. Subscription bomb emails are categorically similar. Misclassification of bomb emails of any category would drastically decrease the accuracy of the classifier. Naive Bayesian based spam filter failed miserably to detect subscription bomb attack. In case-1, both of these methods failed to classify any regular emails. Address locator filter improves the performance significantly of all classifier in practical situation. During an ongoing attack, rate of receiving regular emails is very small compared to bomb emails. And most of the regular emails are sent from a white-listed domain. In all three attacks, SubStop misclassified only one regular email. Implementation of address location filter and proposed WSVM ensures that, most of the bomb emails are recognized correctly, keeping the regular emails unmarked. Our proposed model maintains the highest accuracy, b_Acc and precision in real-time subscription bombing attack.

The time filter works as a switch in our architecture. It ensures any fidgeting of user's inbox only if a bomb attack is imminent. Time filter preserves computational resources and allows the system to work in real-time. The performance of WSVM and SubStop in Table 8 delineate the contribution of address locator filter in the architecture.

WSVM enables our model to adaptively handle the wide heterogeneity in practical email contents by turning the class weight in the training. The training of the model and attack simulation are different in characteristics. There are plenty of regular emails available online for training. Furthermore, the content of regular emails are more diverse compared to subscription bomb emails. We used 4360 regular emails and 528 subscription bomb emails, for training and validations. However, a user receives significantly higher number of bomb emails compared to regular email during an attack. This difference in distribution is the reason for tuning class weight for training. Our experiments demonstrate the advantage compared to SVM with fixed or static weights.

5.4. *Performance on personalized training*

We gathered the data of two more compromised inbox from real time subscription bomb attack. The users granted us permission to test our system on their Gmail account. The bomb emails were marked by the users. We collected the 1000 regular emails from user's inbox that were received at least one week before the attack. The address locator filter was kept off in this experiment to show the stark difference in performance of classifier in two situation. We trained our WSVM classifier with the regular emails collected and bomb emails we have in our database mentioned in Section IV-C. We separately stored the emails that were received during the bomb attack and used those emails for performance evaluation. The details of the experiments are delineated in Table 9.

Table 8
Robustness and superior performance (specially in b_Acc) of SubStop under real-time bombing attacks.

	case-1			case-2			case-3		
Quality Parameter	Accuracy	b_Acc	Precision	Accuracy	b_Acc	Precision	Accuracy	b_Acc	Precision
NB	25.47%	62.38%	100%	37.89%	48.89%	99.53%	31.72%	53.30%	99.75%
SVM	99.92%	90%	99.92%	94.7%	47.55%	99.53%	98.32%	49.30%	99.68%
DL	97.04%	98.5%	98.16%	94.16%	97.07%	94.14%	85.42%	92.69%	85.38%
Proposed WSVM	99.92%	90%	99.92%	97.70%	68.97%	99.72%	98.99%	74.83%	99.84%
Address locator+NB	25.47%	62.38%	100%	45.45%	68.67%	99.80%	37.82%	72.51%	100%
Address locator+SVM	99.92%	90%	99.92%	95.7%	57.55%	99.53%	98.87%	59.30%	99.68%
SubStop	**99.92%**	**90%**	**99.92%**	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**

Table 9
Using personalized emails for training.

	Training set			Testing set	
Attack scenario	Regular email	Bomb email	Address locator list size	Regular email	Bomb bomb
User-1	1000	528	17	31	317
User-2	1000	528	9	14	459

Table 10
Performance comparison on test set of proposed classifier trained with publicly available data and personal data.

	User-1		User-2	
Quality Parameter	Accuracy	b_Acc	Accuracy	b_Acc
Public data	85.51%	71.23%	93.02%	64.89%
Personal data	99.14%	95.59%	99.58%	93.75%

Using personal data for training shows significant improvement compared to commonly available public data as shown in Table 10. In both cases, regular emails were all detected accurately. User 1 and 2 had 42 and 33 bomb emails respectively misclassified when the classifier was trained with publicly available data. The effect of misclassifaction yield poor b_Acc. However, utilizing personal emails for training enhanced the metrics. User 1 and 2 had only 3 and 2 bomb emails misclassified. No regular email was misclassified in either cases.

5.5. Existing anti-spam plug-ins

We installed two spam filters named Spameo [27] and Anti Spam [28], in the form of Google Chrome extensions or plug-ins. Out of the 314 emails sent during the first subscription bombing attack, the extensions were able to block to only 2. The rest 312 emails made their way to the inbox. This shows that the existing solutions are not well equipped to handle a subscription bombing attack.

The most important take out from the experiment is how less human involvement is needed when we utilized the proposed technique in building the filtering process for these kind of attacks. To mitigate a similar kind of attack via unsubscribe services like unroll.me took much longer for us, because of manually unsubscribing from all kind of sender domains. On the contrary, after just authorizing our framework, the layered framework required very less human intervention, which in turn improved the results further. After processing through all the bombarded emails, the framework can notify the user of successfully detecting the regular and subscription bomb emails. Each email detection roughly took around 47ms and for the whole attacks to process, our solution took approximately 16s (for over 300+ bombarded emails at the same time). This proves that not only the solution is extremely efficient but highly fast as well. We aim on achieving faster results in our future works.

6. Related work

In the state of the art literature, spam has been studied extensively in relation to modern age Internet security threats which consists of unsolicited junk emails and phishing emails as well. In several years of spam existence, both independent researchers and industrial organizations have tried to study the behavioral patterns of spam and tried to design implementations which albeit not being universal, served their intended purposes of getting rid of junks.

In recent studies [21] there is another form of email bombing presented barring the two traditional ones - zip bombing. This is essentially email bombing with zip attachments. Usually, every email server scans for any kind of attachments sent through emails. However, this attack places text files as zip attachments with millions and billions of characters and hence require the spamming filter to utilize a large amount of its processing power in detecting the spam emails. However, subscription bombing falls within mass mailing or link list email bombs categories. Zip bombing is not the focus of this work.

Dev et al. [7] and Houle and Pandey [8] are a couple of studies dedicated to mass mailing attacks. However, both the papers deal with existing attack scenarios and present more of a case study on the plight of an average email user. There is no concrete framework in place to flag subscription bombing attacks, in general.

A detailed analysis on the evolution of advanced spammers is given in Sawaya et al. [29]. Provider based email security is proposed in [30]. The research work particularly serve as a dossier on the effectiveness of such security methodology. Kaushik et al. [31] point out that current email control mechanisms are effective in dropping unsolicited emails, however, are prone to dropping desirable emails in the process due to the coarseness.

Anti-phishing techniques have been researched adequately. Almomani et al. [32] provides an amazing and thorough survey on the current methods employed worldwide to deal with the demons of spam and phishing emails. Hamid and Abawajy [33] proposed a cluster based approach in profiling phishing emails. This helped in categorizing phishing emails and measures could be taken accordingly to eliminate the attacks. Che et al. [34] proposed a content based solution in dealing with phishing attacks. Şentürk et al. [35] implemented novel data mining techniques to detect and prevent phishing. eBay toolbar [36], SpoofGuard [37], IE phishing filter [38], CallingID [39], NetCraft [40], CloudMark [41] etc. are few tools implemented at the user side to filter spam emails. These kind of tools directly detect phishing websites based on prior study and understanding of spams incorporated in their programming design. Typically these techniques alert the users for a potential spam attack and leaves the ultimate decision making to the user himself/herself.

Recently, deep learning based classifications are outperforming traditional machine learning algorithms. Abundance of training data is an essential requirement for deep learning techniques. Long-Short-Term-Memory (LSTM) based active learning is presented in [12] and [18] for spam classification. Deep Neural Network (DNN) based anti-phishing filter is implemented in [42]. Any complex algorithm especially based on deep learning requires a significantly higher number of training data. However, we only have subscription bomb email data from one compromised inbox since these are not available in a public library. Training deep learning based algorithms requires tuning on the model design as well as multiple hyper-parameters. Our proposed system effectively tunes one hyper-parameter. There is a significant risk of overfitting using deep learning based classifiers due to lack of sufficient validation data. We presented the comparative performance of DL based classifier [19] with the proposed SubStop method. Even though, the DL based classifier showed promising performance in training phase, it showed inferior performance in real-time attacks.

7. Conclusion and future work

In this paper, we investigated a recent real-time subscription bombing attack. Subscription bombs are a serious threat to both current and monetary affairs in the internet age. Several users including big financial organizations have fallen prey to this deadly yet simplistic form of attack. Based on our study and research, we found out that the existing solutions for dealing with these kind of attacks fall short comprehen-

sively and the victims lose out on a considerable amount of valuable time. According to our dossier of the subscription bombing attack, the unsolicited emails varied from origins to language to patterns to even source email domains.

We attempted to provide a solution 'SubStop' - to this ever increasing crippling threat. At first, we propose a moving window technique to put an initial red flag for subscription bombing attacks, which is then passed on to an Address Location filter. This helps in categorizing the emails in these kind of attacks and separate them for better and accurate processing later on. Then, we leveraged and modified the Weighted Support Vector Machine algorithm in designing and developing a robust defensive mechanism for the hard to detect bomb emails. Our experiment and results show how efficient our proposed method is in defending against such attacks. In future, we aim to collect data from more compromised inboxes and conduct experiments with more sophisticated algorithms.

In our future work, we plan to extend the proposed system architecture as a web-based plug-in feature to incorporate our full proposed framework. We will provide an installable file which users can download and install in their own machines. Once, the required authorization has been done after signing in the email client, our parser will detect and block any kind of unwanted emails involved in a subscription bombing attack.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

Funding: This work was supported in part by the National Science Foundation (NSF) [No. CNS-1816908].

Appendix A. Python Code

A sample python code to launch a subscription bombing attack to any email address.

```
import os
import smtplib
import getpass
import sys
server = input(''Enter Mail Server: '')
user = input(''E-mail Address: '')
passwd = getpass.getpass(''Password: '')
to = input(''\nTarget E-mail Address: '')
subject = input(''Subject: '')
body = input(''Body: '')
emails = int(input(''Number Of Emails: ''))
if server == ''gmail'':
 smtp_server = ''smtp.gmail.com''
 port = 587
elif server == ''yahoo'':
 smtp_server = ''smtp.mail.yahoo.com''
 port = 995
else:
 print(''Script to launch email bombing attack for research purposes.'')
 sys.exit()
try:
 server = smtplib.SMTP(smtp_server,port)
 server.ehlo()
 if smtp_server == ''smtp.gmail.com'':
 server.starttls()
 server.login(user,passwd)
 for i in range(1, emails+1):
 msg = ''From: '' + user + ''\nSubject: '' +subject + ''\n'' + body
 server.sendmail(user,to,msg)
 print(''\rSending Emails{\ldots} :
 sys.stdout.flush()
 server.quit()
 print(''\n Target Email Address Has Been Bombed Successfully!'')
except KeyboardInterrupt:
 print(''[-] Canceled'')
 sys.exit()
except smtplib.SMTPAuthenticationError:
 print(''\n[!] Incorrect Credentials.'')
 sys.exit()
```

References

[1] R. Tomlinson, http://www.computinghistory.org.uk/det/6116/First-e-mail-sent-by-Ray-Tomlinson/, 1971.
[2] M. Xie, H. Yin, H. Wang, An effective defense against email spam laundering, in: Proceedings of the 13th ACM Conference on Computer and Communications Security, in: CCS '06, Association for Computing Machinery, New York, NY, USA, 2006, pp. 179–190, doi:10.1145/1180405.1180428.
[3] GARTNER, Gartner Survey Shows Phishing Attacks Escalated in 2007, 2007. Public Report, Available at http://www.gartner.com/it/page.jsp?id=565125.
[4] J. Marsden, Z. Albrecht, P. Berggren, J. Halbert, K. Lemons, A. Moncivais, M. Thompson, Facts and stories in phishing training: A replication and extension, in: Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems, in: CHI EA '20, Association for Computing Machinery, New York, NY, USA, 2020, pp. 1–6, doi:10.1145/3334480.3381435.
[5] C. for Internet Security, MS-ISAC Security Primer-Email Bombs, 2018. Available at https://www.cisecurity.org/white-papers/ms-isac-security-primer-email-bombs/.
[6] M. McLuhan, This is Marshall McLuhan: The medium is the message., December 2004. Public Report, Available at http://www.messagelabs.com/intelligence/2004report.
[7] J. Dev, E. Rader, S. Patil, Why johnny can't unsubscribe: Barriers to stopping unwanted email, in: Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems, in: CHI '20, Association for Computing Machinery, New York, NY, USA, 2020, pp. 1–12, doi:10.1145/3313831.3376165.
[8] C. Houle, R. Pandey, A layered approach to defending against list-linking email bombs, in: 2018 APWG Symposium on Electronic Crime Research (eCrime), IEEE, 2018, pp. 1–9.
[9] N. Moradpoor, B. Clavie, B. Buchanan, Employing machine learning techniques for detection and classification of phishing emails, in: Computing Conference, 2017, IEEE, 2017, pp. 149–156.
[10] W. Niu, X. Zhang, G. Yang, Z. Ma, Z. Zhuo, Phishing emails detection using cs-svm, in: 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.
[11] P. George, P. Vinod, Machine learning approach for filtering spam emails, in: Proceedings of the 8th International Conference on Security of Information and Networks, in: SIN '15, Association for Computing Machinery, New York, NY, USA, 2015, pp. 271–274, doi:10.1145/2799979.2800043.
[12] Z. Chen, R. Tao, X. Wu, Z. Wei, X. Luo, Active learning for spam email classification, in: Proceedings of the 2019 2nd International Conference on Algorithms, Computing and Artificial Intelligence, in: ACAI 2019, Association for Computing Machinery, New York, NY, USA, 2019, pp. 457–461, doi:10.1145/3377713.3377789.
[13] C.D.C.M. University, Email Bombing and Spamming, 2017. Available at https://resources.sei.cmu.edu/asset_files/WhitePaper/2002_019_001_496605.pdf.
[14] D. Services, How to mitigate email bombing, 2020. Available at https://services.dartmouth.edu/TDClient/1806/Portal/KB/ArticleDet?ID=82545.
[15] R. Anderson, Email Spam Detection Using Python & Machine Learning, 2019. Available at https://medium.com/@randerson112358/email-spam-detection-using-python-machine-learning-abe38c889855.
[16] M. Sahami, S. Dumais, D. Heckerman, E. Horvitz, A bayesian approach to filtering junk e-mail, in: Learning for Text Categorization: Papers from the 1998 workshop, volume 62, Madison, Wisconsin, 1998, pp. 98–105.
[17] H. Drucker, D. Wu, V.N. Vapnik, Support vector machines for spam categorization, IEEE Trans. Neural Networks 10 (5) (1999) 1048–1054.
[18] G. Liu, J. Guo, Bidirectional lstm with attention mechanism and convolutional layer for text classification, Neurocomputing 337 (2019) 325–338, doi:10.1016/j.neucom.2019.01.078.
[19] I. AbdulNabi, Q. Yaseen, Spam email detection using deep learning techniques, Procedia Comput Sci 184 (2021) 853–858, doi:10.1016/j.procs.2021.03.107. The 12th International Conference on Ambient Systems, Networks and Technologies (ANT) / The 4th International Conference on Emerging Data and Industry 4.0 (EDI40) / Affiliated Workshops
[20] Mailbait, MailBait: Fill Your Inbox, 2020. Available at http://mailbait.info/index.html.
[21] T. Bass, A. Freyre, D. Gruber, G. Watt, E-mail bombs and countermeasures: cyber attacks on availability and brand integrity, IEEE Netw 12 (2) (1998) 10–17.
[22] G. Van Rossum, F.L. Drake, Python language reference manual (2003).
[23] M. Jakobsson, F. Menczer, Untraceable email cluster bombs: on agent-based distributed denial of service, arXiv preprint cs/0305042 (2003).
[24] Messaging, Malware Mobile Anti-Abuse Working Group, M3AAWG: M3AAWG Recommendation on Web Form Signup Attacks, 2017. Available at http://www.m3aawg.org/WebFormAttacks.
[25] Unroll.me, https://unroll.me/, June 2012.
[26] Google, Mark or unmark Spam in Gmail, 2020. Available at https://support.google.com/mail/answer/1366858}report_spam.
[27] spameo.com, Spameo Anti Spam Filter (Gmail), 2018. Google Chrome Extension.
[28] BlockSender.io, Anti Spam 2014 (Gmail and Google Apps), 2014. Google Chrome Extension.
[29] Y. Sawaya, A. Kubota, A. Yamada, Understanding the time-series behavioral characteristics of evolutionally advanced email spammers, in: Proceedings of the 5th ACM Workshop on Security and Artificial Intelligence, in: AISec '12, Association for Computing Machinery, New York, NY, USA, 2012, pp. 71–80, doi:10.1145/2381896.2381908.

[30] I.D. Foster, J. Larson, M. Masich, A.C. Snoeren, S. Savage, K. Levchenko, Security by any other name: On the effectiveness of provider based email security, in: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, in: CCS '15, Association for Computing Machinery, New York, NY, USA, 2015, pp. 450–464, doi:10.1145/2810103.2813607.

[31] S. Kaushik, W. Winsborough, D. Wijesekera, P. Ammann, Email feedback: A policy-based approach to overcoming false positives, in: Proceedings of the 2005 ACM Workshop on Formal Methods in Security Engineering, in: FMSE '05, Association for Computing Machinery, New York, NY, USA, 2005, pp. 73–82, doi:10.1145/1103576.1103586.

[32] A. Almomani, B. Gupta, S. Atawneh, A. Meulenberg, E. Almomani, A survey of phishing email filtering techniques, IEEE communications surveys & tutorials 15 (4) (2013) 2070–2090.

[33] I.R.A. Hamid, J.H. Abawajy, Profiling phishing email based on clustering approach, in: 2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), IEEE, 2013, pp. 628–635.

[34] H. Che, Q. Liu, L. Zou, H. Yang, D. Zhou, F. Yu, A content-based phishing email detection method, in: 2017 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C), 2017, pp. 415–422.

[35] Ş. Şentürk, E. Yerli, İ. Soğukpınar, Email phishing detection and prevention by using data mining techniques, in: 2017 International Conference on Computer Science and Engineering (UBMK), IEEE, 2017, pp. 707–712.

[36] EBay, eBay Toolbar, 2004. Available at http://download.cnet.com/eBay-Toolbar/3000-125124-10153544.html?tag=contentMain;downloadLinks.

[37] N. Chou, R. Ledesma, Y. Teraguchi, J.C. Mitchell, Client-side defense against web-based identity theft, Computer Science Department, Stanford University. Available: http://crypto.stanford.edu/SpoofGuard/webspoof.pdf (2004).

[38] Microsoft, IP Phishing Filter, 2016. Available at http://support.microsoft.com/kb/930168.

[39] CallingID, Your Protection from Identity Theft, Fraud, Scams and Malware, 2004. Available at https://www.israeldefense.co.il/en/company/calling-id.

[40] Netcraft, Netcraft Toolbar, 2006. Available at http://toolbar.netcraft.com/.

[41] Jordan Ritter, CloudMark, September 2001. Available at http://www.cloudmark.com/en/products/cloudmark-desktopone/index.

[42] R. Hassanpour, E. Dogdu, R. Choupani, O. Goker, N. Nazli, Phishing e-mail detection by using deep learning algorithms, in: Proceedings of the ACMSE 2018 Conference, in: ACMSE '18, Association for Computing Machinery, New York, NY, USA, 2018, doi:10.1145/3190645.3190719.