CADUE: Content-Agnostic Detection of Unwanted Emails for Enterprise Security

Mohamed Nabeel
mnabeel@hbku.edu.qa
Qatar Computing Research Institute
Qatar

Enes Altinisik
ealtinisik@hbku.edu.qa
Qatar Computing Research Institute
Qatar

Haipei Sun
hsun15@stevens.edu
Stevens Institute of Technology
USA

Issa Khalil
ikhalil@hbku.edu.qa
Qatar Computing Research Institute
Qatar

Hui (Wendy) Wang
hwang4@stevens.edu
Stevens Institute of Technology
USA

Ting Yu
tyu@hbku.edu.qa
Qatar Computing Research Institute
Qatar

ABSTRACT

End-to-end email encryption (E2EE) ensures that an email could only be decrypted and read by its intended recipients. E2EE's strong security guarantee is particularly desirable for the enterprises in the event of breaches: even if attackers break into an email server, under E2EE no contents of emails are leaked. Meanwhile, E2EE brings significant challenges for an enterprise to detect and filter unwanted emails (spams and phishing emails). Most existing solutions rely heavily on email contents (i.e., email body and attachments), which would be difficult when email contents are encrypted. In this paper, we investigate how to detect unwanted emails in a content-agnostic manner, that is, without access to the contents of emails at all.

Our key observation is that the communication patterns and relationships among internal users of an enterprise contain rich and reliable information about benign email communications. Combining such information with other metadata of emails (headers and subjects when available), unwanted emails can be accurately distinguished from legitimate ones without access to email contents. Specifically, we propose two types of novel enterprise features from enterprise email logs: *sender profiling features*, which capture the patterns of past emails from external senders to internal recipients; and *enterprise graph features*, which capture the co-recipient and the sender-recipient relationships between internal users. We design a classifier utilizing the above features along with existing meta-data features. We run extensive experiments over a real-world enterprise email dataset, and show that our approach, even without any content-based features, achieves high true positive rate of 95.2% and low false positive rate of 0.3% with such stringent constraints.

CCS CONCEPTS

• **Security and privacy** → **Phishing**; *Spoofing attacks*; Intrusion/anomaly detection and malware mitigation.

RAID '21, October 6–8, 2021, San Sebastian, Spain

ACM ISBN 978-1-4503-9058-3/21/10…$15.00
https://doi.org/10.1145/3471621.3471862

KEYWORDS

Phishing, Spam, Enterprise logs, End-to-end email encryption

ACM Reference Format:
Mohamed Nabeel, Enes Altinisik, Haipei Sun, Issa Khalil, Hui (Wendy) Wang, and Ting Yu. 2021. CADUE: Content-Agnostic Detection of Unwanted Emails for Enterprise Security. In *24th International Symposium on Research in Attacks, Intrusions and Defenses (RAID '21), October 6–8, 2021, San Sebastian, Spain.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3471621.3471862

1 INTRODUCTION

End-to-end email encryption (E2EE) ensures that emails could only be decrypted and read by their intended recipients. As enterprises increasingly pay attention to security and privacy and comply with privacy regulations, E2EE's strong security guarantee is particularly desirable in the event of breaches: sensitive email contents would not be leaked even if attackers break into enterprise email servers. Several efforts have been made to design usable E2EE systems in practice. For example, notable progress has been made to deploy E2EE over Gmail using the browser extension called *Secure Mail for Gmail* [4]. Several email services now offer E2EE [7], including ProtonMail, Microsoft Outlook, Tutanota, MailFence, PreVeil, and HushMail, to name just a few.

Yet, despite its strong security and privacy guarantee, E2EE remains narrowly adopted in enterprises till today [5]. Besides interoperability and management issues, E2EE brings significant challenges for enterprises to detect and filter unwanted emails and to monitor suspicious traffic [6]. By unwanted emails, we refer to any email that is deemed as insecure, risky, or inappropriate as defined by the policy of an enterprise, which typically includes spams and phishing emails. Existing solutions to detect and filter unwanted emails rely heavily on email contents to identify distinguishing indicators, e.g., malicious URLs [22], suspicious text patterns [47], and display patterns resembling those from well-known organizations [10]. However, all such indicators would not be available under E2EE.

Indeed, intrusion detection over encrypted traffic (including encrypted emails) is a pressing issue that attracts much attention from both industry and academia. One major effort is to design novel encryption schemes that allow an enterprise to search keywords [13, 19, 27] or other patterns [15] over encrypted data. Though with great potential, such techniques have not been adopted

in practice yet due to their current limited search capabilities and high computation, communication, and deployment overheads.

In this paper, we investigate a different approach. We assume that email contents are not encrypted with searchable encryption, and thus email filtering systems could not gain any information from encrypted email contents. Our goal is to develop techniques for enterprises to detect unwanted emails in a *content-agnostic* way, i.e., without relying on email contents at all. This problem is inherently challenging from several aspects. First, email contents provide rich information, such as file attachments, URLs, and personal wording styles, that can be used to distinguish benign emails from unwanted ones. Prior works have shown that features derived from email contents lead to accurate detection of different types of unwanted emails [10, 22, 23, 34, 46–48]. It is perceivable that the effectiveness of existing approaches could degrade significantly without such content-based features. Second, although some existing content-agnostic detection solutions (e.g., [18, 20]) come up with features only based on email headers, these features are not resilient against evasion. In particular, as email headers are prone to forgery, relying on email header information alone is not robust when dealing with adversaries capable of crafting evasive email headers. In fact, some of the solutions (e.g., [18]) are only partially content-agnostic and still rely on features indirectly derived from email contents. Third, while sender reputation can help detect certain types of unwanted emails (e.g., spams) when their senders are of a bad reputation (e.g., the senders appear in IP blacklists [1]), maintaining a large scale server reputation service is expensive and tedious [20]. Outdated reputation systems make the detection of unwanted emails error-prone and evasible. Fourth, while common anti-spoofing techniques such as SPF [25] and DKIM [14] can help identify spoofed sender and forged emails, consistent with current adoption rates [9], we observe that many of the emails in our dataset do not have such protection. Hence, additional measures are necessary to identify unwanted emails.

Meanwhile, we note that many existing works (e.g., [22, 23, 46, 47]) treat each email independently, without considering the context under which emails are received or how they are related to other emails or email lists. Typically, the training process starts from a collection of benign and spam/phishing emails coming from diverse and often unrelated sources (e.g., reported by users from different organizations as well as from both working and personal accounts). This setting is different from the enterprise setting considered in this work. Instead of dealing with arbitrary emails, we focus on filtering incoming emails for a single enterprise. Unlike personal email accounts, where the types of engaged communications could be extremely broad and diverse, benign emails received by enterprise email accounts are inherently shaped by users' roles and their activities in an enterprise. For example, users in the same department or working on the same project tend to receive the same or similar emails, while it would be less likely for users who have never communicated with each other before. In other words, the communication patterns and relationships among internal user accounts contain rich and reliable information about benign email communications. Combining such communication patterns with the meta information of emails (headers and possibly subjects), it is possible to effectively detect unwanted emails without relying on email contents.

In this work, we design two types of novel enterprise features to capture important communication patterns. The first is called *sender profile features*, which record diverse properties of past emails from a sender. The intuition is that a benign sender's emails are driven by specific business purposes that would likely be quite different from unwanted emails. The second type is called *enterprise graph features*. These features are derived from two graphs constructed from internal emails (i.e., emails within an enterprise only). Unlike social network graphs [26, 43], these two graphs capture the communications among internal users with respect to their co-recipient relationship (internal users receiving the same emails) and their sender-recipient relationship (emails between internal users). Enterprise graph features indicate the legitimacy of an email based on its consistency with the structures of the two graphs.

We combine the above enterprise features with existing header and subject features in the literature and design a content-agnostic classifier for unwanted emails in an enterprise setting. We evaluate the accuracy of the classifier using real-world email logs collected from a local enterprise with over 1,500 employees. The email logs contain over 550K emails (including both external and internal ones) collected in a one-month period. We evaluate the performance of the classifier on the email logs under different settings to reflect potential restrictions in an enterprise setting (e.g., when subjects are also encrypted, and thus subject features are not available). We further conduct experiments to compare with closely related work in the literature.

Highlights of experimental results. Our experiments show that, when combining header features with our novel pioneering enterprise features, our approach results in a high performance classification model: with a low false positive rate of 0.3%, we achieve a 95.2% true positive rate. Further, we observe that, even with only enterprise features (i.e., without relying on email contents and email headers), our classifier still achieves commendable performance: with a 0.8% false positive rate, we could achieve a 95.7% true positive rate. This result is important as it shows that, even when header and subject features could be manipulated by attackers, our proposed enterprise features complement them well, and thus significantly improve the robustness of our approach.

Naturally, without access to email contents, our classifier could not achieve the same detection accuracy as existing content-based solutions, which in one aspect reflects the inherent challenge imposed by E2EE. Yet, our experimental results show that the proposed approach is promising in addressing this challenge and it is possible for enterprise to enjoy the strong privacy/security offered by E2EE while effectively detecting and filtering harmful emails.

2 RELATED WORK

The problem of detecting unwanted emails has been actively studied in both academia and industry. Quite a few approaches have been proposed over the years. We categorize these approaches into two types: (1) content-based techniques which rely on both email header and body features; and (2) content-agnostic techniques which do not rely on email contents.

2.1 Content-based Email Detection

Sheng *et al.* [42] extract links from an email's content along with the sender's address, and compare them with a blacklist to identify

phishing emails. Ho *et al.* [23] designs an anomaly detector to identify credential spear-phishing emails by analyzing historical email and system logs. They rely on the presence of URLs in the email content to identify spear-phishing emails. Stringhini *et al.* [44] designed a system named IDENTITYMAILER to detect spear-phishing emails sent by compromised accounts in an enterprise. Though, similar to our approach, IDENTITYMAILER utilizes features derived from an account's communication patterns, it still heavily relies on email content features to capture a user's writing habits, e.g., the frequent use of certain keywords or phrases, or the use of URLs. A follow-up work by Duman et al. [16] designs EmailProfiler to build a behavior profile for each email sender to identify spear-phishing emails at the recipient's side. However, their sender profiles heavily rely on stylometric features from email contents (199 features) in addition to metadata from email headers (23 features). As email contents are not accessible in our setting, these features cannot be derived from E2EE emails.

A plethora of techniques apply advanced machine learning to analyze email contents to detect spams and phishing emails. Among these approaches, NLP plays a significant role to extract semantic features [47], syntax features [34], and contextual features [48]. Vazhayil *et al.* [46] extract features from the email content, and train a few classifiers including decision trees, logistic regression, and SVM. Ho *et al.* [22] use random forest to detect URL-based lateral phishing emails. Almomani *et al.* [10] design a neural network based classifier to detect phishing emails. The classifier uses sixteen features; fifteen of them are derived from email contents such as whether the email contents include HTML and JavaScript, and whether the pictures are used in links. Since all the above techniques rely on email contents, they could not be applied when E2EE is deployed in an enterprise and email contents are not accessible for feature extraction.

2.2 Content-agnostic Email Detection

There have been a few efforts on detecting spam or phishing emails using content-agnostic approaches. Khamis *et al.* [24] train an SVM model to detect spam emails only utilizing header features. However, their features are not robust and the accuracy of the trained model is below 90%. SNARE [20] is a system to assess the reputation of email senders with only metadata of emails, e.g., timestamps, the IPs of the sender and receiver machines, message length and the number of recipients. SNARE is designed to be used by entities who have access to a large number of emails beyond the scope of a single enterprise, so that it can observe a large number of diverse spam emails to discover potential patterns. For example, the sender IP density features and spammer time-of-day features used in SNARE could be observed and effective when SNARE has access to a large number of spam emails from the same spammer network. In the enterprise setting considered in this paper, an enterprise could only observe a smaller portion of spam emails from spammers, which renders the above features not applicable. Similarly, since SNARE is not designed for the enterprise setting, it does not consider features about communication patterns between users. Meanwhile, simple origin based approaches such as IP blacklisting are not robust and hence are not reliable [38][50]. IPs are volatile identifiers. A large fraction of unsolicited emails come from botnets with diverse IP addresses [38]. Further, IPs may not represent the real email senders since they may be sent from IPs behind Network Address Translation appliances (NATs) or using proxy servers.

```
Return-Path: <alice@first.gov>
Received: from m15-35.first.gov (unknown [220.181.15.35]) by
↪  newmx32.second.com (NewMx) with SMTP id for
↪  <bob@second.com>; Mon, 16 Sep 2019 02:25:35 +0800
Received: from ( [155.246.151.34] ) by ajax-webmail-wmsvr35
↪  (Coremail) ; Mon, 16 Sep 2019 02:25:34 +0800 (CST)
X-Originating-IP: [155.246.151.34]
Date: Mon, 16 Sep 2019 02:25:34 +0800 (CST)
From: Alice <alice@first.gov>
To: Bob <bob@second.com>
CC: Carol <carol@third.edu>
Subject: Quarterly Security Report
X-Mailer: Coremail Webmail Server Version XT5.0.10 build
↪  20190724(ac680a23) Copyright (c) 2002-2019
MIME-Version: 1.0
Message-ID: <27d1c1ff.16d362c4@first.gov>

Content-Type: text/plain; charset=UTF-8
Please submit your quarterly security report at
https://example.url.net/ by today.
```

Figure 1: An example of raw emails with content

A recent work by Gascon *et al.* [18] adopts a similar setting to ours. They build sender profiles based on the character traits that the sender leaves in the header of benign emails, and later mark those emails that show deviations from the profile as spoofed. They profile all the senders to each recipient independently by having access to the inbox of that recipient. However, their approach targets solely at sender spoofing attacks. It cannot determine the status of emails from new senders who have no established profiles. Importantly, their approach is not completely content-agnostic because some of their features require access to the content, such as the `"text-quoted"` and the `"quoted-printable"` features. In Section 6.7, we experimentally compare our approach with theirs and provide more insights and details.

3 BACKGROUND AND APPROACH OVERVIEW

3.1 Email Structure and E2EE

SMTP (Simple Mail Transport Protocol) is the dominant protocol today to exchange emails between mail servers across organizational boundaries. An enterprise typically filters incoming emails over SMTP before they reach its internal email servers. Figure 1 shows the typical structure of an email per the SMTP specification, which is composed of two parts: the *header* and the *body* (i.e., the content). The header contains key information for email delivery, such as the email addresses and names of the sender and recipients, subjects, delivery dates, the communicating SMTP servers and their corresponding timestamps. The email content contains everything else in an email, including attachments. With the introduction of MIME extensions, besides unstructured text, email contents nowadays could also include non-textual information such as images, documents, and videos.

Since an email header contains vital information for mail delivery and needs to be updated when an email is delivered from one email server to another, email headers typically are not encrypted

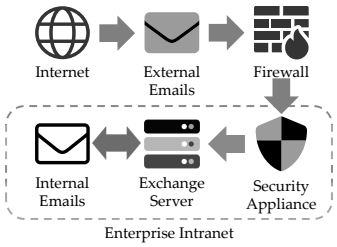


Figure 2: The architecture of enterprise email systems

in today's email encryption protocols and systems [17, 31, 37]. They largely focus on encryption of email contents, which motivates this work to investigate content-agnostic techniques to filter unwanted E2EE emails. Sometimes, to further enhance privacy, some commercial E2EE systems (e.g., ProtonMail and Tutanota) also support encrypting or stripping some header fields (for example, subjects, and the IPs and timestamps of the email servers involved in the delivery of an email) while still compliant with SMTP. The technique proposed in this paper could be easily adapted to different email encryption settings, as will be shown in our experiments (section 6).

3.2 Enterprise Email Systems

An enterprise email system handles emails from the external Internet as well as those inside the enterprise Intranet. All the emails from the external Internet go through a firewall before reaching the Intranet. Security applications are typically installed behind the firewall, and filter out unwanted emails so that only benign emails can reach the Intranet. The filtered emails are normally stored in log files. Most enterprises archive the email logs for a considerable period of time, ranging from six months to several years depending on legal regulations and its security policy. Figure 2 illustrates the architecture of a typical email system in an enterprise.

Typically, an enterprise has its own email domain name (e.g. paypal.com), and its users have their unique email addresses under the enterprise domain name (e.g. alice@paypal.com)[1]. An incoming email is either *external* if the sender's domain is different from that of the enterprise, or *internal* if its sender domain address is the same as that of the enterprise. All incoming external emails have to go through the firewall before they reach the Intranet, while internal emails are transmitted within the Intranet. In this paper, we assume that attackers are from outside the enterprise and focus on filtering incoming external emails. Meanwhile, we will leverage internal emails to design features about the communication pattern among internal users to help identify unwanted external emails.

3.3 Threat Model

In this work, we focus on an enterprise unwanted email threat model, where an adversary tries to trick enterprise users into performing an action, such as opening an email attachment, clicking a link, or leaking personal information, by sending one or more unwanted emails to them. Therefore, the unwanted emails include phishing, spear-phishing, spam, or other unsolicited emails deemed harmful by an enterprise. Our threat model assumes that there are one or more attackers *outside* an enterprise who send unwanted emails to the enterprise. During the period where the profiling is performed, we assume that sender email addresses as well as all internal emails are not spoofed. Thus, the enterprise features learned from the historical data reflect benign behaviors. Meanwhile, we assume that attackers could send emails with spoofed sender email addresses or sender IPs. In other words, an attacker may send unwanted emails in the name of legitimate external users who have interacted with the enterprise before. Our threat model assumes that the attacker cannot craft malicious emails that mimic the enterprise features of legitimate ones.

[1]Though an enterprise may have multiple email domain names, for simplicity and without loss of generality, in this work we assume each enterprise has only one email domain name.

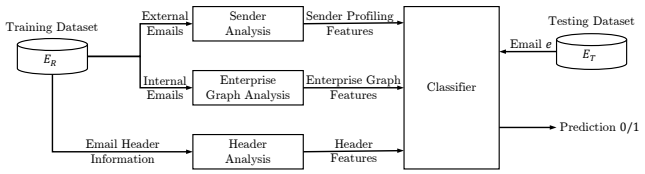


Figure 3: The architecture of CADUE

3.4 Overview of Our Approach

We design CADUE, a Content-Agnostic email Detection system for Unwanted Emails in the enterprise setting. CADUE takes a collection of emails E_R, which consists of a set of labeled external emails and a set of past internal emails of an enterprise as the input, and derives a set of features from E_R for a binary classifier $\mathcal{H}$. For any unlabeled external email e in the testing dataset E_T, $\mathcal{H}$ labels e as 1 if it is predicted as an unwanted email, or 0 otherwise. Our focus is not to design a new classification model for detection. Instead, our goal is to extract a set of content-agnostic features $\vec{x}$ from E_R, and build a state-of-the-art classifier $\mathcal{H}$ with existing classification models (e.g., random forest) to label E_T based on their features $\vec{x}$.

Specifically, we categorize the extracted features $\vec{x}$ into two types: (1) the *non-enterprise* features, which are extracted directly from the headers of the emails in E_R. These features only rely on an email itself; and (2) the *enterprise* features, which capture the external and internal communication patterns between employees of an enterprise. Enterprise features depend on not only the email itself but also historical external and internal emails.

To extract these features, CADUE builds the following three components (illustrated in Figure 3):

- *Header analysis*: This component analyzes the header information of the emails in E_R, and outputs header features. We use header features for our baseline classifier.
- *Sender analysis*: This component groups the *external* emails in E_R by their senders, and extracts the *sender profiling features* from the groups. In other words, sender profiling features capture the historical interaction of an external sender.
- *Enterprise graph analysis:* This component constructs enterprise communication graphs from the *internal* emails in E_R and uses these graphs to extract the *enterprise graph features*. As discussed before, though email contents are not accessible, an email's recipients could reveal the intention of the email, and thus could be useful to distinguish benign emails from unwanted one.

Some prior works [2, 28, 45, 49] have considered similar header features. However, our enterprise features are novel. To our best knowledge, only a few works in the literature focus on email filtering in an enterprise setting [18, 22, 23]. Most of the features in [23] and [22] rely on email contents (e.g., the URLs in the email). Their content-agnostic features are mainly derived from the `From` field in email headers. The problem setting in [18] is different from ours in that it targets solely at one type of unwanted emails – external spoofed emails, while our technique covers a broad range of emails deemed risky by enterprise policies. Furthermore, none of the features in the above works capture the communication patterns and relationships among internal users.

4 ENTERPRISE FEATURES

In this section, we discuss the details of enterprise features. Based on the sources from which these features are derived, we categorize them into two types:

- **Sender profiling features** that are extracted from the *external* emails. These features describe the patterns of how external senders communicate with internal users by emails.
- **Enterprise graph features** that are extracted from the *internal* enterprise emails. These features capture the email communication patterns of internal users.

4.1 Sender Profiling Features

Consider the training dataset E_R, and a new given email e, we first select three types of emails from E_R:

- $E_{SNDR} \subseteq E_R$ consists of all emails in E_R that only have the same sender as e;
- $E_{SNDR+SUB} \subseteq E_R$ includes all emails in E_R that have the same sender and subject as e; and
- $E_{SNDR+RCVR} \subseteq E_R$ contains all emails in E_R that have the same sender and recipient as e.

For each type of emails, we derive a set of features for the input email e. In total, we have 18 features (summarized in Table 1). We call these 18 features the *sender profiling* features.

Next, we discuss the details of these features as well as the intuition behind them.

Features derived from E_{SNDR}. Our first observation is that unwanted emails, especially spam emails, often behave in a distinctive pattern - they are often sent in a large number by the same sender during a short time duration. We design three features (Features 1 - 3 in Table 1) to catch this pattern. Feature 1 measures the average number of emails per day sent by the sender, and feature 2 counts the number of broadcast emails in the training data. Intuitively, a sender who sends out a large number of broadcast emails are more suspicious. Since the number of emails (for both Features 1 and 2) could be quite large for some senders, instead of returning the actual count k, we take the logarithmic scale of k and return a score s_k defined as:

$$s_k = \log(1 + k) \tag{1}$$

Feature 3 in Table 1 measures the average time interval between the consecutive emails received within a time window of T days in E_{SNDR}. To measure the average time interval, first we calculate the *averaged daily time interval*. Suppose there are n emails in E_{SNDR} on the i-th day $e_{i,1}, e_{i,2}, \ldots, e_{i,n}$ sorted by their timestamps. We use the function $t(e)$ to return the timestamp of an email e. Then the averaged daily time interval at the i-th day d_i is calculated as:

$$d_i = \begin{cases} \frac{1}{n-1} \sum_{j=2}^{n} t(e_{i,j}) - t(e_{i,j-1}), & \text{if } n > 1 \\ 86,400, & \text{if } n \in \{0, 1\} \end{cases} \tag{2}$$

In particular, when there is no email or only one email in a day, we set the time interval $d_i = 86,400$ (representing 86,400 seconds, i.e., 24 hours). Based on the average daily time interval, we calculate the average time interval s_{intv} as the average of d_i in past T days (from current date d_c to d_{c-T}), where T is a user-specified parameter for the time window size:

$$s_{intv} = \begin{cases} \frac{\sum_{i=1}^{T} d_{c-i} \cdot \mathbb{1}(d_{c-i} \neq 86400)}{\sum_{i=1}^{T} \mathbb{1}(d_{c-i} \neq 86400)}, & \text{if } \exists 1 \leq i \leq T, d_{c-i} \neq 86400 \\ 86400, & \text{otherwise} \end{cases} \tag{3}$$

To make the score representative, the value of T should not be too small. In our experiments, considering the number of days we have in the dataset, we use $T = 14$, i.e., a 2-week time window, as it leads to the best performance.

Besides the features derived from the frequency of emails, we also design a *reputation-based* feature, namely Feature 4 in Table 1, to measure the reputation of the sender based on its email history. This feature returns the number of unwanted emails in E_{SNDR} sent by the same sender of the input email e. Since the count can be large, we take its logarithmic scale (Equation 1).

Next, we design four features (Features 5 - 8 in Table 1) that measure the similarity between the input email and past benign emails from the same sender. Intuitively, the higher the similarity, the less likely the input email e be an unwanted one. Since email contents are not available, we measure email similarity based on four header fields:

$$HF = \{\texttt{user\_agent}, \texttt{path}, \texttt{message\_id}, \texttt{helo}\}$$

The `user_agent` field indicates the name and version of the sender's email client. The `path` field contains the IP addresses of all hops along the transmission path of the email. The `message_id` field contains a hash string as a prefix (possibly with delimiters between hash strings), an "@" character, and a domain address as the suffix. The hash string and the domain address are generated by the sender's email server. The `helo` field contains the domain address of the sender's email server. For each header field $f \in HF$, we derive a feature that measures the similarity between the input email e and the emails in E_{SNDR} on the field f by calculating the *similarity score* s_{sim}. Formally, let E^-_{SNDR} be the set of emails in E_{SNDR} that are labeled as benign (negative), and T_β be the collection of unique values of the field f. For each value $\gamma \in T_\beta$, we measure the similarity between the input email e and γ (at field f) as $sim(\hat{\gamma}, \gamma)$, where $\hat{\gamma}$ is the value of field f in the input email e. The function $sim()$ can be defined as any string similarity metric, e.g., Levenshtein distance and Jaccard similarity. We use Jaccard similarity in our experiments. Finally, we compute the similarity score s_{sim} between the input email e and E^-_{SNDR} on the field f as:

$$s_{sim} = \begin{cases} \max_{\gamma \in T_\beta} sim(\gamma, \hat{\gamma}) & \text{if } |T_\beta| > 0 \\ 0 & \text{if } |T_\beta| = 0 \end{cases} \tag{4}$$

Emails	No.	Feature	Domain	Description
E_{SNDR}	1	SENDER_NUM_EMAIL	$\mathbb{R}^+$	Average number of emails per day (at logarithmic scale).
	2	SENDER_NUM_BC	$\mathbb{R}^+$	Number of broadcast emails per day (at logarithmic scale).
	3	SENDER_TIME_INTV	$\mathbb{R}^+$	Average time interval between consecutive emails (Equation 3).
	4	SENDER_PAST_DISTRUST	$\mathbb{R}^+$	Number of unsolicited emails in the training data (at logarithmic scale).
	5	SENDER_SIM_UA	$[0, 1]$	Similarity to historical value(s) of `user_agent` field.
	6	SENDER_SIM_PATH	$[0, 1]$	Similarity to historical value(s) in `path` field.
	7	SENDER_SIM_MSGID	$[0, 1]$	Similarity to historical value(s) in `msg_id` field.
	8	SENDER_SIM_HELO	$[0, 1]$	Similarity to historical value(s) in `helo` field.
	9	SENDER_SIM_FIELDS	$[0, 1]$	Similarity to header fields of same sender emails.
	10	SENDER_EMAIL_SUBNET_FREQUENCY	$[0, 1]$	Frequency of emails per sender /24 subnets.
$E_{SNDR+SUB}$	11	EMAIL_IS_SBCAST	$\{0, 1\}$	Whether the email is a broadcasting email with one single recipient.
$E_{SNDR+RCVR}$	12	RECVER_NUM_EMAIL	$\mathbb{R}^+$	Average number of emails per day (at logarithmic scale).
	13	RECVER_NUM_BC	$\mathbb{R}^+$	Number of broadcast emails per day (at logarithmic scale).
	14	RECVER_TIME_INTV	$\mathbb{R}^+$	Average time interval between emails (Equation 3).
	15	RECVER_SIM_UA	$[0, 1]$	Similarity to historical value(s) in `user_agent` field.
	16	RECVER_SIM_PATH	$[0, 1]$	Similarity to historical value(s) in `path` field.
	17	RECVER_SIM_MSGID	$[0, 1]$	Similarity to historical value(s) in `msg_id` field.
	18	RECVER_SIM_HELO	$[0, 1]$	Similarity to historical value(s) in `helo` field.

Table 1: Sender profiling features (E_{SNDR}: all emails that have the same sender as the given email e; $E_{SNDR+SUB}$: all emails that have the same sender and subject as e; $E_{SNDR+RCVR}$: all emails that have the same sender and recipient as e)

We take the best similarity (i.e., maximum similarity) out of all comparisons because an email is highly likely to legitimate if it is similar to at least one benign email.

Feature 9, `SENDER_SIM_FIELDS`, measures the similarity of the header field structure between the input email e and the historical emails from the same sender E_{SNDR}. A binary vector is composed of the 44 binary Header features from the Table in the Appendix. Then, the L_1 distance is computed between e and each email e_i in E_{SNDR} as $L_1(e, e_i)$. Assuming n historical emails from a sender, Feature 9 is computed as the normalized average of the L_1 distances, $\frac{1}{n \times max(L(e,e_i))} \sum_{j=1}^{n} L_1(e_i, e)$. The higher the score, the higher the probability that e is benign.

Feature 10, `SENDER_EMAIL_SUBNET_FREQUENCY`, captures the network infrastructure diversity of the same senders using the /24 subnets. The intuition here is that benign senders have less diverse subnets compared to unwanted email senders who may utilize different subnets (e.g., using botnets) to deliver their unwanted emails. Assume the number of unique /24 subnets in E_{SNDR} is u and the number of emails is n, then, the sender score is computed as u/n. Feature 10 of email e is computed as the score of its sender. The lower the score, the higher the probability that e is benign.

Features derived from $E_{SNDR+SUB}$. From this set of emails, we design one feature (Feature 11 in Table 1). This feature captures an important pattern of unwanted emails when E2EE is deployed: when a sender sends an unwanted email to multiple recipients, it often sends an individual copy of the email to each recipient within a short time window, with each copy including the same subject and the same sender. Thus, Feature 12 is valued as a binary indicating whether there are more than two emails in $E_{SNDR+SUB}$ that satisfy the following two conditions: (1) each email has a single recipient; and (2) the emails are sent within α seconds, where $\alpha > 0$ is a user-specified threshold. In our experiments, we use $\alpha = 3,600$ (i.e., 1 hour) as it delivers the best performance.

Features derived from $E_{SNDR+RCVR}$. Even for the same sender, his/her communication pattern may vary for different recipients. Therefore, we focus on the pairs of senders and recipients, aiming to derive the communication patterns for each pair. To do this, we design seven features (Features 12 - 18 in Table 1) derived from $E_{SNDR+RCVR}$. Similar to features 1 - 10 derived from E_{SNDR}, these seven features capture the communication patterns of each sender-receiver pair. We omit the details of these seven features.

4.2 Enterprise Graph Features

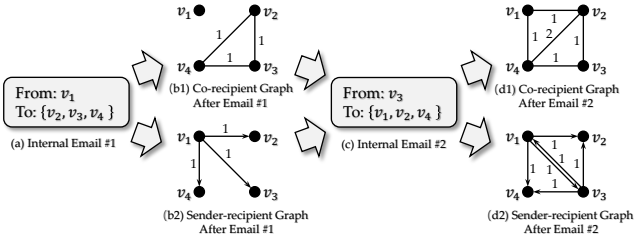


Figure 4: An example of the sender-recipient graph and co-recipient graph

Given an email e with sender s and recipients $R = \{r_1, \ldots, r_k\} (k > 1)$, intuitively, e is more likely to be unwanted if it is rare for all recipients in R to appear together in the recipient lists of past benign emails. We assume that all internal emails are benign, which is a reasonable assumption as internal emails only involve trusted senders (i.e., the employees). Based on this assumption, in the enterprise setting, how users are included in the same emails is not random. It is common that there exist implicit working communities, e.g., departments, working units, or project teams, in the enterprise setting. Such community structures will guide the email co-recipient relationship among users. Therefore, we derive a few features from the internal emails that can reflect such communities,

Graph	Cat.	No.	Feature	Domain	Description
Sender-recipient Graph G_{SR}	CB	1	SR_RANDOMWALK	[0, 1]	Random walk relation score among recipients in G_{SR}.
	CB	2	SR_TRANSCLOSURE	[0, 1]	Transitive closure relation score among recipients in G_{SR}.
	EI	3	SR_PAGERANK	[0, 1]	Average page rank score of recipients in G_{SR}.
Co-recipient Graph G_{CR}	CB	4	CR_RANDOMWALK	[0, 1]	Random walk relation score among recipients in G_{CR}.
	CB	5	CR_TRANSCLOSURE	[0, 1]	Transitive closure relation score among recipients in G_{CR}.
	EI	6	CR_PAGERANK	[0, 1]	Average page rank score of recipients in G_{CR}.

Table 2: Enterprise Graph Features (CB: community-based feature; EI: employee importance feature)

and thus capture the likelihood that all users in R can appear as the recipients in a benign email.

There are two types of email communication relationships among enterprise employees: *sender-recipient relationship*, i.e., one sends emails to another; and *co-recipient relationship*, i.e., users appear in the recipient lists of the same emails. Accordingly, we build two graphs, the *sender-recipient graph (G_{SR})* and the *co-recipient graph (G_{CR})* to model these two types of relationships respectively.

From these two graphs, we derive six enterprise graph features (shown in Table 2). We categorize these features into two types:

- *Community-based features*: In the enterprise setting, a benign email tends to be sent to users in the same working communities instead of those who belong to different communities. Furthermore, it is often the case that this community relationship is transitive. Based on these observations, we derive *community-based* features to reflect the degree that recipients in the input email belong to the same community according to either the sender-recipient relationship or the co-recipient relationship.
- *Employee-importance feature*: Email communications can reveal different levels of importance of employees. A benign email tends to be sent to the users of comparable importance. An email that addresses recipients of significantly different importance (e.g., CEO and interns) is unusual and more likely to be unwanted. Therefore, we derive *employee-importance* features from both the sender-recipient graph and the co-recipient graph.

We design three algorithms, namely *random walk* (RW) [35], *transitive closure* (TC) [32], and *page rank* (PR) [33], to derive the features from both graphs. All the three algorithms take the recipient list R of an email e as well as an enterprise graph (either G_{SR} or G_{CR}) as the input, and output a *relation score* in the range $[0, 1]$ to quantify the likelihood that the recipients in R appear in the recipient list of a benign email, based on their prior communications (either sender-recipient or co-recipient) within the enterprise. Among these three algorithms, RW and TC extract the community-based features [36], while PR extracts the employee-importance based features. Next, we describe how to construct the sender-recipient graph and the co-recipient graph, and then explain how to extract features from these two graphs.

Graph construction. First, we construct the *sender-recipient graph* $G_{SR} = \{V, E_{SR}\}$. In G_{SR}, each vertex $v \in V$ represents an employee in the enterprise. There is a directed edge $e(v, v') \in E_{SR}$ if user v sends an email to user v'. The weight w of the edge $e(v, v')$ is set as the total number of emails that user v sends to v'. Figure 4 (b2) shows an example of the sender-recipient graph constructed from an internal email illustrated in Figure 4 (a). Figure 4 (d2) further shows how the graph is updated when a new internal email comes (illustrated in Figure 4 (c)).

We further construct a *co-recipient graph* $G_{CR} = \{V, E_{CR}\}$, in which each vertex $v \in V$ corresponds to an employee in the enterprise. There is an undirected edge $e(v, v') \in E_{CR}$ if v and v' appear together in the recipient list of any internal email. The weight on each edge $e(v, v')$ is the total number of internal emails that include both v and v' in the recipient list. Intuitively, the co-recipient graph captures the fact that, if two employees are frequently included in the recipient list of the same email, they are more likely to belong to the same working community (e.g., the same department or the same project team). Figure 4 (b1) illustrates the co-recipient graph constructed from the email in Figure 4 (a), and how the graph is updated (Figure 4 (d1)) when having a new internal email (illustrated in Figure 4 (c)).

Extraction of Community-based Features. We design two scoring schemes, namely *random walk* (RW) and *transitive closure* (TC), to measure the degree that all the recipients R in the input email e belong to the same community, according to either the sender-recipient relationship or the co-recipient relationship. To do this, both RW and TC first measure the pairwise relationship score for each pair of users v, v' in the recipient list R, which quantifies the likelihood that v and v' appear in the same email considering their past communications with other employees. Then both RW and TC return a global relationship score of R that is *aggregated* from all the pairwise relationship scores. RW and TC differ on how they measure the pairwise relationship scores, as TC requires and utilizes the transitive property of the graph but RW does not. Next, we explain these two algorithms in detail.

Random walk (RW) scoring scheme: We do random walk with 100,000 steps on G_{SR} and G_{CR} respectively, and obtain two features (Features 1 and 4 in Table 2). Specifically, RW takes a graph G (either G_{SR} or S_{CR}), and a source node $v_i \in G$ as the input. Starting from v_i, we traverse randomly to one of its neighbors v_j, with probability proportional to the weight of the edge $e(v_i, v_j)$. We continue the traversal and finally terminate after L steps, where L is a large number specified by the users. A node may be visited multiple times during the traversal. We use $M[i, j]$ to denote the number of times v_j is reached in a random walk starting from v_i. Note that this relationship is not symmetric, i.e., $M[i, j]$ and $M[j, i]$ may not be equal.

Given an email e and its recipient list R, we compute the *pairwise co-recipient score* $s_{i,j}$ as:

$$s_{i,j} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } v_i \text{ is isolated in } G \text{ and } i \neq j \\ \frac{M[i,j]}{\max_{v_k \in R} M[i,k]} & \text{otherwise.} \end{cases} \quad (5)$$

We compute $\frac{M[i,j]}{\max_{v_k \in R} M[i,k]}$ to make $s_{i,j}$ fall into the range $[0,1]$. Finally, we compute the co-recipient score of the recipient list R by random walk $s_{rw}(R)$ as

$$s_{rw}(R) = \min_{v_i, v_j \in R} s_{i,j} \quad (6)$$

We consider the minimum pairwise co-recipient score as the score for the whole recipient list as it captures the worst likelihood that the whole recipient list is included in a benign email. Intuitively, the lower the score is, the smaller the likelihood that the recipients in R all appear in the same recipient list of a benign email.

Transitive closure (TC) scoring scheme. We compute transitive closure over G_{SR} and G_{CR} respectively, and derive two features (Features 2 and 5 in Table 2). Similar to RW, TC also computes the pairwise co-recipient scores for each pair of recipients in R. It then derives the co-recipient score of all users in R by aggregating all pairwise co-recipient scores.

Specifically, let A be the adjacency matrix of the input graph G (either G_{SR} or G_{CR}). A is normalized with $A[i,j] = \frac{w(i,j)}{\sum_{t=1}^{m} w(i,t)}$, where m is the total number of nodes in the graph, and $w(i,j)$ is the weight of edge $e(i,j)$ if the edge exists, or 0 otherwise. The transitive closure of A is given by $A^+ = \sum_{i=1}^{m-1} A^i$, where

$$A^i = \underbrace{A \times A \ldots A}_{i}$$

We then compute the pairwise co-recipient score $s_{i,j}$ from A^+ by Equation 5 (replacing $M[i,j]$ by $A^+[i,j]$). Finally, we compute the score $s_{tc}(R)$ based on co-recipient scores between users in the recipient list R. Similar to RW, we consider the minimum pairwise co-recipient score as the score for the whole recipient list. Formally, the score $s_{tc}(R)$ is measured as: $s_{tc}(R) = \min_{v_i, v_j \in R} s_{i,j}$.

Extraction of Employee-Importance Features. While RW and TC are driven by underlying user communities, we design another scoring scheme named page rank (PR) to derive two features (Features 3 and 6 in Table 2). PR does not rely on the assumption of existence of communities to measure the co-recipient likelihood of the users.

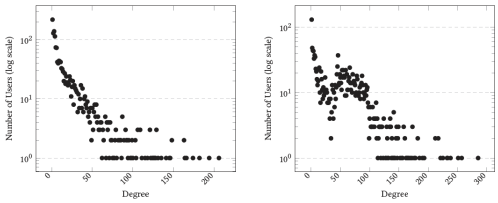
Page rank (PR) scoring scheme. Our PR algorithm applies the standard page rank algorithm [33], with the damping parameter of 0.85, on the input graph G (either G_{SR} or G_{CR}) to obtain a score IS_i for each node v_i. The score reflects the importance of user i based on past internal email communications. We then compute the co-recipient score $s_{pr}(R)$ of all the users in the recipient list R as

$$s_{pr}(R) = \min_{v_i \in R} IS_i, \quad (7)$$

We only measure the lowest PR of the recipients based on the intuition that the lower the PR of a user, the less likely he/she receives emails together with other users in R. We omit the pseudo code of PR due to its simplicity. While we rely on well-known graph algorithms, it is worthwhile noting that linking multiple enterprise logs to create the G_{SR} and G_{CR} and derive discriminating features from them are novel.

5 HEADER FEATURES

Besides the enterprise features, from prior work [2, 28], we derive 47 features from the email header. We use these features as the baseline content-agnostic approach. They are categorized into three groups:



(a) Sender-recipient Graph (b) Co-recipient Graph

Figure 5: Node degree distribution of enterprise graphs

- *SUBJECT (SUBJ)* features: derived from the `subject` field in the email header.
- *NON-SUBJECT (NS)* features: derived from all fields in the header except the `subject`.
- *DEPENDENCY (DEP)* features: cross-field derived features, specifically, checking consistency between some of the header fields.

All SUBJ and NS features are adapted from the features used by Apache SpamAssassin [2], one of the most popular open-source anti-spam platforms. DEP features are adapted from [2] and [28]. We include the details of all the header features in the Appendix.

6 EVALUATION

6.1 Dataset: Statistics and Characteristics

Our experiments are conducted on a real-world dataset that consists of the email logs collected from a local enterprise in 25 days. The email logs include more than 550,000 emails, which include both external emails of 3,963 users outside of the enterprise and internal emails among 1,546 enterprise employees. The email logs include the following information: (i) 587,873 internal email communications from exchange server logs, which are used to build the sender-recipient graph and the co-recipient graph, and (ii) a ground truth of emails from external senders, which consists of 15,399 benign email communications and 911 unwanted email communications. The ground truth is obtained from the security operation analysts in the enterprise, who have full access to email contents along with other information to filter unwanted emails. We treat those emails filtered by the security analysts as true positives and those passed as true negatives. Table 3 includes the details of the dataset. Due to security and privacy reasons, before giving us access to the data, the enterprise removed all content of emails and anonymized sensitive header fields with a keyed hash while preserving the uniqueness.

# of Int. Emails (Benign)	# of Users		# of Incoming Ext. Emails (Ground Truth)	
587,873	5,509		16310	
	Internal	External	Benign	Unwanted
	1,546	3,963	15,399	911

Table 3: Statistical information of email dataset (Int. for internal, Ext. for external)

	# of nodes	# of edges	Avg. node degree
G_{SR}	1,678	17,082	20.2598
G_{CR}	1,722	42,318	49.1498

Table 4: Statistical Information of Communication Graphs

Communication graph characteristics. The key statistics of G_{SR} and G_{CR} is shown in Table 4. of each node The degree distributions of G_{SR} or G_{CR} are shown in Figure 5, which confirms our assumption that both G_{SR} or G_{CR} exhibit social graph characteristics. The node degree distributions of both graphs are close to power-law, which characterizes social graphs.

Community analysis. We use *Infomap* [41], an algorithm for community discovery on social graphs, to analyze the communities in G_{SR} or G_{CR}. Infomap discovered 209 and 87 communities in G_{SR} or G_{CR} respectively. Then we count the number of communities covered by the recipients for each unwanted email. Our analysis shows that most of the recipients of benign emails cover less than 4 communities in both G_{SR} or G_{CR}. Meanwhile, we observe that the recipients of unwanted emails tend to cover more communities than those of benign ones in both graphs. We observe from our dataset that unwanted emails with 4 or 5 receipts are on average delivered to 3.125 communities whereas the emails with similar number of recipients are delivered to 1.84 communities. This supports our intuition that unwanted emails are sent to multiple user communities whereas benign ones are sent to a fewer communities. With the above sample, we also measure the variations in the importance of the co-recipients in benign and unwanted emails. We observe that, on average, the standard deviation of the co-recipient importance in unwanted emails is 3.72 times higher than that in benign emails. This observation reinforces our intuition that the emails addressing recipients of significantly different importance are more likely to be unwanted.

6.2 Selection of Machine Learning Models

Model	Acc.	TPR	FPR
SVM	0.968	0.969	0.025
Logistic Regression	0.968	0.925	0.034
Random Forest	**0.974**	**0.952**	**0.003**
Neural Networks	0.957	0.925	0.011

Table 5: Performance of four machine learning models

Setup of training and testing data. We temporally split the 25-day ground truth data into training and testing datasets. The training dataset consists of the first 18 days, while the testing data set consists of the last 7 days. The training dataset includes 587,873 internal emails and 11,603 external emails (11,084 benign and 519 unwanted). The testing dataset includes 4,707 external emails (4,315 benign and 392 unwanted). At the collection point of the internal emails, emails for mailing lists are already exploded to individual emails and mailing list relationships are indirectly captured by enterprise graph features. Note that internal mail server is configured such that external users are not allowed to send emails to enterprise mailing lists. The training and testing data have similar distributions of benign and unwanted emails (91.67% benign email in the training data and 95.52% benign emails in the testing data).

Machine learning models. We experimented with four different machine learning models: SVM, Logistic Regression, Random Forest and fully connected Deep Neural Networks. The performance results with a balanced dataset (with discriminate threshold 0.5) are depicted in Table 5. The results show that Random Forest has the best performance among all the four models, and hence is selected to be the classifier for our approach in all experiments. To avoid overfitting in the Random Forest model, we generate 500 decision trees with features randomly picked with replacement on each tree, and limited the decision tree depth to 20 [12].

Baseline approach. We use as a baseline a Random Forest classifier that uses all the header features (Table 10 in Appendix). This is corresponding to the approach that applies existing content-based approaches over E2EE. Clearly all content-based features could not be utilized, yet all the header features would still be available. We experimentally evaluate how this approach compares with our approach and what benefits enterprise features bring.

6.3 Feature Importance

We divide all the features into four categories: (1) Sender profiling features (Table 1); (2) Enterprise graph features (Table 2); (3) Non-subject features: the header features from SpamAssassin [2] (with both content and subject features excluded); and (4) Subject features: the header features from SpamAssassin including subject features (with only content features excluded). We train and test the classifier using 5-fold cross-validation with balanced datasets, and calculate the average importance [21] of each feature. Figure 6 depicts the importance scores of the top-30 important features. All the top 4 features are enterprise features. Additionally, one of the graph enterprise features is the third most important feature, whose score is significantly higher than the next more important one. This clearly demonstrates the important role that enterprise features play in the detection of unwanted emails.

The rank and score of the top-3 features per feature category is shown in Table 6. In the category of sender profiling features, the `SENDER_SIM_FIELDS` feature is the most important. This reveals that the structural composition of email headers is very effective in distinguishing benign and unwanted emails. The second important feature, namely `SENDER_EMAIL_SUBNET_FREQUENCY` feature indicates that adversaries tend to use diverse infrastructures to deliver unwanted emails. Such agile tactics are usually used to evade detection and blocking based on infrastructure reputation [20]. Finally, the high rank of the `SENDER_SIM_UA` feature indicates that the unwanted emails are likely to utilize similar email agents.

In the category of enterprise graph features, the top-3 features (`CR_RANDOMWALK`, `SR_RANDOMWALK`, and `CR_TRANSCLOSURE`) are all among the top-15 most important features, with the first being among the top 3. This shows that enterprise graph features are important for the detection of unwanted emails. We note that two of the features are generated from the co-recipient graph. This indicates that the co-recipient relationship is more important than the sender-recipient relationship for the detection of unwanted emails.

We further take a closer look at the performance of the six enterprise graph features. Among these six enterprise graph features, `CR_RANDOMWALK` has the highest importance score (0.1223), and `CR_PAGERANK` has the lowest score (0.0019), which is much lower than that of `CR_RANDOMWALK`. We also group the six features by the algorithms (RW, TC, PR) that they are generated from, and compute the average importance score of features in each group. It turns out that the features generated by RW have the highest average importance score (0.0691). TC ranks the second (average score 0.0069), and PR is the last (average score 0.0019). This demonstrates that the community-based features are more important than the employee-based features.

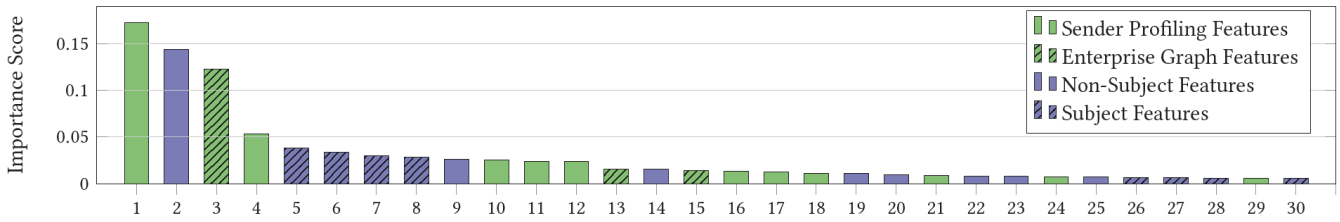


Figure 6: Importance of top-30 features

Category	Name	Ranking	Score	Category	Name	Ranking	Score
Sender Profiling	SENDER_SIM_FIELDS	1	0.173	Non-Subject	NS_FROM_MIXED	2	0.144
Sender Profiling	SENDER_EMAIL_SUBNET_FREQUENCY	4	0.054	Non-Subject	NS_DATE_INVALID	9	0.026
Sender Profiling	SENDER_SIM_UA	10	0.024	Non-Subject	NS_TO_SORTED	14	0.015
Category	**Name**	**Ranking**	**Score**	**Category**	**Name**	**Ranking**	**Score**
Enterprise Graph	CR_RANDOMWALK	3	0.122	Subject	SUBJ_CAPS_PERCENTAGE	5	0.038
Enterprise Graph	SR_RANDOMWALK	13	0.016	Subject	SUBJ_FREE	6	0.034
Enterprise Graph	CR_TRANSCLOSURE	15	0.014	Subject	SUBJ_GUARANTEED	7	0.030

Table 6: Top-3 important features in each category

Recall that all the features in thenon-subject and subject categories are adapted from those of the popular SpamAssassin [2] email filtering system and a previous work [28]. In the non-subject category, NS_FROM_MIXED is the second-most important feature, which reveals that a considerable proportion of unwanted email senders use automatically generated email addresses with numbers appended to compose new addresses. The NS_DATE_INVALID and NS_TO_SORTED features are crafted to detect inconsistencies among the format of the sending timestamp and the sorting of multiple recipients in the to field. However, it is important to note here that these features are not robust as a careful adversary can easily avoid such inconsistencies to evade detection.

Subject features are also highly ranked. SUBJ_CAPS_PERCENTAGE, SUBJ_FREE, and SUBJ_GUARANTEED are ranked 5, 6, and 7, respectively. The features indicate that the unwanted emails tend to use capital letters, use the words "guaranteed" and "free" in email subjects to draw attention. As we will see in Section 6, the high ranks of these features explain the sharp drop in performance for the classifiers that solely rely on header features, when subject features become unavailable under E2EE settings.

6.4 Prediction Accuracy

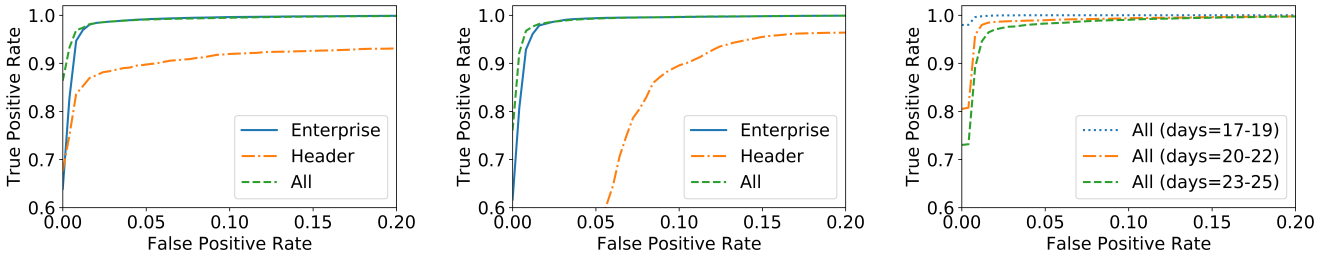
We experimentally evaluate the prediction accuracy of our proposed classifiers. As mentioned earlier, most end-to-end email encryption solutions by default encrypt email contents but leave email headers and subjects unencrypted [8]. We call this setting the *regular setting*. Some commercial solutions also offer capabilities to encrypt or strip certain fields in headers for better privacy. One obvious option is to encrypt email subjects, which many existing solutions support [29, 30, 37]. Sometimes the IPs and timestamps of email servers involved in email delivery are also encrypted or stripped (e.g., ProtonMail and Tutanota). In this experiment, we also consider this *enhanced setting*, in particular, assuming subjects and IPs and timestamps of email delivery paths are encrypted. Clearly, in the enhanced setting, all the subject features cannot be derived. Further, some of the sender profiling features (No.6 and 16 in Table 1) could not be built either. We evaluate the prediction accuracy of our classifiers under both settings.

Specifically, for each setting, we train 3 different Random Forest classifiers, namely:

- **Enterprise** classifier, which is trained with enterprise social graph features and sender profiling features alone. Note that in the enhanced setting, all the features related to IP addresses (namely No. 6 and 16 in Table 1) are disabled;
- **Header** classifier, which is trained with all the header features alone and considered as the baseline. In the enhanced setting, all features in subject category in Table 10 are disabled;
- **All** classifier, which is trained with all the above features.

For each setting, two evaluation metrics are computed: (i) the true positive rate (TPR), which represents the ratio of correctly identified unwanted emails to the total number of unwanted emails, and (ii) the false positive rate (FPR), which represents the total number of benign emails mistakenly identified as unwanted to the total number of benign emails. The relationship between TPR and FPR for various discriminate thresholds is tracked using ROC curves. The area under the ROC (AUC) is an indicator of the quality of the classifier; the higher the AUC, the better the performance.

Regular setting. Figure 7 (a) shows the zoomed-in version of the ROC curves of the different classifiers in the regular setting. Each ROC curve is generated by varying the discriminate threshold on the corresponding classifier output, with each discriminate threshold averaged over 100 runs. The figure shows relatively poor performance of the **Header** baseline classifier (AUC = 0.930). For example, when the FPR = 0.6%, the TPR = 78.1%. On the other hand, the performance of the **Enterprise** classifier is much better than that of the **Header** classifier (AUC = 0.984). For example, when the FPR = 0.6%, the TPR = 87.6%. This clearly shows the effectiveness of enterprise features in detecting unwanted emails. When the header features are integrated with the enterprise features, the performance is further enhanced as shown by the ROC curve of the



(a) ROC curves for regular setting

(b) ROC curves for enhanced setting

(c) **All** regular ROC curves for different epochs

Figure 7: Prediction Accuracy of CADUE

All classifier (AUC = 0.984). For example, the TPR = 96.7% when the FPR = 0.6%.

Enhanced setting. The prediction accuracy of the three classifiers under the enhanced setting is shown in Figure 7(b). The figure shows that the **Header** classifier has a significant performance drop due to the masked subject features (AUC 0.910). For example, when the FPR = 0.6%, the TPR = 5.3 %; and to achieve TPR of 95.1%, the FPR soars to 13.2%. This indicates that the prediction accuracy of the **Header** classifier heavily relies on the subject features which become unavailable in the enhanced settings. Meanwhile, the **Enterprise** classifier achieves a similar performance as that under the regular setting (AUC 0.982). For example, when the FPR = 0.6%, the TPR = 85.8%), as it does not rely on subjects. Even though some of the sender profiling features could not be built due to the unavailability of delivery path information in the enhanced setting, they are complemented well by other robust enterprise features. For the **All** classifier, it is particularly interesting to see that, even though the effectiveness of header features is significantly reduced in the enhanced setting, the prediction accuracy of the combined features is still close to that under the regular setting (AUC = 0.984), due to the stable performance of enterprise features. For example, with FPR = 0.6%, the TPR = 95.2%.

We further zoom-in to a particular setting where the discriminate threshold is 0.5, and report the prediction performance of all classifiers under this setting in Table 7. Among the three classifiers, the **All** classifier always achieves the best prediction accuracy.

Setting	Classifier	TPR	FPR	AUC
Regular	Enterprise	0.957	0.008	0.974
	Header (baseline)	0.888	0.035	0.926
	All	0.952	0.003	0.974
Enhanced	Enterprise	0.954	0.010	0.972
	Header (baseline)	0.893	0.100	0.896
	All	0.953	0.005	0.974

Table 7: Prediction performance of different classifiers (discriminate threshold = 0.5)

With only header and enterprise features but not contents, the accuracy of CADUE is a bit lower than that of the existing techniques that utilize content features, which reflects the inherent challenge that E2EE imposes on unwanted email detection. In practice, CADUE can be tuned for appropriate tradeoff between true positives and false positives, depending on the use case. For example, under the regular setting, with a false positive rate of 0.02%, CADUE could still achieve close to 87.3% true positive rate. Even under the more restrictive enhanced setting, with the same low false positive rate, 78.4% true positive rate could be achieved.

6.5 Concept Drift

Class	Training (1-14)	Ep. 17-19	Ep. 20-22	Ep. 23-25
Benign	7499	2217	2182	1123
Unwanted	298	211	171	169

Table 8: Training and epoch test sets

Concept drift refers to the phenomenon that the data tends to change over time in unpredictable manner. As a result, predictions of the models trained in the past may become less accurate as time passes. Similar to many cybersecurity problems, due to concept drift, CADUE is likely to experience performance dropping when the trained model becomes older. To evaluate the concept drift, as shown in Figure 8, we re-split our 25-day ground truth data into a training dataset and 3-day epoch testing sets. The training dataset includes the first 14 days (day one through day 14), the first testing epoch includes days 17-19, the second testing epoch includes days 20-22, and the third testing epoch includes days 23-25. Figure 7 (c) shows the performance of the **All** classifier (in the regular setting) using the three testing epochs, with each data point averaged over 100 runs. As expected, the figure shows that the performance degenerates as we move further from the training window. However, the degeneration in performance is marginal as shown by the figure and the particular settings depicted in Table 9 (when the discriminate threshold is 0.5). This indicates that CADUE continues to perform well even after 10 days from its training time. Unfortunately, as we only have access to the email logs for a 25-day period, we could not conduct thorough experiments to study this problem over longer time windows and hence determine the appropriate rate for retraining to deal with more significant concept drifting, including major changes to organizational structures. The **All** classifier under the enhanced setting has a similar trend to that under the regular setting, the detail of which is thus omitted due to space limit.

Epoch	Setting	TPR	FPR	AUC
17-19	Regular	0.987	0.003	0.992
	Enhanced	0.977	0.003	0.987
20-22	Regular	0.965	0.005	0.980
	Enhanced	0.964	0.004	0.980
23-25	Regular	0.913	0.006	0.953
	Enhanced	0.919	0.007	0.956

Table 9: Prediction performance of different epoch test sets with all features (discriminate threshold = 0.5)

6.6 Emails from "Fresh" Senders

Existing profiling based approaches could not handle emails from senders never seen before, as these senders' profile features do not exist (e.g., [18]). In ours, if the email from the new sender is sent to multiple internal users in the enterprise, we could still rely on the enterprise graph features to detect it as unwanted or not. Even if a spammer may change his email address frequently, through enterprise graph features, we could still observe the unusual recipient features and effectively detect unwanted emails. On the other hand, if fresh emails are sent to a usual group of recipients, they are more likely to be marked as benign. When the new sender's email is only sent to a single recipient, while the enterprise graph features are not applicable, our approach can still detect unwanted emails using Header features. We breakdown the classification results considering only fresh emails. Our model achieves a high accuracy of 97.84% with a FPR of 2.0% for fresh emails.

6.7 Comparison with Gascon *et al.*

As mentioned before, the approach proposed by Gascon *et al.* [18] targets at spearphishing emails which pretends to be sent by previous known senders. In practice, spearphishing emails are rare and only account for a very smaller portion of unwanted emails. In fact, none of the senders of the unwanted emails in our dataset have ever sent benign emails before. Hence, Gascon *et al.* 's approach will not detect any of the unwanted emails in our dataset.

We evaluate the FPs of Gascon *et al.* 's approach over our benign dataset. Following the procedure stated in [18], we build a new benign email dataset that comprises the emails of each sender who has more than 5 emails (10510 emails from 495 senders). For each sender, we use the first 5 emails to build its profile, and use the remaining emails for testing. We follow the implementation of [18], after excluding those features that would not be available under E2EE (e.g., `"text-quoted"` and `"quoted-printable"`, which rely on email contents). If an email in the testing set is correctly mapped to its sender profile, we count it as a TN; otherwise, we count it as a FP. For the experiment, we got a TNR of 87.1% and a FPR of 12.9%. These results support our earlier observation that Gascon *et al.* 's approach is not a complete content-agnostic approach, and its performance could be greatly affected when the content-dependent features are unavailable.

7 DISCUSSION AND LIMITATIONS

Feature Robustness. Recall that our approach uses header features, sender profiling features, and enterprise graph features. All the header features are under the control of the email sender, and hence can be manipulated by adversaries to evade filtering [11]. Furthermore, header features from the subject group (Table 10) may not be available if the subject is encrypted. On the other hand, without prior infiltration (as stated in our attack model), the sender profiling features and the enterprise features are out of control of the adversary, and hence cannot be easily manipulated. As the experiments show above, our approach performs well when trained with only sender profiling features and enterprise graph features, and hence is robust against adversarial manipulation of subject features. Furthermore, most existing email filtering solutions rely on email contents and header information, both of which are under the control of the adversaries, and hence could be carefully crafted to evade filtering. Our approach not only has the capability to filter emails without having access to content or header information but, more importantly, is more robust against adversarial manipulation.

Meanwhile, attackers who have compromised the enterprise email server or learned about enterprise users relationships through external sources such as social media networks may mimic some of the enterprise features in order to evade detection. We leave the problem of detecting unwanted emails with attackers having partial knowledge on the enterprise graphs as a future direction.

Updates of enterprise graph. Due to the dynamics of emails, the structure of the enterprise graph may change over time. Such change arises the need for model retraining, which is in fact quite efficient to perform using our automated pipeline compared to deep learning models. One may further improve the performance by incrementally updating the enterprise graphs instead of building from scratch each time. In order to gain the optimal classifier performance without incurring heavy computational cost, we recommend retraining the classifier regularly with a moving window of training data. We leave the problem of identifying the optimal window for training as future work since it requires a larger dataset.

Fresh Senders. We show in Section 6.6, our approach can detect unwanted emails from fresh senders at a relatively high performance metrics as our enterprise features help distinguish those send emails to more than one recipient by using the features derived from the co-recipient relationship. However, if the fresh sender emails are either sent to only one recipient or well crafted to avoid header features, our model may not be able to detect those unwanted emails with high performance. In a real-world deployment, we recommend to exercise a more conservative decision making (e.g. mark suspicious) for fresh emails under the E2EE setting. As additional emails are received from the same sender, one may obtain more confident decisions from our detection model.

8 CONCLUSION

In this paper, we investigate techniques for enterprise to detect and filter unwanted incoming emails when email contents are not accessible due to the adoption of end-to-end email encryption. Besides email header features, we propose a set of novel enterprise features, including sender profiling features and enterprise graph features, based on the observation that historical email communication patterns in an enterprise contain rich and distinguishing information that could help tell unwanted emails from legitimate ones. We then design a content-agnostic classifier that combines enterprise features with traditional header features. Through extensive experiments over a real email dataset collected from a large local enterprise, we show that our classifier achieves high detection accuracy without utilizing any email content information.

Recently the research community pays increasing attention to analyzing encrypted network traffic for a variety of purposes, including classifying web access under anonymous communication, and identifying network/mobile applications under https. In the context of enterprise email filtering, built on top of our current techniques, it would be interesting future work to investigate whether certain features could be extracted from encrypted email contents to further improve the accuracy of email filtering. The challenge is that existing research on analyzing encrypted traffic mostly targets applications/protocols with clear traffic patterns, even when they are encrypted. Email contents, on the other hand, are unstructured.

REFERENCES

[1] [n.d.]. SpamHaus IP Blocklist. http://www.spamhaus.org.
[2] 2003. SpamAssassin. https://spamassassin.apache.org/.
[3] 2003. SpamAssassin Tests Performed. https://bit.ly/3gz1nm5.
[4] 2017. https://techconnecto.com/end-to-end-encryption-gmail/.
[5] 2017. https://blog.mailfence.com/end-to-end-email-encryption/.
[6] 2017. E2EE Emails. https://bit.ly/3qKvz2s.
[7] 2020. https://gdpr.eu/email-encryption/.
[8] 2020. OpenPGP. https://www.openpgp.org/.
[9] 2021. DMARC Adoption Rate. https://www.dmarc360.com/.
[10] A. Almomani, T. Wan, A. Altaher, A. Manasrah, E. ALmomani, M. Anbar, E. ALomari, and S. Ramadass. 2012. Evolving fuzzy neural network for phishing emails detection. *JSC* 8, 7 (2012), 1099.
[11] B. Barnes. 2002. E-Mail Impersonators. https://bit.ly/3gw9sIq.
[12] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.
[13] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou. 2014. Privacy-Preserving Multi-Keyword Ranked Search over Encrypted Cloud Data. *IEEE TPDS* 25, 1 (2014), 222–233.
[14] T. Hansen D. Crocker and M. Kucheraw. 2020. DomainKeys Identified Mail (DKIM) Signatures. https://tools.ietf.org/html/rfc6376.
[15] Nicolas Desmoulins, Pierre-Alain Fouque, Cristina Onete, and Olivier Sanders. 2018. Pattern Matching on Encrypted Streams. In *Advances in Cryptology – ASIACRYPT 2018*, Thomas Peyrin and Steven Galbraith (Eds.). Springer International Publishing, Cham, 121–148.
[16] Sevtap Duman, Kubra Kalkan-Cakmakci, Manuel Egele, William K. Robertson, and Engin Kirda. 2016. EmailProfiler: Spearphishing Filtering with Header and Stylometric Features of Emails. In *40th IEEE Annual Computer Software and Applications Conference, COMPSAC 2016, Atlanta, GA, USA, June 10-14, 2016*. 408–416.
[17] Simon Garfinkel. 1995. *Pretty Good Privacy*.
[18] H. Gascon, S. Ullrich, B. Stritter, and K. Rieck. 2018. Reading Between the Lines: Content-Agnostic Detection of Spear-Phishing Emails. In *Research in Attacks, Intrusions, and Defenses*. Cham, 69–91.
[19] P. Golle, J. Staddon, and B. Waters. 2004. Secure Conjunctive Keyword Search over Encrypted Data. In *Applied Cryptography and Network Security*. Berlin, Heidelberg, 31–45.
[20] Shuang Hao, Nadeem Ahmed Syed, Nick Feamster, Alexander G. Gray, and Sven Krasser. 2009. Detecting Spammers with SNARE: Spatio-Temporal Network-Level Automatic Reputation Engine. In *Proceedings of the 18th Conference on USENIX Security Symposium* (Montreal, Canada) *(SSYM'09)*. USENIX Association, USA, 101–118.
[21] T. Hastie, R. Tibshirani, and J. Friedman. 2009. *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media. 367–368 pages.
[22] Grant Ho, Asaf Cidon, Lior Gavish, Marco Schweighauser, Vern Paxson, Stefan Savage, Geoffrey M. Voelker, and David Wagner. 2019. Detecting and Characterizing Lateral Phishing at Scale. In *Proceedings of the 28th USENIX Conference on Security Symposium* (Santa Clara, CA, USA) *(SEC'19)*. 1273–1290. http://dl.acm.org/citation.cfm?id=3361338.3361427
[23] G. Ho, A. Sharma, M. Javed, V. Paxson, and D. Wagner. 2017. Detecting Credential Spearphishing Attacks in Enterprise Settings. In *USENIX* (Vancouver, BC, Canada). 469–485.
[24] S. A. Khamis, C. F. M. Foozy, M. F. Ab Aziz, and N Rahim. 2020. Header Based Email Spam Detection Framework Using Support Vector Machine (SVM) Technique. In *ICSCDM*. Springer, 57–65.
[25] S. Kitterman. 2020. Sender Policy Framework (SPF). https://tools.ietf.org/html/rfc7208.
[26] A. Li, Z. Qin, R. Liu, Y. Yang, and D. Li. 2019. Spam Review Detection with Graph Convolutional Networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management* (Beijing, China) *(CIKM '19)*. Association for Computing Machinery, New York, NY, USA, 2703–2711. https://doi.org/10.1145/3357384.3357820
[27] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou. 2010. Fuzzy Keyword Search over Encrypted Data in Cloud Computing. In *2010 Proceedings IEEE INFOCOM*.
[28] Y. N. Liu, Y. Han, X. D. Zhu, F. He, and L. Y. Wei. 2014. An expanded feature extraction of e-mail header for spam recognition. In *AMR*, Vol. 846. 1672–1675.
[29] Posteo Mail. 2020. https://posteo.de/.
[30] Tutanota Mail. 2020. https://tutanota.com/.
[31] Mailfence. 2020. https://mailfence.com/.
[32] Esko Nuutila. 1998. Efficient transitive closure computation in large digraphs. (1998).
[33] Lawrence Page. 2001. Method for node ranking in a linked database. US Patent 6,285,999.
[34] Gilchan Park and Julia M Taylor. 2015. Using syntactic features for phishing detection. *arXiv preprint arXiv:1506.00037* (2015).
[35] Karl Pearson. 1905. The problem of the random walk. *Nature* 72, 1867 (1905), 342.
[36] P. Pons and M. Latapy. 2005. Computing Communities in Large Networks Using Random Walks. In *ISCIS*. 284–293.
[37] ProtoMail. 2020. Secure email. https://protonmail.com/.
[38] A. Ramachandran, N. Feamster, and S. S. Vempala. 2007. Filtering spam with behavioral blacklisting. In *CCS*. 342–351.
[39] J. Rao and D. Reiley. 2012. The economics of spam. *JEP* 26, 3 (2012), 87–110.
[40] P. Resnick. 2001. RFC 2822. https://tools.ietf.org/html/rfc2822l.
[41] M. Rosvall and C. Bergstrom. 2008. Maps of random walks on complex networks reveal community structure. *NAS* 105, 4 (2008), 1118–1123.
[42] S. Sheng, B. Wardman, G. Warner, L. Cranor, J. Hong, and C. Zhang. 2009. An empirical analysis of phishing blacklists. In *CEAS*. California, USA.
[43] A. Soliman and S. Girdzijauskas. 2017. AdaGraph: Adaptive Graph-Based Algorithms for Spam Detection in Social Networks. In *NS*. Cham, 338–354.
[44] G. Stringhini and O. Thonnard. 2015. That ain't you: Blocking spearphishing through behavioral modelling. In *DIMVA*. Springer, 78–97.
[45] Dan Sullivan. 2005. *The definitive guide to controlling malware, spyware, phishing, and spam*. Realtimepublishers. com.
[46] A. Vazhayil, N. Harikrishnan, R. Vinayakumar, K. Soman, and A. Verma. 2018. PED-ML: Phishing email detection using classical machine learning techniques. In *IWSPA*. Tempe, AZ, USA, 1–8.
[47] R. Verma and N. Hossain. 2013. Semantic feature selection for text with application to phishing email detection. In *ICISC*. Springer, 455–468.
[48] Rakesh Verma, Narasimha Shashidhar, and Nabil Hossain. 2012. Detecting phishing emails the natural language way. In *ESRCS*. Springer, 824–841.
[49] C. Wang and S. Chen. 2007. Using header session messages to anti-spamming. *Computers & Security* 26, 5 (2007), 381–390.
[50] Y. Zhauniarovich, I. Khalil, T. Yu, and M. Dacier. 2018. A Survey on Malicious Domains Detection through DNS Data Analysis. *ACM Comput. Surv.* 51, 4 (2018), 67:1–67:36.

A APPENDIX

A.1 Header Features

Table 10 summarizes the header features. Next, we describe these header features in details. For better readability, we append the relevant category name (SUBJ, NS, and DEP) to the feature name.

SUBJECT features: The first 19 features (features No. 1 - 19 in Table 10) are extracted from the SUBJECT of the email. Out of these 19 features, the first 15 (features No. 1 - 15 in Table 10) check whether the subject contains specific keywords from a blacklist suggested by Apache SpamAssassin [2, 3]. The SpamAssassin blacklist compiles the keywords that were observed as frequently being appeared in spam and promotional emails. Each of these 15 features is binary, returning 1 if a particular keyword (e.g., "hello", "save" and "free") appears in the `subject` field and 0 otherwise. The SUBJ_HAS_USERNAME feature (No. 16 in Table 10) is also a binary feature with value 1 if the user's nickname in the `from` field is shown in the `subject` field. This feature is crafted to capture an observed behavior in some spam and promoting emails in which adversaries include users' nickname in the subject as a social motivation to increase the chances of opening the email. The SUBJ_CODED feature (No. 17) returns 1 if the subject contains non-ASCII characters, otherwise returns 0. This feature is motivated by the observation that some promoting emails include emojis in the subject. The SUBJ_CAPS_PERCENTAGE feature (No. 18) computes the percentage of capital letters in the `subject` field, as subjects of unwanted emails (e.g., promotional ones) are likely to contain more capital letters than normal emails, may be to draw attention. The last feature (No. 19) computes the percentage of white spaces in the `subject` field. It is mainly designed to model unwanted (e.g., spam) emails that use white spaces to separate letters, especially in keywords that appear frequently in the `subject` field. Adversaries usually use the white spaces trick to evade detection by systems that employ keyword detection against a blacklist. Finally, note that

Group	No.	Feature	Domain	Description
SUBJECT [2]	1	SUBJ_ACCOUNT	{0, 1}	If `subject` field has keyword "account".
	2	SUBJ_APPROVED	{0, 1}	If `subject` field has keyword "approve" or "approval".
	3	SUBJ_BUY	{0, 1}	If `subject` field has keyword "buy".
	4	SUBJ_EARN	{0, 1}	If `subject` field has keyword "earn".
	5	SUBJ_FAMILY	{0, 1}	If `subject` field has keyword "family".
	6	SUBJ_FREE	{0, 1}	If `subject` field has keyword "free".
	7	SUBJ_GAPPED	{0, 1}	If `subject` field is gapped by underlines or spaces.
	8	SUBJ_GUARANTEED	{0, 1}	If `subject` field has keyword "guarantee".
	9	SUBJ_HELLO	{0, 1}	If `subject` field has keyword "hello".
	10	SUBJ_MONEY	{0, 1}	If `subject` field has keyword "money".
	11	SUBJ_ONLY	{0, 1}	If `subject` field has keyword "only".
	12	SUBJ_OWN	{0, 1}	If `subject` field has keyword "own".
	13	SUBJ_PLING_QUERY	{0, 1}	If `subject` field has keyword "?" or "!".
	14	SUBJ_SAVE	{0, 1}	If `subject` field has keyword "save" or "saving".
	15	SUBJ_STATEMENT	{0, 1}	If `subject` field has keyword "statement".
	16	SUBJ_HAS_USERNAME	{0, 1}	If user nickname in `from` field shows in subject.
	17	SUBJ_CODED	{0, 1}	If `subject` field contains non-ASCII characters
	18	SUBJ_CAPS_PERCENTAGE	[0, 1]	Percentage of capital letters in `subject` field.
	19	SUBJ_SPACE_PERCENTAGE	[0, 1]	Percentage of white space in `subject`.
NON-SUBJECT [2]	20	NS_CC_NUMBER	$\mathbb{N}$	Number of recipients in `cc` field.
	21	NS_CC_EMPTY	{0, 1}	If the `cc` field exists but empty.
	22	NS_DATE_INVALID	{0, 1}	If the format of sending timestamp is invalid.
	23	NS_DATETZ_INVALID	{0, 1}	If the format of sending timezone is invalid.
	24	NS_FROM_2ADDR	{0, 1}	If `from` field contains more than one email address.
	25	NS_FROM_CODED	{0, 1}	If `from` field is coded with non-ASCII char set after decoding.
	26	NS_FROM_FREE	{0, 1}	If `from` field contains keyword "free".
	27	NS_FROM_NOREPLY	{0, 1}	If `from` field contains keyword "no-reply".
	28	NS_FROM_OFFERS	{0, 1}	If `from` field contains keyword "offer".
	29	NS_FROM_MIXED	{0, 1}	If `from` field contains numbers mixed in with letters.
	30	NS_FROM_NO_LOWER	{0, 1}	If `from` field doesn't have lower case letter.
	31	NS_FROM_NOADDR	{0, 1}	If `from` field doesn't have a valid address.
	32	NS_FROM_NOUSER	{0, 1}	If `from` field doesn't have a user nickname.
	33	NS_INREPLYTO	{0, 1}	If `in_reply_to` field exists.
	34	NS_MAILFROM_BOUNCE	{0, 1}	If `mailfrom` field has keyword "bounce".
	35	NS_MSGID_NO_AT	{0, 1}	If `message_id` field does not have "@".
	36	NS_MSGID_NO_HOST	{0, 1}	If `message_id` field does not have a valid domain.
	37	NS_REPLYTO_MIXED	{0, 1}	If `reply_to` field exists and contains numbers mixed with letters.
	38	NS_REPLYTO_NOADDR	{0, 1}	If `reply_to` field exists but no valid address.
	39	NS_TO_MISSING	{0, 1}	If `to` field is missing.
	40	NS_TO_NO_ADDR	{0, 1}	If the `to` field does not exist or has no valid address.
	41	NS_TO_SORTED	{0, 1}	If more than 2 recipients in `to` field and in alphabetical order.
	42	NS_WEBMAIL_TRUE	{0, 1}	If `is_webmail` field is true (email was sent from a web interface).
DEPENDENCY [2, 28]	43	DEP_MAILFROM_FROM [28]	[0, 1]	Similarity between `mailfrom` field address and field `from` address.
	44	DEP_MAILFROM_HELO [28]	[0, 1]	Similarity between `mailfrom` field and `helo` field.
	45	DEP_MAILFROM_REPLYTO [28]	[0, 1]	Similarity between `mailfrom` field and field `reply_to` (if exists).
	46	DEP_MSGID_HELO [28]	[0, 1]	Similarity between `message_id` field and field `helo`.
	47	DEP_IN_FUTURE [2]	{0, 1}	If the sending timestamp is in the future of receiving time stamp.

Table 10: Header Features

all these 19 SUBJECT features will not be available if the subject is encrypted.

NON-SUBJECT features: The next 23 features (No. 20 - 42) in Table 10 does not use any information from the `subject` field, and hence dubbed NON-SUBJECT features. These features are grouped according to the fields from which they are derived. Features 20 and 21 are derived from the `cc` field. In particular, NS_CC_NUMBER (No. 20) returns the number of recipients in the `cc` field. This is based on observations that unwanted emails (e.g., spam) are typically sent to a large number of recipients by cc'ing them. The NS_CC_EMPTY

feature (No. 21) captures the behavior in which some unwanted emails include empty `cc` field as observed in [28, 45]. The next two features (No. 22 and 23) are extracted from the `date` field as recommended by the integrity analysis in [45]. Each of these two features will return a value of 1 if the format of the timestamp and time zone is invalid according to RFC 2822 [40], respectively. The next 9 features (No. 24 - 32) are extracted from the `from` field. These features mainly model the coding of the `from` field (No. 25), check if it contains some specific keyword (No. 26 - 28), if it has some character patterns (No. 29 - 30), and if it has a valid address (No. 31) and a user nickname (No. 32). The remaining features are extracted from `in_reply_to` field (No. 33), `mailfrom` field (No. 34 feature), `message_id` field (No. 35 & 36 features), `reply_to` field (No. 37 & 38 features), `to` field (No. 39 - 41 feature), and `is_webmail` field (No. 42 feature). It is worth mentioning that NS_TO_SORTED (No. 41) captures the fact that the alphabetical order among multiple recipients in the `to` field is a potential indicator of a spam email, since a non-spam email typically contains a list of addresses in random order. Furthermore, NS_WEBMAIL_TRUE (No. 42) is designed to account for the cases when webclients are used to send the unwanted emails. The intuition is that adversaries tend to use webmail services as one of the economic ways to send unwanted emails [39].

DEPENDENCY features SUBJECT and NON-SUBJECTS features are derived from individual fields and only can indicate spam emails. To detect phishing emails, we also derive five features from the dependency between a pair of fields. Among these five features, four of them (No. 43 - 46 in Table 10) model the content similarity between a pair of header fields. The content similarity can be measured by any string similarity metrics, e.g., Levenshtein distance and Jaccard similarity. In particular, we consider the consistency between the `from` and `mailfrom` fields (feature No. 43), between the `mailfrom` and `helo` fields (feature No. 44), between the `mailfrom` and `reply_to` fields (feature No. 45), and between the `message_id` and `helo` fields (feature No. 46). Finally, we design the fifth feature (No. 47 feature) to capture the temporal dependency between the send and receive timestamps. Intuitively, the sending timestamp should precede that of receiving. However, a phishing email may not show such temporal dependency.