High Precision Detection of Business Email Compromise

Asaf Cidon, ***Barracuda Networks and Columbia University;*** **Lior Gavish, Itay Bleier, Nadia Korshun, Marco Schweighauser, and Alexey Tsitkin,** ***Barracuda Networks***

https://www.usenix.org/conference/usenixsecurity19/presentation/cidon

This paper is included in the Proceedings of the 28th USENIX Security Symposium.

August 14–16, 2019 • Santa Clara, CA, USA

978-1-939133-06-9

Open access to the Proceedings of the 28th USENIX Security Symposium is sponsored by USENIX.

High Precision Detection of Business Email Compromise

Asaf Cidon[1,2] and Lior Gavish, Itay Bleier, Nadia Korshun, Marco Schweighauser and Alexey Tsitkin[1]

[1]Barracuda Networks, [2]Columbia University

Abstract

Business email compromise (BEC) and employee impersonation have become one of the most costly cyber-security threats, causing over $12 billion in reported losses. Impersonation emails take several forms: for example, some ask for a wire transfer to the attacker's account, while others lead the recipient to following a link, which compromises their credentials. Email security systems are not effective in detecting these attacks, because the attacks do not contain a clearly malicious payload, and are personalized to the recipient.

We present BEC-Guard, a detector used at Barracuda Networks that prevents business email compromise attacks in real-time using supervised learning. BEC-Guard has been in production since July 2017, and is part of the Barracuda Sentinel email security product. BEC-Guard detects attacks by relying on statistics about the historical email patterns that can be accessed via cloud email provider APIs. The two main challenges when designing BEC-Guard are the need to label millions of emails to train its classifiers, and to properly train the classifiers when the occurrence of employee impersonation emails is very rare, which can bias the classification. Our key insight is to split the classification problem into two parts, one analyzing the header of the email, and the second applying natural language processing to detect phrases associated with BEC or suspicious links in the email body. BEC-Guard utilizes the public APIs of cloud email providers both to automatically learn the historical communication patterns of each organization, and to quarantine emails in real-time. We evaluated BEC-Guard on a commercial dataset containing more than 4,000 attacks, and show it achieves a precision of 98.2% and a false positive rate of less than one in five million emails.

1 Introduction

In recent years, email-borne employee impersonation, termed by the FBI "Business Email Compromise" (BEC), has become a major security threat. According to the FBI, US organizations have lost $2.7 billion in 2018 and cumulatively $12 billion since 2013 [13]. Numerous well-known enterprises have fallen prey to such attacks, including Facebook, Google [41], and Ubiquiti [44]. Studies have shown that BEC is the cause of much higher direct financial loss than other common cyberattacks, such as ransomware [11, 13]. BEC attacks have also ensnared operators of critical government infrastructure [39]. Even consumers have become the targets of employee impersonation. For example, attackers have impersonated employees of real-estate firms to trick home buyers to wire down payments to the wrong bank account [1, 7, 17].

BEC takes several forms: some emails ask the recipient to wire transfer money to the attacker's account, others ask for W-2 forms that contain social security numbers, and some lead the recipient to follow a phishing link, in order to steal their credentials. The common theme is the impersonation of a manager or colleague of the target [12]. In this work, we focus on attacks where the attacker is external to the organization, and is trying to impersonate an employee. In §6 we discuss other scenarios, such as where the attacker uses a compromised internal email account to impersonate employees [18, 19].

Most email security systems are not effective in detecting BEC. When analyzing an incoming email, email security systems broadly look for two types of attributes: *malicious* and *volumetric*. Examples of malicious attributes are an attachment that contains malware, a link pointing to a compromised website, or an email that is sent from a domain with a low reputation. There are various well-known techniques to detect malicious attributes, including sandboxing [49], and domain reputation [2, 48]. Volumetric attributes are detected when the same email format is sent to hundreds of recipients or more. Examples include the same text or sender email (e.g., spam), and the same URL (e.g., mass phishing campaigns). However, employee impersonation emails do not contain malicious or volumetric attributes: they typically do not contain malware, are not sent from well-known malicious IPs, often do not contain a link, and are sent to a small number of recipients (with the explicit intent of evading volumetric filters). When employee impersonation attacks do contain a link, it is typically

a link to a fake sign up page on a legitimate website that was compromised, which does not appear on any IP black lists. In addition, the text of the attacks is tailored to the recipient, and is typically not caught by volume-based filters.

Our design goal is to detect and quarantine BEC attacks in real-time, at a low false positive rate (1 in a million emails) and high precision (95%). We make the observations that popular cloud email systems, such as Office 365 and Gmail, provide APIs that enable account administrators to allow external applications to access historical emails. Therefore, we design a system that detects BEC by relying on historical emails available through these APIs.

Prior work on detecting impersonation has been conducted either on very small datasets [10, 14, 20, 45]), or focused on stopping a subset of BEC attacks (domain spoofing [14] or emails with links [20]). In addition, most prior work suffers from very low precision (only 1 in 500 alerts is an attack [20]) or very high false positive rates [10, 45]), which makes prior work unsuitable for detecting BEC in real-time.

The main challenge in designing a system that can detect BEC at a low false positive rate is that *BEC emails are very rare* as a percentage of all emails. In fact, in our dataset, less than one out of 50,000 emails is a BEC attack. Therefore, in order to achieve low false positives, we design a system using supervised learning, which relies on a large training set of BEC emails. However, bootstrapping a supervised learning systems presents two practical challenges. First, it is difficult to label a sufficiently large training dataset that includes millions of emails. Second, it is challenging to train a classifier on an *imbalanced dataset*, in which the training dataset contains almost five orders of magnitude fewer positive samples (i.e., BEC attacks) than negative samples (i.e., innocent emails).

In this paper, we present how we initially trained BEC-Guard, a security system that automatically detects and quarantines BEC attacks in real-time using historical emails. BEC-Guard is part of a commercial product, Barracuda Sentinel, used by thousands of corporate customers of Barracuda Networks to prevent BEC, account takeover, spear phishing and other targeted attacks. BEC-Guard does not require an analyst to review the detected emails, but rather relies on offline and infrequent re-training of classifiers. The key insight of BEC-Guard is to split the training and classification into two parts: header and body.

Instead of directly classifying BEC attacks, the *impersonation classifier* detects impersonation attempts, by determining if an attacker is impersonating an employee in the company by inspecting the header of the email. It utilizes features that include information about which email addresses employees typically utilize, how popular their name is, and characteristics of the sender domain. The *content classifiers* are only run on emails that were categorized as impersonation attempts, and inspects the body of the email for BEC. For emails that do not contain links, we use a k-nearest neighbors [43] (KNN) classifier that weighs words using term frequency-inverse document frequency [28, 42] (TFIDF). For emails with links, we train a random forest classifier that relies on the popularity as well as the position of the link in the text. Both of the content classifiers can be retrained frequently using customer feedback.

To create the initial classifiers, we individually label and train each type of classifier: the labels of the impersonation classifier are generated using scripts we ran on the training dataset, while the content classifiers are trained over a manually labeled training dataset. Since we run the content classification only on emails that were detected as impersonation attempts, we need to manually label a much smaller subset of the training dataset. In addition, to ensure the impersonation classifier is trained successfully over the imbalanced dataset, we develop an under-sampling technique for legitimate emails using Gaussian Mixture Models, an unsupervised clustering algorithm. The classifiers are typically re-trained every few weeks. The dataset available for initial training consists of a year worth of historical emails from 1500 customers, with an aggregate dataset of 2 million mailboxes and 2.5 billion emails. Since training the initial classifiers, our dataset has been expanded to include tens of millions of mailboxes.

BEC-Guard uses the APIs of cloud-based email systems (e.g., Office 365 and Gmail), both to automatically learn the historical communication patterns of each organization within hours, and to quarantine emails in real-time. BEC-Guard subscribes to API calls, which automatically alert BEC-Guard whenever a new email enters the organization's mailbox. Once notified by the API call, BEC-Guard classifies the email for BEC. If the email is determined to be BEC, BEC-Guard uses the APIs to move the email from the inbox folder to a dedicated quarantine folder on the end-user's account.

To evaluate the effectiveness of our approach, we measured BEC-Guard's performance on a dataset of emails taken from several hundred organizations. Within this labeled dataset, BEC-Guard achieves a a precision of 98.2%, a false positive rate of only one in 5.3 million. To summarize, we make the following contributions:

- First real-time system for preventing BEC that achieves high precision and low false positive rates.
- BEC-Guard's novel design relies on cloud email provider APIs both to learn the historical communication patterns of each organization, and to detect attacks in real-time.
- To cope with labeling millions of emails, we split the detection problem into two sets of classifiers run sequentially.
- We use different types of classifiers for the header and text of the email. The headers are classified using a random forest, while the text classification relies primarily on a KNN model that is not dependent on any hard-coded features, and can be easily re-trained.
- To train the impersonation classifier on an imbalanced dataset, we utilize a sampling technique for the legitimate emails using a clustering algorithm.

BEC Objective	Link?	Percentage
Wire transfer	No	46.9%
Click Link	Yes	40.1%
Establish Rapport	No	12.2%
Steal PII	No	0.8%

Table 1: The objective of BEC attacks as a percentage of 3,000 randomly chosen attacks. 59.9% of attacks do not involve a phishing link.

Role	Recipient %	Impersonated %
CEO	2.2%	42.9%
CFO	16.9%	2.2%
C-level	10.2%	4.5%
Finance/HR	16.9%	2.2%
Other	53.7%	48.1%

Table 2: The roles of recipients and impersonated employees from a sample of BEC attacks chosen from 50 random companies. C-level includes all executives that are not the CEO and CFO, and Finance/HR does not include executives.

2 Background

Business email compromise, also known as employee impersonation, CEO fraud, and whaling,[1] is a class of email attacks where an attacker impersonates an employee of the company (e.g., the CEO, a manager in HR or finance), and crafts a personalized email to a specific employee. The intent of this email is typically to trick the target to wire money, send sensitive information (e.g., HR or medical records), or lead the employee to follow a phishing link in order to steal their credentials or download malware to their endpoint.

BEC has become one of the most damaging email-borne attacks in recent years, equaling or surpassing other types of attacks, such as spam and ransomware. Due to the severity of BEC attacks, the FBI started compiling annual reports based on US-based organizations that have reported their fraudulent wire transfers to the FBI. Based on the FBI data, between 2013 and 2018, $12 billion have been lost [13]. To put this in perspective, a Google study estimates that the total amount of ransomware payments in 2016 was only $25 million [11].

In this section, we review common examples of BEC, and provide intuition on how their unique characteristics can be exploited for supervised learning classification.

2.1 Statistics

To better understand the goals and methodology of BEC attacks, we compiled statistics for 3,000 randomly selected BEC attacks in our dataset (for more information about our dataset, see §4.2). Table 1 summarizes the objectives of the attacks. The results show that the most common BEC in the sampled attacks is try to deceive the recipient to perform a wire transfer to a bank account owned by the attacker, while about 0.8% of the attacks ask the recipient to send the attacker personal identifiable information (PII), typically in the form of W-2 forms that contain social security numbers. About 40% of attacks ask the recipient to click on a link. 12% of attacks try to establish rapport with the target by starting a conversation with the recipient (e.g., the attacker will ask the recipient whether they are available for an urgent task). For the "rapport" emails, in the vast majority of cases, after the initial email is responded to the attacker will ask to perform a wire transfer.

An important observation is that about 60% of BEC attacks do not involve a link: the attack is simply a plain text email that fools the recipient to commit a wire transfer or send sensitive information. These plain text emails are especially difficult for existing email security systems, as well as prior academic work to detect [20], because they are often sent from legitimate email accounts, tailored to each recipient, and do not contain any suspicious links.

We also sampled attacks from 50 random companies in our dataset, and classified the roles of the recipient of the attack, as well as the impersonated sender. Table 2 presents the results. Based on the results, the term "CEO fraud" used to describe BEC is indeed justified: about 43% of the impersonated senders were the CEO or founder. The targets of the attacks are spread much more equally across different roles. However, even for impersonated senders, the majority (about 57%) are not the CEO. Almost half of the impersonated roles and more than half of targets are not of "sensitive" positions, such as executives, finance or HR. Therefore, simply protecting employees in sensitive departments in not sufficient to protect against BEC.

2.2 Common Types of BEC

To guide the discussion, we describe the three most common examples of BEC attacks within our dataset: wire transfer, rapport, and impersonation phishing. In §6 we will discuss other attacks that are not covered by this paper. All three examples we present are real BEC attacks from within our dataset, in which the names, companies, email addresses and links have been anonymized.

Example 1: Wire transfer example

```
From: "Jane Smith" <jsmith@acrne.com>
To: "Joe Barnes" <jbarnes@acme.com>
Subject: Vendor Payment

Hey Joe,

Are you around? I need to send a wire
transfer ASAP to a vendor.

Jane
```

In Example 1, the attacker asks to execute a wire transfer. Other similar requests include asking for W-2 forms, medical information or passwords. In the example the attacker spoofs the name of an employee, but uses an email address that

[1] We refer to this attack throughout the paper as *BEC*.

Example 2: Rapport example

```
From: "Jane Smith" <jsmith@acme.com>
Reply-to: "Jane Smith" <ceo.executive@outlook.com>
To: "Joe Barnes" <jbarnes@acme.com>
Subject: At desk?

Joe, are you available for something urgent?
```

Example 3: Spoofed Name with Phishing Link

```
From: "Jane Smith" <greyowl1234@comcast.net>
To: "Joe Barnes" <jbarnes@acme.com>
Subject: Invoice due number 381202214

I tried to reach you by phone today but I
couldn't get through. Please get back to me
with the status of the invoice below.

Invoice due number 381202214:
[http://firetruck4u.net/past-due-invoice/]
```

does not belong to the organization's domain. Some attackers even use a domain that looks similar to the target organization's domain (e.g., instead of acme.com, the attacker would use acrne.com). Since many email clients do not display the sender email address, some recipients will be deceived even if the attacker uses an unrelated email address.

Example 2 tries to create a sense of urgency. After the recipient responds to the email, the attacker will typically ask for a wire transfer. The email has the from address of the employee, while the reply-to address will relay the response back to the attacker. Email authentication technologies such as DMARC, SPF and DKIM can help stop spoofed emails. However, the vast majority of organizations do not enforce email authentication [25], because it can be difficult to implement correctly and often causes legitimate emails to be blocked.[2] Therefore, our goal is to detect these attacks without relying on DMARC, SPF and DKIM.

Example 3 uses a spoofed name, and tries to get the recipient to follow a phishing link. Such phishing links are typically not detected by existing solutions, because the link is unique to the recipient ("zero-day") and will not appear in any black lists. In addition, attackers often compromise relatively reputable websites (e.g., small business websites) for phishing links, which are often classified as high reputation links by email security systems. The link within the email will typically lead the recipient to a website, where they will be prompted to log in a web service (e.g., an invoicing application) or download malware.

3 Intuition: Exploiting the Unique Attributes of Each Attack

The three examples all contain unique characteristics, which set them apart from innocent email messages. We first describe the unique attributes in the header of each example, and then discuss the attributes of the email body and how they can be used to construct the features of a machine learning classifier. We also discuss legitimate corner cases of these attributes that might fool a classifier and cause false positives.

Header attributes. In Example 1 and 3, the attacker impersonates the name of a person, but uses a different email address than the corporate email address. Therefore, if an email contains a name of an employee, but uses an email address that is not the typical email address of that employee, there is a higher probability that the sender is an imposter.

However, there are legitimate use cases of non-corporate emails by employees. First, an employee might use a personal email address to send or forward information to themselves or other employees in the company. Ideally, a machine learning classifier should be able to learn all the email addresses that belong to a certain individual, including corporate and personal email addresses. Second, if an external sender has the same name as an internal employee, it might seem like an impersonation.

In Example 2, the attacker spoofs the legitimate email address of the sender, but the reply-to email address is different than the sender address, which is unusual (we will also discuss the case where the attacker sends a message from the legitimate address of the sender without changing the reply-to field in §6). However, such a pattern has legitimate corner cases as well. Some web services and IT systems, such as LinkedIn, Salesforce, and other support and HR applications, "legitimately impersonate" employees to send notifications, and change the reply-to field to make sure the response to the message is recorded by their system.

Other header attributes might aid in the detection of BEC attacks. For example, if an email is sent at an abnormal time of day, or from an abnormal IP or from a foreign country. However, many BEC attacks are designed to seem legitimate, and are sent in normal times of day and from seemingly legitimate email addresses.

Body attributes. The body of Example 1 contains two unique semantic attributes. First, it discusses sensitive information (a wire transfer). Second, it is asking for a special, immediate request. Similarly, the text of Example 2 is asking whether the recipient is available for an urgent request. Such an urgent request for sensitive information or availability might be legitimate in certain circumstances (for example, in an urgent communication within the finance team).

The unique attribute in the body of Example 3 is the link itself. The link is pointing to a website that does not have anything to do with the company: it does not belong to a web service the company typically uses, and it is not related to the company's domain.

Finally, all three examples contain certain textual and visual elements that are unique to the identity of the sender. For example, Example 1 contains the signature of the CEO and all of the emails contain a particular grammar and writing

[2] Many organizations have legitimate systems that send emails on their behalf, for example, marketing automation systems, which can be erroneously blocked if email authentication is not setup properly.

style. If any of these elements deviate from the style of a normal email from a particular sender, they can be exploited to detect an impersonation. Since in many BEC emails the attackers take great care in making the email appear legitimate, we cannot overly-depend on detecting stylistic aberrations.

As shown above, each of the examples has unique anomalous attributes that can be used to categorize it as a BEC attack. However, as we will show in §7, none of these attributes *on its own* is sufficient to classify an email with a satisfactory false positive rate.

Leveraging historical emails. Much of prior work in detecting email-borne threats relies on detecting malicious signals in the email, such as sender and link domain reputation [2, 48], malicious attachments [49], as well as relying on link click logs and IP logins [20]. However, as Table 1 and the examples we surveyed demonstrate, most BEC attacks do not contain any obviously malicious attachments or links. Intuitively, access to the historical emails of an organization would enable a supervised learning system to identify the common types of BEC attacks by identifying anomalies in the header and body attributes. We make the observation that popular cloud-based email providers, such as Office 365 and Gmail, enable their customers to allow third party applications to access their account with certain permissions via public APIs. In particular, these APIs can enable third-party applications to access historical emails. This allows us to design a system that uses historical emails to identify BEC attacks.

4 Classifier and Feature Design

In this section, we describe BEC-Guard's design goals, and its training dataset. We then describe the initial set of classifiers we used in BEC-Guard, and present our approach to training and labeling.

4.1 Design Goals

The goal of BEC-Guard is to detect BEC attacks in real-time, without requiring the users of the system to utilize security analysts to manually sift through suspected attacks. To meet this goal, we need to optimize two metrics: the false positive rate, and the precision. The false positive rate is the rate of false positives as a percentage of total received emails. If we assume an average user receives over 100 emails a day, in an organization with 10,000 employees, our goal is that it will be infrequent to encounter a false positive (e.g., once a day for the entire organization). Therefore, our target false positive rate is less than one in a million. The precision is the rate of true positives (correctly detected BEC attacks) as a percentage of attacks detected by the system, while the false positive rate is a percentage of false positives of all emails (not just emails detected by the system). If the precision is not high, users of BEC-Guard will lose confidence in the validity of its predictions. In addition to these two metrics, we need to ensure high coverage, i.e., that the system catches the vast majority of BEC attacks.

4.2 Dataset and Privacy

We developed the initial version of BEC-Guard using a dataset of corporate emails from 1,500 organizations, which are actively paying customers of Barracuda Networks. The organizations in our dataset vary widely in their type and size. The organizations include companies from different industries (healthcare, energy, finance, transportation, media, education, etc.). The size of the organization varies from 10 mailboxes to more than 100,000. Overall, to train BEC-Guard, we labeled over 7,000 examples of BEC attacks, randomly selected from the 1,500 organizations.

To access the data, these organizations granted us permission to access to the APIs of their Office 365 email environments. The APIs provide access to all historical corporate emails. This includes emails sent internally within the organization, and from all folders (inbox, sent, junk, etc.). The API also allows us to determine which domains are owned by each organization, and even whether an email was read.

Ethical and privacy considerations. BEC-Guard is part of a commercial product, and the 1,500 customers that participate in the dataset provided their legal consent to Barracuda Networks to access their historical corporate emails for the purpose identifying BEC. Customers also have the option of revoking access to BEC-Guard at any time.

Due to the sensitivity of the dataset, it was only exposed to the five researchers who developed BEC-Guard, under strict access control policies. The research team only accessed historical emails for the purposes of labeling data to develop BEC-Guard's classifiers. Once the classifiers were developed, we permanently deleted all of the emails that are not actively used for training the classifiers. The emails used for classification are stored encrypted, and access to them is limited to the research team.

4.3 Dividing the Classification into Two Parts

The relative rare occurrence of BEC attacks influenced several of our design choices. Our first design choice was to rule out unsupervised learning. Unsupervised learning typically uses clustering algorithms (e.g., k-means [15]) to group email categories, such as BEC emails. However, a clustering algorithm would typically categorize many common categories (e.g., social emails, marketing emails), but since BEC is so rare, it results in low precision and many false positives. Therefore, supervised learning algorithms are more suitable for detecting BEC at a high precision. However, using supervised learning presents its own set of challenges.

In particular, BEC is an extreme case of *imbalanced data*. When sampled uniformly, in our dataset, "legitimate" emails are 50,000× more likely to appear than the BEC emails. This presents two challenges. First, in order to label a modest number of BEC emails (e.g., 1,000), we need to label a corpus

on the order of 50 million legitimate emails. Second, even with a large number of labeled emails, training a supervised classifier over imbalanced datasets is known to cause various problems, including biasing the classifier to prefer the larger class (i.e., legitimate emails) [24,26,47,51]. To deal with this extreme case of imbalanced data, we divided the classification and labeling problem into two parts. The first classifier looks only at the metadata of the email, while the second classifier only examines the body and subject of the email.

The first classifier looks for *impersonation* emails. We define an impersonation as an email that is sent with the name of a person, but was not actually sent by that person. Impersonation emails include malicious BEC attacks, and they also include emails that legitimately impersonate an employee, such as internal systems that send automated emails on behalf of an employee. The impersonation classifier only analyzes the metadata of the email (i.e., sender, receiver, CC, BCC fields). The impersonation classifier detects both spoofed name (Example 1 and 3) and spoofed emails (Example 2). The second set of classifiers, the *content* classifiers, only classify emails that were detected as impersonation emails, by examining the email's subject and body to look for anomalies. We use two different content classifiers that each look for different types of BEC attacks.[3] The two content classifiers are: the text classifier, which relies on natural language processing to analyze the text of the email, and the link classifier, which classifies any links that might appear in the email.

All of our classifiers are trained globally on the same dataset. However, to compute some of the features (e.g., the number of time the sender name and email address appeared together), we rely on statistics that are unique to each organization.

4.4 Impersonation Classifier

Table 3 includes the main features used by the impersonation classifier. The features describe the number of times specific email addresses and names have appeared before in the sender and reply-to fields, as well as statistics about the sender's identity.

To demonstrate why it is helpful to maintain historical statistics of a particular organization, consider Figure 1. The figure depicts the number of email addresses that were used by each sender in an organization with 44,000 mailboxes over three months. 82% of the users had emails sent from only one address, and the rest had emails that were sent from more than one address. The reason that some of the senders used a large number of email addresses, is that they were repeatedly impersonated in BEC attacks. For instance, the CEO is a common target for impersonation. and is often targeted dozens of times. However, this signal alone is not sufficient to detect impersonation,. For example, some of the senders that have a large number of email addresses represent shared mailboxes (e.g., "IT" or "HR"), and are legitimate.

[3]There is no inherent advantage in using multiple content classifiers in terms of the false positive rate or precision. We decided to use two different content classifiers, because it made it easier for us debug and maintain them separately.

Feature	Description
Sender has corp domain?	Is sender address from corp domain?
Reply-to != sender address?	Reply-to and sender addresses different?
Num times sender and email	Number of times sender name and email address appeared
Num times reply-to address	Number of times reply-to address appeared
Known reply-to service?	Is reply-to from known web service (e.g., LinkedIn)?
Sender name popularity	How popular is sender name

Table 3: Main features used by the impersonation classifier, which looks for impersonation attempts, including spoofed names and emails.

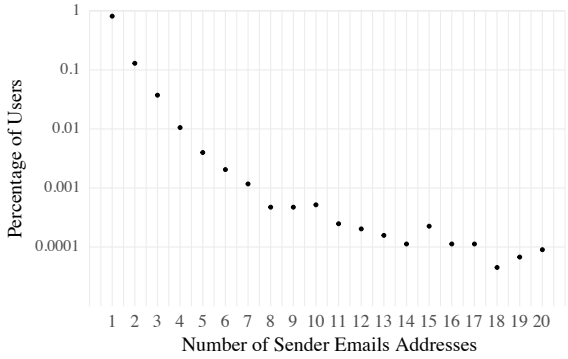

Figure 1: Number of unique emails addresses that were observed for each user in an organization with 44,400 mailboxes. The X axis is the number of unique email addresses that were observed, as a percentage (in the Y axis) of the total number of users of the organization.

Hence, several of the features in the impersonation classifier rely on the historical communication patterns of the organization. This influenced BEC-Guard's architecture. In addition, we maintain a list of known web services that "legitimately" send emails with reply-to addresses that are different than the sender address (e.g., LinkedIn, Salesforce), in order to capture the response. The original list of commonly-used services was populated from a list of the domains of the major web services. We then augmented this list with additional services when we encountered them during the labeling process (in §6 we discuss possible evasion techniques related to this list of legitimate reply-to senders). The sender name popularity score is computed offline by maintaining a list of how frequently names appear across different organizations in our dataset. The more popular a name, the higher the likelihood that a name with an email address the employee typically does not use is another person (a name collision).

Name and nickname matching. In order to detect name spoofing, the impersonation classifier needs to match the

sender name with a name of an employee. However, names can be written in various forms. For example: "Jane Smith" can be written as: "Smith, Jane", "Jane J. Smith" or "Jane Jones Smith". In addition, we need to deal with special characters that might appear in names, such as ì or ä.

To address these problems, BEC-Guard normalizes names. It stores employee name as <first name, last name> tuples, and checks all the variants of the sender name to see if it matches a name of an employee with a corporate email address. These variants include stripping the middle name or initial, reversing order of the first name and surname, and stripping suffixes. Suffixes include examples like "Jr." or when the email address is sent as part of the sender name. In addition, we match the first name against a publicly available list of nicknames [36], to catch cases for example when the attacker sends an email as "Bill Clinton", and the name of the employee is stored as "William Clinton".

Content classifiers. Our system uses two content classifiers: the *text classifier* and *link classifier*. The text classifier catches attacks similar to Example 1 and 2, and the link classifier stops attacks that are similar to Example 3. By design, the content classifiers are meant to be updated more frequently than the impersonation classifier, and should be easily retrained based on false negatives and false positives reported by users.

Text classifier. In BEC attacks similar to Example 1 and 2, the body contains words that are indicative of a sensitive or special request, such as "wire transfer" or "urgent". Therefore, our first iteration of the text classifier was designed to look for specific words that might imply a special request or a financial or HR transaction. The features of the classifiers described the position in the text of a list of sensitive words and phrases. However, over time, we noticed this approach suffered from several problems. First, a classifier that relies on hard-coded keywords can miss attacks when attackers slightly vary a specific word or phrase. Second, to successfully retrain the classifier, we had to modify the lists of keywords that it looks for, which required manually updating the keyword list on a daily basis.

Instead, we developed a text classifier that learns expressions that are indicative of BEC on its own. The first step is to pre-process the text. BEC-Guard removes information from the subject and body of the email that would not be useful for classifying the email. It removes regular expression patterns that include salutations ("Dear", "Hi"), pre-canned headers, as well as footers ("Best,") and signatures. It also removes all English stopwords, as well as any names that may appear in the email.

The second step is to compute the frequency-inverse document frequency [42] (TFIDF) score of each word in the email. TFIDF represents how important each word is in an email, and is defined as:

$$TF(w) = \frac{\text{num times } w \text{ appears in email}}{\text{num words in email}}$$

$$IDF(w) = \frac{log(\text{num emails})}{\text{num emails with } w}$$

Where w is a given word in an email. $TF(w) \cdot IDF(w)$ gives a higher score to a word that appears frequently in a specific email, but which is relatively rare in the whole email corpus. The intuition is that in BEC emails, words for example that denote urgency or a special request would have a high TFIDF score, because they appear frequently in BEC emails but less so in legitimate emails.

When training the text classifier, we compute the TFIDF score of each word in each email of the training set. We also compute the TFIDF for pairs of words (bigrams). We store the global statistics of the IDF as a *dictionary*, which contains number of emails in the training set that contain unique phrases encountered in the training of the text classifier. We limit the dictionary size to 10,000 of the top ranked words (we evaluate how the size of the dictionary impacts classification precision in §7.2).

The feature vector of each email is equal to the the number of words in the dictionary, and each number represents the TFIDF of each one of the words in the dictionary. Words that do not appear in the email, or that do not appear in the dictionary have a TFIDF of zero. The last step is to run a classifier based on these features. Table 4 presents the top 10 phrases (unigram and bigram) in the BEC emails in our dataset. Note that the top phrases all indicate some form of urgency.

Top phrases in BEC emails by TFIDF	
1. got moment	6. need complete
2. response	7. ASAP
3. moment need	8. urgent response
4. moment	9. urgent
5. need	10. complete task

Table 4: The top 10 phrases of BEC emails, sorted by their TFIDF ranking from our evaluation dataset (for more information on evaluation dataset see §7.1). The TFIDF was computed for each word in all of the BEC emails in our evaluation dataset.

Link classifier. The link classifier detects attacks similar to Example 3. In these attacks, the attacker tries to get the recipient to follow a phishing link. As we described earlier, these personalized phishing links are typically not detected by IP blacklists, and are usually unique to the recipient. In this case, since the content classifier only classifies emails that were already classified as impersonation emails, it can mark links as "suspicious", even if they would have a high false positive rate otherwise. For example, a link that points to a

small website, or one that was recently registered, combined with an impersonation attempt would have a high probability of being a BEC email.

Feature	Description
Domain popularity	How popular is the link's least popular domain
URL field length	Length of least popular URL (long URLs are more suspicious)
Domain registration date	Date of domain registration of least popular domain (new domains are suspicious)

Table 5: Main features used by the link request classifier, which stops attacks like in Example 3.

Table 5 describes the main features used by the link request classifier. The domain popularity is calculated by measuring the Alexa score of the domain. In order to deal with link shorteners or link redirections, BEC-Guard expands the URLs before computing their features for the link classifier. In addition, several of the URL characteristics require determining information about the domain (popularity and score). For the domain popularity feature, we cache a list of the top popular domains, and update it offline. To determine the domain registration date, BEC-Guard does a real-time WHOIS lookup. Note that unlike the impersonation classifier, which needs to map the distribution of email address per sender name, none of the features of the text and link classifier are organization-specific. This allows us to easily retrain them based on user reported emails.

4.5 Classifier Algorithm

The impersonation and link classifiers use random forest [5] classification. Random forests are comprised of randomly formed decision trees [40], where each tree contributes a vote, and the decision is determined by the majority of the trees. Our system uses random forests rather than individual decision trees, since we found they provide better precision, but for offline debugging and analysis we often visualize individual decision trees. We decided to use KNN for the text classifier, because it had slightly better coverage than random forests. However, we found that since the text classifier uses a very large number of features (a dictionary of 10,000 phrases), its efficacy was similar across different classifiers. In §7.2 we evaluate the performance of the different classifier algorithms.

In addition, we have explored deep-learning based techniques, such as word2vec [34] and sense2vec [46], which expand each word to a vector that represents its different meanings. We currently do not use such deep-learning techniques, because they are computationally heavy both for training and online classification.

Detecting impersonation of new employees. When a new employee joins the organization, the impersonation classifier will not have sufficient historical information about that employee, since they will not have any historical emails. As that employee receives more emails, BEC-Guard will be start compiling statistics for the employee. A similar problem may also arise in organizations that periodically purge their old emails. In practice, we found that the classifier performs well after only one month of data.

4.6 Labeling

In order to label the initial training set, we made several assumptions about the BEC attack model. First we assumed attackers impersonate employees using their name (under a set of allowed variations, as explained above). Second, we assumed the impersonation does not occur more than 100 times using the same email address. Third, we assumed the attacker uses an email address that is different than the corporate address, either as the from address or the reply-to address. We discuss other types of attacks that do not fit these assumptions, as well as how attackers may evade these assumptions in §6. Under these constraints, we fully covered all of the possible attacks and manually labeled them. In addition, we incorporated missed attacks reported from customers (we discuss this process in §7.3).

The reason we assumed a BEC email does not impersonate an employee using the same email address more than 100 times is that BEC-Guard is designed with the assumption that the organization is already using at a spam filter, which provides protection against volume-based attacks (e.g., the default spam protection of Office 365 or Gmail). Therefore, an attacker that would send an email from an unknown address more than 100 times to the same recipient would likely be blocked by the spam filter. In fact, in our entire dataset, which is only composed of post spam-filtered emails, we have never witnessed an attacker using the email address to impersonate an employee more than 20 times. Note that we only used this assumption for labeling the original training set, and do not use it for ongoing retraining (since retraining is based on customer reported attacks).

Impersonation classifier. In order to label training data for the impersonation classifier, we ran queries on the headers of the raw emails to uncover all emails that might contain BEC attacks under our labeling assumptions (see above). We then labeled the results of all the queried emails as impersonation emails, and all the emails that were not found by the queries as legitimate emails.

Content classifiers. The training dataset for the content classifiers is constructed by running a trained impersonation classifier on a fresh dataset, which is then labeled manually. The initial training set we used for the content classifiers included 300,000 impersonation emails from randomly selected organizations over a year of data. Even within this training data set, we were able significantly further limit the number of emails that needed to be manually labeled. This is due to the fact that the vast majority of these emails were obviously

not BEC attacks, because they were due to a legitimate web services that impersonates a large number of employees (e.g., a helpdesk system sending emails on behalf of the IT staff).

After constructing the initial dataset, training content classifiers is very straightforward, since we continuously collect false negative and false positive emails from users and add them into the training set. Note that we still manually review these samples before retraining as a measure of quality control, to ensure that adversaries do not "poison" our training set, as well as to make sure that users did not label emails erroneously.

Sampling the dataset. Naïvely training a classifier over an imbalanced dataset typically biases the classifier to prefer the majority class. Specifically, it can result in a classifier that will simply always choose to predict the majority class, i.e., legitimate emails, and will thus achieve very high accuracy (i.e., $accuracy = (tp+tn)/(tp+tn+fp+fn)$, where tp is true positives, tn is true negatives, fp is false positives, and fn is false negatives). Since BEC is so rare in our dataset, a classifier that always predicts that an email is legitimate would achieve a high accuracy. This problem is especially acute in the case of our impersonation classifier, which needs to do the initial filtering between legitimate and BEC emails. In the case of content classifiers, we did not have to sample the dataset, because it deals with a much smaller training dataset.

There are various methods of dealing with imbalanced datasets, including over-sampling the minority class and under-sampling the majority class [6,24,27,29,30], as well as assigning higher costs to incorrectly predicting the minority class [9,38].

Our second major design choice was to under-sample the majority class (the legitimate emails). We made this decision for two reasons. First, if we decided to over-sample the BEC attacks, we would need to do so by a large factor. This might overfit our classifier and bias the results based on a relatively small number of positive samples. Second, over-sampling makes training more expensive computationally.

A naïve way to under-sample would be to uniformly sample the legitimate emails. However, this results in a classifier with a low precision, because the different categories of legitimate emails are not well represented. For example, uniformly sampling emails might miss emails from web services that legitimately impersonate employees. The impersonation classifier will flag these emails as BEC attacks, because they are relatively rare in the training dataset.

The main challenge in under-sampling the majority class is how to represent the entire universe of legitimate emails with a relatively small number of samples (i.e., comparable or equal to the number of BEC email samples). To do so, we cluster the legitimate emails using an unsupervised learning algorithm, Gaussian Mixture Models (GMM). The clustering algorithm splits the samples into clusters, each of which is represented by a Normal distribution, projected onto the impersonation classifier feature space. Figure 2 illustrates an example with two features and 14 legitimate email samples. In this example, the samples are split into three clusters. To choose a representative sample of legitimate emails, we randomly pick a certain number of samples from each cluster, proportional to the number of legitimate emails that belong to each cluster. If for example our goal is to use a total of 7 samples, we would choose 4 samples from the first cluster, 2 samples from the second cluster, and 1 sample from the third cluster, because the original number of samples in each cluster is 8, 4, and 2, respectively.

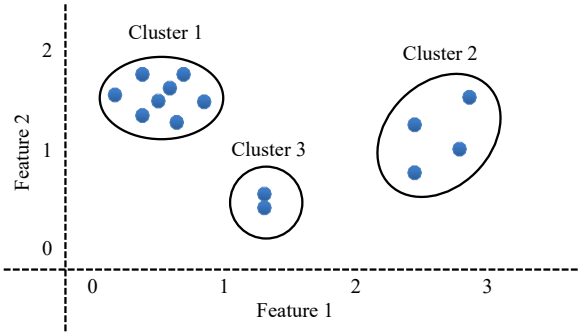

Figure 2: Depiction of running clustering algorithm on a set legitimate emails in a two-dimensional feature space with three clusters. After clustering the legitimate emails, we choose the number of samples from each cluster in proportion to the size of the cluster.

We chose the number of clusters that guarantee a minimal representation for each major "category" of legitimate email. We found that using 85 clusters was sufficient for capturing the legitimate emails in our dataset. When we tried using more than 85 clusters, the clusters beyond the 85th one would be nearly or entirely empty. Even after several iterations of retraining the impersonation classifier, we have have found that 85 clusters are sufficient to represent our dataset.

5 System Design

BEC-Guard consists of two key stages: an online classification stage and an offline training stage. Offline training is conducted periodically (every few days). When a new email arrives, BEC-Guard combines the impersonation and content classifiers to determine whether the email is BEC or not. These classifiers are trained ahead of time in the offline training stage. We describe the key components of our system design in more detail below.

Traditionally, commercial email security solutions have a gateway architecture, or in other words, they sit in the data path of inbound emails and filter malicious emails. As described above, some of BEC-Guard's impersonation classifier features rely on historical statistics of internal communications. The gateway architecture imposes constraints on detecting BEC attacks for two reasons. First, a gateway typically cannot observe internal communications. Second, the gateway usually does not have access to historical communications, so it would require several months or more of observing the communication patterns before the system would be able to detect

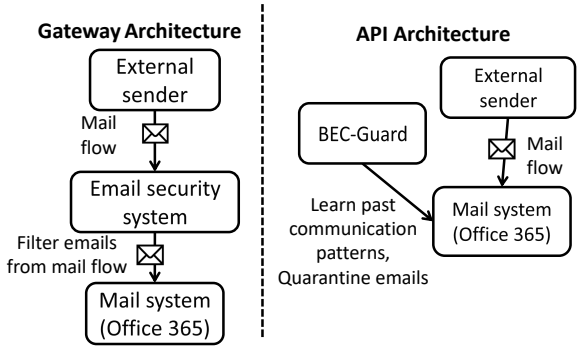

Figure 3: Comparison between the architecture of traditional email security systems, which sit as a gateway that filters emails before they arrive in the mail system, and BEC-Guard's architecture, which relies on APIs for learning the historical communication patterns of each organization, and detecting attacks in real-time.

incoming BEC attacks. Fortunately, cloud-based email services, such as Office 365 and Gmail, provide APIs that enable access to historical communications, as well as to monitor and move emails in real-time. BEC-Guard leverages these APIs both to gain access to historical communication, and also to do near real-time BEC detection. Figure 3 compares the gateway architecture with BEC-Guard's API based architecture. We describe BEC-Guard's design and implementation using the Office 365 APIs.

Warmup phase. We name the process of analyzing each organization's historical communications, the *warmup* phase. In order to start the warmup, the organization enables BEC-Guard to get access to its Office 365 account with an authentication token using OAuth with an Office 365 administrator account. This allows BEC-Guard to access the APIs for all the users associated with the account. Once authenticated, BEC-Guard starts collecting statistics necessary for the impersonation classifier (e.g., number of times a certain user sent an email from a certain email address). The statistics collected by BEC-Guard go back one year. We found that the classifier performs well with as little as one month of historical data.

Online classification. After the warmup phase, BEC-Guard is ready to detect incoming BEC attacks in real-time. To do so, BEC-Guard waits for a webhook API call from any of the users in the organization's Office 365 account. The webhook API calls BEC-Guard anytime there is any new activity for a specific user. When the webhook is triggered, BEC-Guard checks if there is a new received email. If so, BEC-Guard retrieves the email, and classifies it, first using the impersonation classifier, using a database that contains the historical communication statistics unique to each organization. Then, only if it was classified as an impersonation email, BEC-Guard classifies the email using the content classifiers.

If at least one of the content classifiers classifies the email as a BEC attack, BEC-Guard quarantines the email. This is performed by removing the email from the folder where it was received by the user (typically the inbox folder), and moving it into a designated quarantine folder in the end user's mailbox. Since the email is quarantined on the server side, when the user's email clients synchronize the email it will also get quarantined on the user's email clients. In addition, the vast majority of emails get quarantined by BEC-Guard before they are synchronized to the user's email client.

6 Evasion

In this section we discuss attacks that are currently not stopped by BEC-Guard, and evasion techniques that can be used by attackers to bypass BEC-Guard and how they can be addressed.

BEC-Guard is a live service in production, and has evolved rapidly since it was first launched in 2017. We have deployed additional classifiers to augment the ones described in this paper in response to some of the evasion techniques presented below, and the existing classifiers have been retrained multiple times. Another benefit of the API-based architecture is that if we find some attacks were missed by an evasion we can go back in time and find them, and update the system accordingly. The email threat landscape is rapidly changing, and while it is important that the detectors maintain high precision, it is equally important that the security system can be easily adapted and retrained.

6.1 Stopping Other Attacks

BEC-Guard focuses on stopping BEC attacks, in which an external attacker impersonates an employee. However, there are other types of BEC that are not covered by BEC-Guard.

Account takeover. When attackers steal the credentials of an employee, they can login remotely to send BEC emails to other employees. We term this use case "account takeover". There are several approaches to detecting account takeover, including monitoring internal emails for anomalies (e.g., an employee suddenly sending many emails to other employees they typically do not communicate with), monitoring suspicious IP logins, and monitoring suspicious inbox rule changes (e.g., an employee suddenly creates a rule to delete outbound emails) [18–20]. This scenario is not the focus of BEC-Guard, but is covered by our commercial product.

Impersonating both sender name and email without changing reply-to address. It is possible that external attackers could send emails that impersonate both the sender's name and email address, without using a different reply-to address. We have not observed such attacks in our dataset, but they are possible, especially in the case where the attacker asks the recipient to follow a link to steal their credentials. Similar to account takeover, such attacks can be detected by looking for abnormal email patterns. Another possible approach, used by Gascon et al., is to look for anomalies in the actual MIME header [14].

Impersonation of external people. BEC-Guard's impersonation classifier currently relies on having access to the historical inbound email of employees. In order to detect impersonation of external people that frequently communicate

with the organization, BEC-Guard can incorporate emails that are sent from external people to the company.

Text classification in any language. BEC-Guard is currently optimized to catch BEC in languages that appear frequently in our dataset. Both the impersonation classifier and the link classifier are not language-dependent, but the text classifier relies on the TFIDF dictionary is dependent on the language of the labeled dataset. There are a few possible ways to make BEC-Guard's text classifier completely language agnostic. One is to deliberately collect sufficient samples in a variety of languages (either based on user reports or generate them synthetically), and label and train on those emails. Another potentially more scalable approach is to translate the labeled emails (e.g., using Google Translate or a similar tool).

Generic sender names. BEC-Guard explicitly tries to detect impersonations of employee names. However, attackers may impersonate more generic names, such as "HR team" or "IT". This attack is beyond the scope of this paper, but we address it using a similar approach to BEC-Guard in order to detect these attacks: we combine our content classifiers with a new impersonation classifier, which looks for sender names that commonly occur across different organizations, but are sent from a non-corporate email address or have a different reply-to address.

Brand impersonation. Similar to the "generic sender" attack, attackers often impersonate popular online services (e.g., Google Drive or Docusign). These types of attacks are out of the scope for this paper, but we detect them using a similar methodology of combining content classifier, with an impersonation classifier that looks for an anomalous sender (e.g., the sender name has "Docusign", but the sender domain has no relation to Docusign).

6.2 Evading detection

Beyond BEC attacks that BEC-Guard is not designed to detect (as noted above), there are other several ways attackers can try to evade BEC-Guard. We discuss these below and discuss how we have adapted BEC-Guard to address them.

Legitimizing the sender email address. Any system that uses signals based on anomaly detection is vulnerable to attackers that invest extra effort in not appearing "anomalous". For example, when labeling our dataset, we assume that the impersonated employee was not impersonated by the same sender email address more than 100 times. While this threshold is not hard coded into the impersonation classifier, it was a threshold we used to filter emails for the initial training set, and therefore may bias the classifier. Note that we have never observed an attacker impersonating an employee with the same email more than 20 times.

We believe this assumption is valid since BEC-Guard assumes that the organization is already using a volume-based security filter (e.g., the default spam protection of O365 or Gmail or another spam filter), which would pick up a "volumetric" attack. Typically these systems would flag an email that was sent at once from an unknown address to more than 100 employees as spam.

However, a sophisticated attacker may try to bypass these filters by sending a large number of legitimate emails from the impersonated email address to a particular organization, and only after sending hundreds of legitimate emails they would send a BEC using that address. Of course the downside of this approach is that it would require more investment from the attacker, and increase the economic cost of executing a successful BEC campaign. One way to overcome such an attack, is to add artificial samples to the impersonation classifier that have higher thresholds, in order to remove the bias. Of course this may reduce the overall precision of BEC-Guard.

Using infrequent synonyms. Another evasion technique is to send emails that contain text that is different or has a lower TFIDF than the labeled emails used to train our text classifier. For example, the word "bank" has a higher TFIDF, than the word "fund". As mentioned before, one way to overcome these types of attacks is to cover synonyms using a technique, such as word2vec [34].

Manipulating fonts. Attackers have employed various font manipulations to avoid text-based detectors. For example, one technique is to use fonts with a size of zero [35], which are not displayed to the end user, but can be used to obfuscate the impersonation or meaning of the text. Another technique is to use non-Latin letters, such as letters in Cyrillic, which appear similar to the Latin letters to the end user, but are not interpreted as Latin by the text-based detector [16].

In order to deal with these types of techniques, we always normalize any text before feeding it to BEC-Guard's classifiers. For example, we ignore any text with a font size of zero. If we encounter Cyrillic or Greek in conjunction with Latin text, we normalize the non-Latin letters to match the Latin letter that is closest in appearance to it. While these techniques are heuristic based, they have proved effective in stopping the common forms of font-based evasion.

Hiding text in an image. Instead of using text within the email, attackers can hide the text within an embedded image. We have observed this use case very rarely in practice, most likely because these attacks are probably less effective. Many email clients do not display images by default and even when they do, the email may seem odd to the recipient. Therefore, we currently do not address this use case, but a straightforward way to address it would be to use OCR to extract the text within the image.

Using a legitimate reply-to address. As mentioned in §4.4 BEC-Guard relies on a list of legitimate reply-to domains to reduce false positives. This list could potentially be exploited. For example, attackers could craft a LinkedIn or Salesforce profile with the same name of the employee being impersonated and send an impersonation email from that service.

	Precision	FP	Recall
BEC-Guard (Combined)	98.2%	0.000019% (1 in 5,260,000)	96.9%
Impersonation Only	11.7%	0.016% (1 in 6,300)	100%

Table 6: Precision, false positive rate, and recall of BEC-Guard compared to the impersonation classifier alone.

While this is indeed a potential evasion technique, these third party services often have their own anti-fraud mechanisms to stop impersonation. In addition, we believe an impersonation attempt is less likely to succeed if it going through a third-party service, since it would probably seem much less natural than simply sending an email from the email account of the employee. Regardless, we have never seen this evasion technique being used by attackers.

7 Evaluation

In this section, we evaluate the efficacy of BEC-Guard. We first analyze the end-to-end performance of BEC-Guard, using a combination of the impersonation and content classifiers. We then break down the performance of each set of classifiers, and analyze the performance of different classifier algorithms. We also try to estimate the extent of unknown attacks that are not caught by BEC-Guard, by comparing the number of reported missed attacks by customers to the number of true positives.

7.1 End-to-end Evaluation

For the end-to-end evaluation, we randomly sampled emails that were processed by BEC-Guard in June 2018. We manually labeled the emails, and evaluated BEC-Guard's classifiers on the labeled data. We labeled the emails for the evaluation dataset similar to the way we labeled the training data for BEC-Guard's classifiers (see §4.6). We first ran a set of queries that uncover all the BEC attacks that we could find under our labeling assumptions. We then manually labeled the resulting emails, and found 4,221 BEC emails. The entire process took about a week of work for one person. The emails that were not labeled as BEC attacks were assumed to be innocent (In §7.3 we discuss emails that might have been missed by our labeling process).

To evaluate the classifiers, we randomly split the evaluation dataset in half: we used half of the emails for training, and the rest to test the classifiers. The dataset includes 200 million emails from several hundred organizations.

To test the end-to-end efficacy of BEC-Guard, we ran the content classifiers only on the emails that were detected as impersonation emails by the impersonation classifier. Table 6 summarizes the efficacy results. The recall of BEC-Guard is high within the emails we labeled: 96.9% of the BEC emails we labeled were successfully classified by the impersonation classifier as well as one of the content classifiers. The combined false positive rate is only one in 5.3 million emails are falsely detected, which is above our design goal of 1 in a million email. The precision is 98.2%.

Text classifier			
Algorithm	Precision	FP	Recall
Logistic Regression	97.1%	$6.1{\cdot}10^{-5}$%	98.4%
Linear SVM	98.3%	$3.6{\cdot}10^{-5}$%	98.7%
Decision Tree	96.0%	$8.5{\cdot}10^{-5}$%	97.1%
Random Forest	99.2%	$1.7{\cdot}10^{-5}$%	96.4%
KNN	98.9%	$2.3{\cdot}10^{-5}$%	97.5%

Table 7: Text classifier algorithm efficacy using a dictionary of 10,000 words. There is very little difference between the efficacy of the algorithms for the text classifier.

Link classifier			
Algorithm	Precision	FP	Recall
Logistic Regression	33.3%	$85.7{\cdot}10^{-5}$%	96.0%
Linear SVM	92.3%	$3.2{\cdot}10^{-5}$%	90.8%
Decision Tree	94.9%	$2.3{\cdot}10^{-5}$%	96.3%
Random Forest	97.1%	$1.3{\cdot}10^{-5}$%	96.0%
KNN	92.5%	$3.3{\cdot}10^{-5}$%	93.5%

Table 8: Link classifier algorithm efficacy. Random forest provides superior results over the other algorithms.

The false positives of the combined classifiers were due to unlikely incidents where the impersonation classifier detected the email (e.g., due to a personal email address) that also contained anomalous content (e.g., an employee uses a personal email to forward links with low popularity domains to a colleague). Another common false positive occurs when employees leave the organization, and request W-2 forms for tax purposes or other personal information. We plan on addressing such false positives by incorporating features that would indicate whether a sender is no longer an employee of the organization (e.g., if they have stopped sending emails from their corporate address). The false negatives are mostly due to instances where the URL is not deemed suspicious, because it belongs to a domain that got compromised that had a relatively high domain popularity, or because the text of the email is not classified as suspicious. The latter case is typically because the attacker did not use phrases that were similar to any of the BEC attacks that were used to train the text classifier. For example, one of the false negatives asked the recipient for gift card information, which was not a request that was used in any prior attacks.

We also ran the impersonation classifier on the evaluation dataset. Its precision is 11.7%, and its false positive rate is 0.016%. Organizations that are only concerned about recall and have the ability to tolerate a relatively large number of false alerts can run the impersonation classifier on its own. The vast majority of false positives of the impersonation classifier are due to employees using their personal or university (alumni) email addresses.

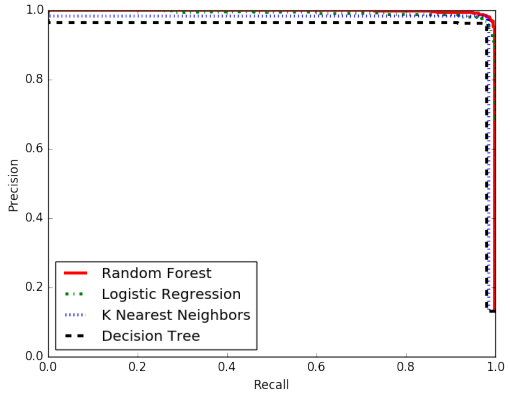

Figure 4: ROC curve of text classifier with different algorithms. All four algorithms perform very similarly, and reach a precision cliff at about 99% recall.

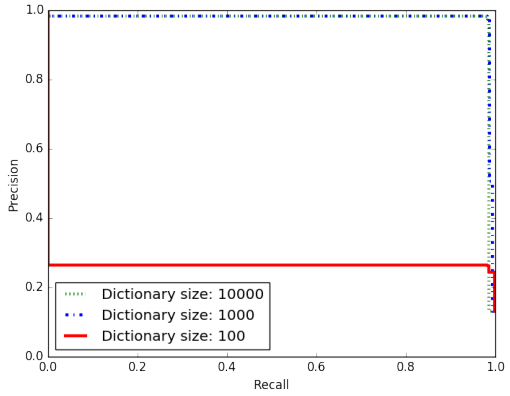

Figure 5: ROC curve of text classifier using KNN with different dictionary sizes. A dictionary size of 1,000 already provides most of the benefit.

7.2 Classifier Algorithms

Table 7 compares the results of the text classifier using different classifier algorithms. As the results show, there is a very small difference between the different classifiers. This is primarily due to the fact that we use a dictionary with a large number of features (10,000). Table 8 shows the results for the link classifier. In the case of the link classifier, random forest more clearly provides superior results than the other classifiers, including KNN. The link classifier is more sensitive to the classification algorithm, because it uses a smaller number of features. Figure 4 presents the ROC curve for four of the classifier algorithms that have a probabilistic output. The ROC curve shows the how each classifier can be tweaked to trade-off precision for recall. All four algorithms behave almost identically: they provide a high level of precision, until a recall level close to 99% where their precision drops. Note that to generate the ROC curves we ran the text classifier only on the emails that were already classified as impersonations. Therefore, its minimum precision in the ROC curve is equal to about 11.7%, which is equal to the precision of the impersonation classifier.

Org	TPs	FNs	Reason
A	31	1	Generic Sender Name
B	4	1	Misclassified Content
C	12	1	External Impersonation
D	8	1	External Impersonation
E	5	1	Misclassified Content
Total	60	5	

Table 9: True positives (TPs) and reported false negatives (FNs) among five organizations, where the administrator has reported at least one false negative.

To analyze the effect of the dictionary size on the classification, Figure 5 plots the efficacy of the text classifier using KNN with different dictionary sizes. The graph shows that most of the marginal benefit is achieved with a dictionary size of 1,000. We observed no noticeable difference in efficacy when using a dictionary larger than 10,000.

7.3 Evaluating Missed Attacks

A general limitation of evaluating imbalanced datasets is that it is difficult to accurately estimate the true false negative rate. In our evaluation dataset, we can only estimate the false negative rate in relation to the data that we labeled. If we missed an attack during labeling, and it was not detected by the classifiers, we would not count it as a false negative.

To deal with "unknown" attacks, our production system allows users to report attacks that it did not detect. We estimate the number of missed attacks *among organizations that have reported missed attacks*. We selected five random organizations that reported missed attacks, and analyzed their detections in the month during which they reported missed attacks. Table 9 provides the number of true and missed detections among these five organizations, as well as the reason for each false negative.

In organization A the attack was missed because the email did not impersonate an employee name, but rather the sender name had a generic title (e.g., "Accountant"). BEC-Guard only detects the impersonation of an employee's name. As we explained in our labeling assumptions (see §4.6), BEC-Guard is only designed to detect attacks that explicitly impersonate an employee name. We speculate that this type of email would be less successful, because the recipient might find it unusual to get an email from a sender name with a generic title, which is not normally used in their company. Nevertheless, our commercial product utilizes other detectors that find "generic titles" as well (see §6). In organization B and E the impersonation classifier successfully detected an impersonation, but the text classifier did not deem the text of the email as suspicious. In both instances, we have since retrained BEC-Guard's text classifiers using the reported emails. In the case of organization C and D, the reported missed email was due to the impersonation of an external colleague (e.g., a vendor the company works with that got impersonated). In §6 we

discuss how to extend BEC-Guard to detect such attacks.

8 Related Work

The growing threat of BEC is widely known and has been described in many in industry and government reports [13, 22, 23]. However, the existing academic work uses very small or synthetic datasets, and suffers from high false positives. In addition, since existing related work is based on limited datasets, it fails to address many of the real-world issues discussed in our paper, such as dealing with the imbalanced dataset, the usage of personal email addresses by employees or "legitimate" impersonations. We believe the reason for the small body of related work is that BEC primarily affects corporate users (not consumers), and it is generally difficult for academic researchers to obtain access to corporate email data.

EmailProfiler [10] builds a behavioral model on incoming emails in order to stop BEC. However, it is based on only 20 mailboxes, has no examples of real-world attacks and does not report false positive rates. In addition, there is prior work on systems that detect emails, which compromise employee credentials with a phishing link [20,45]. There is some overlap between BEC attacks and emails that compromise credentials: in our dataset, 40% of BEC attacks try to phish employee credentials with links. However, the remaining BEC attacks do not contain a phishing link that compromises credentials, and cannot be detected by these systems.

Gascon et al. [14] design a model to stop emails that spoof the domain of the receiver. Similar to BEC-Guard, they base their model on the historical communication patterns of senders. However, in our dataset, spoofing emails represent only about 1% of BEC attacks. Therefore, their model would not catch the other 99% of BEC attacks. The reason domain spoofing represents a small percentage of our dataset, is our dataset only contains emails that were already filtered by an existing spam filter (e.g., Office 365's default filter). Domain spoofing emails contain a mismatch between the sender and reply-to domains, or between the sender domain and the from email envelope. For this reason, traditional spam filters already stop a large number of spoofing emails [33]. In addition, their model is based on a dataset of only 92 mailboxes.

DAS [20] uses unsupervised learning techniques to identify that result in credential theft, which are a subset of BEC attacks. However, it cannot detect attacks that contain only plain text, and is based on a dataset from a single organization with only 19 known attacks. It also suffers from a 0.2% precision, and a much higher false positive rate than BEC-Guard. Similarly, IdentityMailer [45] tries to prevent employee credential compromise by modeling employee behavior, and detecting anomalies in outbound emails. Once an anomaly is detected, the employee is asked to re-authenticate with two-factor authentication. However, their technique suffers from very high false positive rates (1%-8%, compared with 1 in millions of emails in BEC-Guard), and the analysis is based on a small corpus of emails.

Another contemporaneous study done at Barracuda Networks by Ho et al. [18, 19] examines the behavior of attackers using compromised accounts and possible ways to detect account takeover incidents. The techniques presented in this paper are complimentary with the other study, and focus on a different type of attack.

Finally, there is a large body of work on adverserial learning in the context of spam detection [3, 4, 8, 21, 31, 32, 37, 50] that is relevant to our work. In the future, we plan to incorporate some of the evasion techniques introduced in past work, including randomization and the use of honey pots to trick adversaries.

9 Conclusions

BEC is a significant cyber security threat that results in billions of dollars of losses a year. We present the first system that detects a wide variety of BEC attacks at a high precision and false positives, and is used by thousands of organizations. BEC-Guard prevents these attacks in real-time using a novel API-based architecture combined with supervised learning.

One of the main lessons we have learned in developing and deploying BEC-Guard, is that attackers constantly adapt their tactics and approaches. While our supervised learning approach does require continuously retraining our classifiers, and is not fully generalizable, we have found the general approach of using historical email patterns via an API-based architecture has been very useful in quickly developing new classifiers for evolving threats. We have employed a similar approach to the one described in this paper in other contexts, such as detecting brand impersonation, generic sender names and account takeover.

Acknowledgments

We thank Grant Ho, our shepherd, Devdatta Akhawe, and the anonymous reviewers for their thoughtful feedback.

References

[1] R Anglen. First-time phoenix homebuyer duped out of $73k in real-estate scam, 2017. https://www.azcentral.com/story/news/local/arizona-investigations/2017/12/05/first-time-phoenix-homebuyer-duped-out-73-k-real-estate-scam/667391001/.

[2] Manos Antonakakis, Roberto Perdisci, David Dagon, Wenke Lee, and Nick Feamster. Building a dynamic reputation system for DNS. In *Proceedings of the 19th USENIX Conference on Security*, USENIX Security'10, pages 18–18, Berkeley, CA, USA, 2010. USENIX Association.

[3] Marco Barreno, Blaine Nelson, Anthony D. Joseph, and J. D. Tygar. The security of machine learning. *Machine Learning*, 81(2):121–148, Nov 2010.

[4] Marco Barreno, Blaine Nelson, Russell Sears, Anthony D. Joseph, and J. D. Tygar. Can machine learning be secure? In *Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security*, ASIACCS '06, pages 16–25, New York, NY, USA, 2006. ACM.

[5] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, Oct 2001.

[6] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: Synthetic minority over-sampling technique. *J. Artif. Int. Res.*, 16(1):321–357, June 2002.

[7] A. Cidon. Threat spotlight: Spear phishing for mortgages. hooking a big one., 2017. https://blog.barracuda.com/2017/07/31/threat-spotlight-spear-phishing-for-mortgages-hooking-a-big-one/.

[8] Nilesh Dalvi, Pedro Domingos, Mausam, Sumit Sanghai, and Deepak Verma. Adversarial classification. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 99–108, New York, NY, USA, 2004. ACM.

[9] Pedro Domingos. Metacost: A general method for making classifiers cost-sensitive. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 155–164. ACM, 1999.

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

[11] Luca Invernizzi Elie Bursztein, Kylie McRoberts. Tracking desktop ransomware payments end to end. Black Hat USA 2017, 2017. https://www.elie.net/talk/tracking-desktop-ransomware-payments-end-to-end.

[12] FBI. Cyber-enabled financial fraud on the rise globally, 2017. https://www.fbi.gov/news/stories/business-e-mail-compromise-on-the-rise.

[13] FBI. Business email compromise, the 12 billion dollar scam, 2018. https://www.ic3.gov/media/2018/180712.aspx.

[14] Hugo Gascon, Steffen Ullrich, Benjamin Stritter, and Konrad Rieck. Reading between the lines: Content-agnostic detection of spear-phishing emails. In Michael Bailey, Thorsten Holz, Manolis Stamatogiannakis, and Sotiris Ioannidis, editors, *Research in Attacks, Intrusions, and Defenses*, pages 69–91, Cham, 2018. Springer International Publishing.

[15] John A Hartigan and Manchek A Wong. Algorithm AS 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108, 1979.

[16] Alex Hern. Unicode trick lets hackers hide phishing URLs, 2017. https://www.theguardian.com/technology/2017/apr/19/phishing-url-trick-hackers.

[17] L Hernandez. Homebuyers lose life savings during wire fraud transaction, sue Wells Fargo, realtor and title company, 2017. https://www.thedenverchannel.com/money/consumer/homebuyers-lose-life-savings-during-wire-fraud-transaction-sue-wells-fargo-realtor-title-company.

[18] Grant Ho, Asaf Cidon, Lior Gavish, Marco Schweighauser, Vern Paxson, Stefan Savage, Geoffrey M. Voelker, and David Wagner. Detecting and characterizing lateral phishing at scale. In *26th USENIX Security Symposium (USENIX Security 19)*. USENIX Association, 2019.

[19] Grant Ho, Asaf Cidon, Lior Gavish, Marco Schweighauser, Vern Paxson, Stefan Savage, Geoffrey M. Voelker, and David Wagner. Detecting and Characterizing Lateral Phishing at Scale (Extended Report). In *arxiv*, 2019.

[20] Grant Ho, Aashish Sharma, Mobin Javed, Vern Paxson, and David Wagner. Detecting credential spearphishing in enterprise settings. In *26th USENIX Security Symposium (USENIX Security 17)*, pages 469–485, Vancouver, BC, 2017. USENIX Association.

[21] Ling Huang, Anthony D. Joseph, Blaine Nelson, Benjamin I. P. Rubinstein, and J. Doug Tygar. Adversarial machine learning. In *AISec*, 2011.

[22] Infosec Institute. Phishing data – attack statistics, 2016. http://resources.infosecinstitute.com/category/enterprise/phishing/the-phishing-landscape/phishing-data-attack-statistics/.

[23] SANS Institute. From the trenches: Sans 2016 survey on security and risk in the financial sector, 2016. https://www.sans.org/reading-room/whitepapers/analyst/trenches-2016-survey-security-risk-financial-sector-37337.

[24] Nathalie Japkowicz. The class imbalance problem: Significance and strategies. In *Proc. of the Int'l Conf. on Artificial Intelligence*, 2000.

[25] M. Korolov. Report: Only 6% of businesses use DMARC email authentication, and only 1.5% enforce it, 2016. https://www.csoonline.com/article/3145712/security/.

[26] Miroslav Kubat, Robert C Holte, and Stan Matwin. Machine learning for the detection of oil spills in satellite radar images. *Machine learning*, 30(2-3):195–215, 1998.

[27] Miroslav Kubat, Stan Matwin, et al. Addressing the curse of imbalanced training sets: one-sided selection. In *ICML*, volume 97, pages 179–186. Nashville, USA, 1997.

[28] M. Lan, C. L. Tan, J. Su, and Y. Lu. Supervised and traditional term weighting methods for automatic text categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(4):721–735, April 2009.

[29] David D Lewis and Jason Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of the eleventh international conference on machine learning*, pages 148–156, 1994.

[30] Charles X Ling and Chenghui Li. Data mining for direct marketing: Problems and solutions. In *KDD*, volume 98, pages 73–79, 1998.

[31] Daniel Lowd. Good word attacks on statistical spam filters. In *Proceedings of the Second Conference on Email and Anti-Spam (CEAS*, 2005.

[32] Daniel Lowd and Christopher Meek. Adversarial learning. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, KDD '05, pages 641–647, New York, NY, USA, 2005. ACM.

[33] Microsoft. Anti-spoofing protection in Office 365, 2019. https://docs.microsoft.com/en-us/office365/securitycompliance/anti-spoofing-protection.

[34] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc., 2013.

[35] Yoav Nathaniel. ZeroFont phishing: Manipulating font size to get past Office 365 security, 2018. https://www.avanan.com/resources/zerofont-phishing-attack.

[36] C Northern. Nickname and diminutive names lookup, 2017. https://github.com/carltonnorthern/nickname-and-diminutive-names-lookup.

[37] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami. The limitations of deep learning in adversarial settings. In *2016 IEEE European Symposium on Security and Privacy (EuroS P)*, pages 372–387, March 2016.

[38] Michael Pazzani, Christopher Merz, Patrick Murphy, Kamal Ali, Timothy Hume, and Clifford Brunk. Reducing misclassification costs. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 217–225, 1994.

[39] N. Perlroth. Hackers are targeting nuclear facilities, Homeland Security Dept. and F.B.I. say, 2017. https://www.nytimes.com/2017/07/06/technology/nuclear-plant-hack-report.html.

[40] J. Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[41] J.J. Roberts. Facebook and Google were victims of $100m payment scam, 2017. http://fortune.com/2017/04/27/facebook-google-rimasauskas/.

[42] G. Salton and M. J. Mcgill. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA, 1986.

[43] Z. Song and N. Roussopoulos. K-nearest neighbor search for moving query point. pages 79–96, 2001.

[44] United States Securities and Exchange Commission. Form 8-k, 2015. https://www.sec.gov/Archives/edgar/data/1511737/000157104915006288/t1501817_8k.htm.

[45] Gianluca Stringhini and Olivier Thonnard. That ain't you: Blocking spearphishing through behavioral modelling. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 78–97. Springer, 2015.

[46] Andrew Trask, Phil Michalak, and John Liu. sense2vec - A fast and accurate method for word sense disambiguation in neural word embeddings. *CoRR*, abs/1511.06388, 2015.

[47] Gary M Weiss and Haym Hirsh. Learning to predict rare events in event sequences. In *KDD*, pages 359–363, 1998.

[48] Colin Whittaker, Brian Ryner, and Marria Nazif. Large-scale automatic classification of phishing pages. In *NDSS '10*, 2010.

[49] C. Willems, T. Holz, and F. Freiling. Toward automated dynamic malware analysis using CWSandbox. *IEEE Security Privacy*, 5(2):32–39, March 2007.

[50] Gregory L. Wittel and S. Felix Wu. On attacking statistical spam filters. In *Proceedings of the Conference on Email and Anti-Spam (CEAS)*, 2004.

[51] Gang Wu and Edward Y Chang. Class-boundary alignment for imbalanced dataset learning. In *ICML 2003 workshop on learning from imbalanced data sets II, Washington, DC*, pages 49–56, 2003.