A Sender-Centric Approach to Detecting Phishing Emails

Fernando Sanchez and Zhenhai Duan
Florida State University
{*sanchez, duan*}*@cs.fsu.edu*

***Abstract*—Email-based online phishing is a critical security threat on the Internet. Although phishers have great flexibility in manipulating both the content and structure of phishing emails so that they can be made as close as possible to legitimate messages, phishers have much less flexibility in concealing the information related to the sender of a phishing message. Such sender-related information is often inconsistent with the target institution of the phishing email, and thus, can help detect phishing emails. In this paper we advocate and develop a sender-centric approach to detecting phishing emails by focusing on the information related to the sender of a message instead of the content or structure of the message. Our evaluation studies based on real-world email traces show that the sender-centric approach is a feasible and effective method in detecting phishing emails. For example, using an email trace consisting of both phishing and legitimate messages, the sender-centric approach can detect 98.7% of phishing emails with a zero false positive rate, that is, the approach can correctly classify all legitimate messages while achieving a high detection rate of phishing emails.**

I. Introduction

Email-based online phishing is a critical security threat on the Internet, which greatly deteriorates the usefulness and trustworthiness of the Internet email system. Unlike regular spam emails that are mainly annoyous, phishing emails represent a more severe security threat because of their criminal nature and their direct financial impacts on Internet users. In general, phishing attacks attempt to impersonate legitimate institutions to communicate with their clients. Given the great flexibility that phishers have in manipulating both the content and structure of phishing emails, phishers can create phishing messages that are as close as possible to legitimate messages from target institutions, and lure unsuspecting users into providing them with private personal information, such as online banking credentials and social security numbers.

In response, a large number of anti-phishing techniques have been developed in recent years, including various browser toolbars and (content-based) spam filters [21], [17], [3], [6]. However, both the number and sophistication of phishing attacks have been on a continuous rise on the Internet. For example, a recent report from RSA [15] showed that the number of phishing attacks in the year of 2011 increased 37% compared to that in the year of 2010, and approximately one in every 300 emails delivered on the Internet in the year of 2011 was a phishing message. The increase of the number of phishing attacks in recent years can be attributed to a few factors, including the availability of sophisticated tools to automate phishing attacks and potential great financial gains in performing phishing attacks. For example, an RSA report estimated that, on average, phishers can obtain $\$4\,500$ in stolen funds in each phishing attack [15].

On the other hand, despite the advances in the sophistication of phishing attacks, phishing emails often contain suspicious information that can separate phishing emails from legitimate emails. In particular, although phishers have great flexibility in manipulating both the content and structure of phishing emails, they have much less flexibility in completely concealing the information related to the sender of a phishing email, for example, the message delivery path of the email as carried in the `Received:` header fields [10]. Such sender-related information is often inconsistent with the target institution of the phishing email, and thus, can help detect phishing emails. For example, it looks suspicious if a message concerning a US account at the Bank of America was originated from or traversed a foreign country. In this paper we advocate and develop a sender-centric approach to detecting phishing emails by focusing on the information related to the sender of a message instead of the content or structure of the message. In Section III, we will detail the sender-related information carried in an email.

In developing the sender-centric approach to detecting phishing emails in this paper, we will focus on phishing emails that target banks. As reported by RSA [15], more than half of the phishing attacks in the year of 2011 targeted financial institutions, and moreover, such phishing attacks normally have direct financial consequence if they are successfully carried out. Legitimate banking messages have certain properties that can simplify the initial design of the sender-centric approach. For example, the network domain names of banks are relatively stable, and banks tend to host their own mail and web servers. We note that the sender-centric approach can be extended to handle non-banking related phishing attacks (we discuss the extensions in Section V).

The sender-centric approach we develop in this paper is a two-step system. In the first step, the system will separate banking messages from non-banking messages. We note that, in this step, the banking messages we identify can be either legitimate or phishing emails. The goal of this step is not to determine if a message is a phishing email; but rather, to isolate (legitimate or phishing) banking messages from non-banking messages, so that we can focus on only banking messages in the second step of the system. Given that phishers always try to craft phishing messages as close to the legitimate

messages as possible, it is relatively easy to separate banking messages from non-banking messages. If phishers do not mimic legitimate banking messages, users can easily recognize such phishing emails and the effectiveness of phishing attacks will be greatly impacted. To this end, we develop a simple Support Vector Machine (SVM)-based classifier to separate banking messages from non-banking messages [16], utilizing a set of features extracted from email messages.

After banking messages have been separated from non-banking messages, in the second step of the system, we develop a set of heuristic rules to identify suspicious sender-related information so as to detect phishing emails. Given the concerned bank of a banking message, we examine if the sender-related information (for example, the delivery path of the message) is consistent with the concerned bank. As we have discussed above, although phishers can easily manipulate both the content and structure of a phishing email to mimic a legitimate banking message, it is much harder for them to completely conceal the sender-related information, and to make it consistent with the target bank.

In addition to presenting the details of the sender-centric approach to detecting phishing emails, we also evaluate the performance of the approach using a number of real-world data sets, including both legitimate and phishing banking emails, and non-banking emails. The evaluation studies show that the sender-centric approach is a feasible and effective system in detecting phishing emails. For example, using a data set containing both banking and non-banking messages, in the first step of the system, we can successfully separate banking messages from non-banking messages with a 98.87% accuracy. In the second step of the system, using a data set containing both legitimate and phishing banking messages, we can detect 98.7% of phishing emails with a zero false positive rate, that is, we do not misclassify any legitimate banking messages as phishing emails while maintaining a high detection rate of phishing emails.

The remainder of the paper is organized as follows. In Section II we briefly discuss the related work. In Section III we describe the design of the sender-centric approach to detecting phishing emails. We present the evaluation studies in Section IV. We discuss the implications and potential evasion techniques of the sender-centric approach in Section V, and conclude the paper in Section VI.

II. Related Work

In this section we discuss related work on detecting phishing attacks. Over the years, a large number of spam filters have been developed [1], [7], [17]. Although spam filters can be used to detect phishing emails, they normally target the filtering of general spam emails instead of phishing emails. Given that phishing emails are normally crafted as close as possible to legitimate emails, general spam filters can only achieve limited success in detecting phishing emails without misclassifying legitimate emails.

In recent years, a number of content-based filters have been designed specifically for phishing emails. In [3] the authors developed a scheme to filter phishing emails based on the structural properties of phishing emails. It mainly considered the linguistic properties that separate phishing emails from other emails. A scheme was developed in [6] to filter phishing emails based on the features of phishing emails including IP-based URLs and the age of domain names. However, identifying a set of content and structural properties that can separate legitimate messages from phishing messages is a challenging problem. As we have discussed above, in order to convince users that a message is from a target institution, a phisher will often craft the phishing email as close to the legitimate message as possible, in terms of both content and structure. It is challenging to develop a system to have a high detection rate of phishing emails while maintaining a low (ideally, zero) false positive rate.

Moreover, filtering phishing emails based on the content (and structure) of emails allows for an arms race where a phisher can mimic those properties of a legitimate message to evade phishing filters. Instead of relying on the content and structure of emails to detect phishing emails, the sender-centric approach detects phishing emails based on the sender-related information, which phishers have much less flexibility to manipulate.

Many web browser-based toolbars have been developed (see [21] and references therein). However, as reported in [21], existing anti-phishing toolbars have poor performance in terms of both false positive and false negative rates. In addition, it is often too late to detect a phishing attack when a fraudulent web site has been visited, and users may be exposed to undesirable consequences (for example, with spyware or malware being installed on their machines). Ideally, phishing attacks should be identified at the first stage of an attack, which normally uses emails to reach potential victims. Moreover, browser-based toolbars only target phishing scams involving web sites as part of the attack vector. Many recently popular phishing attacks do not use web sites as part of the attack vector. Instead, they ask the recipients to directly reply to the email senders.

MDMap developed in [5] is a system that reveals the suspicious sender-related information using a geographical map to assist average Internet users to identify potential phishing emails. MDMap requires users to be included in the process of phishing detection. The sender-centric approach developed in this paper is a fully automated phishing detection system. In addition, we also develop a more comprehensive set of sender-related information compared to that of MDMap to detect phishing emails.

III. Methodology

In this section, we describe the sender-centric approach to detecting phishing emails. We first provide an overview of the approach, and then discuss the two components of the approach in details.

A. Overview of Sender-Centric Approach

As we discussed in Section I, we focus on detecting phishing emails that target banks. The sender-centric approach

we develop is a two-step system. In the first step, we try to separate (legitimate or phishing) banking messages from non-banking messages. Given that phishers always craft phishing messages as close as possible to legitimate messages, legitimate banking (LB) messages and phishing banking (PB) messages will be extremely similar to each other in terms of both content and structure. Moreover, they tend to have some distinguishing features separating them from non-banking messages. To this end, we will develop a simple Support Vector Machine (SVM) based classifier to separate the two types of messages [16], based on a set of features extracted from the content and structure of emails.

Although content-based filters have various limitations in accurately detecting spam messages and are constantly evaded by spammers, they do not have such limitations in the situation we use them. We note that spammers can constantly evade content-based filters by adapting the content and structure of spam messages. However, in phishing attacks, they must make phishing emails as close as possible to legitimate messages. If a PB message is drastically different from LB messages, a user can already identify it as suspicious and ignore it. As a consequence, we can use content-based filters to effectively separate banking messages from non-banking messages. We will detail the SVM-based classifier in Section III-B.

After banking messages have been separated from non-banking messages, in the second step of the system, we develop a set of heuristic rules to determine if the sender-related information carried in a message is consistent with the concerned bank. We note that although phishers have great flexibility in manipulating both the content and structure of a phishing email, they cannot completely conceal the sender-related information. For example, the first external mail server information can always be determined (the first external mail server of a message is the external mail server that delivers the message into the mail server belonging to the destination network). Moreover, in certain types of phishing attacks, the phishers also want the recipients to send sensitive information back to the attacker by replying to the message, and in these cases, the `Reply-To:` or the `From:` address must be valid and belong to the phisher. In addition, in some other phishing attacks, a fraudulent web site needs to be hosted somewhere on the Internet. We verify any sender-related information carried in the email to determine if it is consistent with the concerned bank, including email addresses (such as `Reply-To:` address), claimed message delivery path (i.e., the `Received:` headers), and URL links in the message body, if any. In Section III-C we will detail the heuristic rules to detect phishing emails and email fields used to determine the sender-related information.

The two-step system of the sender-centric approach puts phishers into a dilemma and helps to effectively detect phishing emails. As we have discussed above, if phishers do not craft phishing emails as close as possible to legitimate messages, users can detect and ignore these phishing emails. On the other hand, if they make their phishing emails as close as possible to legitimate messages, the SVM-based classifier can easily classify them as banking messages, and the second step of the system can detect them as phishing emails with a high probability. Although phishers have great flexibility in manipulating both the content and structure of a phishing message, it is much harder for them to manipulate the delivery infrastructure of the message (and the hosting web server if it is required as part of a phishing attack) to be consistent with the delivery infrastructure of the corresponding legitimate messages. We note that the sender-centric approach can be implemented as an additional component of existing spam filters such as SpamAssassin [17].

B. Separating Banking Emails from Non-Banking Emails

In this subsection, we describe the first step of the sender-centric approach, that is, the SVM-based classifier to separate banking emails from non-banking emails. We will first provide the necessary background on SVM, and then discuss the features that we use to classifier the two types of messages.

1) Support Vector Machine: SVM is a popular machine learning method due to its robust performance in a wide range of problem domains. In essence, SVM is a linear learning algorithm by transforming input data (where the decision function may not be linear) into a high-dimensional feature space (where the decision function is linear), using a mapping function $\phi(x)$. In reality, the mapping function is never performed by using the so called "kernel trick", which is used to compute the dot product of two data points in the mapped high-dimensional feature space. There are a number of well studied kernel functions for this purpose. To a degree, kernel function of two data points measures the similarity (closeness) between the data points.

Given a set of training data containing m data points (x_i, y_i), for $i = 1, 2, \ldots, m$, where $x_i \in R^n$, and $y_i = \{-1, +1\}$, SVM aims to identify the linear separating hyperplane with the maximum margin of separation between the two classes of the data points in the mapped high-dimensional feature space. This choice of separating hyperplane has the property of providing optimal generalization in the sense that the classification error on unseen data points can be minimized, when we use the trained model to predict the class of the unseen data points. Formally, SVM requires to solve the following (primal) optimization problem

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}\xi_i \\ \text{subject to} \quad & y_i(\phi(x_i)\mathbf{w} + b) \geq 1 - \xi_i \\ & \xi_i \geq 0 \\ & \text{for } i = 1, 2, \ldots, m \end{aligned}$$

The above formulation of the optimization problem is the C-classification soft margin classifier in order to accommodate the situation where a linear separating hyperplane does not exist. We may also want to use the soft margin classifier to prevent the overfitting issue so that the model can be better generalized to predict new data points. In this formulation, ξ_i

is the slack variable to what degree we allow a training data point (x_i, y_i) to fall between the margin of the two classes, and C is the penalty parameter controlling the error terms. Note also that in reality, the above primal optimization problem is not directly solved to obtain the separating hyperplane; instead, the Lagrangian dual optimization problem is solved to obtain the model. We refer interested readers to the details of SVM to [16].

2) Message Classification Features: In this section we describe the features that we use to separate banking messages from non-banking messages. As we have discussed, phishing emails targeting banks will be made as similar to legitimate banking messages as possible, so that unwitty recipients can be tricked to provide sensitive personal information to attackers. On the other hand, banking messages have some distinguishing features that separate them from non-banking messages. In order to separate banking messages (legitimate or phishing) from non-banking messages, we collect a set of 31 message features, and group them into three categories. In the following we describe the message features that we use in separate banking messages from non-banking messages.

- **Message format**: Banking messages tend to use some formatting of the content to display the logo or design of the corresponding bank. For this reason, it is common for banking messages to be in the HTML format. We use a binary feature to denote if a message is in the HTML format.
- **Number of URLs**: Banking messages tend to model some graphical representation of the institution's web page. As a consequence, images or hyperlinks to the actual web site are normally included in the body of the message. We use the number of URLs used in the message as a feature.
- **Keywords**: Given the nature of banking messages, they often contain some distinctive words, such as *account*, *bank*, *customer*, etc. We collect a set of 29 keywords in this category, as listed in Table I. Notice that for each keyword we use only the stemmed word (or regular expression) [2]. Similarly, we transform the content of a message to stemmed words first, before applying the SVM-based classifier.

We note that some of the features have been proposed and used in other anti-phishing schemes [3], [6]. However, they are used in a different context. In these schemes, they are used to separate phishing messages from legitimate messages. While in our case, they are used to separate banking messages from non-banking messages. We note that classifying between phishing and legitimate messages using these features often result in a low detection rate or a high false positive rate, given that both the content and structure of phishing emails and legitimate (banking) messages are similar. On the other hand, banking messages and non-banking messages have distinct features, and therefore, we do not have the limitations in using these features as in the previous work.

TABLE I
MESSAGE CLASSIFICATION KEYWORDS.

account	access	bank	credit
click	ident	inconvini	inform
limit	password	helpdesk	servic
recent	statement	updat	confirm
verifi	user	custom	client
log(in)?	usernam	member	secur
SSN	suspend	restrict	hold
disput			

C. Heuristics to Detect Phishing Emails

After a message has been classified as a banking message using the SVM-based classifier, in the second step we develop a set of heuristic rules to determine if the sender-related information carried in the message is consistent with the concerned bank of the message. We have a set of 3 heuristic rules to detect phishing emails, and we denote them as R1, R2, and R3, respectively. The heuristic rules developed in this section are based on a few properties of banking messages. As we discussed in Section I, banks normally have their own network domain names and these domain names are relatively stable. This is especially true for large banks such as Bank of America and CitiBank, which are often the target of phishing attacks. Second, banks tend to host their own mail servers and web servers within their own network domains.

A banking message is passed to the three rules sequentially. If a message is flagged as a phishing email by any of the rules, it does not need to be passed to the remaining rules. A message is classified as legitimate if it can successfully pass all three rules. In the following we will describe them one by one, and detail the fields of a message that we consider as related to the sender of the message.

1) R1: Emails Accounts from Public Email Service Providers: Given the ease to obtain a (free) email account from public email service providers (ESP) such as Yahoo! Mail, Hotmail, and Gmail, phishers increasingly rely on public ESP accounts to communicate with potential victims in recent times. To mislead recipients of such messages, phishers often use the name of the target bank as part of the email account name and the full user name of the email account. Email clients normally only display the sender's name instead of email address, which further helps phishers to mislead unwitty recipients. This provides a convenient approach for phishers that do not have their own infrastructures or their own mail servers. On the other hand, it is unlikely that a bank will use ESP accounts to communicate with their clients. Therefore, our first rule R1 will claim any banking messages containing ESP accounts in any of the following three header fields as phishing: `From:`, `Reply-To:`, and `Return-Path:`.

2) R2: Sender Geographical Locations: One way for phishers to be effective and to hide their true identities is to utilize a wide spread distribution of resources, both for sending phishing messages and for hosting the fraudulent web sites mimicking the legitimate web server. Such wide spread resources can be obtained, for example, from networks of compromised

Algorithm 1 R2: Geographic location consistency.

1: Given a message m
2: //Step 1: Email addresses
3: Obtain CC of `Return-Path:`, `From:`, `Reply-To:` headers
4: **if** (CC of the 3 headers are inconsistent) **then**
5: **return** m as phishing
6: **end if**
7: //Step 2: URLs
8: Extract URLs url_i ($i = 1, 2, \ldots, n$)
9: **for** ($i = 1$ to n) **do**
10: **if** ($Country(url_i)$! $= CC$) **then**
11: **return** m as phishing
12: **end if**
13: **end for**
14: //Step 3: Message delivery path
15: Extract `Received:` fields hr_i ($i = 1, 2, \ldots, k$)
16: //up to and include the first external mail server
17: **for** ($i = 1$ to k) **do**
18: Extract $From$ host from hr_i
19: **if** ($Country(From)$! $= CC$) **then**
20: **return** m as phishing
21: **end if**
22: Extract By host from hr_i
23: **if** ($Country(By)$! $= CC$) **then**
24: **return** m as phishing
25: **end if**
26: **end for**
27: **return** m as $PASSED$

machines or botnets [20]. For example, a phishing message claiming to be from a bank in the USA, could have been generated and sent from a compromised machine in Europe, and redirect the recipient to a fake site hosted in Asia. On the other hand, legitimate banks will be themselves responsible for all the line to service their clients, from the generation of the messages to the hosting of their web sites. We note that another potential reason for phishers to utilize machines in a foreign country than that of the target bank is to avoid or minimize the potential legal consequence in performing phishing attacks.

We develop our second rule **R2** to detect this kind of inconsistencies based on the geographic location of hosts or domains involved in the origin and delivery of email messages. In this paper, we consider the geographic location of a host or domain as the country to which the corresponding IP address or domain name has been registered. That is, the geographic location is considered at the country level. We make this decision for two reasons. First, based on our preliminary studies of phishing emails, a large portion of phishing emails were originated from a foreign country than that of the target bank. Second, it is relatively easy to obtain the accurate country level information of a domain or IP address. Should this phishing behavior be changed after the sender-centric approach is deployed, finer-grained geographic location information can be used (see also rule **R3** in the next subsection). Algorithm 1 summarizes **R2**. In the algorithm (and other algorithms in this section), *CC* stands for Country Code [18]. We say two concerned fields are consistent if CCs of the two fields are the same (that is, they are in the same country).

When a phisher attempts to impersonate a legitimate bank, he often fakes the sending address, for example the `From:` field, to make it look like that it belongs to the target institution. Our **R2** rule first obtains the geographic location of the network domain of the claimed email addresses found on the headers, in order, `Return-Path:`, `From:`, `Reply-To:` (Lines 3 to 6). The location for all three should be consistent, that is, in the same country. We note that a subset of phishing emails rely on email communications between the phishers and recipients to carry out the phishing attack, instead of relying on redirecting recipients to a fraudulent web site. For such phishing attacks, at least one of the above three email addresses must be a valid email address belonging to the phisher. We observe an increasing number of phishing emails relying on this technique.

Another common practice used by phishers is to host a fake web site to which a victim will be redirected to update her information. Normally these web sites will be hosted on compromised servers, or leased servers in countries with less restrictive laws. For this reason, it is likely that the country where the sites are hosted is different from where the target institution is. We examine all URLs found on the body of the message to determine if they are consistent with the (country-level) geographic location of the claimed sender email addresses (Lines 8 to 13).

The other resource in hands of phishers is the large pool of compromised machines used to generate and send the messages. Like with hosting sites, these sending machines are widely distributed around the world, and it would be harder, in terms of cost, to coordinate a campaign with only machines within the geographic proximity of the target bank. (Indeed, currently, phishers may prefer to avoid using machines within the same jurisdiction as that of the target bank.) As the last component of this rule, we verify if the claimed hosts in the message delivery path as carried in the `Received:` header fields (up to and include the first external mail server) are consistent with the geographic location of the claimed sender email addresses (Lines 15 to 26).

We note that when we look for the country code of a host, two possible errors can occur. One is that the country code cannot be determined (perhaps the corresponding network domain has been taken down); another is that the corresponding country code does not match with that of the sending email address of the message. We consider both cases as being inconsistent and flag a message as phishing. Table II lists the detailed errors that cause a message to be flagged as phishing. The *Steps* in the table correspond to steps in Algorithm 1.

3) R3: Authorized Sender: In this rule, we attempt to determine the bank from which the message claims to be, and verify if the sending machine (i.e., the first external mail server) is an authorized one for this institution. To determine the bank from which a message claims to come, we extract

TABLE II
TYPE OF ERRORS IN R2.

Step	Error
1	CC of email addresses does not match CC of email address is undefined
2	CC of a URL on the body does not match CC of URL is undefined
3	CC on delivery path does not match CC on delivery path is undefined

Algorithm 2 R3: Authorized sender

```
1: Given a message m and a list of banks BL
2: // Step 1: determining bank
3: for (mh in (Return-Path:, From:, Subject, Reply-to:, Body URLs)) do
4:    if (mh match in BL) then
5:       Bank b = best match from BL
6:       break
7:    end if
8: end for
9: if (undefined bank b) then
10:   return m as phishing
11: end if
12: // Step 2: FEMTA is authorized?
13: Extract FEMTA mta from Received: headers
14: SPF query (q) using domain for b and host mta
15: if (q == pass) then
16:   return m as PASSED
17: else if (q == neutral or none) then
18:   //Step 3: message delivery path
19:   CC ← Country code for Bank b
20:   Extract Received: fields hr_i (i = 1, 2, ..., k)
21:   // Up to and include FEMTA
22:   for (i = 1 to k) do
23:      Extract From host from hr_i
24:      if (Country(From) != CC) then
25:         return m as phishing
26:      end if
27:      Extract By host from hr_i
28:      if (Country(By) != CC) then
29:         return m as phishing
30:      end if
31:   end for
32: else
33:   return m as phishing
34: end if
35: return m as PASSED
```

certain information from the message, and use the *Levenshtein edit distance* [11] to compare with a list of known banks. Once the bank has been identified, we test if the sending machine is authorized to originate messages for the institution's domain relying on the Sender Policy Framework (SPF) [19]. Note that we do not test SPF on the claimed domain, but we use the known domain for the particular bank.

We take this approach to determine the bank and to determine if the sending machine is authorized to send messages on behalf of the bank domain for a number of reasons, based on our investigation of personal legitimate banking messages and a large number of banks. First, each bank may have multiple network domain names, due to various reasons, for example, merging of banks or partitioning of functionality. Therefore, there may not be a one to one mapping between the bank name and the network domain name belonging to the bank. Second, we also cannot just rely on SPF to determine if the sending machine is authorized for the concerned bank, given that phishers can register a domain name that is close to the domain name of the target bank and configure the SPF records to point to the sending machine under control of phishers. The approach we develop can properly handle both situations.

If we cannot decide if the machine is authorized based on SPF (for example, the institution has not published SPF records), we verify if the delivery path of the message is consistent with the country where the target bank belongs (similar to what we have done in **R2**). A description of this rule is shown in Algorithm 2. In the algorithm, FEMTA stands for the first external mail server (i.e., the external mail server that delivers the message to the destination network). In Lines 3 to 11, we try to determine the bank using the message header fields `Return-Path:`, `From:`, `Subject:`, `Reply-To:`, and URLs in the message in that order. Given that multiple banks may match the message, we choose the one that best matches the message.

In Lines 13 to 16, we use SPF to determine if the sending machine is authorized to send messages for the concerned bank. In the rest of the algorithm (Lines 17 to 34), we check if the message delivery path is consistent with the identified bank. This step is similar to that of Algorithm 1. The difference is that, in Algorithm 2, we compare the message delivery path against the identified bank, instead of the claimed email addresses as in Algorithm 1.

It is important to note that this rule attempts to identify the bank that a phisher is attempting to impersonate. Since phishers need to convince users about the false origin of a message, identifying the proper institution is not a complex task. Using this match, the only way an attacker can be successful will be if the real bank mail server is compromised to send the messages.

Similarly, we also have multiple cases where the rule can flag a message as phishing. First, if we cannot identify the bank, we flag the corresponding message as phishing. This should not occur if the bank list is sufficiently large. Next, a message can be flagged as phishing if the SPF query returns a failure message. Finally, if we cannot use the response from SPF, for example, if the bank has not published its SPF records, a message can be flagged as phishing if a host on the message delivery path does not match with the country location of the bank, or if we cannot obtain the country location for the host in the delivery path. Table III summarizes the different types of errors in R3 that can cause a message to be flagged as phishing. The *Steps* in the table correspond to steps in Algorithm 2.

TABLE III
TYPE OF ERRORS IN R3.

Step	Message
1	Bank cannot be identified
2	SPF query returned a fail code
3	CC on delivery path does not match CC on delivery path is undefined

TABLE IV
DATA SETS

Data Set	# of Messages
Nazario	1001
EasyHam	1400
PersonalPB	50
PersonalLB	10

Together, the three heuristic rules can greatly limit the flexibility of phishers in misleading recipients the where-about of the real sender of a phishing message.

IV. PERFORMANCE EVALUATION

In this section we conduct experimental studies to evaluate the performance of the sender-centric approach. We first describe the data sets used in the experimental studies, and then we describe the performance metrics and software packages we use to carry out the studies. Given that the sender-centric approach is a two-step system, we correspondingly organize the evaluation studies into two groups. In the first one, we study the performance of the SVM-based classifier to separate banking messages from non-banking messages. In the second group, we investigate the performance of the three heuristic rules to separate PB messages from LB messages.

A. Data Sets

We rely on four different data sets to carry out the evaluation studies of the sender-centric approach. The first one is a phishing corpus containing phishing messages collected between 2005 and 2008 by Jose Nazario [13]. We manually pre-processed the corpus to remove the phishing messages that are not related to banks. To simplify our evaluation studies, we also removed the phishing messages that are not written in English. (Note that this is just for the convenience of our evaluation studies. The sender-centric approach can be extended to handle messages not written in English.) After removing these messages, we have a total of 1001 phishing messages from the corpus, and we refer to this set of phishing messages as the Nazario archive.

The second data set we used is the most recent EasyHam corpus from the SpamAssassin project [14], which contains 1400 non-spam messages and is referred to as the EasyHam archive. Finally, from the phishing and legitimate bank messages contained in our own personal mailboxes, we manually select messages that are (claimed) from a bank and written in English. From this we obtain 50 phishing banking messages, and 10 legitimate banking messages. We refer to them as the PersonalPB and PersonalLB archives, respectively. We note that the phishing messages in PersonalPB were collected between 2008 and 2011, which are newer than the ones in Nazario. Table IV summarizes these data sets.

B. Performance Metrics and Software Packages

In this section we describe the performance metrics and software packages we use in the experimental studies. We focus on describing how the experiments in the first group are carried out to evaluate the performance of the SVM-based classifier to separate banking messages from non-banking messages, and the corresponding performance metrics and software packages. We then briefly discuss the experiments in the second group to separate PB messages from LB messages.

For an experiment in the first group, we merge the Nazario and the EasyHam archives to form a new data set containing both banking and non-banking messages. We partition the new data set into two subsets: a training set and a test set. We use the training set to train the SVM-based classifier, and then apply the trained SVM model to the test set to evaluate the performance of the trained SVM-based classifier. We use three common metrics to evaluate the performance of the SVM-based classifier. The first one is *detection accuracy* (or simply accuracy), which measures the percentage of total messages that are classified correctly. More formally (all numbers are with regard to the test set)

$$\text{Accuracy} = \frac{\text{\# of messages classified correctly}}{\text{Total \# of messages}}. \quad (1)$$

The second one is *false positive rate (FPR)*, which measures the percentage of non-banking messages that are misclassified. The last metric is *false negative rate (FNR)*, which measures the percentage of banking messages that are misclassified. More formally (similarly, all numbers are with regard to the test set)

$$\text{FPR} = \frac{\text{\# of non-banking messages misclassified}}{\text{Total \# of non-banking messages}}, \quad (2)$$

$$\text{FNR} = \frac{\text{\# of banking messages misclassified}}{\text{Total \# of banking messages}}. \quad (3)$$

For validation purposes, we repeat each experiment forty times (with different partitioning of training and test sets) and report both the average and standard deviation for each of our metrics.

Next we describe the software packages we use and the configurations. We use the SVM classifier included in the `e1071` package of the R programming language [9], which in essence provides a wrapper interface to the widely used SVM implementation `libsvm` [4]. For all our experiments we use the C-classification SVM with the Gaussian Radial Basis Function (RBF) kernel $k(x, x') = exp(-\gamma\|x - x'\|^2)$, given the reported robust performance of this kernel function.

For training the SVM classifier, we need to specify two parameters, the γ value in the kernel function, and C the penalty value (Section III-B). We rely on the `tune` interface

included in the package to identify the optimal values of the two parameters in a specified range ($C = (2^2, 2^8)$, and $\gamma = (2^{-2}, 2^2)$). The `tune` interface aims to identify the optimal parameters to minimize the classification error, by performing a grid search over the specified parameter range. Note that the tuning is only applied to the training data set. After obtaining the optimal values for the parameters C and γ, we re-train the SVM classifier using the optimal parameters to obtain the final SVM model used to predict the machines in the test set.

Recall that words in email messages are stemmed before being used in any experiments. The stemming of words was performed using the *Lingua::Stem::Snowball* package in Perl [2].

For the experiments in the second group, we use email messages contained in the Nazario, PersonalPB, and PersonalLB archives, which are all (legitimate or phishing) banking messages. We use the similar performance metrics, namely accuracy, false positive rate, and false negative rate as we have defined above, but with banking messages replaced by PB messages and non-banking messages replaced by LB messages in the definitions, respectively. In order to obtain geographic information of an IP address or network domain, we use the MaxMind GeoIP Perl API [12]. To find the *Levenshtein edit distance* we use the *String::Approx* [8] Perl package, using the default 10% *approximateness*.

C. Performance of SVM-based Classifier

The first step of the sender-centric approach is to classify between banking and non-banking messages. For this we have developed an SVM-based classifier using the feature set described in III-B. In this section we evaluate the performance of the SVM-based classifier using the combined data set containing both the Nazario and EasyHam archives. The combined data set contains 2401 message in total. From this data set, we randomly select $1/3$ (800) for training and the remaining $2/3$ (1601) for testing. The objective of the experimental studies is to investigate how well we can separate banking messages from non-banking messages.

Table V shows the performance of the SVM-based classifier. As we can see from the table, we can obtain a very high accuracy rate ($98.94\% \pm 0.0024$, the average accuracy is 98.94% with a 0.0024 standard deviation) and low false positive and false negative rates ($1.14\% \pm 0.0021$ and $0.96\% \pm 0.0059$, respectively). These results show that the SVM-based classifier allows us to successfully separate banking messages from non-banking messages. In addition, we note that the parameter of the SVM-based classifier can be adjusted to further lower the false positive rate or completely remove any false positives. However, this will normally result in a higher false negative rate. In the context of separating banking messages from non-banking messages, we argue that it is reasonable and acceptable to have a higher false negative rate in order to lower or remove false positives. For this reason, in the following studies of the performance of the phishing detection rules, we will use all the phishing messages in the Nazario archive, instead of only the ones that can be classified as banking messages by the SVM-based classifier.

TABLE V
PERFORMANCE OF SVM-BASED CLASSIFIER.

Metric	Result % (std. deviation %)
Accuracy	98.94 (0.24)
False positive rate	1.14 (0.21)
False negative rate	0.96 (0.59)

D. Performance of Heuristics in Detecting Phishing Emails

In this section we evaluate the performance of the heuristic rules to detect phishing emails from a set of (legitimate and phishing) banking messages. In this section we apply the three heuristic rules on all messages in the Nazario, PersonalPB, and PersonalLB archives. For each message we report the first rule that flags the message as phishing and the reason why it is flagged.

We first show the results when the three rules are applied on the messages contained in the Nazario archive, which are all phishing banking messages. Table VI shows the performance of the heuristic rules in detecting phishing banking messages using the Nazario archive. As we can see from the table, the heuristic rules using sender-related information are an effective mechanism to identify phishing emails. Out of 1001 PB messages contained in the archive, 988 (98.7%) are classified successfully as phishing messages. Figure 1 shows the percentage of messages flagged by the rules (cumulative). From the figure we can see that, for the 988 phishing messages that are successfully flagged, most of them (956) are flagged by the first two rules.

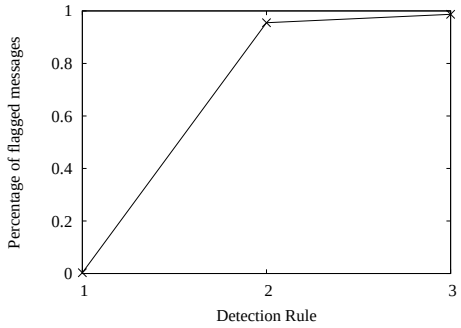

Fig. 1. Performance on Nazario archive.

In order to fully understand the effectiveness of each rules and the specific reason why a message is flagged, we show the breakdown of the messages based on the specific reasons

TABLE VI
PERFORMANCE ON NAZARIO ARCHIVE.

Data set size	Flagged messages (%)
1001	988 (98.7%)

TABLE VII
PERFORMANCE ON NAZARIO ARCHIVE (BREAKDOWN).

Row	Rule	Count	Reason
1	R1	3	Free email provider address
2	R2	73	CC of email addresses not match
3	R2	563	CC of URL undefined
4	R2	250	CC of URL not match
5	R2	9	CC of delivery path undefined
6	R2	58	CC of delivery path not match
7	R3	1	Bank cannot be identified
8	R3	31	SPF query returned fail code
9	-	13	Not flagged as phishing

that they are flagged as phishing in Table VII. We note that, a message can be flagged as phishing by multiple rules; however, the table only taxonomizes the messages according to the first rule (and the specific reason) by which a message is flagged. As we can see from the table, the main reason for detecting phishing messages is URLs inconsistencies. This result makes sense with the fact that phishers use servers to host their services in temporary or compromised hosts at various locations. As we can see from the table, 813 messages (rows 3 and 4) get flagged because they either contain URLs for which we cannot obtain the geographic location, or a URL that does not match the country from the sender email address. We also note that 563 phishing messages contain URLs for which we cannot determine the country of the corresponding domains. This is likely caused by the fact that the phishing archive is a few years old, and certain phishing domains have been taken down. For real-time phishing detection, we are more likely to determine the country of URLs contained in a phishing message. In order to understand if such messages can be flagged by later rules, we also apply **R3** on these messages. 562 out of these 563 can be flagged by **R3**. This means that even if we relax the requirement on URLs in a message in **R2**, we can still flag most of these messages in **R3**.

We also note that in fact many phishing messages (98 messages in rows 5, 6 and 8) were likely originated from bots or compromised machines. We can assume this because these messages were delivered from paths that are inconsistent or where SPF fails. Finally, we have another significant group of 73 messages (row 2) that have different email address among its headers, and the domains are registered to different countries. This reflects some kind of forging of addresses to avoid being detected by some filters or use fake addresses that may look similar to a bank's name.

We also perform the same studies on the newer PersonalPB archive. As shown in Figure 2, all the messages are flagged as phishing by the first two rules. Similarly, we examine the particular rule and the specific reason by which a message is flagged as phishing, as shown in Table VIII. We can notice that for this newer archive there is a significant increase in messages that come from free email service providers, which shows that phishers are increasingly relying on public email service providers to carry out phishing attacks, instead of registering and maintaining their own domains.

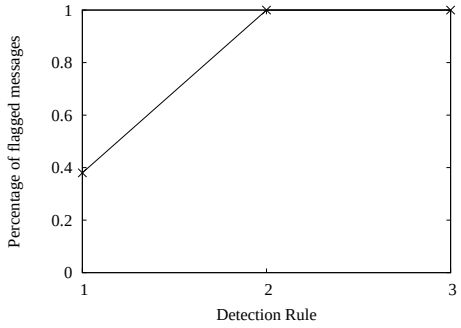

Fig. 2. Detection on PersonalPB

TABLE VIII
PERFORMANCE ON PERSONALPB ARCHIVE (BREAKDOWN).

Row	Rule	Count	Reason
1	R1	19	Free email provider address
2	R2	13	CC of email addresses not match
3	R2	1	CC of a URL undefined
4	R2	3	CC of a URL not match
5	R2	1	CC of delivery path undefined
6	R2	13	CC of delivery path not match

In the last evaluation study, we investigate the performance of the heuristic rules using the PersonalLB archive, which contains only legitimate banking messages. All the messages in the archive passed the 3 rules without being flagged. In particular, all the involved banks have their own dedicated network domains, all messages are delivered from their own mail servers to the recipient domains.

V. DISCUSSION

The design of the current sender-centric approach has only focused on detecting phishing messages related to banks. PB messages need to imitate the legitimate banking messages, in terms of both content and structure, which makes easy the task of separating banking messages from non-banking messages. The second step of the system explores the fact that it is hard for phishers to match the infrastructure of a legitimate bank, and in contrast, it is more likely for them to depend on (likely compromised) resources widely distributed on the Internet. This geographic distribution of resources allows us to effectively differentiate between legitimate and phishing banking messages.

While the set of heuristic rules have been designed with banking messages in mind, they can be easily extended to classify other kind of phishing messages that target online retailers and online payment companies, which only support the communications between the companies and the clients, but not communications between clients. The sender-centric approach, as currently designed, cannot easily classify phishing messages targeting online companies that support communications between clients, such as online auction systems like E-Bay. Such companies allow for buyers to directly contact a seller, after winning a bid, and the email address of the

seller is included on the *Reply-to:* field. In order to handle these messages, the sender-centric approach needs to ignore this field. It is likely that ignoring this field will not affect the effectiveness of the sender-centric approach. We plan to carry out a detailed study on this in our future work using a broader data set containing various kinds of phishing and non-phishing emails.

We also note that it is possible for phishers to evade, in part, the developed system. For example, they could attempt to impersonate a small bank that has not published SPF records and use (likely compromised) machines geographically close to the location of the bank to send the messages and host their phishing site. While such kind of attacks is possible, we argue that our system greatly limits the flexibility that phishers could use to deploy their operations. This limitation could make impractical for a phisher to operate. For the example, the cost of phishing becomes much higher at several levels. First, since they can only impersonate small banks, the pool of possible users becomes much limited. Next, they will need to use an infrastructure much more limited because the distribution of bots, or servers hosting their phishing site, will be reduced to those in the geographic proximity of the target institution, making phishing much more vulnerable to identification and prosecution.

Another way that phisher could attempt to evade our system is to compromise a bank's system to either send phishing messages or host the fake site. We note, that if a phisher is capable of compromising the bank infrastructure for such kind of attacks, they probably are able to obtain private user information from the bank directly, instead of launching phishing attacks.

VI. Conclusion

In this paper we have developed a sender-centric approach to detecting phishing emails. We note that, although phishers have great flexibility in manipulating both the content and structure of phishing emails, they have much less flexibility in concealing the information related to the sender of a phishing message. Such sender-related information is often inconsistent with the target institution of the phishing email, and thus, can help detect phishing emails. In this paper we have advocated and developed a sender-centric approach to detecting phishing emails by focusing on the information related to the sender of a message instead of the content or structure of the message. Our evaluation studies based on real-world email traces showed that the sender-centric approach is a feasible and effective method in detecting phishing emails.

References

[1] S. Ahmed and F. Mithun. Word stemming to enhance spam filtering. In *Proceedings of First Conference on Email and Anti-Spam (CEAS)*, July 2004.

[2] O. Bartunov and T. Sigaev. CPAN Lingua::Stem::Snowball. http://search.cpan.org/~creamyg/Lingua-Stem-Snowball-0.952/lib/Lingua/%Stem/Snowball.pm.

[3] M. Chandrasekaran, K. Narayanan, and S. Upadhyaya. Phishing email detection based on structural properties. In *New York State Cyber Security Conference*, 2006.

[4] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[5] P. Dwyer and Z. Duan. MDMap: Assisting Users in Identifying Phishing Emails. In *Proceedings of 7th Annual Collaboration, Electronic Messaging, Anti-Abuse and Spam Conference (CEAS)*, Redmond, WA, July 2010.

[6] I. Fette, N. Sadeh, and A. Tomasic. Learning to detect phishing emails. In *Proceedings of the 16th International Conference on World Wide Web*, Alberta, Canada, May 2007.

[7] J. Goodman, G. V. Cormack, and D. Heckerman. Spam and the ongoing battle for the inbox. *Communications of the ACM*, 50(2):25–33, Feb. 2007.

[8] J. Hietaniemi. CPAN String::Approx. http://search.cpan.org/~jhi/String-Approx-3.26/Approx.pm.

[9] A. Karatzoglou and D. Meyer. Support Vector Machines in R. *Journal of Statistical Software*, 15(9), Apr. 2006.

[10] J. Klensin. Simple Mail Transfer Protocol. RFC 5321, Oct. 2008.

[11] V. I. Levenshtein. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady*, 10, Feb. 1966.

[12] MaxMind. MaxMind geoip perl api. http://www.maxmind.com/app/perl.

[13] J. Nazario. Phishing Corpus. http://monkey.org/~jose/phishing/.

[14] S. Project. Ham Email Corpus. http://spamassassin.apache.org/publiccorpus/.

[15] RSA. RSA online fraud report, Jan. 2012.

[16] B. Scholkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. The MIT Press, 2002.

[17] SpamAssassin. The Apache SpamAssassin project. http://spamassassin.apache.org/.

[18] Wikipedia. Country code. http://en.wikipedia.org/wiki/Country_code.

[19] M. Wong and W. Schlitt. Sender policy framework (spf): Authorizing use of domains in e-mail, version 1. RFC 4408, Apr. 2006.

[20] Y. Xie, F. Xu, K. Achan, R. Panigrahy, G. Hulten, and I. Osipkov. Spamming botnets: Signatures and characteristics. In *Proc. ACM SIGCOMM*, Seattle, WA, Aug. 2008.

[21] Y. Zhang, S. Egelman, L. F. Cranor, and J. Hong. Phinding phish: Evaluating anti-phishing tools. In *Proceedings of 14th Annual Network and Distributed System Security Symposium (NDSS)*, 2007.