Blocking Spam By Separating End-User Machines from Legitimate Mail Server Machines

Fernando Sanchez
Florida State University
sanchez@cs.fsu.edu

Zhenhai Duan
Florida State University
duan@cs.fsu.edu

Yingfei Dong
University of Hawaii
yingfei@hawaii.edu

ABSTRACT

Spamming botnets present a critical challenge in the control of spam messages due to the sheer volume and wide spread of the botnet members. In this paper we advocate the approach for recipient mail servers to filter messages directly delivered from remote end-user (EU) machines, given that the majority of spamming bots are EU machines. We develop a Support Vector Machine (SVM) based classifier to separate EU machines from legitimate mail server (LMS) machines, using a set of machine features that cannot be easily manipulated by spammers. We investigate the efficacy and performance of the SVM-based classifier using a number of real-world data sets. Our performance studies show that the SVM-based classifier is indeed a feasible and effective approach in distinguishing EU machines from LMS machines. For example, training and testing on an aggregated data set containing both EU machines and LMS machines, the SVM-based classifier can achieve a 99.27% detection accuracy, with very small false positive rate (0.44%) and false negative rate (1.1%), significantly outperforming eight DNS-based blacklists widely used today.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and protection*; H.4.3 [**Information Systems Applications**]: Communications Applications—*Electronic mail* ; I.5.4 [**Pattern Recognition**]: Applications—*Text processing*

General Terms

Security, Algorithms, Performance

Keywords

Content-Independent Spam Control, Spamming Bot, Machine Classification, Learning

1. INTRODUCTION

CEAS '11, September 1-2, 2011, Perth, Australia.

Given the importance of controlling spam to improve the trustworthiness and usability of the Internet email system, many anti-spam systems have been developed over the years (see [2, 9, 18, 19, 21, 23] and the references therein). In response, spammers have also developed advanced spamming infrastructures and techniques to infiltrate and evade the deployed anti-spam systems. In particular, the spamming infrastructure has evolved from spammers' own mail servers, to open relays [1], and nowadays to spamming botnets (networks of compromised machines used for sending spam) [10, 24]. Various studies have shown that spam messages sent from botnets accounted for above 80% of all spam messages on the Internet in recent years [10, 12]. For example, the MessageLabs Intelligence 2010 annual security report showed that approximately 88.2% of all spam in 2010 were sent from botnets.

Spamming botnets present a significant challenge in the control of spam messages because of the sheer volume and wide spread of their members. These two natures of spamming botnets render many anti-spam techniques less effective, such as the reactive DNS-based blacklists (DNSBL) and reputation-based spam filtering schemes. Given that spammers have a large pool of spamming bots to use, it is essentially impossible for reactive DNSBLs to maintain a complete and up-to-date list of IP addresses involved in spamming. Similarly, given the large number of members in a spamming botnet, each spamming bot only needs to send a small portion of the total spam messages in a spam campaign. From an individual recipient network domain's perspective, the number of spam messages sent from a spamming bot could be too small to trigger a reputation system to respond. More importantly, the majority of existing anti-spam schemes allow for the arms race between the spammers and the anti-spam community, and they encourage spammers to recruit large number of spamming bots and explore novel usage of these spamming bots.

Rather than allowing for the arms race between spammers and the anti-spam community in the war of botnet-based spamming, in this paper we will develop a novel scheme that targets the root cause of the problem to discourage (and ideally, to prohibit) spammers from using botnets to send spam messages. A key observation that motivates this approach is that the majority of spamming bots are end-user (EU) machines instead of legitimate mail server (LMS) machines, given that legitimate (mail) server machines are normally well protected and less likely to be compromised. A legitimate email message is normally composed on an EU machine

and then delivered to the local mail server of the sender network domain, from where it is further delivered to the recipient mail server. In contrast, spam messages originated from spamming botnets are normally directly delivered from the EU machines (where a customized mail server software is installed) to the recipient mail servers. By blocking messages directly delivered from remote EU machines, we can effectively prohibit spammers from using compromised machines to send spam messages.

In this paper we aim to develop a *lightweight yet effective* scheme to distinguish EU machines from LMS machines so that messages delivered from remote EU machines can be blocked. Many features can be used to determine if a sending machine is an EU machine or an LMS machine. However, in this paper we focus on the features of a sending machine that cannot be easily manipulated by a spammer, and are already available at or can be easily obtained by a recipient mail server. In particular, we consider two types of features associated with a sending machine: the operating system (OS) and the hostname lexical structure of the sending machine. Based on the OS and hostname lexical features of sending machines, we develop a lightweight Support Vector Machine (SVM) based classifier to separate EU machines from LMS machines [17].

In addition to presenting the SVM-based classifier, we also evaluate the efficacy and effectiveness of the classifier using real-world data sets, and compare the performance of the developed classifier with eight commonly used DNSBL systems. The evaluation studies show that our SVM-based classifier has a very high detection accuracy (percentage of machines that are classified correctly) with very low false positive and false negative rates. For example, on an aggregated data set containing both EU machines and LMS machines, the SVM-based classifier can achieve a 99.27% detection accuracy, with a false positive rate of 0.44% and false negative rate of 1.1%, significantly outperforming all eight DNSBLs considered in the study.

The remainder of the paper is structured as follows. In Section 2 we present the design of the SVM-based classifier including the machine features used in the study. In Sections 3 and 4 we describe the data sets used for the performance studies and the results of the performance studies of the SVM-based classifier, respectively. In Section 5 we discuss the related work, the potential techniques that spammers may develop to evade the SVM-based classifier, and the limitations of the SVM-based classifier. We conclude the paper and discuss future work in Section 6.

2. METHODOLOGY

As we have discussed in Section 1, the majority of spam messages were sent from spamming botnets in recent years, and the majority of the spamming bots are likely to be EU machines, given that server machines are normally well protected. Based on these observations we can effectively block spam messages and discourage spammers from using spamming botnets if we can distinguish EU machines from LMS machines and block messages directly delivered from a remote EU machine.

In this paper we develop a Support Vector Machine (SVM)-based classifier to separate EU machines from LMS machines [17], and then messages directly delivered from remote EU machines can be filtered by the system deploying the SVM-based classifier. We refer to the message filtering system using the SVM-based classifier as the SVM-based filtering system (or simply the SVM-based system, when there is no confusion). In this system, we train the SVM-based classifier using a data set including both EU machines and LMS machines, based on the history of the emails received by the local system. In building the SVM-based classifier, a set of machine features that cannot be easily manipulated by spammers will be used.

When an incoming email delivery request comes, the recipient mail server can use the SVM-based classifier to determine if the sending machine is an EU machine or an LMS machine. Based on the type of the sending machine, the recipient mail server can process the message differently depending on the local configured policy. For example, one policy could be to directly reject the message delivery request from the remote machine if it is an EU machine. Another possible policy is to feed the machine classification information to a more comprehensive spam filtering system such as SpamAssassin [18], which can use this classification information as one of the factors to determine the likelihood of the incoming message being a spam. In this paper we only focus on the machine classification problem and will not discuss further how classification results may be used by individual recipient mail servers.

In the following we will first present a brief description of SVM and then we will discuss the machine features that we use in the design of the SVM-based classifier.

2.1 Support Vector Machines

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 two 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 accommo-

date 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 [17].

2.2 Machine Features

In applying SVM to classify the machines (EU machine vs. LMS machine), we only consider the machine features that cannot be easily manipulated by spammers. A number of studies have confirmed that EU machines and LMS machines have different distributions of installed operating systems (OS) [5, 14]. In addition, our previous study also showed that EU machines and LMS machines have quite different hostname naming structures [16]. Therefore in this study, we consider two types of machine features: OS and hostname lexical structure of a machine. In total, we extract 53 features from these two types of machine features. In the following we will discuss these machine features.

OS features: We extract two OS features related to a machine: 1) whether or not the OS type of a machine can be determined, and 2) the type of the OS on the machine (if we can determine the OS type). Due to various set-ups and configurations on the remote network, we may not always be able to obtain the OS fingerprint of a remote machine. Moreover, due to the limitation of OS fingerprint tools, we may not be able to determine the specific type of an OS, even if we can obtain the OS fingerprint. We consider this as one of the features instead of ignoring the machines that we cannot determine the OS type. If we can determine the OS type of a machine, we further classify the OS into eight categories: Linux, BSD, Solaris, AIX, IronPort, Windows Server, Windows 98/2000/XP, and other. The type "other" means that the OS is not one of the first 7 OS types.

Hostname lexical features: we extract 51 features related to the hostname of a machine. In the following we briefly discuss them.

- Number of dots on a hostname. This feature informs the length of a hostname in terms of number of labels contained in the hostname. In general, EU machines (in particular, EU machines on broadband networks) have a longer hostname than LMS machines.

- Number of dashes on the local name portion of a hostname. Note that for this feature we only consider the local name portion of a hostname, given that some networks have dashes in their domain names. Similarly, due to certain DNS naming conventions, EU machines tend to have more dashes in their local names [16, 20].

- If a hostname encodes IP address as part of the hostname. Due to some suggested DNS naming convention [20], it is common for broadband service providers to include IP address of a client machine into the hostname of the machine. Examples include 192-168-10-5.example.com and 192.168.10.5.example.com. We use a number of regular expressions to match hostnames containing an IP address.

- If the reverse DNS lookup of the machine IP address resolves to a valid hostname. Note that, a large portion of EU machines do not have a valid reverse DNS hostname [15, 16].

- Common keywords found on the local name portion of a hostname. Each keyword is considered as a machine feature. We extract two common keywords from hostnames of LMS machines: `server` and `relay`. We extract 11 common keywords from hostnames of EU machines in data sets we have: `dhcp`, `host`, `rev(ip)?`, `broadband`, `bb`, `ip`, `user`, `cust(omer)?`, `ppp`, `catv`, and `pool`. This ad-hoc method gives us a total of 13 features. Note that two of the features are regular expressions (`rev(ip)?` and `cust(omer)?`).

- Keywords on the local name portion of a hostname as suggested by the Internet draft [20]. This Internet draft describes suggestions for a common DNS naming scheme for automatically generated DNS resource records for large networks such as Internet service providers. We extract 34 keywords to be used on local names of automatically generated hostnames for machines in networks where the naming convention is followed. Example keywords include `mx`, `mail`, and `dsl`. Note that, some of the features are regular expressions.

Table 1: Summary of Data Sets.

Data set	# of machines	# of domains
LMS	8282	1765
EU1	6339	505
EU2	19729	1623

3. DATA SETS

In this section we describe the data sets used in our study. In order to investigate the performance of the SVM-based classifier in classifying machines, we use three data sets. One of them contains LMS machines (we refer to it as the LMS data set), and another two contain EU machines (we refer to them as the EU1 and EU2 data sets, respectively). In the following we describe how the data sets are collected and processed. Table 1 shows a summary of the three data sets in terms of number of machines and network domains in each data set.

First, we describe the LMS data set. An email trace was collected at a number of mail servers on the Florida State University (FSU) campus network from 10/01/2008 to 10/15/2008 (inclusive). During the course of the email trace collection, the mail servers received about 53 M messages destined for 82 sub-domains in the FSU campus network, of which about 47 M, or about 88.7%, were spam. This email trace contains about 6.4 M connecting remote machines (which requested to deliver messages to FSU mail servers).

In order to ensure that the LMS machines selected are from relatively trustworthy network domains (not spammer network domains), we identify LMS machines in the following way based on the FSU email trace. We first group

connecting machines into their corresponding network domains. We rely on the public suffix list maintained at [13] to determine the network domain given a hostname. We then rank the network domains according to the ham ratio (the percentage of messages that are non-spam). In this study, we only consider the network domains that have sent to FSU at least 100 messages and the ham ratio is at least 90%. Given the relatively large number of email messages delivered from these network domains and high ham ratio, we consider them as trustworthy well-behaved network domains. In this way we obtain 1765 trustworthy network domains.

We then look up the MX and SPF records of these network domains. For each retrieved SPF record, we only include the entries that identify a unique machine using either an IP address or a hostname. We do not include the entries in a SPF record that identify a range of IP addresses such as $a.b.0.0/16$, which are likely to be a placeholder for future use instead of indicating real mail servers currently being used. In this way we obtain 8282 LMS machines. If an entry in a SPF record only contains an IP address, we also try to obtain the corresponding reverse DNS hostname. We then use the tool `nmap` to collect the OS fingerprint of these machines.

We next describe the EU1 data set. The EU1 data set is obtained based on a spam trace collected by Cortez *et al* [5]. The spam trace was collected by advertising several spam traps in the popular websites in the US. The trace contains 12137 spam messages, collected over a time span from 04/21/2009 to 11/09/2009 (inclusive). In addition to the content of the spam messages, the spam trace also contains auxiliary information, including the OS fingerprint of the sending machine (not all messages contain OS fingerprint of the sending machine), and if the sending machine was blacklisted by any of eight DNSBLs at the time when the spam message was delivered. We will provide the details of the eight DNSBLs when we study the performance of the SVM-based classifier.

We would like to obtain a set of (compromised) EU machines from this spam trace. However, a spam message in the trace may not be delivered from an EU machine (it could be from a spammer's own mail server, or relayed by some other mail servers). In order to obtain a set of EU machines, we adopt the following heuristic [16]. If a message only traverses a single external hop before being delivered into the recipient network domain, the sending machine is considered as an EU machine. As we have discussed above, spamming bots tend to directly deliver a spam message from themselves to the recipient mail server, while legitimate messages are composed on an EU machine, and then delivered to the local mail server of the sending network domain before being forwarded to the recipient mail server.

To be more precise, we use network-level consistent (NLC) path to determine the number of hops that a message traverses [16]. At a high level, an NLC path is a portion of the message delivery path as carried in the `Received:` header. An NLC path requires that the `from-domain` of the current `Received:` header belongs to the same network prefix as the `by-domain` of the previous `Received:` header. In essence, NLC path allows a mail server to use two different IP address for the incoming process and the forwarding process in the mail server. (See [16] for the details on NLC path.) For simplicity, we only consider the /16 network prefix in this study.

After obtaining the NLC message delivery path of a message, we claim that the sending machine be an EU machine if the NLC path has only one single external hop (i.e., directly delivered from the originating machine to the recipient network domain). In this way we obtain 6339 EU machines from 505 network domains. If a valid reverse DNS hostname is carried in the corresponding `Received:` header for the external sending machine, we also include the hostname in the data set for the corresponding IP address. If no valid reverse DNS hostname is carried in the corresponding `Received:` header for the external sending machine, we claim that the sending machine does not have a reverse DNS hostname at the time of the message delivery. We do not perform a reverse DNS lookup on the corresponding IP address, given that the configuration may have changed from the time when the message was delivered to the current time.

The last data set EU2 is obtained in a similar fashion as EU1 but based on a different spam trace, which contains one month (January 2011) spam messages collected by a public spam archive site [7]. This spam trace contains 32274 spam messages. Similarly, we consider the sending machine of a message whose NLC path only contains one external hop to be an EU machine. In this way, we are able to obtain 19729 EU machines from 1623 network domains. As we have done for EU1, we also include the corresponding hostname of a sending machine to the data set, if a valid reverse DNS hostname to the IP address is carried in the corresponding `Received:` header. Unfortunately, this spam archive does not have any OS fingerprint information, nor the information if the sending machine is listed by any DNSBLs.

4. PERFORMANCE EVALUATION

In this section we will conduct experimental studies to investigate the performance of the SVM-based classifier in classifying EU machines from LMS machines. We will first discuss the performance metrics and the software packages we use, and then we will perform three sets of experimental studies. We will first study the performance of the SVM-based classifier when all 53 features are considered and both the test set and training set are drawn from the same data set. We also compare the performance of the SVM-based classifier with eight DNSBLs coming with the spam trace associated with data set EU1. We then evaluate the importance (weight) of individual machine features in affecting the performance of the SVM-based classifier. In the last performance study, we investigate how well the SVM-based classifier can perform when the training set and the test set are drawn from two different data sets.

4.1 Performance Metrics and Software Packages

Given a data set, we partition it 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 percentages of total machines (including both EU machines and LMS machines) that are classified correctly. More formally (all numbers are with regard to the

test set)

$$\text{Accuracy} = \frac{\text{\# of EU and LMS classified correctly}}{\text{Total \# of machines}}.$$

The second one is *false positive rate*, which measures the percentage of LMS machines that are misclassified. The last metric is *false negative rate*, which measures the percentage of EU machines that are misclassified. More formally (similarly, all numbers are with regard to the test set)

$$\text{False positive rate} = \frac{\text{\# of LMS misclassified}}{\text{Total \# of LMS machines}},$$

$$\text{False negative rate} = \frac{\text{\# of EU misclassified}}{\text{Total \# of EU machines}}.$$

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 [11], which in essence provides a wrapper interface to the widely used SVM implementation `libsvm` [3]. 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 (see Section 2). 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 $\gamma = (2^{-2}, 2^2)$, respectively. 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.

4.2 Performance with All Features

In the first set of performance studies, we consider all 53 features and also compare the performance of the SVM-based classifier with eight DNSBLs coming with the spam trace associated with EU1.

For these studies, we merge the EU1 and LMS data sets to build an aggregated data set that includes both EU machines and LMS machines (there are totally 14621 machines. See Table 1). We then randomly sample 1/3 (4873) of the machines to form the training set, and the remaining 2/3 (9748) of the machines form the test set. We use the procedure outlined in the previous subsection to train the SVM-based classifier on the training set, and apply the resulting SVM model to classify the machines in the test set.

Table 2 shows the performance of the SVM-based classifier in this study. As we can see, the SVM-based classifier can achieve a very high detection accuracy (99.27%) with very small false positive and false negative rates (0.44% and 1.1%, respectively). The results show that the SVM-based classifier is indeed feasible and effective in distinguishing EU machines from LMS machines, when the training set and the test set are from the same source. This observation indicates that the SVM-based classifier can be deployed by individual network domains to effectively block messages from remote spamming bots, when it is trained using the sender machine information as observed by the corresponding network domains.

Table 2: Performance of the SVM-based classifier with all 53 features.

Metric	Result
Accuracy	99.27% (9677/9748)
False positive rate	0.44% (24/5487)
False negative rate	1.1% (47/4261)

Next we compare the performance of the SVM-based classifier with the eight DNSBLs coming with the spam trace used to create the EU1 data set. Given that the spam trace contains the information if a sending machine was blacklisted by any of the eight DNSBLs at the time when the message was delivered, we can compute the detection rate of each of the eight DNSBLs, that is, the percentage of the sending machines being blacklisted by a DNSBL. We similarly compute the detection rate of the SVM-based classifier. To make the comparison fair, we compute the detection rate of both the SVM-based classifier and the eight DNSBLs only using the EU machines included in the test set coming from the aggregated data set. Table 3 shows the result.

Table 3: Performance comparison between the SVM-based classifier and DNSBLs.

Schemes	Detection rate (%)
SVM classifier	98.90
zen-spamhaus.org	82.61
sbl-xbl.spamhaus.org	69.63
cbl.abuseat.org	64.47
dnsbl.sorbs.net	49.99
bl.spamcop.net	47.34
psbl.surriel.com	36.00
blackholes.five-ten-sg.com	28.35
dul.dnsbl.sorbs.net	19.62

From the table we can see that the SVM-based classifier significantly outperforms all eight DNSBLs. The SVM-based classifier has a detection rate of 98.90%, while the detection rate of the eight DNSBLs ranges from 19.62% to 82.61%. The DNSBL that achieves the highest detection rate is the `zen` blacklist from the Spamhaus project [19], which combines all the blacklists maintained by the Spamhaus project.

4.3 Feature Relevance

In this subsection we study the importance (weight) of the 53 features in affecting the performance of the SVM-based classifier. For this study we use the F-score tool [4]. Intuitively, the F-score of a feature measures the discriminativeness of the feature in determining the classes that data points belong to. The higher the value of an F-score of a feature, the *more likely* that the feature is more discriminative. Note that the F-scores of a set of features do not directly determine the absolute rank of the features in affecting the performance of an SVM classifier. However, they do provide us with some guidelines on the potential importance of the features on affecting the performance of an SVM classifier.

Table 4 shows the top 10 features according to their F-scores obtained based on the training set of the aggregated data set. As we have discussed, the higher ranked features

tend to have more impacts on the performance of the SVM-based classifier. From the table we can see that both features *if IP address is encoded in hostname* and *if an IP address has a reverse DNS hostname* play a critical role in affecting the performance of the SVM-based classifier. This result is intuitively understandable in that EU machines (with dynamically allocated IP addresses) tend to encode IP addresses in their hostnames or not have a reverse DNS hostname at all. To a degree, this ranking result confirms the effectiveness of the F-score tool.

From the table we can also see that the F-scores of the ranked features decrease quickly, which indicates that the top few features largely determine the overall performance of the SVM-based classifier. We also note that 11 out of the 53 features we extracted have an F-score of 0 (not shown in the table); it is likely that they do not play any significant role in determining the performance of the SVM-based classifier. All these 11 features are keywords we extracted from the Internet draft on suggested naming schemes for large networks [20]. They did not appear often in our data set. This could be that the corresponding suggestions have not been commonly followed or it is due to the limitation of our data collection vantage points.

Table 4: Top 10 features in terms of F-scores.

Rank	Feature	F-score
1	IP address encoded in hostname	3.51
2	If reverse DNS hostname exists	1.25
3	Number of dots in hostname	0.60
4	If OS type can be determined	0.15
5	Keyword "mx" in local name	0.12
6	Keyword "mail" in local name	0.11
7	Type of OS	0.10
8	Local name matches "dsl"	0.05
9	Local name matches "dyn(amic)?"	0.03
10	Local name matches "adsl"	0.027

Using the ranking of the features based on the F-scores, we study how the number of features affect the performance of the SVM-based classifier. Figure 1 shows the performance of the SVM-based classifier as a function of the number of features (features are ranked according to the F-scores). As we can see from the figure, we can already achieve a detection accuracy of 97.53% (with a false positive rate of 1.9% and a false negative rate of 3.26%), using only the top 5 features. In general, adding more features to an extent slightly improves the performance of the SVM-based classifier. After the number of features reaches certain threshold (around 11 in our data set), the performance does not change much. It is also worth noting that the performance of the SVM-based classifier fluctuates slightly after the number of features reaches the threshold (for all three performance metrics).

In summary, features do not play the same role in contributing to the performance of the SVM-based classifier. It is critical to identify the most discriminative features in order to achieve a good performance.

4.4 Performance Across Data Sets

It is common that different spammers may control different spamming botnets and possess different databases of spamming target email addresses (for both, membership may overlap but not the same). As a consequence, distinct recipient mail servers may receive spam message delivery requests from different set of sending machines. It is therefore interesting to see if an SVM-based classifier trained using data trace in one network domain can be applied to distinguish EU machines from LMS machines at a different network domain. It could have critical importance for DNSBLs that rely on their own spam traps to collect spamming bots to construct the blacklist of spamming IP addresses, should they want to deploy the developed SVM-based classifier.

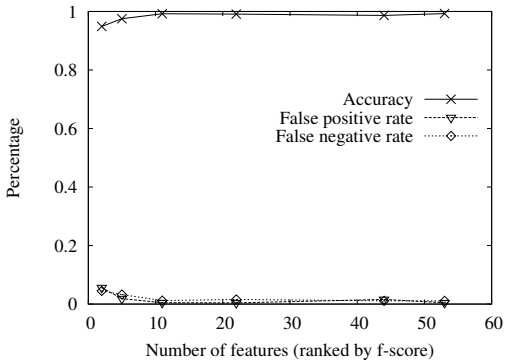

Figure 1: Impact of features on performance of the SVM-based classifier.

To understand how well the SVM-based classifier can perform in this environment, we apply the trained SVM model using the aggregated data set to classify machines contained in the EU2 data set. Note that the EU2 data set does not contain any OS related information. Therefore, we remove the two OS features from the aggregated data set and retrain the SVM classifier by following the same procedure as in Subsection 4.2. In particular, we use the same 1/3 of the aggregated data set as the training set to build the SVM model. Applying the model on the test set of the aggregated data set (2/3 of the data), we obtain an accuracy of 98.37%, with a false positive rate of 0.56% (31/5487) and a false negative rate of 3% (128/4261).

We then treat the entire EU2 data set as the new test set and apply the trained SVM model (on the training set of the aggregated data set) to investigate the performance of the SVM-based classifier on data set across different network domains. Note that the EU3 data set does not contain any LMS machines. The SVM-based classifier can achieve a detection accuracy of 91.81% (18114/19729) with a false negative rate of 8.19% (note that the accuracy is also the detection rate in this case, given that there are no LMS machines in the data set). This shows that the SVM classifier can achieve robust performance even if it is used in the situations where the classification has to be performed in a network different from the network where the SVM model is trained.

5. RELATED WORK AND DISCUSSION

In this section we first discuss the related literatures and systems that are most relevant to our work. We then discuss potential techniques that a spammer may develop to try to evade the SVM-based system, and the limitations of the developed SVM-based classifier.

5.1 Related Work

The closest work to the SVM-based classifier developed in this paper is the SpamAssassin Botnet plugin [15], which identifies if a message has been submitted by a spamming bot based on the domain name characteristics of the sending machine. A set of rules (regular expressions) are used in the identification of spamming bots, including if a reverse DNS lookup of an IP address resolves to a valid domain name, if the corresponding hostname contains keywords such as "client", and if the hostname contains the IP address of the sending machine, among others. Our SVM-based classifier is a more comprehensive machine classification scheme. Instead of relying on static rules of domain name characteristics of sending machines to identify spamming bots, we use a machine learning algorithm to determine the nature of a sending machine with a richer set of features. The SVM-based classifier can be viewed as a generalization of the SpamAssasssin Botnet plugin.

In our previous work [16], we have performed a preliminary study on the naming structures of both legitimate mail server machines and spamming (EU) machines. That study showed that LMS machines and spamming machines have strikingly different naming structures. For example, about 34% of LMS machines have "mail" as their local names in their domain names. In contrast, about 45% of spamming machines only have IP addresses and do not have a domain name. Moreover, about 23% of spamming machines contain IP addresses in their domain names. This study indicates that it is possible to distinguish EU machines from LMS machines based on simple features associated with the sending machines. The current work extends our previous study to develop a SVM-based classifier to identify EU machines.

Another approach to identifying machines that should not be used to send email messages to a remote network domain is the Policy Block List (PBL) maintained by the Spamhaus project [19]. PBL is a database of end-user IP addresses that are maintained by both the participating network domains and the Spamhaus team. PBL contains various (end-user) IP addresses that should not initiate a message delivery request to a remote network domain, for example, dynamic IP addresses. However, PBL is far from complete based on our recent experience with the database. A possible reason is that the voluntary participation of network domains is still quite limited.

Sender authentication schemes such as SPF and DKIM [22, 8] help to verify if a message has been originated from the claimed network domain. However, although they are critical in preventing spammers from spoofing an existing network domain belonging to others, they cannot prevent spammers from using spamming botnets by registering their own network domains and pointing to the spamming bots. As an example, a spammer can register a new domain and set up an SPF record to point to a spamming bot. Then from SPF's perspective, the spamming bot is a valid mail server for the new domain belonging to the spammer.

Recently, a number of reputation based spam filtering schemes have been developed that use the network-level features to determine the likelihood of an incoming message to be a spam [5, 9]. Example network-level features include the Autonomous System (AS) number of the sender network, sender-receiver geographic distance, the sender IP neighborhood spamming density, the number of traceroute hops from receiver to sender, the geographic coordinates of the sender, among others. These schemes have a different motivation and take a fundamentally different approach from ours. While they aim to detect the likelihood of an incoming message to be a spam based on the network-level features associated with the sender, we try to classify machines into EU machines and LMS machines, and block messages from EU machines. In addition, these network-level reputation systems may have an undesirable side effect to classify a message sent from a legitimate mail server as a spam message if the mail server is surrounded by a large number of spamming machines.

5.2 Potential Evasion Techniques

After the SVM-based classifier is deployed on the Internet, spammers may try to develop schemes to evade to the system to continue using spamming bots to send spam messages. In the following we will outline a few potential evasion techniques and examine how likely they can succeed.

An obvious technique a spammer can try to evade the system is to relay the spam messages using the local legitimate mail server, instead of directly delivering the messages to the remote network domain, where the SVM-based classifier is deployed. By *local legitimate mail server*, we mean the mail server that is used by the owner of the (compromised) machine. It could be the local mail server on a campus network, a smart host on an ISP network, or a web-based mail server of an email service provider. After botnets become an infeasible solution for spammers to send spam messages, they will look for new approaches to spamming, including sending spam messages via legitimate mail servers. However, sending spam via legitimate mail servers does increase the difficulty and raise the cost of spamming.

First, we note that relaying through the legitimate mail server requires the spammer to obtain the authentication information such as user account and password, which applies to local mail servers, smart hosts, and web-based email servers. Although arguably it is not hard to obtain the user account and password information after a machine has been compromised, it does add another layer of difficulty for spammers to use spamming bots. Another technique for spammers to obtain user accounts and passwords is phishing [6], which itself has become a major security problem on the Internet.

Second, many spam messages are addressed to non-existing destination email addresses, because, for example, the email address database of the spammer contains outdated address information, or the database is built from a dictionary by permuting common names. This will result in many returned undeliverable messages. Depending on the specifics of the spamming set-up, the owner of the compromised machine may receive such returned emails and more likely to notice that the machine has been compromised and his authentication information has been stolen.

In addition, legitimate mail servers (such as web-based email service providers) can help their users to identify the possibility that their authentication information has been stolen if information such as login history can be provided to the users. Currently, such history information is available to law enforcement under search warrant. In order to minimize undeliverable messages and to make the messages more appealing to the recipients, spammers may only send (spam) messages to the contacts of the account owner. One potential technique a user can use to monitor such behavior

is to add a separate account owned by the same user to his own contact list.

Third, many legitimate mail servers nowadays employ a spam filter to outgoing messages in addition to the spam filter on incoming messages. Relaying via a legitimate mail server makes it easier for the mail server to detect the abnormal spamming behavior of the corresponding account and inform the owner.

A second evasion technique that a spammer may try is to relay spam messages via open relays [1]. However, as open relays are being actively detected and blocked, forwarding spam messages via open-relays is not a reliable approach for spammers. Moreover, if the open relays is on an EU machine, it will be automatically detected by the SVM-based classifier and all the message delivery requests from it will be rejected by the system.

5.3 Limitations

The SVM-based classifier may have an undesirable impact on small home business owners who run mail servers on their home networks. Given that the SVM-based system filters messages directly sent from EU machines, the messages originated from these mail servers will be filtered, if these machines are on broadband networks and not configured properly. A simple approach to addressing this problem is for these mail servers on home networks to forward outgoing messages via smart hosts provided by the corresponding network service provider. Relaying outgoing messages via smart hosts of the corresponding network service provider is a common practice nowadays, and it is the required approach to sending outgoing messages on some networks.

A key premise of the SVM-based classifier is that sufficient differences exist between EU machines and LMS machines so that we can identify simple features to separate the two. In this study we considered two types of features, the OS and the hostname lexical structure of EU and LMS machines. Various studies have shown that spamming bots and LMS machines do tend to have different distributions of operating systems [5, 14], and spamming bots and LMS machines do tend to have different naming patterns of their hostnames [16, 20]. However, our data sets are limited, and networks may have different practices in managing and naming their machines. It is imperative to investigate the performance of the SVM-based classifier using more diverse and larger data sets, which we plan to do in our future work.

6. CONCLUSION AND FUTURE WORK

In this paper we advocated the technique of filtering messages directly delivered from remote end-user machines as an effective approach to control spam messages sent by spamming botnets. We developed an SVM-based classifier to separate end-user machines from legitimate mail server machines, using a set of machine features that cannot be easily manipulated by spammers. Using real-world data sets, we investigated the efficacy and performance of the SVM-based classifier. Our performance studies confirmed that the SVM-based classifier is indeed a feasible and effective approach in distinguishing end-user machines from legitimate mail server machines, significantly outperforming eight DNSBLs widely used today. As future work, we plan to explore a more comprehensive set of machine features and study the performance of the SVM-based classifier on more diverse and larger data sets.

7. ACKNOWLEDGMENTS

We thank Paulo Cortez at the University of Minho in Portugal for providing us with a spam trace, upon which the data set EU1 was constructed. We also thank Bruce Guenter for making his spam archive publicly available. Fernando Sanchez and Zhenhai Duan were supported in part by NSF Grant CNS-1041677. Yingfei Dong was supported in part by NSF Grants CNS-1041739 and CNS-1018971. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of National Science Foundation.

8. REFERENCES

[1] M. Andreolini, A. Bulgarelli, M. Colajanni, and F. Mazzoni. Honeyspam: Honeypots fighting spam at the source. In *Proceedings of Usenix SRUTI*, Cambridge, MA, July 2005.

[2] E. Blanzieri and A. Bryl. A survey of learning-based techniques of email spam filtering. *Artificial Intelligence Review*, 29(1):63–92, 2008.

[3] 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`.

[4] Y.-W. Chen and C.-J. Lin. Combining SVMs with various feature selection strategies. In *Feature extraction, foundations and applications*. Springer, 2006.

[5] P. Cortez, A. Correia, P. Sousa, M. Rocha, and M. Rio. Spam email filtering using network-level properties. In *ICDM*, pages 476–489, 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] B. Guenter. Spam archive. `http://untroubled.org/spam/`.

[8] T. Hansen, D. Croker, and P. Hallam-Baker. DomainKeys Identified Mail (DKIM) Service Overview. RFC 5585, June 2009.

[9] S. Hao, N. A. Syed, N. Feamster, E. G. Gray, and S. Krasser. Detecting Spammers with SNARE: Spatio-temporal Network-level Automatic Reputation Engine. In *USENIX Security Symposium*, pages 101–118, Montreal, Canada, Aug. 2009.

[10] J. P. John, A. Moshchuk, S. D. Gribble, and A. Krishnamurthy. Studying spamming botnets using botlab. In *6th Symposium on Networked Systems Design and Implementation (NSDI'09)*, Apr. 2009.

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

[12] MessageLabs. Messagelabs intelligence: 2010 annual security report.

[13] Mozilla Foundation. Public suffix list. `http://publicsuffix.org/`.

[14] A. Ramachandran and N. Feamster. Understanding the network-level behavior of spammers. In *Proc. ACM SIGCOMM*, pages 291–302, Pisa, Italy, Sept. 2006.

[15] J. Rudd. Botnet plugin for spamassassin.

[16] F. Sanchez, Z. Duan, and Y. Dong. Understanding forgery properties of spam delivery paths. In

Proceedings of 7th Annual Collaboration, Electronic Messaging, Anti-Abuse and Spam Conference (CEAS), Redmond, WA, July 2010.

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

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

[19] Spamhaus. The spamhaus project. `http://www.spamhaus.org/`.

[20] M. Sullivan and L. Munoz. Suggested Generic DNS Naming Schemes for Large Networks and Unassigned hosts. Internet Draft, Apr. 2006.

[21] B. Taylor. Sender reputation in a large webmail service. In *Proceedings of Third Conference on Email and Anti-Spam (CEAS)*, Mountain View, CA, July 2006.

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

[23] M. Xie and H. Wang. A collaboration-based autonomous reputation system for email services. In *Proc. IEEE INFOCOM*, pages 992–1000, San Diego, CA, Mar. 2010.

[24] 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.