Digital Investigation 12 (2015) S12–S21

Contents lists available at ScienceDirect

Digital Investigation

journal homepage: www.elsevier.com/locate/diin

DFRWS 2015 Europe

Spam campaign detection, analysis, and investigation☆

Son Dinh [a, *], Taher Azeb [a], Francis Fortin [b], Djedjiga Mouheb [a], Mourad Debbabi [a]

[a] NCFTA Canada & Concordia University, 1455 de Maisonneuve Blvd West, Montreal, QC H3G 1M8, Canada
[b] Centre international de criminologie comparée, École de criminologie, Université de Montréal, Montreal, QC H3C 3J7, Canada

Keywords:
Spam
Spam campaign
Spam analysis
Characterization
Frequent pattern tree

A B S T R A C T

Spam has been a major tool for criminals to conduct illegal activities on the Internet, such as stealing sensitive information, selling counterfeit goods, distributing malware, etc. The astronomical amount of spam data has rendered its manual analysis impractical. Moreover, most of the current techniques are either too complex to be applied on a large amount of data or miss the extraction of vital security insights for forensic purposes. In this paper, we elaborate a software framework for spam campaign detection, analysis and investigation. The proposed framework identifies spam campaigns on-the-fly. Additionally, it labels and scores the campaigns as well as gathers various information about them. The elaborated framework provides law enforcement officials with a powerful platform to conduct investigations on cyber-based criminal activities.

Introduction

Electronic mail, or most commonly known as *email*, is ubiquitous and so is its abusive usage. *Spam emails* affect millions of users, waste invaluable resources and have been a burden to the email systems. For instance, according to Symantec Intelligence Report, the global ratio of spam in email traffic is 71.9% (Symantec Intelligence Report, 2013). Furthermore, adversaries have taken advantage of the ability to send countless emails anonymously, at the speed of light, to carry on vicious activities (e.g., advertising of fake goods or medications, scams causing financial losses) or even more severely, to commit cyber crimes (e.g., child pornography, identity theft, phishing and malware distribution). Consequently, spam emails contain priceless cyber security intelligence, which may unveil the world of cyber criminals.

Spam data has been used extensively in various studies to detect and investigate cyber threats such as botnets (Zhuang et al., 2008; Xie et al., 2008; John et al., 2009; Pathak et al., 2009; Thonnard and Dacier, 2011; Stringhini et al., 2011) and phishing attacks (Fette et al., 2007; Bergholz et al., 2008; Moore et al., 2009; Bergholz et al., 2010). Moreover, the accessibility of various data sources, which can be correlated to complement their incompleteness, has brought new opportunities and challenges to researchers. Unfortunately, most studies either use a single data source or work on a static spam data that is collected during a specific time frame (Pitsillidis et al., 2012). More importantly, spammers have been constantly modernizing their arsenals to defeat the anti-spam efforts. Spamming techniques have evolved remarkably from simple programs to sophisticated spamming software, which disseminate template-generated spam through a network of compromised machines. Botnet-disseminated spam emails are usually orchestrated into large-scale campaigns and act as the pivotal instrument for several cyber-based criminal activities. As a consequence, it is critical to perform an analysis of spam data, especially *spam campaigns*, for the

☆ The authors gratefully acknowledge support from the Canadian Radio-television and Telecommunications Commission (CRTC).
* Corresponding author.
E-mail address: tsondt@gmail.com (S. Dinh).

http://dx.doi.org/10.1016/j.diin.2015.01.006

purpose of cyber-crime investigation. Nevertheless, given the stunning number of spam emails, it is implausible to analyze them manually. Therefore, cyber-crime investigators need automatic techniques and tools to accomplish this task.

In this research, we aim at elaborating methodologies for spam campaign detection, analysis and investigation. We also emphasize the importance of correlating different data sources to reveal spam campaign characteristics. More precisely, we propose a framework that: (1) Consolidates spam emails into campaigns. (2) Labels spam campaigns by generating related topics for each campaign from Wikipedia data. (3) Correlates different data sources, namely *passive DNS*, malware, WHOIS information and geolocation, to provide more insights into spam campaign characteristics. (4) Scores spam campaigns based on several customizable criteria.

The identification of spam campaigns is a crucial step for analyzing spammers' strategies for the following reasons. First, the amount of spam data is astronomical, and analyzing all spam messages is costly and almost impossible. Hence, clustering spam data into campaigns reduces significantly the amount of data to be analyzed, while maintaining their key characteristics. Second, because of the characteristics of spam, spam messages are usually sent in bulk with specific purposes. Hence, by clustering spam messages into campaigns, we can extract relevant insights that can help investigators understand how spammers obfuscate and disseminate their messages. Labeling spam campaigns and correlating different data sources reveal the characteristics of the campaigns and therefore, significantly increase the effectiveness of an investigation. Moreover, scoring spam campaigns helps investigators concentrate on campaigns that cause more damage (e.g., malware distribution or phishing).

The remainder of this paper is organized as follows. In Section 2, we present the related work. We discuss existing techniques for detecting spam campaigns in Section 3. In Section 4, we present our framework. Experimental results are presented in Section 5. Finally, we conclude the paper in Section 6.

Related work

Several approaches for clustering spam emails into groups have been proposed in the literature:

URL-based spam email clustering

In this category, spam emails are clustered using features such as the embedded URLs or the source IP addresses of spam emails. F. Li et al. (Li and Hsieh, 2006) propose an approach for clustering spam emails based on identical URLs. The approach also uses the amount of money mentioned inside the content of the email as an extra feature of the cluster. Xie et al. (2008) propose AutoRE, a framework that detects botnet hosts by signatures that are generated from URLs embedded in email bodies. Both of these URL-based clustering approaches are very efficient in terms of performance and number of false positives. However, spammers can easily evade such techniques using dynamic source IP addresses, URL shorten services or polymorphic URLs.

Spam email clustering using text mining methods

Zhuang et al. (2008) develop techniques to unveil botnets and their characteristics using spam traces. Spam emails are clustered together into campaigns using shingling algorithm (Broder et al., 1997). The relationship between IP addresses determines if the campaigns are originated from the same botnet. In Qian et al. (2010), Qian et al. design an unsupervised, online spam campaign detection, namely *SpamCampaignAssassin* (SCA), and apply extracted campaign signatures for spam filtering. SCA utilizes a text-mining framework built on *Latent Semantic Analysis* (LSA) (Landauer et al., 1998) to detect campaigns. Nevertheless, template-generated spam and scalability render text mining ineffective.

Spam email clustering using web pages retrieved from embedded URLs

Anderson et al. (2007) propose a technique called *spamscatter* that follows embedded URLs inside spam emails, renders and clusters the websites into scams using *image shingling*. Konte et al. (2009) analyze the scam hosting infrastructure. The authors extract *scam campaigns* by clustering web pages retrieved from URLs inside spam emails using different heuristic methods. Even though URL content provides sound results, scalability is also an issue. More importantly, actively crawling embedded URLs may alert spammers and expose the spamtraps used to collect data.

Spam email clustering based on email content

Wei et al. (2008) propose an approach based on the agglomerative hierarchical clustering algorithm and the connected components with weighted edges model to cluster spam emails. Only spam emails used for advertising are tested by the authors. In Calais et al. (2008), Calais et al. introduce a spam campaign identification technique based on frequent-pattern tree (FP-Tree) (Han et al., 2000, 2004). Spam campaigns are identified by pointing out nodes that have a significant increase in the number of children. An advantage of this proposed technique is that it can detect obfuscated parts of spam emails naturally without prior knowledge. In another work (Guerra et al., 2008), Calais et al. briefly present *Spam Miner*, a platform for detecting and characterizing spam campaigns. Kanich et al. (2009) analyze spam campaigns using a parasitic infiltration of an existing botnet's infrastructure to measure spam delivery, click-through and conversion. Haider and Scheffer (2009) apply Bayesian hierarchical clustering (Heller and Ghahramani, 2005) to group spam messages into campaigns. The authors develop a generative model for clustering binary vectors based on a transformation of the input vectors.

Spam campaign detection techniques

A straightforward approach for grouping spam emails into campaigns is to calculate the distances between spam

emails and then apply a clustering technique, which generates clusters of emails that are "close" to each other. The distance between two emails can be measured by computing the similarity of their textual contents, e.g., *w-shingling* and the Jaccard coefficient (Broder et al., 1997), *Context Triggered Piecewise Hashing* (CTPH) (Kornblum, 2006) or *Locality-Sensitive Hashing* (LSH) (Damiani et al., 2004). The first metric has been used in many studies (Zhuang et al., 2008; Anderson et al., 2007; Gao et al., 2010) while the second and the third metrics are originally utilized for spam mitigation (spamsum, 2013; Damiani et al., 2004). However, one disadvantage of this approach is its non-scalability because of the enormous amount of spam emails to be analyzed. Additionally, the use of templates for content generation and numerous obfuscation techniques has remarkably increased the diversity of spam emails in one campaign. For these reasons, clustering techniques based on the textual similarity of spam emails have become ineffective.

We approach the problem of identifying spam campaigns based on the premise that spam emails sent in one campaign are disseminated autonomously by the same mean. As a result, they must have the same goal and share common characteristics. However, a spammer may employ many obfuscation techniques on the subject, the content and the embedded URL(s). For instance, we have observed that spammers may alter the email subjects in the same campaign, but these subjects still have the same meaning when interpreted manually. This behavior has also been witnessed in (Pitsillidis et al., 2010). Another observation is that a spam email may contain a random text from a book or a Wikipedia article (Stringhini et al., 2011). The text is written by a human and therefore cannot be easily distinguished using statistical methods. Spammers also utilize fast-flux service networks and randomly generated subdomains (Wei et al., 2009) to obfuscate the embedded URLs. Furthermore, spammers usually insert random strings to the embedded URLs to confuse spam filters, or include tracking data to verify their target email lists and monitor their progress.

Because the obfuscation can be employed at any part of a spam email, our spam campaign detection technique must consider features from the whole spam email. We chose the features for our technique from the header, the textual content, the embedded URL(s) and the attachment(s) of spam emails. Our main spam campaign detection method is based on the *frequent-pattern tree* (FP-Tree), which is the first step of the FP-Growth algorithm (Han et al., 2000, 2004). The FP-Tree technique was originally used by Calais et al. (2008). However, our approach is different in terms of the features used as well as the signals used to identify spam campaigns from the previously constructed FP-Tree. Furthermore, we enhance our method to detect spam campaigns on-the-fly by adopting a technique that incrementally builds the FP-Tree (Cheung and Zaiane, 2003). A detailed explanation of our methodology can be found in the next section.

Our software framework

As depicted in Fig. 1, the main components of our spam campaign detection, analysis and investigation framework are the following:

- *Central Database*

The database is carefully chosen so that it is scalable, flexible and able to store relationships between different entities. Section 4.1 elucidates our choice.

- *Parsing and Features Extraction*

This component parses spam emails, extracts and stores their features in the central database for further analysis. More details about this component are given in Section 4.2.

- *Spam Campaign Detection*

This module takes the features from the central database as its inputs, identifies spam campaigns and saves them back into the database. Section 4.3 explains this module thoroughly.

- *Spam Campaign Characterization*

This component gives insights into spam campaigns to further reduce the analysis time and effort. We combine different data sources, label and score spam campaigns to help investigators quickly grasp the objective of a campaign and concentrate on what they are after. This component is described in detail in Section 4.4.

Central database

Due to the tremendous number of spam emails, NoSQL databases such as document-based ones are considered instead of the traditional SQL-based RDBMS. Document-oriented databases are acknowledged for their flexibility and high scalability. However, most of them do not have a powerful query language like SQL. Moreover, the ability to represent relationships between spam emails, spam campaigns, IP addresses and domain names is essential. These relationships can be stored and processed efficiently in a graph database, which is based on graph theory. In our system, we employ *OrientDB* (OrientDB, 2013), which is an open-source NoSQL database management system. OrientDB has the flexibility of document-based databases as well as the capability of graph-based ones for managing relationships between records. It has the advantages of being fast, scalable and supports SQL-styled queries. Spam emails are stored in the database as documents with different fields, such as header fields, textual content, embedded URL(s), etc. Additionally, spam campaigns, IP addresses, domain names and attachments are also saved as different document types. For each document type, the database maintains a hash table of a specific field to eliminate duplications. The database also stores the relationships between different documents, such as between spam emails and spam campaigns, spam emails and attachments or spam campaigns and IP addresses.

Parsing and feature extraction

Our parser takes spam emails, each of which consists of a header and a body, as its inputs. The header has many different fields, which are name/value pairs separated by

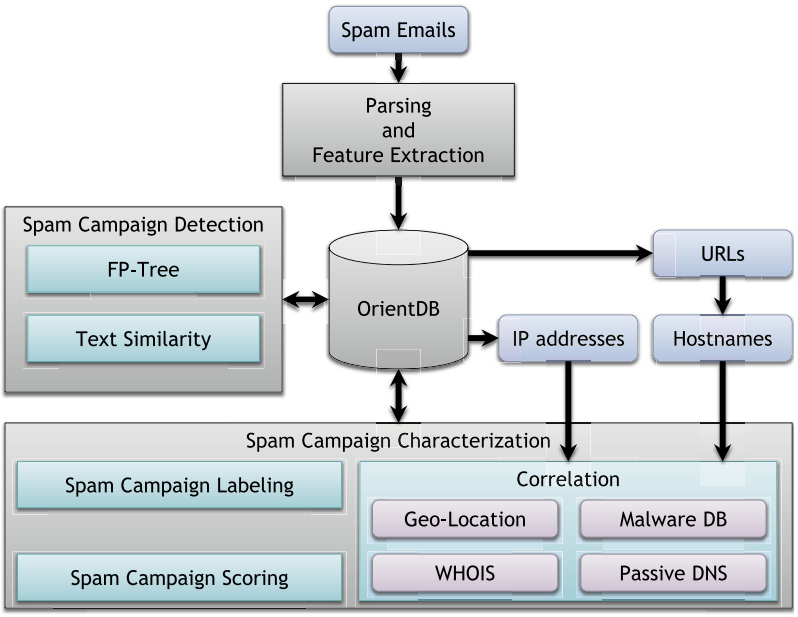


Fig. 1. Architecture of our software framework.

":". The body may have only one part or multiple parts. Relevant features needed for the analysis are extracted. More precisely, we consider the following features:

- *Content Type*: The first part of the Content-Type header field describes the MIME types of the email: text/plain, text/html, application/octet-stream, multipart/alternative, multipart/mixed, multipart/related and multipart/report.
- *Character Set*: The encoding of a spam email can be extracted from the header fields or inferred from the email content. This encoding roughly represents the spam email language. This feature is rarely obfuscated by spammers due to the fact that spam emails must be decoded successfully to be read.
- *Subject*: The whole subject line is considered as a feature. If the subject is encoded using Q-encoding, it is decoded into Unicode.
- *Email Layout*: In the case of text/plain, the layout is a sequence of characters "T", "N" and "U", which stand for text, newline and URL, respectively. For text/html, the layout is the top three levels of the DOM tree. For multipart emails, we use the tree structure of the email body to form the layout.
- *URL Tokens*: The embedded URLs of spam emails are split into tokens: the hostname, the path and the parameters. Each token is considered as a feature.
- *Attachment Name(s)*: Each attachment name is considered as a feature.

Spam campaign detection

Our spam campaign detection method is based on the *Frequent-pattern tree* (FP-Tree) (Han et al., 2000, 2004). We employ the FP-Tree based on the premise that the more frequent an attribute is, the more it is shared among spam emails. Less frequent attributes usually correspond to the parts that are obfuscated by spammers. Thus, building an FP-Tree from email features allows spam emails to be grouped into campaigns. Furthermore, obfuscated features are discovered naturally within the FP-Tree and therefore exposing the strategies of spammers. Two scans of the dataset are needed to construct the FP-Tree. The cost of inserting a feature vector *fv* into the FP-Tree is $O(|fv|)$, where $|fv|$ indicates the number of features in *fv*. The FP-Tree usually has a smaller size than the dataset since feature vectors that share similar features are grouped together. Hence, this structure reduces the amount of data that needs to be processed. In the following, we present our FP-Tree-based spam campaign detection algorithms.

Frequent-pattern tree construction

We define two data structures, *FPTree* and *FPNode*:

- The *FPTree* has one *root* and one method:
  - The *root* is an *FPNode* labeled *null*.
  - The *add(transaction)* method (Algorithm 1) accepts a feature vector as its only parameter.

Algorithm 1 *FPTree.add(fv)* Method

```
1:  function add(fv)
2:      n ← root
3:      for each f in fv do
4:          next_n ← n.search(f)
5:          if next_n is None then
6:              next_n ← New FPNode(f)
7:              Add next_n as a child of n
8:          end if
9:          n ← next_n
10:     end for
11: end function
```

- The *FPNode* has eight properties and two methods:
  - The *tree* property indicates the *FPTree* of this *FPNode*.
  - The *item* property represents the feature.
  - The *root* property is *True* if this *FPNode* is the root; *False* if otherwise.
  - The *leaf* property is *True* if this *FPNode* is a leaf; *False* if otherwise.
  - The *parent* property indicates the *FPNode* that is the parent of this *FPNode*.
  - The *children* property is a list of *FPNode*s that are the children of this *FPNode*.
  - The *siblings* property is a list of *FPNode*s that are the siblings of this *FPNode*.
  - The *add(node)* method adds the *FPNode* "node" as a child of this *FPNode* and adds this *FPNode* as the parent of the *FPNode* "node".
  - The *search*(*item*) method checks to see if this *FPNode* has a child "item".

We build the FP-Tree by creating the *null FPNode*, which is the root of the tree, and adding the feature vectors individually. The features in each feature vector are sorted in descending order based on their occurrences in the dataset. Algorithm 2 comprehensively demonstrates this process.

Algorithm 2 FP-Tree Construction

```
Transactions ← ( )                          ▷ An empty list structure
items ← { }                                 ▷ An empty key:value structure
for each spam email do
    feature_vector ← ( )                    ▷ An empty list structure
    for each feature do
        Add feature to feature_vector
        items[feature] ← items[feature] + 1
    end for
    Add spam_email_ID to the end of feature_vector
    Add feature_vector to Transactions
end for
the_fp_tree ← FPTree()
for each transaction in Transactions do
    if |transaction| > 2 then
        Sort (descending) the items in transaction[1 : −1] based on their counts in items
                ▷ The content_type at the beginning and the spam_email_ID at the end are omitted
        the_fp_tree.add(transaction)
    end if
end for
```

At the end, we obtain the FP-Tree, in which each node represents a feature extracted from spam emails. Each path from one leaf to the root represents the feature vector of each spam message. From the FP-Tree, we can see that two messages that have some common features share a portion of the path to the root. We use this characteristic to cluster spam emails, which have several common features, into campaigns.

Spam campaign identification

From the constructed FP-Tree, we extract spam campaigns by traversing the tree using the depth-first search method. When visiting each node, we check for the following conditions:

1. The number of children of that node must be greater than the threshold min_num_children.
2. The average frequency of the children of that node must be greater than the threshold *freq_threshold*.
3. In the path from that node to the root, there must be one node that does not have its feature type in the *n_obf_features list*.
4. The number of leaves of the sub-tree starting from that node must be greater than a threshold *min_num_messages*.

If a node satisfies all of the above conditions, all the leaves of the sub-tree that have that node as root are the spam emails of a spam campaign. We store the campaign, delete the sub-tree and continue the process until all the remaining nodes in the tree are visited.

The FP-Tree technique is very efficient and naturally reveals the features that are common in the spam emails of a campaign as well as the features that are obfuscated by spammers (Calais et al., 2008). However, the disadvantage of this technique is that it only works with a static set of data. It has been demonstrated that spam campaigns may last for a long period of time (Pathak et al., 2009). Consequently, the ability to identify spam campaigns in their early stage gives investigators an upper hand in their pursuit of the spammers. We improve our FP-Tree-based spam campaign identification algorithm by constructing the FP-Tree on-the-fly. We extract feature vectors from spam emails as soon as they arrive and insert those vectors into the tree. In the following, we provide more details about this process.

Incremental FP-Tree

Since the FP-Tree approach (Han et al., 2000, 2004) was proposed for mining frequent patterns, many studies have been conducted to improve the functionality (Kai-Sang Leung, 2004) as well as the performance (Ong et al., 2003) of this technique. Moreover, some FP-Tree-based incremental mining algorithms have been proposed (Cheung and Zaiane, 2003; Leung et al., 2007). We adopt the ideas from (Cheung and Zaiane, 2003) to construct the incremental FP-Tree as follows:

1. Create a *null* root.
2. Each new transaction is added starting from the root level.
3. At each level:

(a) The items of the new transaction are compared with the items of the children nodes.

(b) If the new transaction and the children nodes have items in common, the node having the highest count is chosen to merge with the item in the transaction. Thus, the count of that node is increased by 1. The item of the merged node is then removed from the transaction. If the count of a descendant node becomes higher than the count of its ancestor, the descendant node has to be

Table 1
Statistics of the dataset.

	1 year (Apr. 2012–Mar. 3013)	1 month (Apr. 2013)
Messages	678,095	100,339
Unique IP addresses	91,370	16,055
Unique hostnames	321,549	74,833
Unique attachments	5537	393

moved in front of its ancestor and merged with the item in the transaction.
(c) This process is repeated recursively with the remainder of the new transaction.

4. Any remaining items in the new transaction are added as a new branch to the last merged node.
5. Run the spam campaign detection process regularly to detect new campaigns or merge with the existing ones.

Spam campaign characterization

Grouping spam emails into campaigns has greatly reduced the amount of data that needs to be analyzed. However, when examining a specific spam campaign, an investigator still needs to skim through a significant number of spam emails to grasp the essential information. Therefore, a summary of the spam content is valuable to the investigator. We employ techniques to automatically label spam campaigns to give the investigator an overall view of them. We also develop a scoring mechanism to rank spam campaigns so that the investigator can concentrate on the highly-ranked ones. Additionally, spam campaigns may reveal some characteristics that are hidden when analyzing spam emails separately. For example, the relationships between different domains and IP addresses or between spamming servers. Therefore, we utilize and correlate various kinds of information, such as WHOIS information, passive DNS data, geo-location and malware database, to assist investigators in tracing spammers. Moreover, we employ visualization tools to illustrate the relationships between spam emails, spam campaigns, domains and IP addresses. The visualization helps to emphasize critical information in the dataset, for instance, an aggressive spamming IP address. In the following, we provide more details about our characterization methods.

Table 2
Distribution of 'raw' MIME types (1 month).

MIME type	Count	Percentage
text/html	52,596	50.2345%
text/plain	32,977	31.49.54%
multipart/alternative	16,270	15.5395%
multipart/mixed	2465	2.3543%
multipart/related	264	0.2521%
multipart/report	60	0.0573%
text/html\n\tcharset = "windows-1250"	19	0.0181%
text/html\n\tcharset = "us-ascii"	15	0.0143%
text/html\n\tcharset = "iso-8859-1"	14	0.0134%
text/html\n\tcharset = "windows-1252"	13	0.0124%
text/html\n\tcharset = "iso-8859-2"	8	0.0076%
application/octet-stream	1	0.0010%

Table 3
Distribution of 'raw' MIME types (1 year).

MIME type	Count	Percentage
text/plain	353,774	52.1431%
text/html	213,578	31.4795%
multipart/alternative	73,436	10.8238%
multipart/mixed	32,797	4.8340%
multipart/related	2,134	0.3145%
multipart/report	1,878	0.2768%
text/html\n\tcharset = "windows-1252"	131	0.0193%
text/html\n\tcharset = "windows-1250"	127	0.0187%
text/html\n\tcharset = "iso-8859-2"	126	0.0186%
text/html\n\tcharset = "us-ascii"	125	0.0184%
text/html\n\tcharset = "iso-8859-1"	108	0.0159%
application/octet-stream	75	0.0111%
text/plain charset = utf-8	35	0.0052%
text/plain charset = us-ascii	29	0.0043%
text/plain charset = "us-ascii"	28	0.0041%
text/plain charset = "utf-8"	26	0.0038%
text/plain charset = "iso-8859-1"	25	0.0037%
text/plain charset = iso-8859-1	25	0.0037%
text/plain\n\tcharset = "windows-1252"	4	0.0006%
text/plain\n\tcharset = "us-ascii"	3	0.0004%
text/plain\n\tcharset = "iso-8859-2"	1	0.0001%
text/plain\n\tcharset = "iso-8859-1"	1	0.0001%
text/plain\n\tcharset = "windows-1250"	1	0.0001%

Correlation module

In this module, we utilize WHOIS information (Harrenstien et al., 1985; Daigle, 2004) and passive DNS data, received from a trusted third party, to gain more insights into the domain names found in spam emails. WHOIS information is associated with each second-level domain while passive DNS data gives the historical relationships between hostnames and IP addresses along with their time-stamps. In addition, we use geo-location and malware data to add valuable information to the IP addresses that we have found in spam campaigns. The IP addresses can be extracted from the `Received` header fields or from the passive DNS data.

Spam campaign labeling

We propose techniques for labeling spam campaigns to provide an overview of their content. First, we employ an automatic labeling technique that has been proposed by Lau et al. (Lau et al., 2011). Even though this method works in some cases, it is not scalable because of the large number of queries to Wikipedia. Thus, we apply another method that utilizes *Wikipedia Miner toolkit* (Milne and Witten, 2013), which stores summarized versions of Wikipedia in offline databases. We create a *Spam Campaign Labeling Server*, which listens to a specific network port, using the

Table 4
Distribution of the correct MIME types.

MIME type	1 year		1 month	
text/html	353,952	52.1694%	52,665	50.3004%
text/plain	214,195	31.5704%	32,977	31.4954%
multipart/alternative	73,436	10.8238%	16,270	15.5395%
multipart/mixed	32,797	4.8340%	2465	2.3543%
multipart/related	2134	0.3145%	264	0.2521%
multipart/report	1878	0.2768%	60	0.0573%
application/octet-stream	75	0.0111%	1	0.0010%

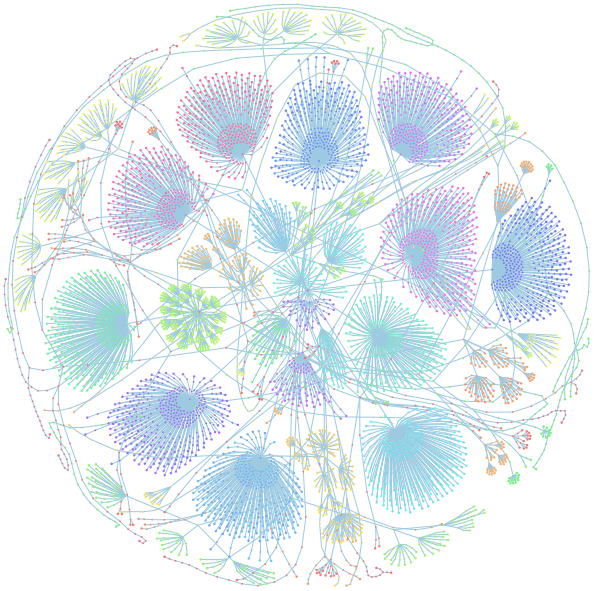


Fig. 2. Visualization of the FP-Tree constructed from a one-day spam data.

libraries from Wikipedia Miner toolkit. We extract frequent words from the content of spam emails in a campaign and feed them to the *Spam Campaign Labeling Server*. The response, which contains a list of relevant topics of that spam campaign, is saved back into the central database.

Spam campaign scoring

In most cases, investigators only need to pursue a particular objective in their investigations. Therefore, we suggest a method that has been verified by a law enforcement official to assign each spam campaign a score. The score of each campaign is determined by the sum of various weighted elements. Each element is the count of an attribute, a characteristic or a signal that may appear in spam emails of a campaign. For example, an investigator who is interested in spammers targeting Canadians may want to put more effort into spam campaigns that have ".ca" domain names in the email addresses of the recipients. We implement the attributes both from our speculation and from the advice of a Canadian law enforcement official. Some of the signals are Canadian IP addresses, ".ca" top-level domain names, IP ranges of Canadian corporations, etc.

Results

In this section, we present the results of our spam campaign analysis. We also show some visualizations that can provide investigators with a spectacular view of the spam campaigns to further understand the behaviors of spammers.

The dataset

Table 1 shows some statistics of our spam dataset that we received from a trusted third party. Table 2 and Table 3

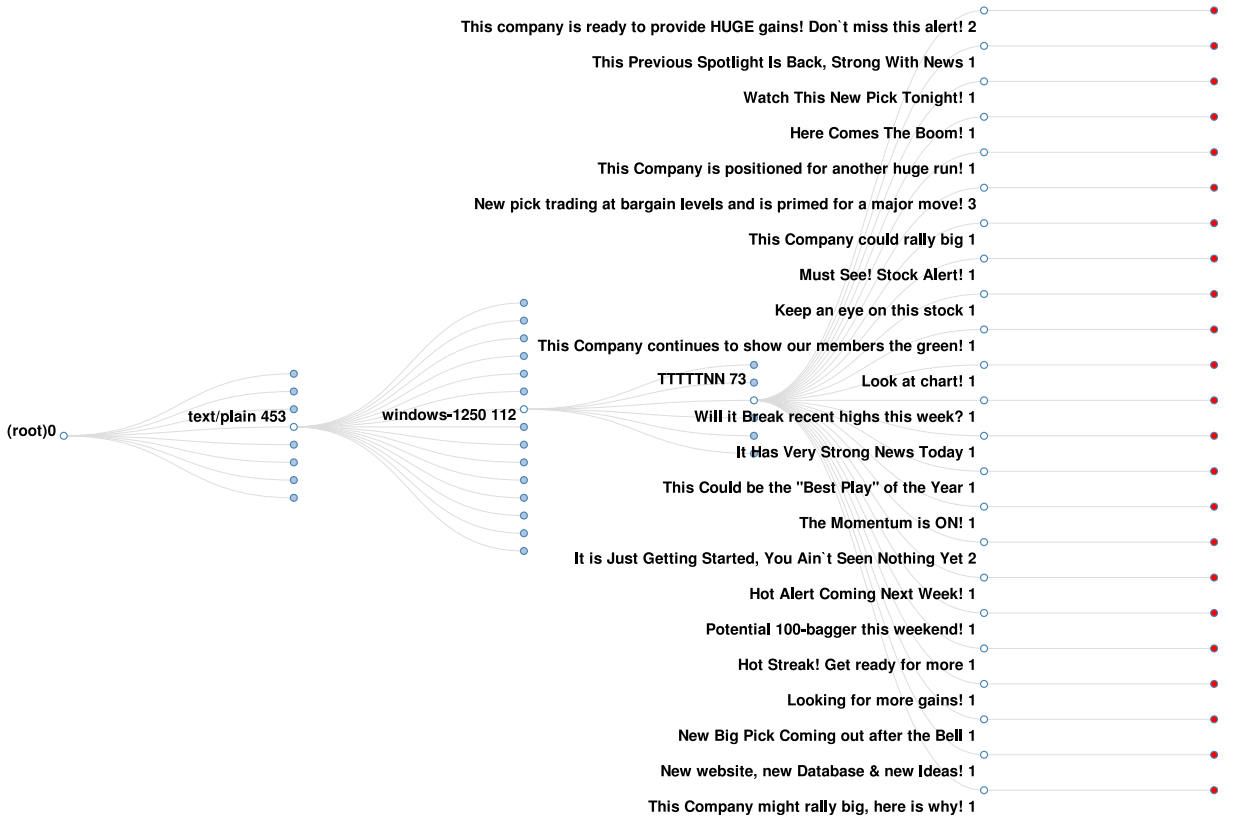


Fig. 3. Visualization of one part of the frequent-pattern tree.

Table 5
Processing time.

	1 year	1 month
Parsing time	10885.2 s	2467.3 s
Feature extracting time	204.6 s	78.0 s
FP-Tree building time	36.7 s	7.8 s
Campaign detecting time	35.5 s	5.4 s

Table 6
Distribution of the decisive features used to identify spam campaigns.

Feature	1 year	1 month
Layout	2,778	641
Hostname	1,036	190
Subject	1,160	134
URL	602	106
Attachment	108	15
Character set	3	2

depict the distribution of the spam email MIME types. The MIME type and the character set are rarely obfuscated by spammers. However, spam messages may not follow the standard rules, which leads to the appearance of some non-standard MIME types. This valuable signal can be used to detect spam emails that are generated by the same automatic mean. The distribution of the correct MIME types is shown in Table 4.

Spam campaign detection

Fig. 2 depicts a full FP-Tree that is built using a *one-day* spam data (Data-Driven Documents) (CodeFlower Source code visualization). This figure clearly shows groups of nodes inside the FP-Tree, proving that the spam emails can be grouped into spam campaigns by using the FP-Tree structure. Fig. 3 illustrates a part of the FP-Tree that we have built (Data-Driven Documents). The node labeled "*TTTTTNN*" represents the layout of the emails. We can clearly see that all the spam emails of the campaign share the same MIME type ("*text/plain*"), the same character set ("*windows-1250*") and the same layout ("*TTTTTNN*"). However, they have different subject lines although those subjects have the same meaning (about stock markets).

Table 5 shows the time needed to extract spam campaigns from our dataset. The program runs on a server that has the Intel® Xeon® X5660 (2.80 GHz) CPU and 32 GB of memory. The most time-consuming task is the parsing of spam email files to insert into the main database (around 3 h for one-year data and 40 min for one-month data). The remaining steps only take around 5 min for one-year data and 1 min 30 s for one-month data to complete. We use one-year data just to demonstrate the efficiency of the FP-Tree method. The following sections present the results using one-month data.

FP-tree features

In Table 6, we show the critical features that are used to identify spam campaigns. Moreover, we perform a more in-depth analysis of the relation between the decisive features and the parameters of our spam campaign detection algorithm. The process is as follows:

- First, the *min_num_messages* parameter is fixed to 5.
- For each step, the *min_num_children* parameter is increased by 1 (from 2 to 20). The number of detected spam campaigns corresponding to the decisive features are recorded 8 times (when the *freq_threshold* parameter is set to 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5 and 5.0, respectively). The *freq_threshold* of 1.5 means that the average frequency of the children nodes is larger or equal to 1.5 times the frequency of the current node.

The results of this analysis are shown in Fig. 4.

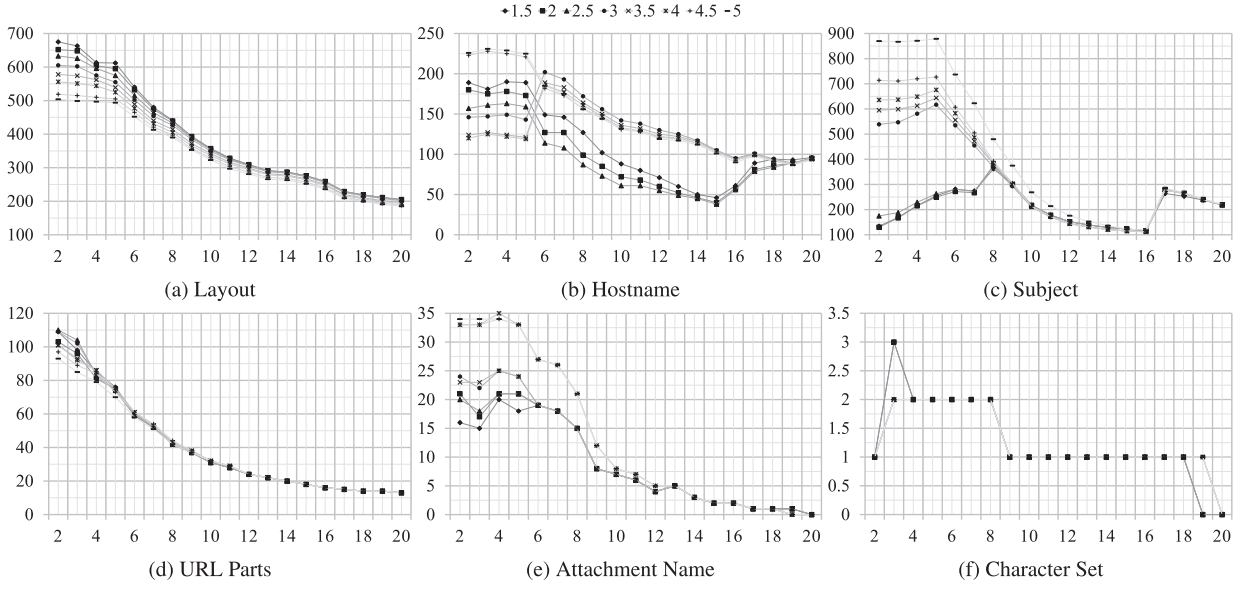
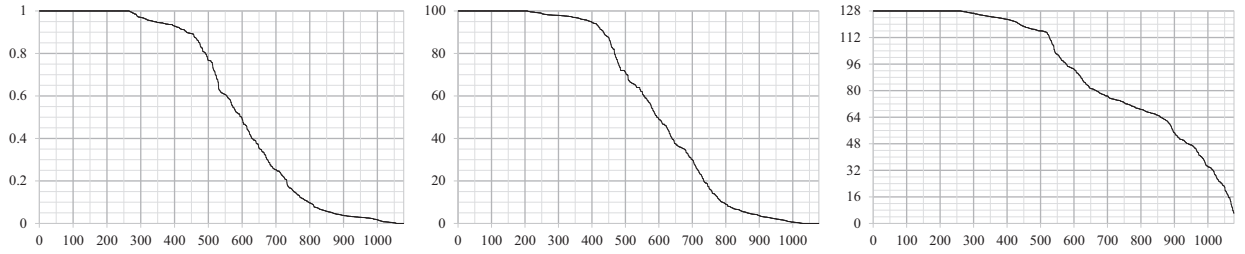


Fig. 4. Relation between the FP-Tree features and the algorithm's parameters.



(a) W-shingling (b) Context Triggered Piecewise Hashing (c) Locality-Sensitive Hashing

Fig. 5. Spam campaign text similarity scores.

Text similarity

After spam campaigns have been identified using the FP-Tree, we calculate the text similarity scores of each campaign. We apply three measurements to generate three similarity scores of all the email pairs in each campaign: *w-shingling* and the Jaccard coefficient (Broder et al., 1997), *Context Triggered Piecewise Hashing* (CTPH) (Kornblum, 2006) and *Locality-Sensitive Hashing* (LSH) (Damiani et al., 2004). A spam campaign has three similarity scores, each of which is the average score of each method. Fig. 5 shows the results of this process. The vertical axis represents the score of each metric while the horizontal axis is the campaigns sorted by their scores. Table 7 summarizes the results.

Calculating text similarity between messages in each spam campaign is a very time-consuming process. It takes 13,339.6 s (around 3 h 42 min), 3942.5 s (around 65 min) and 584,486 s (nearly one week) to complete the calculation of the w-shingling method, the Context Triggered Piecewise Hashing method and the Locality-Sensitive Hashing method, respectively.

Spam campaign characterization

In this subsection, we present some characteristics of the spam campaigns identified from a one-month data. In Fig. 6, we show the spam campaigns along with the spam messages, the related IP addresses and the related hostnames. We select top 5 spam campaigns (with big numbers in the figure) that have the highest scores and have less than 1000 messages for the purpose of this illustration. The scores are computed based on some criteria that have been mentioned in our spam campaign scoring method. In Fig. 7, a word cloud, which illustrates the topics of the spam campaigns, is presented.

Conclusion

Spam campaigns act as the pivotal instrument for several cyber-based criminal activities. Consequently, the analysis of spam campaigns is a critical task for cyber security officers. In this paper, we propose a software framework for spam campaign detection, analysis and investigation. The framework provides law enforcement officials with a platform to conduct investigations on cyber crimes. Our system greatly reduces investigation efforts by consolidating spam emails into campaigns. Moreover, it aids investigators by providing various information about spam campaigns. Spam campaign characteristics are also unveiled by labeling spam campaigns and correlating

Table 7
Summary of campaign text similarity scores.

	w-s		CTPH		LSH	
>=50%	598	54.96%	595	54.69%	859	78.95%
100%	265	24.36%	206	18.93%	260	23.90%
0%	20	1.84%	46	4.23%	0	0.00%
No text	10	0.92%	10	0.92%	10	0.92%

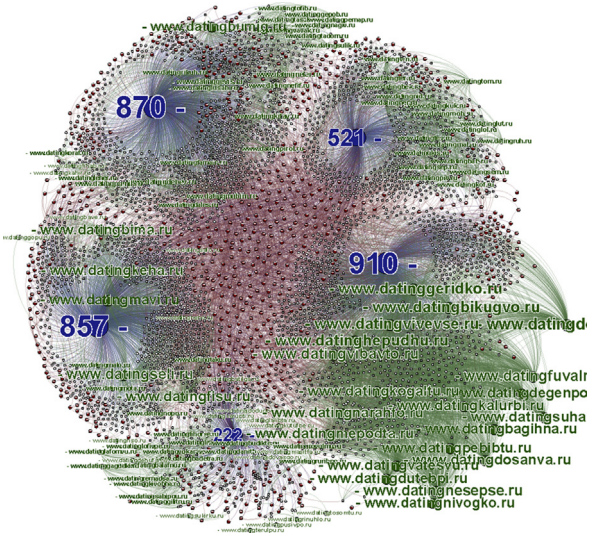


Fig. 6. Five spam campaigns that have the highest scores and less than 1000 emails

Fig. 7. Topics of spam campaigns.

different data sources. Additionally, we employ a feature-rich and scalable database to handle a large number of spam emails, a scoring mechanism to highlight severe spam campaigns and a visualization tool to further assist investigators. Our system has been adopted by a governmental organization and used by law enforcement officials to pursuit spammers, take down spamming servers and reduce spam volume.

References

Anderson DS, Fleizach C, Savage S, Voelker GM. Spamscatter: characterizing internet scam hosting infrastructure. In: Proceedings of 16th USENIX security symposium, SS'07; 2007. pp. 10:1–10:14.

Bergholz A, Chang JH, Paaß G, Reichartz F, Strobel S. Improved phishing detection using model-based features. In: CEAS; 2008.

Bergholz A, De Beer J, Glahn S, Moens M-F, Paaß G, Strobel S. New filtering approaches for phishing email. J Comput Secur 2010;18(1):7–35.

Broder AZ, Glassman SC, Manasse MS, Zweig G. Syntactic clustering of the web. Comput Netw ISDN Syst 1997;29(8):1157–66.

Calais P, Pires DE, Neto DOG, Meira Jr W, Hoepers C, Steding-Jessen K. A campaign-based characterization of spamming strategies. In: CEAS; 2008.

Cheung W, Zaiane OR. Incremental mining of frequent patterns without candidate generation or support constraint. In: Database engineering and applications symposium, 2003. Proceedings. Seventh international, IEEE; 2003. p. 111–6.

CodeFlower Source code visualization. http://redotheweb.com/CodeFlower/.

Daigle L. WHOIS protocol specification. Internet RFC 2004;3912. ISSN: 2070-1721.

Damiani E, di Vimercati SDC, Paraboschi S, Samarati P. An open digest-based technique for spam detection. In: ISCA PDCS; 2004. p. 559–64.

Data-Driven Documents, http://d3js.org/.

Fette I, Sadeh N, Tomasic A. Learning to detect phishing emails. In: Proceedings of the 16th international conference on world wide Web. ACM; 2007. p. 649–56.

Gao H, Hu J, Wilson C, Li Z, Chen Y, Zhao BY. Detecting and characterizing social spam campaigns. In: Proceedings of the 10th ACM SIGCOMM conference on internet measurement. ACM; 2010. p. 35–47.

Guerra P, Pires D, Guedes D, Meira Jr W, Hoepers C, Steding-Jessen K. Spam miner: a platform for detecting and characterizing spam campaigns. In: Proc. 6th Conf. Email Anti-Spam; 2008.

Haider P, Scheffer T. Bayesian clustering for email campaign detection. In: Proceedings of the 26th annual international conference on machine learning. ACM; 2009. p. 385–92.

Han J, Pei J, Yin Y. Mining frequent patterns without candidate generation. ACM SIGMOD Rec 2000;29(2):1–12.

Han J, Pei J, Yin Y, Mao R. Mining frequent patterns without candidate generation: a frequent-pattern tree approach. Data Min Knowl Discov 2004;8(1):53–87.

Harrenstien K, Stahl M, Feinler E. WHOIS protocol specification. Internet RFC 1985;954. ISSN: 2070-1721.

Heller KA, Ghahramani Z. Bayesian hierarchical clustering. In: Proceedings of the 22nd international conference on machine learning. ACM; 2005. p. 297–304.

John JP, Moshchuk A, Gribble SD, Krishnamurthy A. Studying spamming botnets using botlab. In: NSDI, vol. 9; 2009. p. 291–306.

Kai-Sang Leung C. Interactive constrained frequent-pattern mining system. In: Database engineering and applications symposium, 2004. IDEAS'04. Proceedings. International, IEEE; 2004. p. 49–58.

Kanich C, Kreibich C, Levchenko K, Enright B, Voelker GM, Paxson V, et al. Spamalytics: an empirical analysis of spam marketing conversion. Commun ACM 2009;52(9):99–107.

Konte M, Feamster N, Jung J. Dynamics of online scam hosting infrastructure. In: Passive and active network measurement. Springer; 2009. p. 219–28.

Kornblum J. Identifying almost identical files using context triggered piecewise hashing. Digit Investig 2006;3:91–7.

Landauer TK, Foltz PW, Laham D. An introduction to latent semantic analysis. Discourse Process 1998;25(2–3):259–84.

Lau JH, Grieser K, Newman D, Baldwin T. Automatic labelling of topic models. ACL 2011;2011:1536–45.

Leung CK-S, Khan QI, Li Z, Hoque T. Cantree: a canonical-order tree for incremental frequent-pattern mining. Knowl Inform. Syst 2007;11(3):287–311.

Li F, Hsieh M-H. An empirical study of clustering behavior of spammers and group-based anti-spam strategies. In: CEAS; 2006.

Milne D, Witten IH. An open-source toolkit for mining wikipedia. Artificial Intelligence; 2013.

Moore T, Clayton R, Stern H. Temporal correlations between spam and phishing websites. In: Proc. of 2nd USENIX LEET; 2009.

Ong K-L, Ng W-K, Lim E-P. Fssm: fast construction of the optimized segment support map. In: Data warehousing and knowledge discovery. Springer; 2003. p. 257–66.

OrientDB, http://www.orientdb.org/, last accessed in August 2013.

Pathak A, Qian F, Hu YC, Mao ZM, Ranjan S. Botnet spam campaigns can be long lasting: evidence, implications, and analysis. In: Proceedings of the eleventh international joint conference on measurement and modeling of computer systems. ACM; 2009. p. 13–24.

Pitsillidis A, Levchenko K, Kreibich C, Kanich C, Voelker GM, Paxson V, et al. Botnet judo: fighting spam with itself. In: NDSS; 2010.

Pitsillidis A, Kanich C, Voelker GM, Levchenko K, Savage S. Taster's choice: a comparative analysis of spam feeds. In: Proceedings of the 2012 ACM conference on internet measurement conference, IMC'12; 2012. p. 427–40.

Qian F, Pathak A, Hu YC, Mao ZM, Xie Y. A case for unsupervised-learning-based spam filtering. ACM SIGMETRICS Perform Eval Rev 2010;38(1):367–8.

spamsum, http://www.samba.org/ftp/unpacked/junkcode/spamsum/README, (last accessed in August 2013).

Stringhini G, Holz T, Stone-Gross B, Kruegel C, Vigna G. Botmagnifier: locating spambots on the internet. In: USENIX security symposium; 2011.

Symantec Intelligence Report. Report. May 2013. http://www.symantec.com/content/en/us/enterprise/other_resources/b-intelligence_report_05-2013.en-us.pdf.

Thonnard O, Dacier M. A strategic analysis of spam botnets operations, in: proceedings of the 8th annual Collaboration. In: Electronic messaging, anti-abuse and spam conference. ACM; 2011. p. 162–71.

Wei C, Sprague A, Warner G, Skjellum A. Mining spam email to identify common origins for forensic application. In: Proceedings of the 2008 ACM symposium on applied computing. ACM; 2008. p. 1433–7.

Wei C, Sprague A, Warner G, Skjellum A. Characterization of spam advertised website hosting strategy. In: Sixth conference on email and anti-spam, Mountain View, CA; 2009.

Xie Y, Yu F, Achan K, Panigrahy R, Hulten G, Osipkov I. Spamming botnets: signatures and characteristics. ACM SIGCOMM Comput Commun Rev 2008;38(4):171–82.

Zhuang L, Dunagan J, Simon DR, Wang HJ, Osipkov I, Tygar JD. Characterizing botnets from email spam records. LEET 2008;8:1–9.