Classification of Malicious Domain Names using Support Vector Machine and Bi-gram Method

Nhauo Davuth and Sung-Ryul Kim

Konkuk University, South Korea
davuthstory@gmail.com, kimsr@konkuk.ac.kr

Abstract

Everyday there are millions of domains registered and some of them are related to malicious activities. Recently, domain names have been used to operate malicious networks such as botnet and other types of malicious software (malware). Studies have revealed that it was challenging to keep track of malicious domains by Web content analysis or human observation because of the large number of domains. Legitimate domain names usually consist of English words or other meaningful sequences and can be easy to understand by humans, while malicious domains are generated randomly and do not include meaningful words or are not otherwise readable. Recently, a classification method has been proposed to classify malicious domain names. They used many features from DNS queries, including some textual features. However, it seems difficult to collect and maintain those data. Our contribution is that, by using only domain names we could achieve better classification results, thus showing that domain names themselves contain enough information for classification.

Keywords: *malware, botnet, support vector machine, Bi-gram*

1. Introduction

DNS is the protocol that resolves a domain name to the corresponding IP address. It is one of the most important elements of the internet infrastructure and is essential for such basic usage as browsing through a website or sending an email. Attackers have been using DNS to operate malicious networks, such as botnets and other types of malicious software. Bots [13] are new types of malicious codes that install into a compromised machine, which then can be controlled by the bot-master. Bot-master refers to a machine that interacts with the compromised machines in order to send commands via a Command and Control (C&C) channel. Botnet can be used for a variety of harmful purposes, including identity theft, spreading of spam, performing distributed denial of service attacks (DDoS) [19], and tricking the users into fake advertisement. Lately, botnets also have been deployed in a variety of peer-to-peer style or client-server style. In a wide field, the most investigated spam activities have been in the domain of webs and emails [14, 15]. Anyway, malicious activities often depend on DNS service as a first step to spread out their malware.

A framework for DNS based detection and mitigation of malware infection on a network was proposed by Etienne [3] to detect fast flux domains by using DNS queries. In their experiments, they extracted features from DNS query response such as A Record, NS record, Number IP ranges, Number ASNs, Time to Live (TTL) and some textual features. They applied Naive Bayesian statistics in their system. However, the result revealed that they could detect the malicious domain names with the degree of accuracy at about 87 %.

In this paper, we focus on classification of DNS based only on the domain names in order to identify the malicious domains, thus using less information to obtain a better result. By this, we can help to mitigate many internet threats that stem from botnets, phishing sites and malwares hosting services. In order to handle this work, we use the SVM algorithm. SVM [2] is a technique that is suitable for binary classification tasks and even in the case of non-regular data (data not regularly distributed) it can help to evaluate the information. SVMs can gain flexibility in the choice of the form the threshold has. SVMs operate locally, so we can be able to reflect in our frequency features of a single domain name or combinations of a small number of domain names. SVMs are powerful learning approach for solving two-class pattern recognition while Naive Bayesian classification faces a common problem which is referred to as data sparsely problem, particularly when the size of training data is very small. The theoretical conclusion is that SVMs are suitable for textual classification because SVM performs well with high dimensional feature spaces, few irrelevant features (dense concept vector), and sparse instance vectors.

The rest of this paper is organized as follows: background and related work in Section 2; description datasets and feature extracting in Section 3, while SVM and classification model discusses in Section 4, the result will be discussed in Section 5, and conclusion description in Section 6.

2. Background and Related Work

2.1. DNS Concept

Domain Name System (DNS) [17] translates internet domain and hostnames to IP address, uses a tree or hierarchical name structure. At the top of the tree is the root followed by the top-level domain (TLDs) then the domain name and number of lower levels, and each level separated by dot (.). DNS top-level domain splits into two types: generic top-level domain (gTLD) (such as .com or .org) and country code top-level domain (ccTLD) such as .us, .ca or .kr, etc...

2.2. DNS Queries

Clients use a DNS server to find information from them. The request may come directly from the clients, or from an application running on the client. The client sends a query message to the DNS server which contains a fully qualified domain name (FQDN), a query type, such as a particular resource record that client request, and the class for the domain name, which is usually the internet (IN) class. When a client looks up a name used in a program, it queries DNS servers to solve the name. Each query message that client sent contains three specific information request for the server to answer. That is a specific DNS domain name, a specified query type and a specific class for the DNS domain name [17].

2.3. Related Work

In general, botnet detection through DNS analysis follows two kinds of research field: the first is research on behavior of groups of machine in order to determine if they are infected. The second is research on the detection domain name that involved with malicious activities.

Etienne, *et al.*, [3] proposed a technique for the detection and mitigating botnet infection on a network. Their techniques aim to identify botnets behavior without

requiring the system administrator to maintain blacklists or update signature. They use multiple features from DNS queries such as A Record, NS record, Number IP ranges, Number ASN, Time to Live (TTL) and alphanumeric characters from domain names. In their experiment, they applied Naive Bayesian. However, the accuracy is still lower than 87% and the use of many features implies difficulty in collection and storage.

Based on URL extraction from spam mail Ma, *et al.*, studied a number of statistical method form machine learning for classification web site [16]. In their experiment, to obtain the information from the host names, they perform active probing to determine the number of IP addresses that involves with the domains. And they also obtained the IP addresses list; the limitation of this system is that the analysis includes only the domains that are included in spam mails. So the system can't detect other domain names such as command and control servers. Hao, *et al.*, [4] studied DNS lookup patterns measured from the .com top-level domain servers. Their preliminary analysis revealed that the resolution patterns for malicious domain names are sometimes different from those observed for legitimate domains. While they only report some initial measurement results and do not discuss how this method may be leveraged for detection purposes.

3. Data Sets and Feature Extraction

3.1. Data Collection

In our experiment, the raw data that we used is from malicious domains and legitimate domains. We divided them into training datasets and testing dataset.

3.1.1. Malicious dataset: The data we used for malicious example came from several sources. We obtained the list of domain names that were known to have been generated by recent Botnets or involved with malicious activities. Our primary source of blacklisting came from services such as Conficker [5, 6], Tropig [7], and Kraken [10, 11]. We also collected blacklist domain names from Zeus tracker [20]. It exhibits the most sophisticated domain generator by carefully matching the frequency of occurrence of vowel and consonants as well as concatenating the resulting word with common suffixes in the end. We observed that most of those domain names have been pronounceable yet are not in English language dictionary.

3.1.2. Legitimate dataset: The legitimate domain names were obtained from the Google Doubleclick ad planner top1000 most visited list [18], and we collected a list of pre-release domains compiled daily at different times by their partner registrars which is available at namejet.com [8].

3.1.3. Testing dataset: Testing datasets (unknown data) were obtained from domain names that analyzed and knew as malicious domains from www.malwaredomainlist.com [9]. Using testing data, which were already classified, are appropriate to compare the accuracy of our experiment and easily evaluate our approach.

Table 1. Amount of Data Used for Experiment

Legitimate domain	22,435 domain names
Malicious domain	1,533,964 domain names
Unknown domain	40,910 domain names

3.2. Constructing the Dataset

The quality of results produced by the SVM classifier strongly depends on the quality of the training set and the basic idea of this classification method is the training data which has labels corresponding to the domains being malicious or legitimate. In order to use the training data, we have to convert the list of domain names to a dataset. We included about 1.5 millions of malicious domain names and several thousand legitimate domain names. We had conservation that when constructing the list of the domain name, some third-level domains (3LD) that are extracted from URLs may be related to malicious activities, for example, a.x.com. In our experiment, we divided the dataset into two paths (training data and testing data). In a training dataset there are two classes, the first class for malicious labeled by +1 and the second class for legitimate labeled by -1. In the actual dataset, each line contains an instance that represents a training example or a testing example and follows the next format:

<label> <index$_1$>:<value$_1$> <index$_2$>:<value$_2$> <index$_3$>:<value$_3$>... <index$_n$>:<value$_n$>

Every instance can have many pairs of index/value, depending on a feature selected from raw data. <label> is an integer indicating the class label. In the training dataset, every instance must have a label. The labels on testing data are also used to calculate accuracy or error. <index> is an integer starting from 1. <value> is a real number. In our case, the value is the frequency numbers of feature that appear in one data. The pairs <index>:<value> is an integer string which separated by ":" and every each pair of index/value separated by a space. The order of pairs of index/value in an instance should be from the smallest to the biggest, according to the number of indexes.

3.3. Feature Extraction

We extract the features from domain names by using the bi-gram method, and reduce the amount of features by setting threshold conditions:

3.3.1. Bi-gram: A bigram [12] or di-gram is every sequence of two adjacent elements in a string of tokens, which are typically letters, syllables, or words; they are n-grams for *n=2*. The bi-gram's frequency distribution in a string is commonly used for simple statistical analysis of text in many applications, including in computational linguistics, cryptography, speech recognition, and so on. Below is an example of extracting bi-grams from a domain name: yourlogo.co.uk

yo ou ur rl lo og go o. .c co o. .u uk

<Label> <index$_1$>:<value$_1$> <index$_2$>:<value$_2$>…… <index$_n$>:<value$_n$>

<Label> Malicious we used label +1 and with Legitimate we used label -1

<Value> frequency of index for an instance (for example o. is appeared twice so the value is 2).

<Index> ((int)Char$_1$ * 256) + (int)Char$_2$

Example:

(int) o = 111
(int) . = 46
} o. = (111 * 256) + 46 = 28462

The result from extracting yourlogo.co.uk is:

-1 11875:1 11893:1 25455:1 26479:1 27759:1 28462:2 28519:1 28533:1 29292:1 30059:1 30066:1 31087:1

3.3.2. Threshold: In our experiment, we set some threshold values to remove useless features. We assume that f_1 is the frequency of each feature for legitimate domains and f_2 is the frequency of each feature for malicious domains, and we want to select the features that have frequency bigger than 100:

Absolute Value k:

If $f_1 < k$, Remove f_1

If $f_2 < k$, Remove f_2

Ratio Value r:

If $f_1 - f_2 < | r |$, Remove f_1 and f_2

4. SVM Classifier

4.1. SVM

Support Vector Machines (SVM) [2] is a supervised learning method for automatic pattern recognition. SVM learns from a training set to classifier which separates a set of positive examples (domain names) from a set of negative examples of introducing the maximum margin between the two sets. The training set can be described by l points in the n dimensional space. $x_i = R^n$ with two different labels $y_i \in \{-1, +1\}$ depending on the class which is assigned to the point x_i for all $i = 1,2,....l$. The points x_i, which lie on the hyperplane, satisfy $w = x_i + b = 0$, where $w \in R^n$ is the normal vector of the hyperplane. $||w||$ is the Euclidean norm of w. Mathematically, the points x_i can be expressed by two inequalities as follows:

$w.x_i + b \geq +1$ *for* $y_i = +1$ *(1)*

$w.x_i + b \leq -1$ *for* $y_i = -1$ *(2)*

We decided to define the "margin" of a separating hyperplane to be $d(+)$, $d(-)$. $Margin = 2/||w||$ so, in order to find largest margin we have to find the smallest $||w||$. In order to minimize $||w||$, we need to allocate Lagrangian multipliers α where $\alpha_i \geq 0$, $\forall_i$. We wish to find the minimum w and b, and the maximum α.

We then calculate $w = \sum_{i=1}^{l} \propto_i y_i x_i w$, which determines the set of support vectors by finding the indices such that $\alpha_i > 0$.

We then calculate $b = \frac{1}{N_i} \sum_{s \in S} (y_s - \sum_{m \in S} \propto_m y_m x_m . x_s)$. Testing each new point x' can be done by the following: $y' = sgn\ (w.\ x' + b)$.

The first step of SVM is the learning process. After dataset's features were extracted, we apply learning to those datasets to generate a model that will be used to classify in

the next step. To do this, we combined two classes of data with different labels (label -1 for legitimate and label +1 for malicious domain) into a dataset. The learning process will calculate to select points that could be a support vector. A point should be a support vector laying on line H_1 or H_2 with distance from both lines is maximized. After that we can find the middle line of H_1 or H_2 called Hyperplane. The second step is the classification process, in order to be able to use the decision rule for classification. SVM has learned the value of the score parameter w and b on a training sample. In this step, we input a new point of testing dataset with a model in order to calculate the value of it that should belong to class +1 or class -1.

4.2. SVM light

We applied SVM^{light} [1] in our experiments. The optimization algorithm used in SVM^{light} has scalable memory requirements and can handle the problem with many thousands of support vectors efficiently. SVM^{light} consists of a learning model and a classification model. Classification module can be used to execute the learned model to new examples. We used all default parameters in our experiment. In classification mode the target value denoted the class of data points.

5. Result and Discussion

The Accuracy, which refers to the proportion of data classified as accurate type in the total data. Accurate circumstances are True Positive (TP) and True Negative (TN), while false detected situations are False Positive (FP) and False Negative (FN). Accuracy of the system is calculated by the following equation:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} * 100\%$$

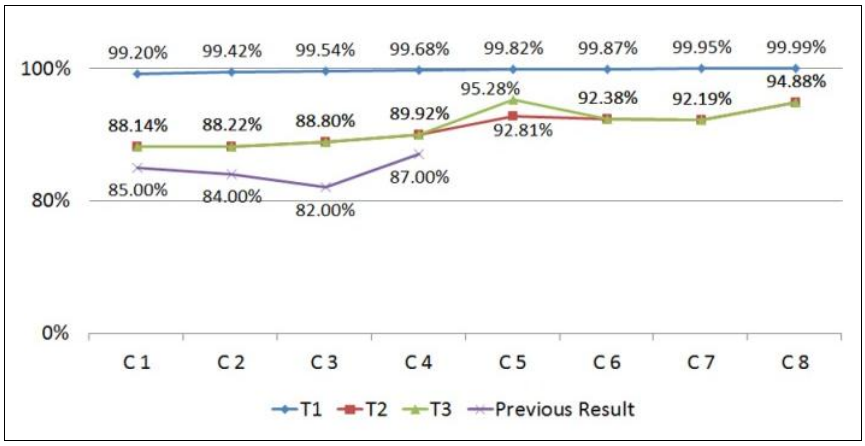

Figure 1. Experiment Result

T1: Tested use data, which generate a model

T2: Testing using unknown data

T3: Testing using unknown data that removed common features same as training set.

Figure 1 presents the performance comparison accuracy of three different test data with which we have combined one, two, three domain names, etc. in experiments C1, C2, C3, etc., respectively.

Through our experiment, the result indicated that it is possible to accurately classify domain names as whether algorithmically generated or not. It can be easily observed that SVMs trained using Bigram method and removed common feature had provided the best performance than a single character. Anyways, we were attempting to investigate the best choice among eight combining instances in training and testing datasets. Figure 1 presents clearly that the more domain names we combine into one instance the better is the accuracy of the experiments. However, even without any combination, we could achieve a better result than the previous work.

6. Conclusion

DNS plays an important role in the operation on the internet. Malicious services often depend on the DNS service as a first step to spread out their malicious codes. Our purpose is to identify malicious domain names, as soon as they appear and help to mitigate many internet threats. In this paper, we analyzed DNS based on domain names by using the SVM classifier. Through our experiment, the results showed that features extracted by bi-gram were performing a lot better than single alphanumeric character; however, the degree of accuracy is still limited. To solve this problem, we set threshold value to remove common features that have low frequency. And then we found that, it performed very well with the accuracy higher than 88.14%. And also we can get a better result, if we combined many domain names into an instance. Anyways, if we compare with other algorithms such as Naïve Bayesian or C5.0, we found that SVM gave an effective result and is suitable to classify the two-class examples with a high dimensional space.

Acknowledgements

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MEST) (No. 20120006492).

References

[1] T. Joachims," SVM light, Making large-Scale SVM Learning Practical", Advances in Kernel Methods - Support Vector Learning, B. Schölkopf and C. Burges and A. Smola (eds.), MIT-Press, **(1999).**
[2] V. Vapnik, "The Nature of Statistical Learning Theory", Support Vector Machine, Springer-Verlag, New York, **(1995)**.
[3] E. Stalmans, "A Framework for DNS Based Detection and Mitigation of Malware Infections on a Network", Information Security South Africa Conference, **(2011)**.
[4] S. Hao, N. Feamster and R. Pandrangi, "An Internet Wide View into DNS Lookup Patterns", VeriSign Incorporated, **(2010)**.
[5] P. Porras, H. Saidi and V. Yegneswaran, "An Analysis of Conficker's Logic and Rendezvous Points", SRI International Technical report, **(2009)**.
[6] P. Porras, H. Saidi and V. Yegneswaran, "Conficker C Analysis", SRI International Technical report, **(2009)**.
[7] B. Stone-Gross, "Your Botnet is my Botnet: Analysis of a Botnet Takeover", ACM Conference on Computer and Communications Security (CCS), **(2009).**
[8] Pre-release domain name, **(2012)**, http://www.namejet.com/Pages /Downloads.aspx.
[9] Malware domain, http://www.malwaredomainlist.com/.
[10] On kraken and bobax botnets, http://www.damballa.com/downloads/r_pubs/Kraken_Response.pdf.
[11] Pc tools experts crack new kraken, http://www.pctools.com/ news/view/id/202/.
[12] Bigram, http://en.wikipedia.org/wiki/Bigram.

[13] M. T. Banday, J. A. Qadri, N. A. Shah, "Study of Botnets and Their Threats to Internet Security", **(2009).**
[14] A. Mukherjee, B. Liu and N. Glance, "Spotting Fake Reviewer Groups in Consumer Reviews", International World Wide Web Conference Committee, **(2012)** April 16–20, Lyon, France.
[15] SOPHOS, Security Threat Report, **(2012)**, http://www.sophos.com.
[16] J. Ma, "Beyond blacklist: learning to detect malicious website from suspicious URLs", SIGKDD conference, **(2009)** Paris, France.
[17] Incognito Software Inc., "Understanding of DNS (the Domain Name System)", **(2007)**, www.incognito.com/resources/nc/wp_understanding_dns.pdf.
[18] Top 1000 sites doubleclick ad planner Google, **(2011)**, http://www.google.com/adplanner/static/top1000/.
[19]A. Buscher and T. Holz, "Tracking DDoS Attacks: Insights into the Business of Disrupting the Web", USENIX Association Berkeley, CA, USA, **(2012)**.
[20] Z. Tracker, "Zeus IP & domain name block list", https://zeustracker.abuse.ch/.

Authors

Nhauo Davuth

Nhauo Davuth is a master degree student of Advanced Technology Fusion Department, Konkuk University, Seoul, Korea. He graduated bachelor degree in computer science and engineering at Royal University of Phnom Penh, Cambodia. His recent research interests are data mining and security in general.

Sung-Ryul Kim

Sung-Ryul Kim is a professor of the division of Internet and Multimedia Engineering at Konkuk University, Seoul, Korea. He received his Ph.D. degree in computer engineering at Seoul National University, Korea. His recent research interests are in cryptographic algorithms, distributed algorithms, security in general, cloud computing, and data mining. ** Corresponding author.