MALICIOUS URL DETECTION USING MULTI-LAYER FILTERING MODEL

RAJESH KUMAR, XIAOSONG ZHANG, HUSSAIN AHMAD TARIQ, RIAZ ULLAH KHAN

School of Computer Science & Engineering, University of Electronic Science and Technology of China
E-MAIL: rajakumarlohano@gmail.com, rerukhan@outlook.com

Abstract:

Malicious URLs are harmful to every aspect of computer users. Detecting of the malicious URL is very important. Currently, detection of malicious webpages techniques includes black-list and white-list methodology and machine learning classification algorithms are used. However, the black-list and white-list technology is useless if a particular URL is not in list. In this paper, we propose a multi-layer model for detecting malicious URL. The filter can directly determine the URL by training the threshold of each layer filter when it reaches the threshold. Otherwise, the filter leaves the URL to next layer. We also used an example to verify that the model can improve the accuracy of URL detection.

Keywords:

Malicious URL; Black-list and White-list Technology; Machine Learning; Multi-layer Filtering Model

1. Introduction

In recent years, the Internet has been playing a bigger and bigger role in people's work and life. Currently, we observed that not every website is user-friendly and profitable. More and more malicious websites began to appear and these malicious websites endangering all aspects of the user. This can lead the user towards economic losses, even some can create confusion over the management of the country. Detecting and stopping malicious websites has become an important control measure to avoid the risk of information security [7-8] [10]. Generally, the only entrance to the website is URL, which can be malicious URL. At this level of entrance, the identification of URL is the best solution to avoid information loss. So the malicious URL identification has always been a hot area of information security.

Spams, malicious webpages and URLs that redirect or mislead the legitimate users to malware, scams, or adult content. It is perhaps correlated with the use of the internet. To identify malicious URLs, ML-based classifiers draw features from webpages content (lexical, visual, etc.), URL lexical features, redirect paths, host-based features, or some combinations of them. Such classifiers usually act in conjunction with knowledge bases which are usually in-browser URL BLs or from web service providers. If the classifier is fed with URL-based features, it is common to set a URL aggregator as a per-processor before extracting features. Mostly using supervised learning paradigm, Naive Bayes [1][13], Support Vector Machine with different kernels [9], and Logistic Regression are popular Machine Learning classifiers for filtering spam and phishing [6]. Meanwhile, GT-based learning to deal with active attackers is also evaluated in spam filtering. In this paper we solve this problem using multi-layer filtering model to do each classification which are able to handle their own relatively good data.

2. Multi-Layer Filter Model

Malicious URL detection [5][11-12] is a typical classification application scenario. The URL may be malicious URL or normal URL. The first part of several machine learning classification algorithms are very useful. A wide range of applications have also been applied to malicious URL detection scenarios. In order to give a brief overview to the advantages of each classifier, this article designed a malicious URL Multi- layer detection model. This model is mainly composed of 4- layer classifier, where these classifiers are also called filters in this model, so the model consists of 4 Layer filter composition.

2.1. Stratified filter

1) Black and white list filter: The model's first-level filter is a black-and-white list filter which will be validated by recognizing the normal URLs and Malicious URLs. The normal URL addresses are stored in the white list file, while the malicious URL addresses are stored in the blacklist file. To detect the URL, we traverse the list of black and white to determine whether the URL in the black list or in the white list.

2) Naive Bayesian filter: The second layer filter in this model is a naive Bayesian filter that trains the model by

dividing into two main steps:

By training the URL samples, we use two-dimensional arrays C1 and C2 to store the probability of each value of malicious website and normal website, such as formula (1) below:

$$c_i = \begin{pmatrix} P_{11}, & P_{12}, & ...P_{1m} \\ \vdots & \vdots & \vdots \\ P_{n1} & P_{n2}, & ...P_{nm} \end{pmatrix} \quad (1)$$

2.2. Alpha N-Bayes threshold training

Let $\alpha_{nbayes} = \max(p1/p2, p2/p1)$ (where P1 and P2 represent respectively what is calculated by the naive Bayesian formula model as malicious URL and normal URL probability value), so the size of nbayes can represent this classification judgment of the credibility. The nbayes describes that the URL belongs to one of the categories. The probability is much greater than the probability of belonging to another class. So, when nbayes arrives at certain threshold size, we can consider that URL is Naive Bayesian good data. If this threshold is set to too small, it may not be reached to the ideal classification accuracy. If this threshold is set to too large, naive shellfish the data that yeast good filters will be very small. This article discusses another group training data to train this threshold. The specific method is discussed briefly in Section 3

Assume that the appropriate training threshold is α_{nbayes}. For the upper filter down to detect URL, put it into the trained naive Bayesian model, and calculate nbayes if $\alpha_{nbayes} > \alpha_{nbayes}$, then we think the URL is a good Bayesian model of good data, otherwise, the URL cannot be considered as a good data for a naive Bayesian model, i.e. it cannot determine the nature of the URL, so record the classification results, and move to the next filter.

2.3. CART decision tree filter

The model's third-level filter is CART Decision Tree Filter.

The model is further split into two steps:

1) model training: The CART tree is constructed by training samples with URLs, and the leaf nodes are decision nodes, and store the CART tree in the file system.

2) cart threshold training: Let $\alpha_{nbayes} = \max(n/m, m/n)$ (n represents the number of malicious URL leaf nodes, m represents the number of normal URL leaf nodes), so the size of a cart can characterize the type of occupation that a leaf node decides. Similar to naive Bayesian filter, this threshold can neither be set too small nor too large. So, to train the threshold, through the same training data group, the specific method is discussed in Section 3.

If $\alpha_{nbayes} > \alpha_{nbayes}$, then we think that URL is CART Decision tree model has a good at data, otherwise, record the classification results, and the URL is filtered to the next filter.

2.4. SVM filter

The final filter of this model is SVM filter. SVM training model is mainly classified models. It Derived classification function for the upper Layer filter down the URL, record classification results, combined with Naive Bayesian Filter and CART Decision Tree filtering collectively determine the classification of the URL.

3. Instance Validation

3.1. Data sources and experimental environment

The malicious URL dataset in this experiment is downloaded from the malicious website lab (Http://www.mwsl.org.cn/), the normal URL dataset is collected from first category directory (http://www.dir001.com/). 10000 samples are taken from each dataset i.e. 10000 from malicious URLs and 10000 from normal URLs. This article uses features extraction and data modeling. Python language is used as the implementation programming. Windows 10 64bit as an operating system and Core i5 with 16GB RAM was used as personal computer.

Table 1 feature vector extraction rules

No	Features
F1	The domain names contained more than 4 consecutive numbers
F2	The domain name contains special characters (#, $, @, ~, _, -)
F3	Top Five domain name (com, en, net, org, cc)
F4	The number of "." in domain name
F5	domain name total length
F6	The Length of longest domain name segment
F7	Meaningful coefficients in primary domain names

3.2. Feature Selection

Garera[2] and Gattani[3] make comparisons of comprehensive study regarding URL feature selection. This article mainly from the perspective of domain name cost, malicious website The creator knows that the domain name of his site has a great risk of being banned, so in the purchase

When buying a domain name, you often buy cheaper or even free domains for cost savings Name, and this domain often has the following characteristics: 1) TLD is not the mainstream Domain name; 2) domain name with special characters; 3) domain name length is very long; 4) The main domain consists of meaningless letters or numbers; 5) There are many "." To confuse the domain name structure.

Based on the above information, this paper chooses seven features, as shown in Table 1.

The calculation method of F7 in Table 1 is as follows:

This text chooses the commonly used 5492 English words and 187,207 Chinese word phonetic together constitute a meaningful word Tree, the string of the URL's main domain name is a meaningful list of the tree of words to match, if the match can be considered match these characters are there Meaning, and then use the rules of the characters and the total length of the characters that have ratio Meaning coefficient. For example, bookdsxihuan.com, because the domain name does not contain 4.

More than one consecutive number, so F1 is set to 0; due to the domain name does not contain Any of the following special words: #, $, @, ~, -, _, so F2 is set to 0; This example contains one of the top five domain names "com", so F3 is set to 1; The domain name contains a "." Therefore F4 is set to 1; F5 is the total length of the domain name, This example is 16; in this case, the longest domain name is "bookdsxihuan" and its length is 12, so F6 is set to 12; the main domain name of this example is "bookdsxihuan" with two A meaningful word "book", "xihuan" match, meaningful length of 10, The primary domain has a length of 12, then it has a meaningful coefficient of 10/12 = 0.83. From this, the feature vector of this domain name is expressed as {0,0,1,1,16,12,0.83}.

Table 2 url data set

Dataset Name	Dataset Description
Training model dataset	8000 malicious URL, 8000 normal URL
Training threshold dataset	1000 malicious URL, 1000 normal URL
Testing dataset	1000 malicious URL, 1000 normal URL

Table 3 test results for various models

Model Name	Accuracy Rate %	Rexall Rate %	Precise rate %
Multi-layer filtering model	79.55	68.80	87.64
Simple Naive Bayes	77.30	66.40	84.91
Single Decision Tree	79.35	69.00	87.01
Single SVM	76.80	79.40	75.48

3.3. Training model and threshold

In this paper, we divide the 20,000 URLs collected in section 3.1 into three in the experiment Part, as shown in Table 2.

If the URL which is to be tested can directly be judged as malicious or normal, so the experimental tuning process does not consider black and white list filter. The blacklist filter can be considered authoritative in this model. The experimental process is as follows:

Perform the data on the three datasets using the feature extraction rules in Section 3.2

Train a separate machine learning with the train-model-set sample set Classifiers, including the naive Bayesian model, the CART decision tree model and SVM model

Build a separate machine learning model into a multilayer filter model, and at the very beginning give a very large threshold pair, so $\alpha_{nbayes}^{new} = 500, \alpha_{cart}^{new} = 10$ Substitute the train-threshold-set sample set into the multi-layer filter model and calculate the check measurement accuracy, and gradually reduce these two thresholds, this article uses the formula in the program $\alpha_{nbayes}^{new} = 1 + 0.8\alpha_{nbayes}^{old}, \alpha_{cart}^{new} = 1 + \alpha_{cart}^{old}$, record each thresholds accuracy, and finally pick the combination of the highest accuracy of the function $(\alpha_{nbayes}^{*}, \alpha_{cart}^{*})$ Substituting the threshold combination $(\alpha_{nbayes}^{*}, \alpha_{cart}^{*})$ into the multi-layer filter model, test this multi-layer filter model with three separate test set sample, sets classifiers and record the accuracy rate, recall rate and accuracy.

4. Experiment Results

The optimal threshold pair trained after Section 3.3 is $(\alpha_{nbayes}^{*} = 400.2, \alpha_{cart}^{*} = 1.203)$, the results are shown in Table 3.

In the multi-layer filter model, Naive Bayesian filter determined 308 URL, decision tree filter determined the 1370 URL, another 322 The URL is jointly detected by three filters. This result is also in line with Table 3. The performance of the separate classification models can be seen in three separate models. Decision tree model is comparatively best performing model than Naive Bayes and SVM. It is also observed in Table 3 that multi-layer filter model performs better than all the three classifier models. Multi-layer filter model can let the classifier to deal with their own good data. Every layer of the model plays a beneficial role for classification of the URLs and ultimately improves the detection of malicious URL in terms of accuracy.

5. Conclusion

In this paper, black and white list technology and machine learning algorithms were used and formed multi-layer filtering model for detection of malicious URLs. The model was trained for each machine learning algorithm i.e. naive Bayesian classification and decision tree classifier threshold and this threshold is used to refer to guide two classifiers for filtering URL. We combined the Naive Bayesian classifier, Decision Tree classifier and SVM classifiers in one multi-layer model to improve the malicious URL detection system in terms of accuracy. We observed from the real examples that multi-layer filtering models does effective detection of malicious URLs.

References

[1] Hugh A. Chipman, Edward I. George, and Robert E. McCulloch. "Bayesian CART Model Search." Journal of the American Statistical Association, Vol. 93(443), pp 935–948, September 1998.

[2] Sujata Garera, Niels Provos, Monica Chew, and Aviel D. Rubin. "A framework for detection and measurement of phishing attacks." In Proceedings of the 2007 ACM workshop on Recurring malicious code - WORM '07, page 1, 2007.

[3] Abhishek Gattani, AnHai Doan, Digvijay S. Lamba, Nikesh Garera, Mitul Tiwari, Xiaoyong Chai, Sanjib Das, Sri Subramaniam, Anand Rajaraman, and Venky Harinarayan. "Entity extraction, linking, classifica- tion, and tagging for social media." Proceedings of the VLDB Endowment, Vol. 6(11), pp 1126–1137, August 2013.

[4] David D. Lewis. Naive (Bayes) at forty: The independence assumption in information retrieval. pages 4–15. 1998.

[5] Justin Ma, Lawrence K. Saul, Stefan Savage, and Geoffrey M. Voelker. "Learning to detect malicious URLs." ACM Transactions on Intelligent Systems and Technology, Vol. 2(3), pp 1–24, April 2011.

[6] Fadi Thabtah Maher Aburrous, M.A.Hossain, Keshav Dahal. "Intelligent phishing detection system for e-banking using fuzzy data mining." Expert Systems with Applications, Vol. 37(12), pp 7913–7921, Dec 2010.

[7] Ankush Meshram and Christian Haas. "Anomaly Detection in Industrial. Networks using Machine Learning: A Roadmap." In Machine Learning for Cyber Physical Systems, pages 65–72. Springer Berlin Heidelberg, Berlin, Heidelberg, 2017.

[8] Xuequn Wang Nik Thompson,Tanya Jane McGill. "Security begins at home: Determinants of home computer and mobile device security behavior." Computers & Security, Vol. 70, pp 376–391, Sep 2017.

[9] Dan Steinberg and Phillip Colla. "CART: Classification and Regression Trees." The Top Ten Algorithms in Data Mining, pp 179–201, 2009.

[10] D. Teal. "Information security techniques including detection, interdiction and/or mitigation of memory injection attacks," Google patents. Oct 2013.

[11] Kurt Thomas, Chris Grier, Justin Ma, Vern Paxson, and Dawn Song. "Design and Evaluation of a Real-Time URL Spam Filtering Service." In 2011 IEEE Symposium on Security and Privacy, pp 447–462. May 2011.

[12] Sean Whalen, Nathaniel Boggs, and Salvatore J. Stolfo. "Model Aggregation for Distributed Content Anomaly Detection." In Proceedings of the 2014 Workshop on Artificial Intelligent and Security Workshop - AISec '14, pp 61–71, New York, USA, 2014. ACM Press.

[13] Ying Yang and Geoffrey I. Webb. "Discretization for Naive-Bayes learning: managing a discretization bias and variance." Machine Learning, Vol. 74(1), pp 39–74, Jan 2009.