Rule-Based Phishing Attack Detection

Ram B. Basnet [a,b,*], Andrew H. Sung [a,b], Quingzhong Liu [c]

[a] Computer Science & Engineering Department, New Mexico Tech, Socorro, NM 87801, USA

[b] Institute for Complex Additive Systems Analysis (ICASA), New Mexico Tech, Socorro, NM 87801, USA

[c] Department of Computer Science, Sam Houston State University, Huntsville, TX 77341, USA

[*]Corresponding Author, {rbasnet, sung}@cs.nmt.edu, qxl005@shsu.edu

***Abstract*— The World Wide Web has become the hotbed of a multi-billion dollar underground economy among cyber criminals whose victims range from individual Internet users to large corporations and even government organizations. As phishing attacks are increasingly being used by criminals to facilitate their cyber schemes, it is important to develop effective phishing detection tools. In this paper, we propose a rule-based method to detect phishing webpages. We first study a number of phishing websites to examine various tactics employed by phishers and generate a rule set based on observations. We then use Decision Tree and Logistic Regression learning algorithms to apply the rules and achieve 95-99% accuracy, with a false positive rate of 0.5-1.5% and modest false negatives. Thus, it is demonstrated that our rule-based method for phishing detection achieves performance comparable to learning machine based methods, with the great advantage of understandable rules derived from experience.**

Keywords- Phishing attack, phishing website, rule-based, machine learning, phishing detection, decision tree

I. INTRODUCTION

Phishing is a criminal mechanism employing both social engineering and technical subterfuge to steal consumers' personal identity data and financial account credentials, according to AntiPhishing Working Group (APWG) [1].

Phishing emails usually act on behalf of a trusted third-party to trick email receivers into performing some actions such as giving away personal information, e.g. bank accounts, social security numbers, usernames and passwords to online banking and popular social networking websites like Facebook, Twitter, etc. Though much research on anti-phishing techniques has been done and new techniques and methodologies are being proposed regularly, online scammers manage to come up with innovative schemes to circumvent existing detection technologies and lure potential victims to their phishing campaigns.

Once the phishing email receivers are lured into a fraudulent website, even the experienced, security-minded users are often easily fooled to fulfill the website's primary goal. Data indicates that some phishing attacks have convinced up to 5% of their recipients to provide sensitive information to spoofed websites [9].

Kroll survey [2] finds that phishing is the top information theft threat to North American companies. The survey also found that the top techniques used for information theft against U.S. companies where phishing.

While payment systems and financial sectors continued to lead the most targeted phishing brands, classifieds emerged as a major non-traditional phishing vector accounting for 6.6% of phishing attacks detected in Q2 2010, growing 142% from Q1, according to APWG quarterly report [1]. Government sector accounted for 1.3% of the phishing attacks in Q2 2010. United States continued its position as the top country for hosting phishing website during the same quarter.

As a result, the design and implementation of effective phishing detection techniques to combat cyber crime and to ensure cyber security, therefore, is an important and timely issue that–as long as the cyber criminals are proceeding unabated in scamming Internet users–requires sustained efforts from the research community.

In this paper, we propose a rule-based approach to detecting phishing webpages and present our preliminary experimental results on temporal data sets using Decision Tree and Logistic Regression learning algorithms.

II. RELATED WORK

There is an extensive recent literature on automating the detection of phishing attack, most importantly, detection of phishing emails, phishing URLs, and phishing webpages. Phishing attack detection techniques based on the machine learning methodology has proved highly effective, due to the large phishing data set available and the advances in feature mining and learning algorithms; see e.g., [3], [4], [5], [6], [7], [10], [24], [25], [26].

Anomaly detection has been used to detect phishing webpage [32] where a number of anomaly based features are extracted from webpage and SVMs is applied on a data set with 279 phishing and 100 legitimate webpages producing 84% classification accuracy.

Besides machine learning (ML) based techniques, there exists a plethora of other approaches in phishing detection. Perhaps, the most widely used anti-phishing technology is the URL blacklist technique that most modern browsers come equipped with [14], [30]. Other popular methods are browser-based plug-ins or add-in toolbars. SpoofGuard [16] is one such tool that uses domain name, URL, link, and images to evaluate the spoof probability on a webpage. The plug-in applies a series of tests, each resulting in a number in the range (0, 1). The total score is a weighted average of the individual test results. Other similar anti-phishing tools include SpoofStick [19], SiteAdvisor [17], Netcraft anti-phishing toolbar [23], AVG Security Toolbar [22], etc.

Visual similarity based methods have been explored for detecting phishing pages. Weynin et al. [21] compare legitimate and spoofed webpages and define visual similarity metrics. The spoofed webpage is detected as a phishing attack if the visual similarity is higher than its corresponding preset threshold. Medvet et al. [11] consider three key page features, text pieces and their style, images embedded in the page, and the overall visual appearance of the page as rendered by the browser.

III. Rule-Based Approach

In this section, we discuss motivation of our approach and the underlying techniques we propose to achieve our goal.

A. Motivation

Though different in goal, our approach is particularly inspired by the approach introduced by the open source intrusion detection and prevention system (IDS/IPS), Snort [31]. Snort monitors networks by matching each packet it observes against a set of rules. As the phishing attacks have been growing rapidly by the day, we feel that there is a need for Snort like phishing attack detection technology at the application level. In this paper, we try to investigate such an approach.

B. Our Approach

Just like a network IDS signature, a rule is a pattern that we want to look for in a webpage. The idea behind the rule-based approach is to make the process of phishing attack detection as intuitive, simple, and user-friendly as possible. One of the main goals of our approach is to make the framework flexible and simple to extend the rule set by incorporating new and emerging phishing tactics as they are encountered. We generate our rule set primarily relying on our observations and the machine learning features proposed in various existing literatures [3], [4], [5], [6], [10] on phishing attack detection. We gather various techniques and tricks used by phishers to lure their potential victims to a forged website and use those heuristics to develop our initial rule set. In this section, we briefly describe various rules that we employ in detecting whether a given webpage is phishing.

A rule is usually written in the following form:

IF *conditions* THEN *actions*

If the *conditions*, also known as *patterns*, are satisfied then the *actions* of that particular rule are fired.

A rule may range from very simple – checking a particular value in the URL – to highly complex and time-consuming that may require to analyze meta-data, query search engines and blacklists and combine several conditions with *AND* and *OR* operators. Depending on their characteristics and the methods used to extract the rules, we broadly group them into the following categories.

C. Search Engine-based Rules

The idea behind using results from top search engines is to leverage their power and effectiveness in continuously crawling and indexing a large number of webpages. In [3], we show that search-engine based features are very effective in determining phishing URLs and essentially demonstrate that search engines' large and continuously growing indexes act as a rudimentary white-list. We develop two rules using search engines.

Rule 1: IF a webpage's URL is not present in all search engines' indexes, THEN the webpage is potentially phishing.

Rule 2: IF a webpage's domain is not present in all search engines' indexes, THEN the webpage is potentially phishing.

To generate Rule 1, we check if a URL exists in the search engines' (Google, Yahoo!, and Bing) indexes. Our rule generator automatically queries the search engines and retrieves top 30 results. If the results do not contain the URL, this rule considers the webpage as potentially a phishing attack. We observed that all three search engines returned the URL as the first result if they have indexed the URL. Intuitively, it makes sense because we search the URL itself not ranked relevant URLs based on keywords. But, to be on the safe side, we use top 30 results as it has been shown that going beyond the top 30 results had little effect [6].

Similarly, Rule 2 is generated by querying the search engines with the domain of a URL. If the top 30 results do not contain the domain, this rule says that the given webpage is potentially phishing.

D. Red Flagged Keyword-based Rule

By examining 80% of randomly selected URLs on DS1 data set, we found that certain groups of words seem to be more popular among phishers, perhaps, to lure unsuspecting users to the forged webpage. Using substring extraction algorithm, we generated a list of 62 word stems that frequently occur in our training data set. We iterate through this keyword list and check if any of the word is found in the URL. Thus, we generate our next rule:

Rule 3: IF a keyword is present in the URL, THEN the webpage is likely phishing.

E. Obfuscation-based Rules

Phishers often obfuscate URLs to trick users into thinking that the malicious URL belongs to a legitimate website users are familiar with. Obfuscating URLs with certain characters such as "-", soft hyphen, Unicode, and visually similar looking characters are very common techniques employed by phishers. We try to identify these tactics and generate rules from them. For example, we check if certain characters such as "-", "_", "=", "@", digits, and non-standard port etc. are present in a webpage's URL.

These tactics used by phishers lead us to our next set of rules.

Rule 4: IF a webpage's URL is IP based (hex-based, octal, or decimal-based), THEN the webpage is potentially a phishing attack.

Rule 5: IF a URL contains any of the following characters [-, _, 0-9, @, ",", ;] OR contains a non-standard port, THEN the webpage is potentially phishing.

Rule 6: IF host part of a URL has 5 or more dots OR length of the URL is longer than 75 characters OR length of the host is longer than 30 characters, THEN the webpage is potentially a phishing attack.

F. Blacklist-based Rule

We employ Google Safe Browsing API [14] to check URLs against Google's constantly updated blacklists of suspected phishing and malware pages and generate our next rule.

Rule 7: IF a URL is in Blacklist(s), THEN it is potentially a phishing webpage.

G. Reputation-based Rule

We generate our next set of rules from historical stats on top IPs and domains that have a bad reputation of hosting the most phishing webpages. We use 3 types of statistics: Top 10 Domains, Top 10 IPs, and Top 10 Popular Targets published by PhishTank [33]. We also use top 50 IP address stat produced by StopBadware.org [13].

Rule 8: IF a URL contains a top phishing target OR its IP or domain is in the statistical reports produced by PhishTank, Stopbadware, etc., THEN the webpage is potentially a phishing attack.

H. Content-based Rules

The rules in this category are rooted in the HTML contents of the phishing webpages. An ingenious phishing webpage resembles the look and feel of the target legitimate website. Nevertheless, the same tactics employed by phishers also give us opportunities to discover our content-based rules. By observing HTML structures of hundreds of phishing webpages, we've generated the following rules:

Rule 9: IF a webpage contains *password* input field AND (the corresponding form content is sent in plain text without using Transport Layer Security (TLS)/Secure Sockets Layer (SSL) OR the form content is sent by using 'get' method), THEN the webpage is potentially phishing.

Rule 10: IF a webpage contains *password* input field AND the corresponding form content is sent to external domain regardless of TLS/SSL, THEN the webpage is potentially phishing.

Rule 11: IF a webpage contains META tag AND the refresh property's destination URL is in external domain OR it belongs to a blacklist, THEN the webpage is potentially phishing.

Rule 12: IF a webpage is redirected by its server AND the page contains *password* field, THEN the webpage is potentially phishing.

Rule 13: IF a webpage has IFrame tag AND its source URL belongs to a blacklist, THEN the webpage is potentially a phishing attack.

Rule 14: IF a webpage contains *password* input field AND the webpage has more external than internal links, THEN the webpage is potentially phishing.

Rule 15: IF a webpage has bad HTML markups AND contains *password* input field, THEN the webpage is potentially a phishing attack.

Figure 1 shows the histograms of Rules 1-15 obtained on data set DS1. The histogram (see Figure 1) confirms that Rule 1 and Rule 2 have high prominence in phishing webpages and are very strong indicators of whether a webpage is phishing. These rules by themselves can detect more than 97% of phishing webpages, while correctly classifying 100% of legitimate webpages. Rule 3 has high prominence in phishing webpages as well compared to non-phishing webpages.

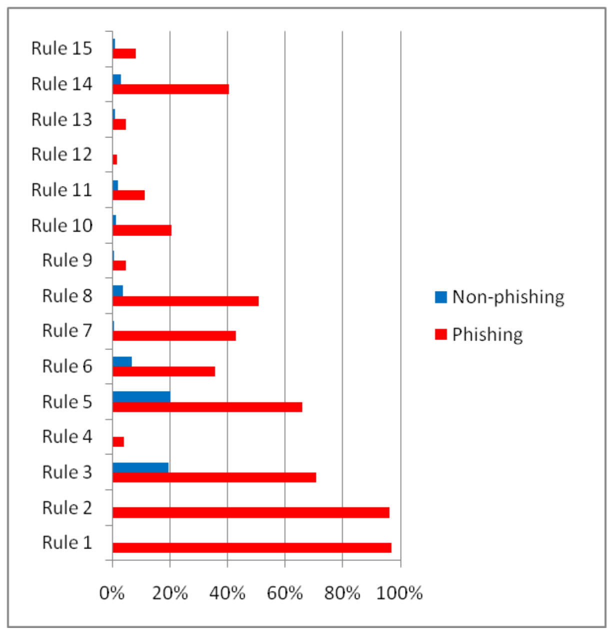

Figure 1: Histogram of Rules 1-15 on DS1 data set

Some phishing webpages (~ 4%) satisfy Rule 4, while not a single non-phishing webpage satisfies it. Though very sparsely present, this rule can be a good indicator of whether a webpage is phishing. Roughly 66% of phishing webpages and surprisingly 20% of non-phishing webpages satisfy Rule 5.

43% of phishing webpages satisfy Rule 7, while 0% of non-phishing webpages satisfy the same. This suggests that Rule 7 is a strong indicator of whether a page is phishing. Rule 8 is present in about 51% of phishing webpages and in roughly 4% of non-phishing webpages. Rule 9 has relatively small presence among phishing webpages but no presence on non-phishing webpages, indicating that this rule is not universally applicable, but still a strong indicator of phishing webpage.

About 21% of phishing and 1% of non-phishing webpages satisfy Rule 10. Relying on this rule alone would miss a large percentage of phishing webpages while it would also misclassify some legitimate webpages as phishing. Rule 12 is satisfied by a very small number of phishing webpages (~1%). However, no single non-phishing webpage satisfy the same suggesting that this rule may not aid in false positives.

Relatively more phishing webpages satisfy Rule 13, 14, and 15 compared to non-phishing webpages.

We point out that these are not the exhaustive list of rules. One of the major advantages of rule-based approach is to be able to quickly tune the rules to ones' needs and easily modify or add rules as and when needed to detect new and ever changing phishing attacks.

IV. Experimental Evaluation

In this section, we briefly describe the data sets we use and present the results of experimental validation of our approach on these data sets. The experiments were carried out on a machine with Core 2 Duo 2 GHz Intel processors and 3 GB RAM.

A. *Data Sets*

For phishing webpages, we wrote Python scripts to automatically download confirmed phishing URLs from PhishTank [8]. PhishTank, operated by OpenDNS, is a collaborative clearing house for data and information about phishing on the Internet. A potential phishing URL once submitted is verified by a number of registered users to confirm it as phishing. We collected first set of phishing URLs from June 1 to October 31, 2010. Phishing tactics used by scammers evolve over time. In order to investigate these evolving tactics and to closely mimic the real-world *in the wild* scenario, we collected second batch of confirmed phishing URLs that were submitted for verification from January 1 to May 3, 2011.

We collected our legitimate webpages from two public data sources. One is the Yahoo! directory[1], the web links in which are randomly provided by Yahoo's server redirection service [34]. We used this service to randomly select a URL and download its page contents along with server header information. In order to cover wider URL structures and varieties and page contents, we also made a list of URLs of most commonly phished targets (using statistics from PhishTank [33]). We then downloaded those URLs, parsed the retrieved HTML pages, and harvested and crawled the hyperlinks therein to also use as benign webpages. We made the assumption, which we think is reasonable, to treat those webpages as benign, since their URLs were extracted from a legitimate sources. These webpages were crawled between September 15 and October 31 of 2010. The other source of legitimate webpages is the DMOZ Open Directory Project[2]. DMOZ is a directory whose entries are vetted manually by editors.

Based on the date on which phishing URLs were submitted to PhishTank for verification, we generated two data sets. The first data set, we refer to it as DS1, contains 11,341 phishing webpages submitted before October 31, 2010 and 14,450 legitimate webpages from Yahoo! and seed URLs. The second data set, we refer to it as DS2, contains 5,456 phishing webpages submitted for verification between January 1 and May 3 of 2011 and 9,636 randomly selected legitimate webpages from DMOZ. Table I summarizes these data sets.

TABLE I
SUMMARY OF DATA SETS

Data Set	Phishing	Non-phishing	Total Samples
DS1	11,341	14,450	25,791
DS2	5,456	9,636	15,092
DS1+DS2	16,797	24,086	40,883

We discarded the URLs that were no longer valid as the page couldn't be accessed to extract features from their contents.

B. *Counting Rules to Detect Phishing Webpages*

In order to detect a phishing webpage based on rules, one naïve yet simple approach is to give equal weight to each rule and count the number of rules satisfied by the page. Using a carefully chosen threshold, if the total number of rules satisfied by an instance is more than the threshold value, we can alert that the webpage is phishing. However, as the histogram shows (see Figure 2), choosing the best threshold value that would give the balanced and best false positive and negative rates is not a trivial task. Histogram in Figure 2 shows rule count from 0 up to 10 as only a very few phishing instances in the data set satisfied more than 10 rules.

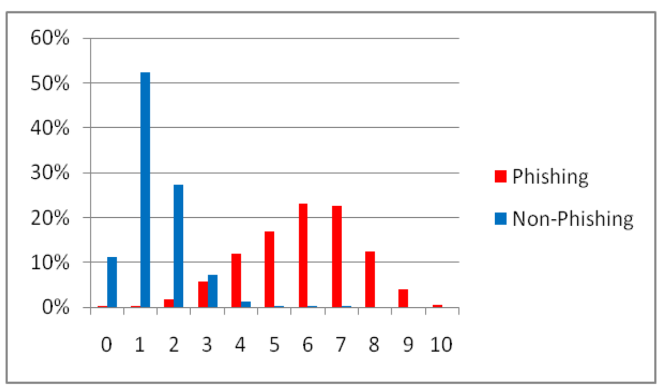

Figure 2: Histogram of rules count on the data set DS1. Horizontal axis is rule count and vertical axis is instance count

[1] http://dir.yahoo.com

[2] http://www.dmoz.org

Nonetheless, we experimented with a few thresholds and present the results, True Positive Rate (TPR), False Positive Rate (FPR), False Negative Rate (FNR), and True Negative Rate (TNR) in Table II.

TABLE II
PERFORMANCE MEASURES FOR VARIOUS THRESHOLD VALUES

Rule Count Threshold	TPR	TNR	FPR	FNR
2	99.55%	63.60%	36.40%	0.45%
3	97.64%	91.04%	8.96%	2.36%
4	**91.87%**	**98.20%**	**1.80%**	**8.13%**
5	79.96%	99.63%	0.37%	20.04%

As the equally weighted count-based approach resulted in unsatisfactory results, we resorted to ML-based approach to automatically prioritize each rule and generate decision rules. The next experiments detail this approach.

C. *Training with Rules*

In these experiments, we used the rules identified in Section III.B as binary features and applied them to train classifiers. We used Decision Tree (DT) and Logistic Regression (LR) algorithms implemented in the WEKA data mining library [15]. We employed 10-fold cross validation method to test the models and perform our analysis.

DT: DT is represented by tree structure where each internal node tests the corresponding attribute, each branch corresponds to attribute value, and each leaf node represents a classification decision [35].

Using a given input/output data set, DT can be learned by splitting the source set into subsets based on an attribute value test. This process is repeated on each derived subset in a recursive manner called recursive partitioning. The recursion is completed when the subset at a node all has the same value of the target variable, or when splitting no longer adds value to the predictions. Once the tree is trained, an unknown sample is classified by successive tests from the root of a DT down to a leaf.

For DT learning, we chose the C4.5 [20] algorithm which is implemented as J48 classifier in WEKA. On data set DS1, we obtained the pruned decision tree of size 19 with 10 leaf nodes (see Figure 3).

Rules 2, 4, 9, 10, 11, and 12 are removed from the model as a result of pruning.

Besides simple to understand and interpret, DT is robust and uses a white box model. DT model can be converted to rules using boolean logic as following: IF (Rule_1 <= 0) AND (Rule_7 <= 0) AND (Rule_14 <= 0) THEN phishing = No. Using this sequence of tests, 14,154 of non-phishing samples from DS1 data set are correctly classified, while 173 phishing webpages are misclassified. Similarly, IF (Rule_1 > 0) THEN phishing = Yes. Using this rule, 10,976 phishing webpages are correctly classified, while producing 0 false positive. The rest of the 8 rules can be genereted and interpreted in the similar manner.

It took 3.35 seconds to build C4.5 model. The detailed test results are shown in Table IV.

```
Rule_1 <= 0
|  Rule_7 <= 0
|  |  Rule_14 <= 0: -1 (14154.0/173.0)
|  |  Rule_14 > 0
|  |  |  Rule_8 <= 0
|  |  |  |  Rule_3 <= 0: -1 (384.0/27.0)
|  |  |  |  Rule_3 > 0
|  |  |  |  |  Rule_5 <= 0: -1 (88.0/20.0)
|  |  |  |  |  Rule_5 > 0
|  |  |  |  |  |  Rule_13 <= 0: +1 (33.0/7.0)
|  |  |  |  |  |  Rule_13 > 0
|  |  |  |  |  |  |  Rule_6 <= 0
|  |  |  |  |  |  |  |  Rule_15 <= 0: +1 (5.0/1.0)
|  |  |  |  |  |  |  |  Rule_15 > 0: -1 (8.0/3.0)
|  |  |  |  |  |  |  Rule_6 > 0: -1 (14.0)
|  |  |  Rule_8 > 0: +1 (45.0/6.0)
|  Rule_7 > 0: +1 (84.0/11.0)
Rule_1 > 0: +1 (10976.0)
```

Figure 3: DT model using C4.5 on data set DS1

LR: LR is a statistical model used for prediction of the probability of occurrence of an event by fitting data to a sigma function logistic curve [18]. Besides high classification accuracy, LR has the advantage of performing automatic feature ranking as well as providing an interpretable linear model of the training data. Because the output of a linear model depends on the weighted sum of the features, the sign and magnitude of the individual parameter vector coefficients can tell us how individual features contribute to a 'phishing' or a 'non-phishing' prediction. Positive coefficients correspond with phishing features while negative coefficients correspond with legitimate non-phishing features. A zero coefficient means that the corresponding feature will not contribute to the prediction outcome.

The coefficients and the odds ratio obtained from LR model on data set DS1 are displayed in Table III.

As indicated by the high odds ratio, Rule 4 is found to be the most useful in detecting whether a webpage is phishing. Similarly, Rules 1, 2, and 12 are strong indicators that a webpage is phishing attack. Interestingly, Rules 10, 13, and 15, on the other hand, seem to indicate that a webpage is non-phishing. LR took 2.33 seconds to build model and gave classification error rate of 1.02%, TPR of 97.88%, and FPR of 0.15%. The performance difference between C4.5 and LR is insignificant.

D. *Data Drift*

Phishing tactics and attack techniques keep changing as attackers come up with novel ways to circumvent the existing filters. Rules developed from observing a particular data set can yield a highly accurate classification results when trained and tested on disjoint sets of the same data source. But do these results hold when testing new phishing

TABLE III
FEATURES AND THEIR COEFFICIENTS USING LOGISTIC REGRESSION

Feature	Logistic Coefficient	Odds Ratio
Rule 1	80.4784	8.94E+34
Rule 2	63.5746	4.07E+27
Rule 3	1.4302	4.1796
Rule 4	138.2439	1.09E+60
Rule 5	1.5243	4.5917
Rule 6	0.3788	1.4606
Rule 7	5.008	149.6037
Rule 8	2.2699	9.6781
Rule 9	0.9032	2.4675
Rule 10	-0.155	0.8564
Rule 11	0.9984	2.714
Rule 12	58.7186	3.17E+25
Rule 13	-0.3982	0.6715
Rule 14	3.2524	25.852
Rule 15	-0.497	0.6084
Constant	-5.8523	

webpages using the same rule set extracted from old phishing webpages? To investigate this question, we tested new phishing data set DS2 against the model obtained by training the C4.5 classifier on old data set DS1. Table IV shows classification results using C4.5 classifier on various combinations of temporal data sets.

TABLE IV
OVERALL ERROR RATES ON TRAINING ON ONE DATA SET AND TESTING ON ANOTHER (POSSIBLY DIFFERENT OR TEMPORAL-BASED) DATA SET

Training	Testing	
	DS1	DS2
DS1	0.98%	4.86%
DS2	1.22%	4.51%
DS1+DS2	0.96%	4.18%

As expected, when trained and tested using the same data set DS1, the error yielded is the lowest due to low FPR (0.21%) and FNR (1.9%). When training and testing sources are completely mismatched, the error ranges from 1.2% to 4.8%. Surprisingly, the error received on training and testing using DS2 is comparatively higher (4.5%) with 1.3% FPR and 10.2% FNR. Although, error rate doesn't significantly decrease with the newer phishing data, the disparity in accuracy emphasizes that rules and training data should be selected judiciously. Thus, it is important to collect data that is representative and retrain the deployed classifier with new data often. Finding an optimal time interval to retrain the system for optimum performance represents an interesting direction for future study but is beyond the scope of this paper.

V. DISSCUSSION

In this section, we discuss some of the limitations of rule-based approach and some possible ways to address them.

A. *Tuning False Positives & Negatives*

Machine learning models allow us to tune the tradeoff between false positives and negatives. Figure 4 shows the results of this experiment as an ROC graph with respect to the decision threshold *t* over an instance of DS1 data set using C4.5 classifier.

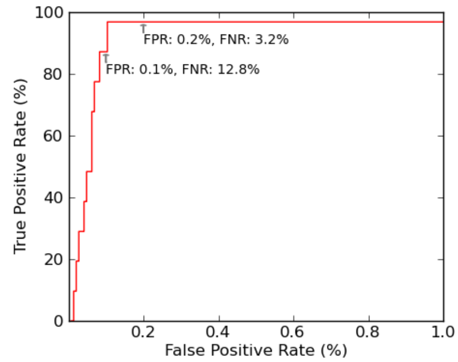

Figure 4: ROC showing tradeoff between false positives and false negatives using C4.5 on DS1 data set

Instead of using decision threshold *t* to minimize the overall error rate, Internet users may want to tune the threshold to have very low false positives at the expense of more false negatives or vice versa. By tuning false positives to conservatively low 0.1%, we can achieve false negatives of 12.8%. By tolerating slightly higher false positives of 0.2%, however, we can achieve significantly lower false negatives of 3.2%.

B. *Limitaions of Our Approach*

The rule set is premature and we emphasize its needs for expansion and thorough scrutiny. The system, if deployed, will likely produce some false alarms, while also missing a good number of phishing webpages. Attackers may thwart the system by minimizing the phishing tricks matching none or a small number of rules on their crafted phishing webpage. One such scenario is when attackers hack a legitimate webpage to host their phishing campaign. Such legitimate webpages are highly likely to appear on search engines' results. Phishers may design flash-based webpages virtually hiding all the HTML contents for analysis. However, expanding our rule set may address such potential attacks. We can add visual similarity-based rules, for instance. An interesting research area would be to expand the rule set using the tactics used in phishing emails and use it to detect phishing attack via emails.

URL-shortening services are growing in popularity thanks to micro-blogging websites such as Twitter[3]. In order to take advantage of the popularity and the obscurity provided by these shortening services, scammers are now establishing their own fake URL-shortening services [27]. Under this scheme, shortened links created on these fake

[3] http://twitter.com

URL-shortening services are further shortened by legitimate URL-shortening sites. These links are then distributed via phishing emails, blogs, micro-blogs, and social networking websites. We use the Python library [12] to automatically detect and expand shortened URLs.

To address this, additional rule could be determined such as: IF a URL is shortened by unsupported URL-shortening service, THEN the webpage is potentially phishing attack. Because our data set doesn't have any webpage satisfying this rule, we do not include it in our current rule set.

VI. Conclusions and Future Work

In this paper, we proposed and evaluated a rule-based phishing attack detection technique. By analyzing a large number of phishing webpages and combining various features used in ML approach, we generated our 15 initial rule set. These rules were then used as features in Decision Tree and Logistic Regression learning algorithms and their performance results were compared. C4.5 and LR gave competitive accuracy of 99% and FPR of 0.5% and FNR of 2.5%. Their performance slightly degraded, however, when tested with new data sets against models trained with old data set.

As future work, we plan to work on refining the rules to improve on false positives and negatives on newer data sets. Then we plan to develop a rule-based, light-weight, real-time phishing attack detection system like Snort; deploy and test the system in the real world.

Acknowledgment

The authors would like to acknowledge the generous support received from ICASA (the Institute for Complex Additive Systems Analysis), a research division of New Mexico Tech.

References

[1] "Antiphishing.org. 2010 2nd Quarter Report," 2011. [Online]. Available: http://apwg.org/reports/apwg_report_q2_2010.pdf.

[2] KROLL, Global Fraud Report. Accessed on April 10, 2011. http://www.kroll.com/about/library/fraud/Oct2010/region_northamerica.aspx.

[3] R. B. Basnet, A. H. Sung, D. Ackley, and Q. Liu, "A Web Mining Approach to Detecting Phishing URLs," *Computers & Security*, Elsevier, (submitted), 2011.

[4] R. B. Basnet, S. Mukkamala, and A. H. Sung, *Detection of phishing attacks: A machine learning approach*. Studies in Fuzziness and Soft Computing, 226:373-383, Springer, 2008.

[5] J. Ma, L. K. Saul, S. Safage, and G. M. Voelker, "Beyond Blacklists: Learning to Detect Malicious Web Sites from Suspicious URLs," in *Proc. ACM SIGKDD Conference*, pp. 1245-1253, Paris, France, June 2009.

[6] Y. Zhang, J. Hong, and L. Cranor, "CANTINA: A Content-Based Approach to Detecting Phishing Web Sites." in *WWW 2007*, Banff, Alberta, Canada, May 2007, ACM Press.

[7] C. Whittaker, B. Ryner, and M. Nazif, "Large-Scale Automatic Classification of Phishing Pages," in *Proc. 17th Annual Network and Distributed System Security Symposium*, CA, USA, March 2010.

[8] PhishTank, Out of the Net, into the Tank. [Online]. Available: http://www.phishtank.com/developer_info.php.

[9] R. Dhamija, J. D. Tygar, and M. Hearst, "Why Phishing Works," *CHI 2006*, Montreal, Quebec, Canada, April 2006.

[10] S. Doshi, N. Provos, M. Chew, and A. D. Rubin, "A Framework for Detection and Measurement of Phishing Attacks," in *Proc. ACM Workshop on Rapid Malcode (WORM)*, Alexandria, VA, Nov. 2007.

[11] E. Medvet, E. Kirda, and C. Kruegel, "Visual-Similarity-Based Phishing Detection," in *Proc. 4th International Conference on Security and Privacy in Communication Networks*, New York, NY, USA, 2008.

[12] PyLongURL. Python Library for LongURL.org. [Online]. Available: http://code.google.com/p/pylongurl/.

[13] StopBadware – IP Address Report – Top 50 by Number of Reported URLs. [Online]. Available: http://stopbadware.org/reports/ip.

[14] Google Safe Browsing API. [Online]. Available: http://code.google.com/apis/safebrowsing/.

[15] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA Data Mining Software: An Update," *SIGKDD Explorations*, Vol. 11, Issue 1, 2009.

[16] N. Chou, R. Ledesma, Y. Teraguchi, D. Boneh, and J. Mitchell, "Client-side defense against web-based identity theft," in *11th Annual Network and Distributed System Security Symposium (NDSS '04),* San Diego, USA, 2004.

[17] McAfee SiteAdvisor Software – Website Safety Ratings and Secure Search. [Online]. Available: http://www.siteadvisor.com/.

[18] J. Friedman, T. Hastie, and R. Tibshirani, "Additive Losgitic Regression: A Statistical view of Boosting," *The Annals of Statistics 2000*, Vol. 28, No. 2, pp. 337-407, 2000.

[19] SpoofStick Home. [Online]. Available: http://www.spoofstick.com/

[20] J. R. Quinlan, "C4.5 Programs for Machine Learning," Morgan Kaufmann Publishers, San Mateo, CA, USA, 1993.

[21] L. Weynin, G. Huan, L. Xiaoyue, Z. Min, and X. Deng, " Detection of Phishing Webpages based on Visual Similarity," in *WWW '05: Special interest tracks and posters of the 14th International Conference on World Wide Web*, pp. 1060-1061, New York, NY, USA, 2005. ACM Press.

[22] AVG Security Toolbar. [Online]. Available: http://www.avg.com/product-avg-toolbar-tlbrc#tba2

[23] Netcraft Anti-Phishing Toolbar. [Online]. Available: http://toolbar.netcraft.com/.

[24] D. Miyamoto, H. Hazeyama, and Y. Kadobayashi, "A Proposal of the AdaBoost-Based Detection of Phishing Sites," in *Proc. 2nd Joint Workshop on Information Security (JWIS)*, Aug. 2007.

[25] I. Fette, N. Sadeh, and A. Tomasic, "Learning to Detect Phishing Emails", *in Proc. 16th International Conference on World Wide Web*, pp. 649-656, May 2007.

[26] R. B. Basnet and A. H. Sung, "Classifying Phishing Emails Using Confidence-Weighted Linear Classifiers," in *Proc. International Conference on Information Security and Artificial Intelligence*, pp. 108-112, Chengdu, China, Dec. 2010.

[27] Symantec.cloud. MessageLabs Intelligence Reports. [Online]. Available: http://www.messagelabs.com/mlireport/MLI_2011_05_May_FINAL-en.pdf. Accessed May 25, 2011.

[28] L. Richardson. Beautiful Soup. [Online]. Available: http://www.crummy.com/software/BeautifulSoup/.

[29] Microsoft Security Intelligence Report, Vol.10, 2011. [Online]. Available: http://www.microsoft.com/security/sir/default.aspx.

[30] SmartScreen Filter – Microsoft Windows. [Online]. Available: http://windows.microsoft.com/en-US/internet-explorer/products/ie-9/features/smartscreen-filter, 2011.

[31] M. Roesch, "Snort – Lightweight Intrusion Detection for Networks." [Online]. Available: http://assets.sourcefire.com/snort/developmentpapers/Lisapaper.txt.

[32] Y. Pan and X. Ding, "Anomaly Based Web Phishing Page Detection," in *Proc. 22nd Annual Computer Security Application Conference, (ACSAC '06),* pp. 381-392, Miami Beach FL, USA, Dec. 2006.

[33] PhishTank – Statistics about phishing activity and PhishTank usage. [Online]. Available: http://www.phishtank.com/stats.php

[34] Yahoo! Inc.: Random Link - random. [Online]. Available: http://random.yahoo.com/fast/ryl.

[35] T. M. Mitchell, "Machine Learning," McGraw Hill, 1997.