BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **18**, No 1
Sofia • 2018 Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.2478/cait-2018-0002

Malicious URLs Detection Using Decision Tree Classifiers and Majority Voting Technique

Dharmaraj R. Patil, J. B. Patil

Department of Computer Engineering, R. C. Patel Institute of Technology, 425405 Shirpur, India
E-mail: *dharmaraj.rcpit@gmail.com*

***Abstract*:** *Researchers all over the world have provided significant and effective solutions to detect malicious URLs. Still due to the ever changing nature of cyber-attacks, there are many open issues. In this paper, we have provided an effective hybrid methodology with new features to deal with this problem. To evaluate our approach, we have used state-of-the-arts supervised decision tree learning classifications models. We have performed our experiments on the balanced dataset. The experimental results show that, by inclusion of new features all the decision tree learning classifiers work well on our labeled dataset, achieving 98-99% detection accuracy with very low False Positive Rate* (*FPR*) *and False Negative Rate* (*FNR*)*. Also we have achieved 99.29% detection accuracy with very low FPR and FNR using majority voting technique, which is better than the well-known anti-virus and anti-malware solutions.*

***Keywords*:** *Static and dynamic analysis; feature extraction; decision tree learning; malicious URLs; Web security.*

1. Introduction

The World Wide Web (WWW) has become the global platform for millions of users all over the world. Today's Web is well matured and has large application area, including e-commerce, online banking, social networking, communication and many more. Rich Web based applications are available over the WWW to provide such types of services. This is the positive side of this technology. Unfortunately, the Web has also become a more dangerous place; the popularity of WWW has also attracted hackers, intruders, attackers, etc., to abuse the Internet and users to perform illegitimate activity for financial profit. Popular types of attacks using malicious URLs include: Drive-by download, phishing, and social engineering and spamming [1]. Niels Provos et al. [2], in 2007 found more than three million URLs that launched drive-by-download attacks. According to Bin Liang et al. [3], 29 of 90 Websites contained malicious code. According to Davide Canali et al. [4] in particular attackers frequently use drive-by-download exploits to compromise a large number of users.

To overcome these issues, research community all over the world has applied many efficient machine learning approaches and achieved significant detection of malicious URLs. These include static analysis approach, dynamic analysis approach, blacklisting-based approach and heuristic-based approach [5]. D. Patil and J. Patil [6] have applied static analysis of URLs approach with 79 static features of URLs and domain names and achieved detection rate between 95-99% with very low False Positive Rate (FPR) and False Negative Rate (FNR). Also, they have applied static analysis approach for the detection of malicious JavaScript code in the Webpages with 77 static JavaScript features and achieved detection rate between 97-99% with very low FPR and FNR in their next study [7]. But, due to the ever changing nature of attack construction techniques applied by the attackers, there are still many open issues. To overcome the limitations of above approaches, dynamic analysis of URLs is more effective for the detection of today's dynamic attack construction techniques used by attackers. Web pages feature selection plays an important role in dynamic analysis, for the effective detection of malicious Web pages.

In this paper, we have applied a hybrid methodology, i.e. combination of static and dynamic approach for the detection of malicious URLs. We have extracted features using static analysis and some dynamic analysis of the URLs. We have extracted 117 static and dynamic features, among which 44 are new features to identify malicious URLs. We have constructed a balanced labeled dataset of 52,082 malicious and benign benchmarks URLs. Our dataset consists of equal distribution of malicious and benign URLs. It consists of 26,041 benign and 26,041 malicious URLs. We have evaluated our methodology using 6 state of the art decision tree learning classifiers including, J48 Decision Tree, Simple CART, Random Forest, Random Tree, ADTree and REPTree. We have built a multi-model classification system for the effective detection of URL as benign or malicious using Majority Voting algorithm. Also, we have compared our detection results with 18 well-known anti-virus and anti-malware solutions. Our experimental results show that, by inclusion of new features all decision tree learning classifiers perform well on our labeled dataset, achieving 98-99% detection accuracy with very low false positives and false negatives, as compared to the well-known anti-virus and anti-malware solutions.

The remainder of this paper is organized as follows. Section 2 gives a brief related work. Section 3 describes the methodology with feature extraction and supervised decision tree learning classifiers. Section 4 describes the experimental results. Section 5 gives discussion and limitations of our system. We present our conclusions in Section 6.

2. Related work

Many researchers all over the world have proposed different approaches for classification and detection of malicious URLs given as below.

Choi, Zhu and Lee [8] have proposed a method using machine learning to detect malicious URLs of all popular attack types like spam, phishing, malware etc. and to identify the nature of attack a malicious URL attempts to launch. They have

used features like lexical, link popularity, Webpage content, DNS, DNS fluxiness and network traffic. They have collected real-life data from various sources like benign URLs from DMOZ Open Directory Project, Yahoo!'s directory, Spam URLs from jwSpamSpy, Web spam dataset, Phishing URLs from PhishTank and Malware URLs from DNS-BH. They have used three machine learning algorithms like the Support Vector Machine (SVM) to detect malicious URLs, RAkEL and ML-kNN learning algorithms for multi-label classification problem to identify attack type. They have evaluated their method on 40,000 benign URLs and 32,000 malicious URLs and achieved the accuracy of 98% in detection of malicious URLs and 93% in identification of attack types.

Eshete, Villafiorita and Weldemariam [5] have presented a lightweight approach, called BINSPECT that combines static analysis and emulation. They have used supervised learning techniques in detection of malicious Web pages that may launch drive-by-download, phishing, injection and malware distribution attacks. They have extracted features like URL features, page-source features and social-reputation features. They have collected a malicious dataset of 71,919 URLs from the malware and phishing blacklist of Google, Phishtank database and the malware and injection attack URL list of MalwareURL. The benign dataset of 414,000 benign URLs is collected from three popular sources like the Alexa Top sites, the Yahoo random URL generation service and the DMOZ directory. According to their experimental evaluation, BINSPECT achieved 97% accuracy with low false signals.

Le et al. [9] have presented a novel two-stage classification model to detect malicious Web pages. They have divided the detection process into two stages. In the first stage they have estimated the maliciousness of Web pages using static features and in the second stage they have used the potential malicious Web pages found in the first stage for final identification of malicious Web pages by extracting run time features of these Web pages. They have extracted the static features from contents or properties of Web pages without rendering fully or executing Web pages. Potential run-time features like foreign contents, script contents and exploit contents are extracted by rendering Web pages fully and executing them on specific systems. They have used scoring algorithm for the classification. They have evaluated their scoring algorithm on the dataset of 20,000 benign Web pages for training and 13,646 instances of benign and malicious for testing. According to their experimental results, this approach reduced 86% of suspicious Web pages without missing attacks.

Basnet and Sung [10] have proposed a machine learning based approach to detect phishing Web pages. They have used many novel content based features and applied cutting-edge machine learning techniques such as 6 batch learning algorithms, Random Forests, Support Vector Machines (SVM) with rbf linear kernels, Naive Bayes, C4.5, Logistic Regression (LR) and a set of five online learning algorithms: Updatable version of Naive Bayes (NB-U), updatable version of LogitBoost (LB-U), Perceptron, Passive-Aggressive (PA) and Confidence-Weighted (CW) algorithms. They have used 179 Web page features such as lexical based features, keyword based features, search engine based feature and reputation

based features to demonstrate their approach. To conduct all the experiments, they used WEKA and CW libraries. The experimental results show that their proposed approach can detect phishing Webpages with an accuracy of 99.9%, false positive rate of as low as 0.00% and false negative rate of 0.06%.

Ma et al. [11] have explored how to detect malicious Web sites from the lexical and host-based features of their URLs. They show that this problem lends itself naturally to modern algorithms for online learning. According to them online algorithms not only process large numbers of URLs more efficiently than batch algorithms, they also adapt more quickly to new features in the continuously evolving distribution of malicious URLs. They developed a real-time system for gathering URL features and pair it with a real-time feed of labeled URLs from a large Web mail provider. According to their experimental analysis, they have achieved detection accuracy of 99% over a balanced dataset.

Garera et al. [12] have focused on studying the structure of URLs employed in various phishing attacks. They described several features that can be used to distinguish a phishing URL from a benign one. These features include page based, domain based, type based and word based. These features are used to model a logistic regression filter that is efficient and has a high accuracy. They have used millions of URLs in their experiments and achieved classification accuracy of 97.3%.

3. Methodology

3.1. Framework of our proposed of Malicious URLs detection system

Fig. 1 shows the framework of our proposed of malicious URLs detection system. It consists of feature extraction phase, training phase and classification phase. The raw malicious and benign URLs from benchmarks sources are fed to the feature extraction script written in Java.

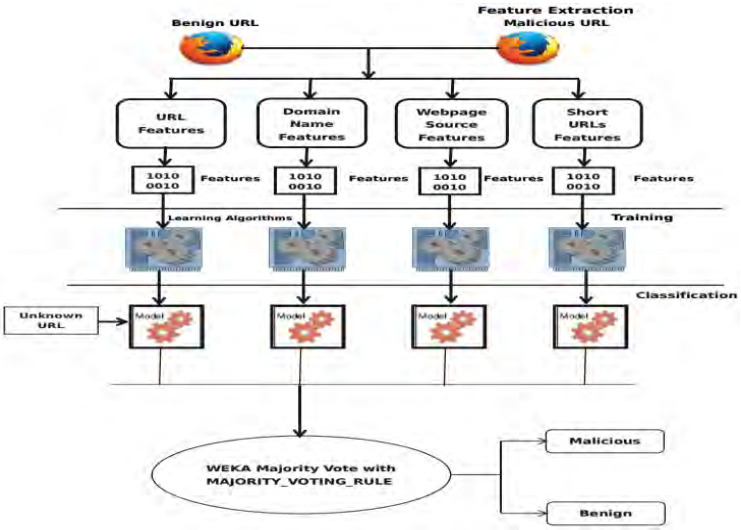

Fig.1. Framework of our proposed of Malicious URLs detection system

We have extracted the 117 static and dynamic features of the benign and malicious URLs. These are numeric and binary features. In our dataset preparation, we have labeled the benign URLs as –1 and malicious URLs as +1. In the training phase, 6 decision tree learning algorithms J48 Decision Tree, Simple CART, Random Forest, Random Tree, ADTree and REPTree are trained using our labeled dataset. This phase provides 6 trained models, which are used in the testing phase. In the testing phase, unknown URLs are tested using the trained model, as benign or malicious. We have evaluated 6 trained decision tree learning models on our dataset, in terms of detection accuracy, False Positive Rate (FPR), False Negative Rate (FNR), precision, recall, F-measure and ROC. Further, we have built a multi-model classification system for the effective detection of URLs as benign or malicious using Majority Voting algorithm. The Majority Voting scheme with MAJORITY_VOTING_RULE allows comparison of different models and makes the overall result more reliable.

3.2. Feature extraction

We have extracted four types of static and dynamic URLs features like, URL features, domain name features, Webpage source features and short URLs features. We have implemented URLs feature extractor in Java. The URL feature extraction is implemented based on the URL class of Java and the features are collected by lexical scanning of the URL string. The domain name features extraction is implemented based on the domain name extraction and scanning of the domain name. The Webpage source features are collected by visiting the page via Selenium WebDriver [13] and an instance of Firefox browser so as to capture the details of what is rendered (HTML) using a feature extraction engine implemented in Java. For each URL visit for feature extraction, a fresh instance of the Firefox browser is created to ensure a unique session for each URL. The short URLs features are extracted by checking the domain names containing the major URL shortening services like bit.ly, goo.gl, tinyurl.com, owl.ly, deck.ly, su.pr and bit.do. The expanded URLs are obtained by making query to the URL shortening services. After getting the original URL from URL shortening services, we have set a threshold value of 30 for the length of URLs i.e. if the length of the returned URL is over 30, it is marked as malicious. Also, we have checked the lexical properties of the returned URL string for deciding it as benign or malicious. We have checked the returned URL string for containing suspicious lexical characters like, _, =, (,), %, & and @.

3.2.1. URL features

We have extracted 63 URL features from the URL string. Among these features 47 are from the literature [5, 6, 8, 14-19] and 16 are new features. These are the lexical properties of the URLs. Lexical features are the textual properties of the URL itself. These features include the general look and feel properties of the URLs. In addition to the lexical features, we have checked the presence of suspicious words in the URLs. These are numeric and binary features. These URL features are given in

Table 1. We have extended the lexical feature set by adding 7 new lexical features. These features are important to differentiate malicious URLs from benign ones.

Table 1. URL features

Sr. No	Feature name	Type
Features used in the literature		
1	Length of URL	numeric
2	Presence of IP address in Hostname	numeric
3	Length of Query string in URL	numeric
4	Number of Tokens in URL	numeric
5	Number of Dots (.) characters	numeric
6	Number of Hyphens (-) sign characters	numeric
7	Number of Underscore (_) sign characters	numeric
8	Number of Equal (=) sign characters	numeric
9	Number of Forward slash (/) sign characters	numeric
10	Number of Question Mark sign (?)characters	numeric
11	Presence of “secure” word in URL string	binary
12	Presence of “account” word in URL string	binary
13	Presence of “webscr” word in URL string	binary
14	Presence of “login” word in URL string	binary
15	Presence of “ebayisapi” word in URL string	binary
16	Presence of “signin” word in URL string	binary
17	Presence of “banking” word in URL string	binary
18	Presence of “confirm” word in URL string	binary
19	Presence of “blog” word in URL string	binary
20	Presence of “logon” word in URL string	binary
21	Presence of “signon” word in URL string	binary
22	Presence of “login.asp” word in URL string	binary
23	Presence of “login.php” word in URL string	binary
24	Presence of “login.htm” word in URL string	binary
25	Presence of “.exe” word in URL string	binary
26	Presence of “.zip” word in URL string	binary
27	Presence of “.rar” word in URL string	binary
28	Presence of “.jpg” word in URL string	binary
29	Presence of “.gif” word in URL string	binary
30	Presence of “viewer.php” word in URL string	binary
31	Presence of “link=” word in URL string	binary
32	Presence of “getImage.asp” word in URL string	binary
33	Presence of “plugins” word in URL string	binary
34	Presence of “paypal” word in URL string	binary
35	Presence of “order” word in URL string	binary
36	Presence of “dbsys.php” word in URL string	binary
37	Presence of “config.bin” word in URL string	binary
38	Presence of “download.php” word in URL string	binary
39	Presence of “.js” word in URL string	binary
40	Presence of “payment” word in URL string	binary
41	Presence of “files” word in URL string	binary
42	Presence of “css” word in URL string	binary
43	Presence of “shopping” word in URL string	binary
44	Presence of “mail.php” word in URL string	binary
45	Presence of “.jar” word in URL string	binary
46	Presence of “.swf” word in URL string	binary
47	Presence of “.cgi” word in URL string	binary

Table 1 (c o n t i n u e d)

Sr. No	Feature name	Type
New features		
1	Number of Semicolon (;) sign characters	numeric
2	Number of Open Parenthesis (() sign characters	numeric
3	Number of Close Parenthesis()) sign characters	numeric
4	Number of Mod Sign (%) sign characters	numeric
5	Number of Ampersand Sign (&) sign characters	numeric
6	Number of At the Rate Sign (@) sign characters	numeric
7	Number of Digits in the URL	numeric
8	Entropy of URL string	real
9	Presence of ".php" word in URL string	binary
10	Presence of "abuse" word in URL string	binary
11	Presence of "admin" word in URL string	binary
12	Presence of ".bin" word in URL string	binary
13	URL without "www"	binary
14	Presence of "personal" word in URL string	binary
15	Presence of "update" word in URL string	binary
16	Presence of "verification" word in URL string	binary

- Shannon entropy of URLs

To demonstrate the randomness factor in URLs, we have used Shannon Entropy as a measure: higher the entropy, higher is the randomness of the instance under consideration. We calculated the entropy measure of each benign and malicious URL separately [20]. The Shannon entropy of the URL string is calculated using following equation:

$$H(x) = -\sum_{i=0}^{n} p(x_i) \log_b p(x_i), \tag{1}$$

where $H(x)$ is the Shannon entropy of string x, b is the base of the logarithm used, and $p(x)$ is the probability mass function.

Table 2 show the average entropy of malicious and benign URLs used in our dataset. From the table it is clear that, malicious URLs have high entropy as compare to benign URLs. It shows that there is more randomness factor in malicious URLs, to mark it as malicious.

Table 2. Average entropy of benign and malicious URLs used in our dataset

Sr. No	Average entropy of Benign URL string	Average entropy of Malicious URL string
1	3.87	4.14

- Suspicious word based features of the URLs

We have added seven new suspicious words in the URL feature set. The word-based features are binary. We tested if the given word is present or absent in a URL. We have used string matching algorithm by Knuth-Morris-Pratt (KMP) to find the presence or absence of the suspicious word in the URL [21]. The frequency distribution of these new suspicious word-based features is given in Table 3. It is clear from the Table 3 that the frequency of the suspicious word features in the malicious URLs is higher than that of benign URLs. Hence, these features help to identify malicious URLs from benign URLs.

Table 3. Distribution of word based features

Sr. No	Feature name	Distribution of word based features presence in URLs	
		Benign (%)	Malicious (%)
1	Presence of ".php" word in URL string	0.03	35.66
2	Presence of "abuse" word in URL string	0.01	5.51
3	Presence of "admin" word in URL string	0.04	6.45
4	Presence of ".bin" word in URL string	0.08	0.13
5	Presence of "personal" word in URL string	0.03	0.19
6	Presence of "update" word in URL string	0.15	2.2
7	Presence of "verification" word in URL string	0.00	0.72

3.2.2. Domain name features

We have used 18 domain name features, among these seven are taken from the literature [6, 8, 5, 12, 18, 22] and 11 are new features. We have extracted the domain names from the URL string a script written in Java. These are numeric, binary and real value features. The domain name features are given in Table 4.

Table 4. Domain name features

Sr. No	Feature name	Type	Description
Features used in the literature			
1	Length of Domain Name	numeric	Length of the domain name string
2	Domain Name contains IP address?	binary	It is 1 if domain contains IP address
3	Is Domain is TLD?	binary	It is 1 if domain is a top-level domain
4	Number of Sub-Domains	numeric	No of sub-domains in the domain name string
5	Number of Yahoo links for domain	numeric	No of Yahoo search results for the domain name
6	Number of Bing links for domain	numeric	No of Bing search results for the domain name
7	Alexa Rank of domain	numeric	Alexa ranking of the domain name
New features			
1	Domain Name is Valid?	binary	It is 1 if domain name is a valid domain name
2	Entropy of Domain Name string	real	Shannon entropy of the domain name string
3	Number of tokens in Domain Name	numeric	No of tokens in the domain name string
4	Length of Longest Domain Token	numeric	Length of longest domain name token
5	Entropy of Longest Domain token	real	Shannon entropy of the longest domain token
6	Average length of domain token	real	Average length of domain token
7	Number of tokens in Path	numeric	No of tokens in the domain name path string
8	Length of Longest Path Token	numeric	Length of longest domain path token
9	Average length of path token	real	Average length of domain path token
10	Domain Name contains suspicious https?	binary	It is 1 if domain name contains suspicious string "https"
11	Domain Name contains suspicious www?	binary	It is 1 if domain name contains suspicious string "www"

- Shannon entropy of domain name

We have used Shannon entropy to demonstrate the randomness factor in domain names of malicious and benign URLs. High entropy indicates the more suspicious nature of the URL. The Shannon entropy of the domain name string is calculated using (1). Table 5 show the average entropy of malicious and benign domain names and longest domain tokens used in our dataset.

Table 5. Average entropy of benign and malicious URL domain names and longest domain tokens

Sr. No	Average entropy of benign URL domain name	Average entropy of malicious URL domain name	Average entropy of longest domain token in benign URL	Average entropy of longest domain token in malicious URL
1	3.25	3.37	2.52	2.89

It is clear that the entropy of domain names and longest tokens in domain names of malicious URLs is higher than benign URLs. This indicates that there is more randomness factor in malicious URLs, to mark it as malicious.

3.2.3. Web page source features

For the effective detection of malicious Web pages, we have used the Web page source features. We have rendered the Web pages with the help of Selenium WebDriver and an instance of Firefox browser, every time for a new URL the Web page is loaded. We have written a script in Java and Selenium WebDriver, which extracts the Web page source features. We have extracted 34 such features among which 19 features are taken from literature [5, 8, 12, 14, 22, 23] and 15 are new features. These are numeric, binary and real value features. These features are given in Table 6.

Table 6. Web page source features

Sr. No	Feature name	Type
Features used in the literature		
1	Number of Blank Lines in a Web Page	numeric
2	Number of Blank Spaces in a Web Page	numeric
3	Number of Words in a Web Page	numeric
4	Average Length of Words in a Web Page	real
5	Number of iFRames in a Web Page	numeric
6	Number of JavaScript in a Web Page	numeric
7	Number of embed Tag in a Web Page	numeric
8	Number of object Tag in a Web Page	numeric
9	Number of meta Tag in a Web Page	numeric
10	Number of div Tag in a Web Page	numeric
11	Number of body Tag in a Web Page	numeric
12	Number of form Tag in a Web Page	numeric
13	Title Tag present? in a Web Page	binary
14	Number of anchor Tag in a Web Page	numeric
15	Number of Hidden elements in a Web Page	numeric
16	Number of External JavaScript Files in a Web Page	numeric
17	Number of Links in a Web Page	numeric
18	Number of Internal Links in a Web Page	numeric
19	Number of External Links in a Web Page	numeric

Table 6 (c o n t i n u e d)

Sr. No	Feature name	Type
New features		
1	Number of image Tag in a Web Page	numeric
2	Number of span Tag in a Web Page	numeric
3	Number of input Tag in a Web Page	numeric
4	Number of CSS styles in a Web Page	numeric
5	Number of audio Tag in a Web Page	numeric
6	Number of applet Tag in a Web Page	numeric
7	The size of Webpage	numeric
8	Credit card number word present? in a Web Page	binary
9	log word present?, in a Web Page	binary
10	pay word present?, in a Web Page	binary
11	free word present?, in a Web Page	binary
12	access word present?, in a Web Page	binary
13	bonus word present?, in a Web Page	binary
14	click word present?, in a Web Page	binary
15	Entropy of Webpage	real

3.2.4. Short URLs features

Today Online Social Networks (OSN) like Twitter, Facebook, WhatsApp, etc., are widely used by millions of users all over the world for communication. Due to the text limitation on OSN, URL shortening services like bit.ly, goo.gl, tinyurl.com, owl.ly, deck.ly, su.pr, bit.do, etc., are widely used; however they are not free from risks [24]. It is also applicable to the Webpages. To deceive the legitimate users' attackers often use such types of URL shortening services to hide their original identity. Considering this in mind, we have extracted two features of short URLs. We have written an URL expander script in Java, once we get the short URL with above URL shortening services; our expander script returns the original URL. We have set the threshold of ≥30 characters for the length of the URL and designed following rules:

```
1. if (expandedURL_length >= 30 && contains suspicious characters)
2. {
3.    URL — > malicious
4. }
5.  else
6. {
7.    URL — > benign
8. }
```

Also, to decide the URL is malicious or benign we have extracted the lexical features, i.e., is URL contains suspicious characters like, _, =, (,), %, & and @. These are numeric and binary features and given in Table 7.

Table 7. Short URLs features

Sr. No	Feature name	Type
1	Length of expanded URL	numeric
2	Is URL is malicious?	binary

3.3. Decision tree methods used for Malicious URLs detection

The problem of identifying malicious URLs is an instance of binary classification. For a given URL, the data point $x \in R^d$ represents its feature vector with d features. Let the set of training sample data be $\{(x_1, y_1), \ldots, (x_i, y_i), \ldots, (x_n, y_n)\}$, where x_i denotes the i-th feature vector; $y \in \{-1, +1\}$ is the label of the i-th feature vector, denoting whether the feature vector represents a benign or not; and n is the size of the data set. During testing, if the predicted label $\hat{y} = +1$ but the actual label $y = -1$, then the error is a false positive. If $\hat{y} = -1$ but $y = +1$, then the error is a false negative.

3.3.1. Decision tree learning

Owing to space limitations, the detail discussion of these algorithms is out of the scope of this paper. We have given the short description of each algorithm is as follow.

- J48 Decision Tree: J48 Decision tree learning is one of the most widely used techniques for classification. J48 is slightly modified C4.5 in WEKA. The C4.5 algorithm generates a classification-decision tree for the given dataset by recursive partitioning of data. The decision is grown using depth-first strategy. The algorithm considers all the possible tests that can split the data set and selects a test that gives the best information gain [35].
- Simple CART: Classification and regression trees are machine-learning methods for constructing prediction models from data. The models are obtained by recursively partitioning the data space and fitting a simple prediction model within each partition [36].
- Random Forest: Random forest is a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. It is an effective classifier in prediction. Random forest generally exhibits a substantial performance improvement over the single tree classifier such as CART and C4.5 [37].
- Random Tree: With k random features at each node, a random tree is a tree drawn at random from a set of possible trees. Here "at random" means that each tree in the set of trees has an equal chance of being sampled. Random trees can be generated efficiently and the combination of large sets of random trees generally leads to accurate models [38].
- ADTree: An alternating decision tree (ADTree) is a machine learning method for classification. It is introduced by Freund and Mason [39]. An ADTree consists of an alternation of decision nodes, which specify a predicate condition and prediction nodes, which contain a single number. An instance is

classified by an ADTree by following all paths for which all decision nodes are true and summing any prediction nodes that are traversed.

- REPTree: REPTree is a fast decision tree classifier which builds a decision/regression tree using information gain as the splitting criterion and prunes it using reduced-error pruning. It only sorts values for numeric attributes once. Missing values are dealt with by splitting the corresponding instances into pieces (i.e., as in C4.5) [40].

3.3.2. Majority voting

We have used WEKA's vote algorithm to obtain the final decision on, whether the URL as malicious or benign. Voting is the simplest ensemble algorithm and is often very effective. It can be used for classification or regression problems. It works by creating two or more sub-models, in our case 6 models. Each sub-model makes predictions which are combined using MAJORITY_VOTING_RULE. The following Fig. 2 gives the working of the majority voting algorithm. It is a meta-classifier for combining similar or conceptually different machine learning classifiers for classification via majority voting. In majority voting, we predict the final class label as the class label that has been predicted most frequently by the classification models. Here, we predict the class label $\hat{y}$ via majority voting of each classifier Cj [26, 27]:

(2) $$\hat{y} = \text{mode}\ \{C_1(x), C_2(x), ..., C_m(x)\},$$

where $\hat{y}$ predicted class label and $C_1(x)$, $C_2(x)$,..., $C_m(x)$ classification models.

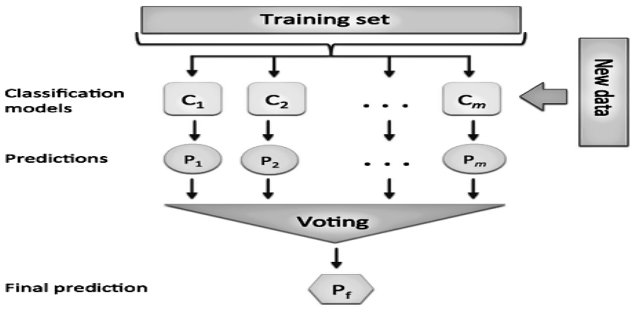

Fig. 2. Majority voting algorithm

4. Experimental setup and evaluation

4.1. Data source and dataset

We have collected URLs from the benchmark sources of URLs for both malicious and benign URLs and divided the dataset into a ratio of 66:34 as training and a testing set, i.e., 66% for training and 34% for testing. The dataset of benign URLs is collected from the Alexa Top sites [28]. We collected 26,041 benign URLs from the above source of benign URLs. For the malicious dataset, we have collected URLs from three benchmark sources, like the malware and phishing blacklist of the PhishTank database of verified phishing pages [29], the malware and injection attack URL list of Malware Domain List [30] and Spam domain blacklist by

jwSpamSpy [31]. We collected 26,041 malicious URLs from the above benchmark sources of malicious URLs including 8,976 phishing URLs, 11,297 malware URLs and 5,721 spam URLs. We have constructed a balanced dataset consisting of equal instances of malicious and benign URLs. The breakdown of the dataset is shown in Table 8.

Table 8. Dataset for training and testing

Task	Benign	Malicious	Total
Training	17,187	17,187	34,374
Testing	8,854	8,854	17,708
Total			52,082

4.2. Evaluation results

4.2.1. Evaluation measures

We have evaluated the performance of 6 decision tree learning classifiers on our URL dataset shown in Table 8. We have used the Weka API of all the learning classifiers, in our experiments [25]. To obtain the best classification results we have used the majority voting scheme. To decide the best performing classifier, we have used the confusion matrix, which contains actual and predicted classifications done by a classification algorithm [32]. We have used the following confusion matrix given in Table 9.

Table 9. Confusion matrix for actual and predicted benign and malicious URLs

Actual \ Predicted	Positive	Negative
Positive	TP	FN
Negative	FP	TN

Using the above confusion matrix we have calculated following measures, to evaluate the performance of the classifiers. A binary classifier predicts all data instances of a test dataset as either positive or negative. This classification (or prediction) produces four outcomes – true positive, true negative, false positive and false negative.

- True Positive (TP): correct positive (malicious URL) prediction
- False Positive (FP): incorrect positive (malicious URL) prediction
- True Negative (TN): correct negative (benign URL) prediction
- False Negative (FN): incorrect negative (benign URL) prediction

Based on the above confusion matrix, the classifier performance measures like accuracy, FPR, FNR, precision, recall and F-measure is calculated using the following equations:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FN} + \text{FF}}, \tag{3}$$

$$\text{FPR} = \frac{\text{FP}}{\text{TN} + \text{FP}}, \tag{4}$$

$$\text{FNR} = \frac{\text{FN}}{\text{TP} + \text{FN}}, \tag{5}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{6}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{7}$$

$$\text{F-measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{8}$$

4.2.2. Significance of new features

To verify whether the features we have introduced are important in enhancing the effectiveness of analysis and detection of malicious URLs, we compared the classification accuracy, False Positive Rate (FPR), False Negative Rate (FNR), precision, recall, F-measure and ROC of the classifiers with and without our newly introduced features on our URL dataset. As shown in Table 10, the use of new features improved the overall performance of all the classifiers, as shown with (↑) for improved accuracy.

Table 10. Overall contribution of new features on the accuracy of classifiers

Classifier	Accuracy without new features (%)	Accuracy with new features (%)	Change (%)
J48 Decision Tree	98.51	99.03	0.53 (↑)
SimpleCart	98.31	99.15	0.84 (↑)
Random Forest	98.98	99.44	0.46 (↑)
Random Tree	97.83	98.18	0.35 (↑)
ADTree	98.02	98.48	0.45 (↑)
REPTree	98.31	99.19	0.89 (↑)
Majority Voting	98.68	99.29	0.61(↑)

Table 11. Detailed performance analysis of machine learning classifiers on our URL dataset with and without new features

Classifier	Accuracy (%)	FPR	FNR	ROC
Without new features				
J48 Decision Tree	98.51	0.029	0.000	0.995
SimpleCart	98.31	0.033	0.001	0.999
Random Forest	98.98	0.020	0.000	0.998
Random Tree	97.83	0.040	0.004	0.979
ADTree	98.02	0.039	0.000	1.000
REPTree	98.31	0.033	0.001	0.999
Majority Voting	98.68	0.026	0.000	0.987
With new features				
J48 Decision Tree	99.03 (↑)	0.018 (↓)	0.002 (↑)	0.998 (↑)
SimpleCart	99.15 (↑)	0.016(↓)	0.001	0.998(↓)
Random Forest	99.44 (↑)	0.011 (↓)	0.000	1.000 (↑)
Random Tree	98.18 (↑)	0.032 (↓)	0.004	0.982 (↑)
ADTree	98.48 (↑)	0.029 (↓)	0.001 (↑)	1.000
REPTree	99.19 (↑)	0.014 (↓)	0.002 (↑)	0.998 (↓)
Majority Voting	99.29 (↑)	0.014 (↓)	0.000	0.993(↑)

As shown in Table 11, by the inclusion of new features the FPR and FNR of classifiers is decreased. The FPR of all seven classifiers is decreased as shown with (↓). The FNR of four of the seven classifiers remains same on both the features set with new features and without new features. The FNR of three of the seven classifiers is slightly increased using new features. Also the ROC area of four of the seven classifiers is increased by using the new features, while it remains same for one of the seven classifiers on both the features set with new and without new features. It is slightly decreased for SimpleCart and REPTree classifiers. The overall performance analysis of all the seven classifiers shows that, it is good indication that our new introduced features are enhancing the effectiveness of analysis and detection of malicious URLs.

Table 12 shows the overall performance of seven classifiers in terms of precision, recall and f-measure with and without inclusion of new features in our URL dataset. It is clear from the table, that there is a significant improvement in all the three performance measures for all 7 classifiers with the inclusion of new features as shown with (↑).

Table 12. Performance analysis of machine learning classifiers in terms of precision, recall and F-measure on our URL dataset with and without new features

Classifier	Precision	Recall	F-measure
Without new features			
J48 Decision Tree	0.986	0.985	0.985
SimpleCart	0.984	0.983	0.983
Random Forest	0.990	0.990	0.990
Random Tree	0.979	0.978	0.978
ADTree	0.981	0.980	0.980
REPTree	0.984	0.983	0.983
Majority Voting	0.987	0.987	0.987
With new features			
J48 Decision Tree	0.990 (↑)	0.990 (↑)	0.990 (↑)
SimpleCart	0.992(↑)	0.992(↑)	0.992(↑)
Random Forest	0.994 (↑)	0.994 (↑)	0.994 (↑)
Random Tree	0.982 (↑)	0.982 (↑)	0.982 (↑)
ADTree	0.985 (↑)	0.985 (↑)	0.985 (↑)
REPTree	0.992 (↑)	0.992 (↑)	0.992 (↑)
Majority Voting	0.993 (↑)	0.993 (↑)	0.993 (↑)

4.2.3. Comparison with antivirus and anti-malware softwares and services

To verify the effectiveness of our approach for the analysis and detection of malicious URLs, we compared the classification accuracy of 18 well-known antivirus and anti-malware softwares and services with our approach, as shown in Table 13. We have used the VirusTotal public API v2.0 in our Java program [33]. VirusTotal, a subsidiary of Google, is a free online service that analyzes files and URLs enabling the identification of viruses, worms, trojans and other kinds of malicious content detected by antivirus engines and website scanners. We have used the Vigneswar Rao's, VirusTotal public API v2.0 client implementation in Java, to design our script written in Java [34]. Virus Total's public API lets to upload and scan files, submit and scan URLs, access finished scan reports and make

automatic comments on URLs and samples without the need of using the HTML website interface. It allows building simple scripts to access the information generated by VirusTotal.

We have extracted the detection statistics of 18 well-performing antivirus and anti-malware softwares and services. We have tested all the 26041 malicious URLs used in our dataset using VirusTotal public API v2.0. Table 13 shows the detection accuracy of 18 well-known antivirus and anti-malware softwares and services on our malicious URLs dataset. Out of 18, Fortinet antivirus outperforms all the remaining antivirus and anti-malware softwares and services in detection accuracy, which has a detection accuracy of 96.5%. The overall detection accuracy of our approach using majority voting classifier with new features is 99.29%, which is far better than all the 18 well-known antivirus softwares. It shows that our approach is more effective in the analysis and detection of malicious URLs.

Table 13. Detection accuracy of well-known antivirus and anti-malware softwares and services on our malicious URLs

Sr. No	Antivirus and anti-malware softwares and services	Detection accuracy (%)
1	**Our approach**	**99.29**
2	Fortinet	96.5
3	Kaspersky	95.72
4	Sophos	79.68
5	Avira	62.17
6	BitDefender	58.95
7	ESET	51.04
8	G-Data	44.15
9	Websense ThreatSeeker	38.17
10	Emsisoft	35.12
11	Phishtank	33.54
12	Dr.Web	30.05
13	Google Safebrowsing	22.77
14	Netcraft	18.72
15	Malware Domain Blocklist	16.53
16	Malwarebytes hpHosts	12.44
17	Malware Patrol	6.93
18	Comodo Site Inspector	6.8
19	CLEAN MX	3.62

5. Limitations of our approach

Considering our approach, it is also not free from limitations. Following are some of the limitations of our malicious URLs detection system:

1. There is need to investigate features from social networks to characterize Malicious URLs.

2. Our methodology lacks analysis and detection of obfuscated JavaScripts in the Webpages, which is the major cause behind attacks like drive-by downloads, XSS, etc.

3. There is need to investigate more features of short URLs for the effective detection.

6. Conclusions

In this paper, we have performed the static and dynamic analysis of URLs for the detection of URL as benign or malicious. We have extracted 117 static and dynamic features of the URLs, among which 44 are new features. We have prepared a labeled dataset of 52,082 URLs, among which 26,041 are malicious and 26,041 are benign. We have evaluated the performance of 6 decision tree learning algorithms in terms of detection accuracy, FPR, FNR, precision, recall, F-measure and ROC on our balanced dataset. Our experimental results show that with inclusion of new features all the decision tree learning classifiers have achieved good detection rate between 98-99% with very low FPR and FNR. In addition, we have compared our approach with 18 well-known antivirus and anti-malware softwares and services in terms of detection accuracy. The experimental analysis show that, our approach outperform all the 18 well-known antivirus and anti-malware softwares and services in terms of malicious URLs detection accuracy with an overall accuracy of 99.29% using majority voting technique.

***Acknowledgements*:** This work is supported by the financial assistance under the scheme "Rajiv Gandhi Science and Technology Commission (RGSTC), 13-IIST/2014, Government of Maharashtra" through North Maharashtra University, Jalgaon, India.

References

1. Patil, D. R., J. B. Patil. Survey on Malicious Web Pages Detection Techniques. – International Journal of u- and e-Service, Science and Technology, Vol. **8**, 2015, No 5, pp. 195-206. **http://dx.doi.org/10.14257/ijunesst.2015.8.5.18**
2. Provos, N., P. Mavrommatis, M. A. Rajab, F. Monrose. All Your iFRAMEs Point to Us. – In: Proc. of 17th Conference on Security Symposium (SS'08), USENIX Association Berkeley, CA, USA, 2008, pp. 1-15.
3. Liang, B., J. Huang, F. Liu, D. Wang, D. Dong, Z. Liang. Malicious Web Pages Detection Based on Abnormal Visibility Recognition. – In: Proc. of International Conference on e-Business and Information System Security (EBISS'09), Wuhan, 2009, pp. 1-5.
4. Canali, D., M. Cova, G. Vigna, C. Kruegel. Prophiler: A Fast Filter for the Large-Scale Detection of Malicious Web Pages. – In: Proc. of 20th International Conference on World Wide Web (WWW'11), Hyderabad, India, 2011, pp. 197-206.
5. Eshete, B., A. Villafiorita, K. Weldemariam. BINSPECT Holistic Analysis and Detection of Malicious Web Pages. – In: Proc. of 8th International ICST Conference, SecureComm, Padua, Italy, 2012, pp. 149-166.
6. Patil, D. R., J. B. Patil. Malicious Web Pages Detection Using Static Analysis of URLs, – International Journal of Information Security and Cybercrime, Vol. **5**, 2016, Issue 2, pp. 31-50.
7. Patil, D. R., J. B. Patil. Detection of Malicious JavaScript Code in Web Pages. – Indian Journal of Science and Technology, Vol. **10**, 2017, No 19, pp. 1-12.
8. Choi, H., B. B. Zhu, H. Lee. Detecting Malicious Web Links an Identifying Their Attack Types. – In: Proc. of 2nd USENIX Conference on Web Application Development (WebApps'11), USENIX Association Berkeley, CA, USA, 2011, pp. 1-12.
9. Le, V. L., I. Welch, X. Gao, P. Komisarczuk. Two-Stage Classification Model to Detect Malicious Web Pages. – In: Proc. of IEEE International Conference on Advanced Information Network.ing and Applications, Biopolis, 2011, pp. 113-120.

10. B a s n e t, R. B., A. H. S u n g. Classifying Phishing Emails Using Confidence-Weighted Linear Classifiers. – In: Proc. of International Conference on Information Security and Artificial Intelligence, 2010.
11. M a, J., L. K. S a u l, S. S a v a g e, G. M. V o e l k e r. Learning to Detect Malicious URLs. – ACM Transactions on Intelligent Systems and Technology, Vol. **2**, 2011, No 3, Article 30, pp. 30(1)-30(24).
http://doi.acm.org/10.1145/1961189.1961202.
12. G a r e r a, S., N. P r o v o s, M. C h e w, A. D. R u b i n. A Framework for Detection and Measurement of Phishing Attacks. – In: Proc. of 2007 ACM Workshop on Recurring Malcode, 2007, pp. 1-8.
13. Selenium WebDriver 2.39. Last accessed on 25th December 2016.
http://www.seleniumhq.org/projects/webdriver/
14. C a n a l i, D., M. C o v a, G. V i g n a, C. K r u e g e l. Prophiler: A Fast Filter for the Large-Scale Detection of Malicious Web Pages. – In: Proc. of 20th International Conference on World Wide Web (WWW'11), Hyderabad, India, 2011, pp. 197-206.
15. W a n g, T., S. Y u, B. X i e. Novel Framework for Learning to Detect Malicious Web Pages. – In: Proc. of International Forum on Information Technology and Applications (IFITA'10), Kunming, 2010, pp. 353-357.
16. C o v a, M., C. K r u e g e l, G. V i g n a. Detection and Analysis of Drive-by-Download Attacks and Malicious JavaScript Code. – In: Proc. of International World Wide Web Conference Committee (IW3C2), Raleigh, North Carolina, USA, 2010.
17. M a, J., L. L a w r e n c e, K. S a u l, S. S a v a g e, G. M. V o e l k e r. Beyond Blacklists: Learning to Detect Malicious Websites from Suspicious URLs. – In: Proc. of 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'09), NY, USA, 2009, pp. 1245-1254.
18. L i u, H., X. P a n, Z. Q u. Learning Based Malicious Web Sites Detection Using Suspicious URLs. Last accessed January 2016.
http://users.eecs.northwestern.edu/~hlc720/349/HTXPZYQ.pdf/
19. Z h a n g, Y., J. H o n g, L. C r a n o r. CANTINA: A Content-Based Approach to Detecting Phishing Web Sites. – In: Proc. of International World Wide Web Conference Committee (IW3C2), Banff, Alberta, Canada, 2007, pp. 639-648.
20. V e r m a, R., A. D a s. What's in a URL: Fast Feature Extraction and Malicious URL Detection. – In: Proc. of 3rd International Workshop on Security and Privacy Analytics, 2017, pp. 55-63.
21. K n u t h, D. E., J. H. M o r r i s, V. R. P r a t t. Fast Pattern Matching in Strings. – In: SIAM Journal on Computing, Vol. **6**, 1977, No 2, pp. 323-350.
22. B a s n e t, R., S. M u k k a m a l a, A. H. S u n g. Detection of Phishing Attacks: A Machine Learning Approach. – Soft Computing Applications in Industry, Vol. **226**, 2008, pp. 373-383.
23. M a r c h a l, S., K. S a a r i, N. S i n g h y, N. A s o k a n. Know Your Phish: Novel Techniques for Detecting Phishing Sites and Their Targets. – In: Proc. of IEEE 36th International Conference on Distributed Computing Systems (ICDCS'16), 2016, pp. 323-333.
24. N e p a l i, R. K., Y. W a n g. You Look Suspicious!!: Leveraging Visible Attributes to Classify Malicious Short Urls on Twitter. – In: Proc. of 49th Hawaii International Conference in System Sciences (HICSS'16), 2016, pp. 2648-2655.
25. Weka 3: Data Mining Software in Java. Last accessed December 2016.
http://www.cs.waikato.ac.nz/ml/weka/
26. EnsembleVote Classifier. Last accessed on 25th January 2017.
http://rasbt.github.io/mlxtend/user_guide/classifier/EnsembleVoteClassifier/
27. How to Use Ensemble Machine Learning Algorithms in Weka. Last accessed on 25th January 2017.
http://machinelearningmastery.com/use-ensemble-machine-learning-algorithms-weka/
28. Alexa: Alexa Top 500 Global Websites. Last accessed on November 2016.
http://www.alexa.com/topsites/
29. PhishTank: Join the Fight against Phishing. Last accessed on November 2016.
https://www.phishtank.com/

30. Malware Domain List. Last accessed on December 2016.
http://www.malwaredomainlist.com/forums/index.php?topic=3270.0/
31. Spam Domain Blacklist (Filtered by jwSpamSpy). Last accessed on December 2016.
http://www.joewein.de/sw/blacklist.htm/
32. Basic Evaluation Measures from the Confusion Matrix. Last accessed on January 2017.
https://classeval.wordpress.com/introduction/basic-evaluation-measures/
33. VirusTotal Public API v2.0. Last accessed on January 2017.
https://www.virustotal.com/en/documentation/public-api/
34. VirusTotal Public API v2.0 Client Implementation in Java. Last accessed on January 2017.
https://vighnesh.me/virustotal/
35. Q u i n l a n, J. R. Induction of Decision Trees. – Machine Learning, Vol. **1**, 1986, No 1, pp. 81-106.
36. B r e i m a n, L., J. H. F r i e d m a n, R. A. O l s h e n, C. J. S t o n e. Classification and Regression Trees. Belmont, California, Wadsworth International Group, 1984.
37. B r e i m a n, L. Random Forests. – Machine Learning, Vol. **45**, 2001, No 1, pp. 5-32.
38. Z h a o, Y., Y. Z h a n g. Comparison of Decision Tree Methods for Finding Active Objects. – Advances in Space Research, Vol. **41**, 2008, No 12, pp. 1955-1959.
39. F r e u n d, Y., L. M a s o n. The Alternating Decision Tree Learning Algorithm. – In: Proc. of International Conference on Machine Learning, Vol. **99**, 1999, pp. 124-133.
40. REPTree. Last accessed on November 2016.
http://weka.sourceforge.net/doc.dev/weka/classifiers/trees/REPTree.html

Received 23.10.2017; *Accepted 23.01.2018*