ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 3, Issue 9, September 2016

# Machine Learning Approach for Detecting Malicious Domain Names

[1]Madeeha M.K, [2]Nataasha N. Raul
[1][2]Department of Computer Engineering
[1][2]Sardar Patel Institute of Technology Mumbai-400058, India

***Abstract*—The Domain Name System (DNS) is an integral component of the internet which efficiently converts domain names into 32 bit IP addresses and vice versa. The internet had always been a favorite target where the attackers staged various types of malicious activities with their effects ranging from small to extreme. The attackers are now a days focusing on attacking domains which involves managing botnet to fire numerous attacks on the victim. Relative to the increase in the number of wicked web users targeting domains for implementing malicious activity there is also an increase in the number of scholars interested in doing research to curb the advancement of such attacks. Many systems have already been introduced various researchers to detect domains involved in malicious activities based on the DNS query and response observation of the domain under consideration. Some of the systems thus developed blacklist those domains that are fraudulent in nature. We propose a machine learning approach in detecting malicious domain names by assigning scores to the domains based on the analysis. The attributes relevant for our experiment that distinguish malicious and benign domains were obtained by analyzing a large number of features from collected DNS query responses. In addition to the features based on domain names we also make use of the Google Page Rank, Alexa Traffic Rank and SSL rating. We assign score to each domain in the range of 0 to 10. Based on the experimental results and observations a low score below and including 5 implies that the domain is involved in malicious activities and scores above 5 implies benign domain.**

***Keywords*—Domain Scoring, Malicious domains, Passive DNS analysis.**

## I. INTRODUCTION

Internet is playing a crucial role in our day to day life and it is no longer a safe place to heedlessly roam around. The advancement in internet technologies has made the life of common man easy and simple but at the same time inviting a lot more threats to his personal and public life in ways which he could not even imagine. As such people with malicious intentions are now a days focusing on the domain name system to perform their attacks. They infect their the domain name sevice thereby using it to maintain their malicious device distributed network. There are several types of attacks addressing domains such as phishing, dns amplification, domain lockup, cache poisoning, typo squatting etc. Domain Name System can be considered as the backbone of Internet. It stores information about all those systems hosted in the Internet assuming the ip address to be static in nature. DNS maps the domain names into otherwise hard to remember

IP addresses. All the information the DNS holds is thus stored in a large number of hierarchical database servers which is maintained as a distributed database. And it is this hierarchical database servers which actually provides the mapping between the domain names and corresponding ip addresses. When a browser requests for IP address corresponding to a host name, DNS resolver sends a request to a DNS server to get the server's IP address. If the requested DNS server is not able to resolve it, that DNS server will then forward request to the root DNS server. The root DNS server will send the address of the DNS server in the next level of hierarchy and check for its availability there. This process repeats a number of times until we obtain the actual IP address. DNS also maintains a cache to store the recent queries so that next time the client makes a request, the server can directly reply from its cache rather than traversing the entire hierarchy[1].

Our proposal is to create a malicious domain detection system which checks for every domain we enter if it is malicious or benign based on a reputation score assigned to each domain on real time. The ultimate goal of the system is to classify the domains into either of the two classes whereby we develop the system such that it assigns low score to malicious websites and a high score to domains that are legitimate. This method of assigning reputation scores to domains helps to dynamically blacklist domains that are malicious in nature. In order to compute and assign scores to each domain that is under consideration, we passively collected DNS requests from the gateway DNS Server of the Enterprise network and also DNS information from various sources such as Security Information Exchange, Alexa, Center for Applied Internet Data Analysis and from Internet Wide Scan Data

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 3, Issue 9, September 2016

Repository were collected. We relied on Malware Domain List, Phish Tank, and several other trusted Internet sources for Malicious domain information collection.

The paper is structured as follows. In Section II, we present the related works which summaries the various previous re- searches done in detecting malicious domains. Section III pro- vides the problem description and an overview of the solution that we propose. Section IV describes in detail our model to detect malicious domains by applying machine learning technique and random forest classifier to the selected set of features. In section V we provide the experimental results and system performance with a large dataset that we have collected during the initial system training phase. Section VI concludes

## II. RELATED WORK

### *A.DNSDataCollectionandClassification*

F. Weimer et al., [2] proposed a method for collecting DNS data by passively analyzing DNS requests using a sensor for packet capture and observed difference between responses of legitimate domain lookups and responses from lookup of malicious domains which is carried out by analyzer. Sensors are distributed across the network. Results showed that there is a difference in behaviour of Fast-Flux networks when compared to benign domains.

Roberto Perdisci et al., [3] presented FluxBuster, a system that detects and tracks malicious flux networks by passive DNS traffic analysis. DNS traffic traces generated by recursive DNS servers located in different networks across the world was monitored on a large-scale. They also collected andanalysed a large number of domains that are suspicious in nature with their source being spam emails and blacklists and could detect any malicious flux networks when they are accessed by the victims. A five month long experiment with FluxBuster showed an accurate detection of malicious flux networks with a low false positive rate.

Zdrnja et al. [4] discusses passive network monitoring to detect DNS anomalies. Database stores the passively collected DNS data and helps to determine historical behavior of partic- ular DNS records and linkage between them. Better anomaly detection will be enabled by additional sensor installation all over the world. The database analysis is an automated process which can easily and quickly detect anomalies and attacks to the system and thus it serves as an early alert system against spam and worm attacks[4].

Plonka et al.[5] in their paper proposes a scalable approach to manage passive DNS data collection using a method to time correlate zone properties and also network related properties. According to the paper if the malicious traffic could be identified in an accurate and right on time manner it could at least minimize the impact of the attack even if it cannot block it before it completes its action. The key problem that pose a challenge in this approach is that there exist no inherent mechanism to achieve this.

Leyla Bilge et al.[6] applied machine learning to passive DNS query analysis and malicious DNS detection. For the purpose of extracting relevant features among a possibly large set of features they made use of genetic algorithm and for classification phase they relied on J48 decision tree. They studied DNS usage for benign and malicious domains, and identified 15 features that can be used to distinguish both. To evaluate the J48 classifier[16], they classified their training set with tenfold cross-validation and percentage split of 66 percentage for training and rest for checking correctness. Area under ROC curve was high for both methods. They got an accuracy of 99 percentage with their classifier.

Felegyhazi et al. et al. [7] proposed a system for black-listing domain names by applying clustering on the collected name server details and registration information of domains. They collected a set of bad domains which was used as seed. They proposed a method based on initial seed domains to infer sets of malicious domins which have not yet been includedin any of the blacklists. They measured the accuracy of their

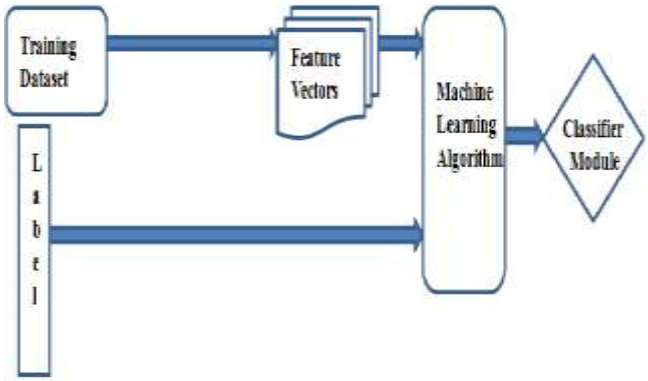


***Fig. 1. Proposed System Architecture: Training***

predictions using a combination of several popular blacklists plus services. They found that from a fairly

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 3, Issue 9, September 2016

modest set of initial seeds they could predict a large set of additional malicious domains, with arguably quite low false positives[7].

### *B. Assigning Reputation Score to Domains*

Manos Anton kakis, in his paper proposes a dynamic DNS based reputation score assigning system[8]. In thispaper, author studied and implemented a method for assigning repu- tation scores to new and unknown domains dynamically. They assigned low score for malicious domains and high score for trusted benign domains. With this reputation score dynamic domain name blacklists could counter cyber attacks really very effectively. The real world evaluation of Notos show that it can correctly classify new domains with a low false positive rate 0.38 percent and high true positive rate 96.8 percent.

## III. PROBLEM DESCRIPTION AND PROPOSED SOLUTION OVERVIEW

The Domain Name System (DNS) is the backbone of internet.It translates the user friendly domain names into corresponding IP addresses and vice-versa. This system helps people to just remember the easy to memories domain names instead of their corresponding complicated 32 bit IP address. Because of its necessary role in the world wide network of an enormously huge number of systems, DNS is additionally a fascinating target for malicious activities such as spamming, phishing, and botnets. Bots makes use of domain name system to resolve names to locate their command and control servers. Similarly, spam mails contain URLs which are actually links to the faulty domains that resolve to scam servers. It is worth to monitor the use of the DNS system for signs that indicate that an explicit domain name is employed as part of a malicious operation[6] .

Our proposal is to develop a system that assigns reputation score to the domains that the user queries based on passive DNS analysis.

## IV. PROPOSED SYSTEM ARCHITECTURE

The overall architecture of the system is shown as two separate diagrams. The first one i.e, Figure 2 corresponds to training and the second one i.e, figure 3 corresponds to the class-tosifter. To start with, we have three modules that act as backbone our system. The data collector module, the classifier module and the score predictor module. The collection of the huge amount of relevant information necessary for the experiment was carried out using the data collection module. Immediately after the data collection phase we carry out a passive analysis on the available data to build a classifier that will be able to identify a domain as malicious or benign. Supervised machine learning techniques was used to distinguish between DNS activities of normal and malicious domains. The random forest algorithm which is an ensemble learning technique was used in the construction of classifier module. A forest as we all know is a collection of trees and here the Random Forest algorithm constructs a forest of trees by selecting a random subset of data and as well as a random subset of features each time for each tree construction and thus end up with a different tree each time. Each tree contributes to the final score computation which we obtain as a probability when using the algorithm. We then multiply this probability measure by 10 to obtain the actual reputation score of the domain that is considered which will be in the range of 0 to 10.

### *A. Data Collection*

The data collection is an important part of a classification experiment. This is the initial step which is actually done offline to train the system to learn from the collected details. To come up with an output prediction the classifier should initially be trained with a data set. So in this data collection process we made use of Alexa and Malware domain list for collecting the list of benign and malicious domain names respectively. We thus create a file containing the list of all trusted and malicious domains and then collect all the necessary properties which we have previously identified as relevant to the experiment. Feature relevance was calculated using Sanford Predictive Modeler 8.0. The features include the TTL or time to live, the number of IP address associated with the domain name, domain registration details such as how long the domain was registered, name server of the domain name, Google page rank, Alexa traffic rank and SSL rating by Qualys SSL Labs. DNS data is obtained from various resources like Security Information Exchange, Alexa, Center for Applied Internet Data Analysis and from Internet Wide Scan Data Repository.

### *B. Classification based on DNS Features*

The goal of the classifier is to classify at a very low false positive rate. Figure 3 shows the classifier. Since we need to use very large data sets for classification we employ ensemble learning methods. The development of ensemble methods have been very influential in the data analysis and machine learning areas. Here many base estimators or trees are built with the learning algorithm whose predictions are

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 3, Issue 9, September 2016

then combined by ensemble learning technique. Thus it improves performance over a single estimator. Averaging methods are used to obtain the final class prediction where the technique is to first build a number of individual estimators independent of each other and finally average their predictions[15]. Generally, the combined figure is far better than any other single base figure since its variance is reduced. Here we use random forests for classification.

Random forests have proved to be very effective and powerful ensemble learning method that could be used for

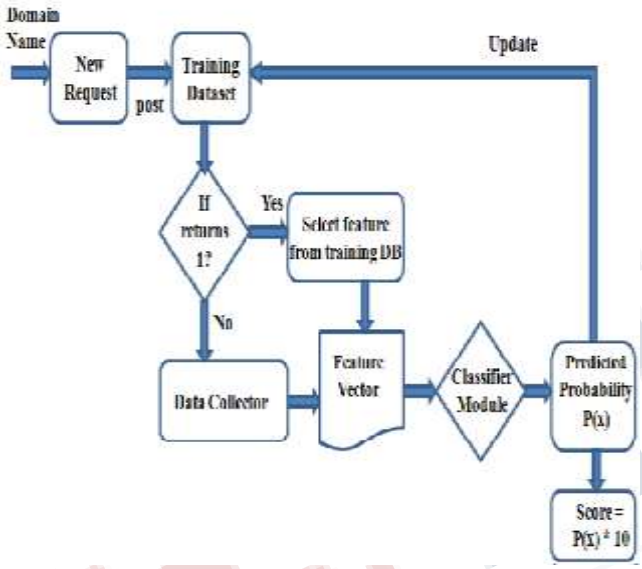


*Fig. 2. Proposed System Architecture: Classifier*

Classification of datasets having high dimensionality [15]. In random forests, every tree within the ensemble is built from a random sample drawn with replacement from the training set. When splitting a node throughout the development of a tree, the split that is chosen might not be the most effective choice among all possibilities. But it is perpetually the most effective choice a random subset of features that are chosen at that instant. Due to randomness, the bias of the forest will slightly increase with respect to the bias of a single non- random tree however due to averaging, its variance decreases, typically quite compensating for the increase in bias, thus yielding an overall better model. The random forest makes use of bagging mechanism to classify very large data set with multiple dimansions quickly and accurately. Random Forests either performs voting from individual trees or combine and find average of all the individual trees probabilistic prediction in order to obtain classification accuracy. Here we make use of the thescikit-learn implementation of random forest which combines classifiers by averaging their probabilistic prediction.

For score computation we use predictproba(X) function where X in our case is the test csv file containing n samples with its features corresponding to n domains. This function predict class probabilities for each domain in the list. The predicted class probabilities of an input sample is computed as the mean predicted class probabilities of the trees in the forest[15]. The class probability of a single tree is the fraction of samples of the same class in a leaf. This function will give us the output probability prediction Prb(X). We can also use this function to compute the probability prediction for individual domain by passing the corresponding feature vector as the input to the function which we will be using in the final score computation phase. Random Forest is used in our work because of the following reasons:

- Large training data set. When training set is large high bias classifiers such as Naive Bayes are not powerful in giving accurate classification models.
- Normal decision trees tends to over fit. This can be reduced in Random Forest by fixing the number of cross folds.
- Random Forest is fast and scalable. Random Forest can provide probabilistic outputs which in our case can be used as score by multiplying the probability by 10.

Each tree in the forest is trained with a different data set as well as a different feature set. This selection of data and feature set is done at random as determined by the random forest algorithm using the bootstrap method. At each split the algorithm tries to reduce the impurity. That is for each tree in the forest first of all it randomly chooses a training data subset. It then splits at each node based on the condition and continues until the stop condition is met or the tree is fully grown. So with each split the bootstrapping process will choose the variable subset. And for each chosen variable we apply the sample data and sort it by the chosen variables. We compute the number of instances present in each interval value for each of the feature in the selected feature set. The Gini impurity will be computed at each split point. Finally it chooses the best split to be at the root node that is the one with minimum Gini impurity will be chosen. This procedure will be repeated for the remaining features in the randomly chosen subset. And once this tree is fully

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 3, Issue 9, September 2016

constructed for the next tree the same steps repeats starting from random data and feature selection till completion of tree. The accuracy estimation is done by the random trees internally while training[15].

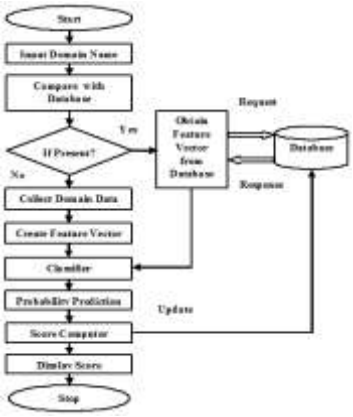


***Fig. 3. Data Flow Diagram of proposed system***

### *C. Scoring Procedure*

We consider the following parameters for the score evalu- ation. The page rank provided by Google and the traffic rank from Alexa, SSL rating from Qualys Labs and DNS based features. DNS based features consists of those features that are extracted by observing DNS traffic.When a new DNS request is given by the user, it is first compared with the database entries to check if it is already present or not. Based on the response we choose the feature vector corresponding to the domain name from database if already present and then it is passed to the classification model. In case the domain is new to the system we pass the domain name to the data collector module, collect the features and construct the feature vector and then pass it to the classifier. We are using Random Forest as classifier. Most of the machine learning methods provide classification result only. But for our work, it is not enough. Probability of the domain name being benign is to be found. Random Forest provides accurate probability estimations

The overall score is calculated as follows:

$$S = 10 * P(X) \quad (1)$$

where S is the score, X is the new request and P is the probability of new request being benign.

## V. EXPERIMENTAL SETUP AND RESULTS

A working implementation of the malicious domain detec- tion and scoring system primarily based on machine learning was designed using Python and MySql. The system comes with a data collection module, classifier module and a score predictor. Fig 3 depicts the flow diagram of the proposed system. Initially we have the data collector module which we used to collect the training data as well as the data related to new incoming request. After the initial data collection phase system is trained to predict the outcome of a new request based on a classification algorithm. This is the offline phase. Fig4 shows the sample output for data collector module. In the first column of the data collector output we have the domain names and the corresponding rows carry the feature vector for each one of them as computed by the system. The final column holds the valid data which determines the class to which the domain belongs. If valid field is set to 1 then we assume the domain to be trusted and if 0 we assume it to be malicious.Now when a user request a new domain name it will be checked in the database if it is already present there and if present the row or feature set corresponding to that will be fetched from the database . If the requested domain is new to our system we will make use of the data collector module to gather and compute all the feature vector values from relevant sources and the DNS query analysis and thus compute the feature vector. This feature vector is then passed on to the classifier module. The Random Forest classifier construct a forest of 100 random trees with randomly selected dataset from the training database and using a randomly chosen feature set. Here to each of the hundred trees the new domain feature vector will be applied on top. The traversal progress as per the values and it differ in different trees. Finally towards theleaf of each tree it will fall into either of the two classes which will be the prediction of that individual tree. Combiningthe predictions from all the hundred trees constructed for this particular request we get the actual class prediction for the domain. Fig 5 shows the classifier sample prediction output for the given set of domains. Last column classifier output holds the predicted value. Multiplying this prediction result with hundred we compute the domains reputation score. If the score thus obtained is greaterthan five then the valid field will be set to 1; implying a trusted entity; and the training database is updated accordingly for the new entry. In case it is lessthan 5 the valid field is set to 0 implying a malicious domain and the database is updated. The threshold value is set as five after performing a number of experiments with a constant test data set and varying training database. The training dataset was changed each time by varying the ratio between the number of trusted and malicious domain vectors. We performed the experiment twenty times with different training sets on the same test data and it was observed that the trusted domains always maintained a score above 5 and malicious ones always less

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 3, Issue 9, September 2016

than five except for a few cases where certain malicious domains went slightly above 5 which could be neglected as error rate compared with the size of data considered for experiment.

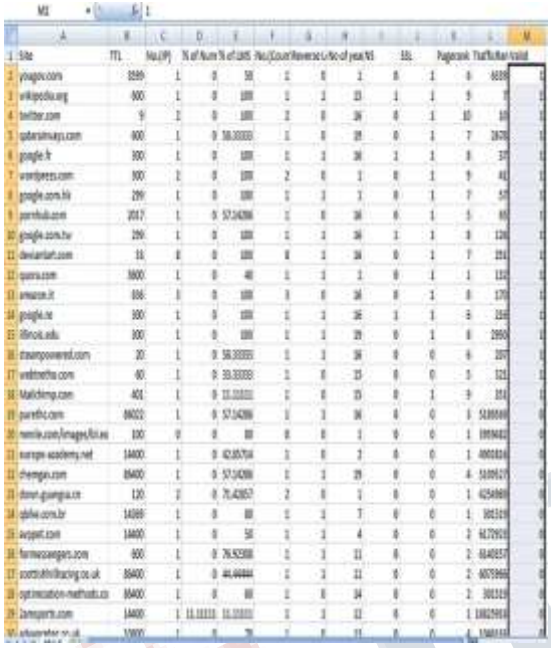

*Fig. 4. Sample Output for Data Collector*

The performance of our proposed malicious domain detection system with random forests was analyzed and compared with ID3 and J48 classifier algorithms. To proceed with evaluation, the system must first be trained with a dataset. For this purpose we have collected a large number of domains including both trusted and malicious from Alexa and malware domain list respectively. For each of the domain name in the list, all the features required for classification were collected.DNS data is obtained from various resources like Security Information Exchange, Alexa, Center for Applied Internet Data

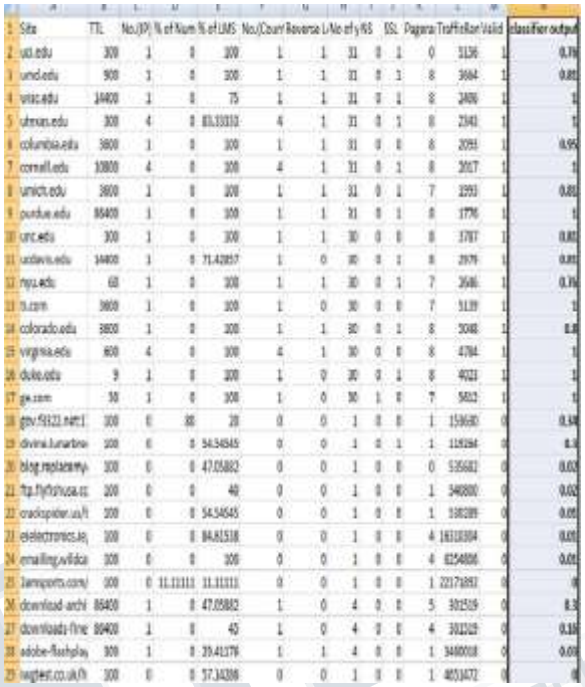

*Fig. 5. Classifier Prediction Results for the Sample Set of Domains*

Analysis and from Internet Wide Scan Data Repository. The entire data is thus saved in our MySql database named training. This data is then applied to the weka tool for performance analysis and comparison. Fig 6 shows the table holding average TPR, FPR, AUC and accuracy values of the three algorithms when applied to our full training dataset, with ten fold cross validation and with 66 percentage split. The results show that with full training dataset applied both the ID3 and Random Forest classifiers have hundred percent accuracy and TPR and J48 has 98.4 percent. With ten fold cross validation and

| Experiment | Detector | TPR | FPR | AUC | Accuracy |
|---|---|---|---|---|---|
| Full Training Data | J48 | 0.983 | 0.02 | 0.993 | 0.983 |
| | ID3 | 0.998 | 0.006 | 1 | 0.997 |
| | RandomForest | 1 | 0 | 1 | 1 |
| 10-Fold Cross Validation | J48 | 0.976 | 0.038 | 0.987 | 0.975 |
| | ID3 | 0.993 | 0.011 | 0.992 | 0.992 |
| | RandomForest | 0.995 | 0.011 | 0.996 | 0.995 |
| 66% Split | J48 | 0.974 | 0.032 | 0.973 | 0.991 |
| | ID3 | 0.985 | 0.028 | 0.978 | 0.984 |
| | RandomForest | 0.998 | 0.013 | 1 | 0.998 |

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 3, Issue 9, September 2016

***Fig. 6. Performance Analysis for three classification algorithms***

percentage split also Random Forests performs best followed by ID3 in either case. With Random Forests we have an overall average TPR of 99 percent and FPR of 0.1 percent. Additionally we have plotted a Receiver Operating Characteristic(ROC) Curve which is a tradeoff between the true positive rate or sensitivity and false positive rate and computed the area under ROC which was found to be 0.998.

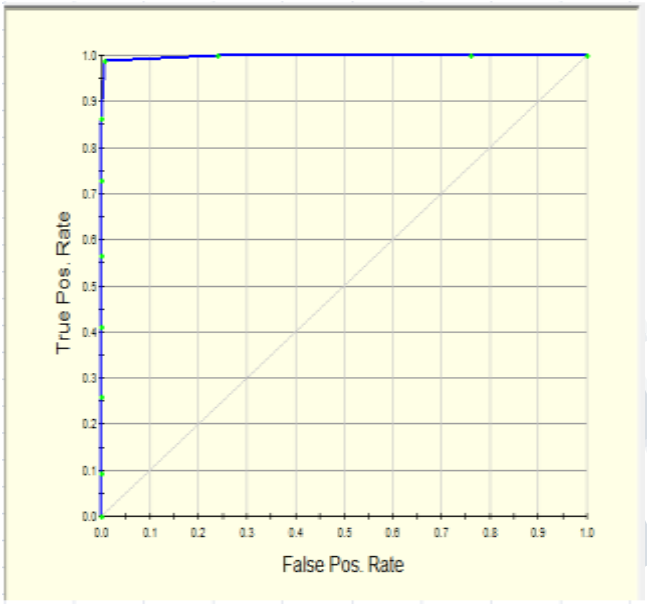


***Fig. 7. ROC Curve***

## CONCLUSSION

Domain Name system is the fundamental component of internet and domain names are a prerequisite to work with the web. Cyber criminals target and host attacks against certain domains and indulge those domains in unethical or criminal activities sometimes even without the knowledge of the owner as in the case with domain hijacking or domain theft. The attackers also use the domain names to control the botnets and communicate with each other. Lot of research work had been done in this regard to identify such faulty domains which had been successful in curbing their growth to an extent but still demanding a better solution. We attempt to build a malicious domain detection system that assigns score to domain names based on their page rank, SSL rating and DNS related features. Our system assigns a reputation score to every domain accessed which is in the range of 0 to 10. In addition to the use of identified features we also incorporate machine learning principle to develop a self-learning classifier system. If deployed in an organization the system can be used to send an alert to the admin on accessing malicious sites.

## ACKNOWLEDGMENT

TheauthorswouldliketothankttheSardarPatelInstitute ofTechnology,Indiaforprovidingthenecessaryfacilitiesfor carryingoutthiswork.

## REFERENCE

[1] Jon Postel. ”Domain name system structure and delegation”. 1994.
[2] Florian Weimer.” Passive dns replication”. In FIRST Conference on Computer Security Incident, pp. 98, 2005.
[3] Roberto Perdisci, Igino Corona, and Giorgio Giacinto. ”Early detection of malicious flux networks via large-scale passive dns traffic analysis”.In IEEE Transactions on Dependable and Secure Computing, pp. 714726, IEEE, 2012.
[4] Bojan Zdrnja, Nevil Brownlee, and Duane Wessels.”Passive monitoring of dns anomalies” In Detection of Intrusions and Malware, and Vulnerability Assessment, pp. 129139, Springer, 2007.
[5] David Plonka and Paul Barford. ”Context-aware clustering of dns query traffic”. In Proceedings of the 8th ACM SIGCOMM conference on Internet measurement, pages 217-230, ACM, 2008.
[6] Leyla Bilge, Engin Kirda, Christopher Kruegel, and Marco Balduzzi ”Exposure: Finding malicious domains using passive dns analysis”. In NDSS, 2011
[7] Felegyhazi, Mark and Kreibich, Christian and Paxson, Vern ”On the Potential of Proactive Domain Blacklisting ”.Proceedings of the 3rd USENIX conference on Large-scale exploits and emergent threats: botnets, spyware, worms, and more, USENIX Association, 2010.
[8] M. Antonakakis, R. Perdisci, D. Dagon, W. Lee, and N. Feamster. ”Building a Dynamic Reputation System for DNS”.In 19th Usenix Security Symposium, USENIX, 2010.
[9] E. Passerini, R. Paleari, L. Martignoni, and D. Bruschi.” Fluxor: Detecting and monitoring fast-flux service networks”. In Detection of Intrusions and Malware, and Vunerability Assessment, 2008.
[10] R. Perdisci, I. Corona, D. Dagon, and W. Lee. ”Detecting Malicious Flux Service Networks through Passive Analysis of Recursive DNS Traces”. In 25th Annual Computer Security Applications Conference, ACSAC, 2009.
[11] Jalalzai, M.H. Shahid, W.B. ; Iqbal, M.M.W. ”DNS security challenges and best practices to deploy secure DNS

with digital signatures." In 12th International Bhurban Conference on Applied Sciences and Technology (IBCAST), 2015.
[12] Sun Bin, Wen Qiaoyan, Liang Xiaoying . "A DNS Based Anti Phishing Approach". In IInd International Conference on Networks Security, Wireless Communications and Trusted Computing ,IEEE, 2010.
[13] Firenet.(October, 2008). DNS / nslookup - How to find the root servers ? Online. Available :https://www.fir3net.com/Networking/Protocols/dnsnslookup-how-to-find-the-root-servers.html. ( December,2015).
[14] Infoblox. (October, 2015). Online. Available : http://cybersecuritysummit.co.uk/wpcontent/uploads/2015/10/Top-Ten-DNS-Attacks. (Accessed: December, 2015).
[15] Breiman, Leo. "Random forests." Machine learning 45.1 pages 5-32,2001.
[16] Nejad, Tayebeh Rouhani and Abadi, Mohammadebrahim Shiri Ahmad."Intrusion detection in computer networks through a hybrid approach of data mining and decision trees."2014.
[17] Chan, Jonathan Cheung-Wai and Paelinckx, Desire "Evaluation of Random Forest and Adaboost tree-based ensemble classication and spectral band selection for ecotope mapping using airborne hyperspectral imagery." Remote Sensing of Environment 112.6, pp. 2999-3011, 2008.
[18] Jaree Thongkam, Guandong Xu, and Yanchun Zhang. "Adaboost algorithm withrandom forests for predicting breast cancer survivability. "In International Joint Conference on Neural Networks (IJCNN), pp 3062-3069,IEEE,2008.
[19] Thomas G Dietterich. "An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization" Machine learning, 40(2):139-157, 2000.