International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
Article Received: *25 July 2023* ***Revised:*** *12 September 2023* ***Accepted:*** *30 October 2023*

Fast Flux Domain Detection Using DNS Traffic

Mohammad Akaram[1*], Faizan Ahamad[2], Shafiqul Abidin[3]

[1*,2,3] Department of Computer Science, Aligarh Muslim UniversityAligarh, India
[1] mohdakaram8266@gmail.com
[2] faizan15august@gmail.com
[3] shafiqulabidin@yahoo.co.in

Abstract—There are many attacks possible that affect the services of DNS server, one such type of attack is Distributed Denial of Service (DDoS). So to avoid such attacks, DNS servers use various types of techniques like load balancing, Round Robin DNS, Content Distribution Networks, etc. But cybercriminals use these techniques to hide their actual and network location from the outside world. One such type of technique is Fast-Flux Service Networks, which is like proxies to the cybercriminals that makes them untraceable. FFSN is a major threat to internet security and used in many illegal scams like phishing websites, malware delivery, illegal adult content, and etc. Fast flux service networks have some limitation as attackers do not have control over the compromised PC's physically.
For the detection of FFSN, broadly two approaches have been proposed, namely, (i) Using passive network traffic, and (ii) Using active network traffic. The problem of detection with active network traffic is that they predict CDN domain as FFSN domain because initially, FFSN looks like CDN. Further, there are many machine learning algorithms have been used to detect FFSN. In this research, we emphasize on two problems, namely, (i) Features used for detecting the FFSN which helps us to distinguish FFSN from the other network efficiently, and (ii) Find the best classifier for detection of FFSN.
This work shows how relevant features extracted from the network traffic help us to distinguish FFSN from benign domains. Further, we try to propose the best threshold values for each feature that efficiently detect FFSN while distinguishing it from other benign domains. In this work, we have used five different machine learning algorithms, namely, Decision Tree, Random Forest, SVM, KNN, and Boosted Tree. Then, we compare the performance of these five machine learning algorithms to find out which is the best one to detect fast flux domain from passive DNS network traffic.

Keywords—RRDNS, CDN, Fast-Flux domains, Botnets, Machine learning Classifiers.

I. INTRODUCTION

In order to service the DNS query request made by a user, DNS servers must be available all the time to process the request and should be fast enough. But due to various attacks possible on DNS, its functionality can be compromised. One of the possible attacks is the DDoS attack. These attacks can be minimized through various ways like using RRDNS [1], CDNs [2] or Fast Flux service networks. What they actually do is, just distribute the load of a particular DNS server to many identical DNS servers (i.e, different physical machines). These technologies are beneficial for the legal organizations but some illegal organizations use these for their own interest. One of them is Fast Flux service networks that use frequently changing IP's, which are associated with a particular domain, because of that it is very hard to find the location of the attacker. Fast Flux service network is a well-known threat, which helps the attacker not to disclose its physical or network location.The basic idea behind Fast Flux is that there are many IP addresses associated with a single domain name, which get changed frequently, so that the actual location of the attacker cannot be known. Fast Flux Service Networks are basically of two types Single Flux Service Networks and Double Flux Service Networks [3]. In case of the Single Flux Service Networks, only IP addresses of the domain name changes, whereas in the case of Double Flux Service Networks IP address of nameservers also gets changed which provide an extra layer of abstraction to the attacker.

An attacker may create a large botnet by infecting the users with malware that then act as bots. Generally, the Fast-Flux "mothership" (same as command and control, i.e, C&C systems found in conventional botnets) controls the bots in the fast-flux service networks which are kept behind the proxy servers or VPNs, because of which tracing of the network location of the fast flux domain is hard. Most of the time users do not even know that they are infected and part of the Fast Flux Service Network.
In Fast-Flux Service Networks, generally, domains have very low TTL value because of that in a single session user makes a request with many bots in the botnet. Honeynet Project was first to uncover Fast-Flux Service Networks, they observed that fast flux domains keep changing IP's every 3 minutes that means if a user is connected to a site more than 3 minutes, he/she actually connects with different compromised PCs after every 3 minutes [4]. If someone wants to shut down a fast flux domain then he must shut down all the IP addresses referenced to that domain because if at least one of the IP addresses returned in DNS response is reachable, the whole scam is working. But in reality, it is not feasible to block all the compromised user's IP addresses as these user's do not have any idea that they indulged in illegal activities. Therefore, it is hard to locate the actual location of the attacker that makes the Fast-Flux Service Networks most threatening attack in today's internet world.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
Article Received: *25 July 2023* ***Revised:*** *12 September 2023* ***Accepted:*** *30 October 2023*

II. TECHNICAL BACKGROUND

A. RRDNS

Round Robin DNS is a technique used for load balancing, fault tolerance, load distribution for DNS servers. As other load balancing techniques depend on different physical systems, this technique uses the DNS server to achieve the power balancing. In this technique multiple IP addresses are given to DNS server such that IP addresses are rotated with respect to the users, i.e, one IP address is given to one user next will be given to next user and so on until the IP addresses are over thereafter it starts again with the first IP address from the pool of IP addresses. That means every single DNS record contains multiples answers to the DNS queries, i.e, multiple IP addresses are associated with a single domain name. Because of that load will be distributed among different DNS servers which provide the same service. The time period of rotation depends on the TTL value of DNS record, lower the TTL value, faster these IP addresses get rotated. But there is a drawback of lowering the TTL value as the load on DNS server will increase. RRDNS real-life use comes where the companies have heavy traffic. It is used for other services as well such as FTP, mail server, etc. Most of the enterprises have multiple mail servers which use round-robin fashion to handle the traffic efficiently. Round Robin DNS is simple to use and implement but it has some drawbacks also, like record caching in DNS hierarchy itself as well as client-side addresses caching and reuse, the combination of which can be difficult to manage [1].

For example, in Fig. 1, when a user wants A records for amazon.in domain name, he/she gets 3 IP addresses which are mapped with amazon.in domain. When the user again tries to access that domain, he/she gets the same set of IP addresses but their order is not same as previous, IP addresses get ordered according to Round Robin Fashion.

```
$ host -t A amazon.in
amazon.in has address 54.239.33.92
amazon.in has address 52.95.116.115
amazon.in has address 52.95.120.67
$ host -t A amazon.in
amazon.in has address 52.95.120.67
amazon.in has address 54.239.33.92
amazon.in has address 52.95.116.115
$ host -t A amazon.in
amazon.in has address 52.95.116.115
amazon.in has address 52.95.120.67
amazon.in has address 54.239.33.92
```

Fig. 1. RRDNS example

B. CDN

Content Distribution Network or Content Delivery Network, as its name suggests, takes the contents (i.e, web pages or DNS servers) and distributes them across the globe which makes the service provided by the servers, faster. It is designed to reduce the network latency, means it removes unnecessary traffic from the internet which flows from long distances by putting CDN servers to different geographical locations so that user get same content but from its nearest CDN server instead of the main deployment of DNS server. In Fig.2, for example, suppose a user in India try to access US-based website so he has to contact the server which resides in the US, but in this case, the network latency increases, as the query has to travel all along from India to US. Instead of which, with the help of CDN servers we provide same content to the user near to its geographical location by deploying multiple servers across the globe. So basically CDN servers analyze the packet and send the query to fastest and nearest DNS server.

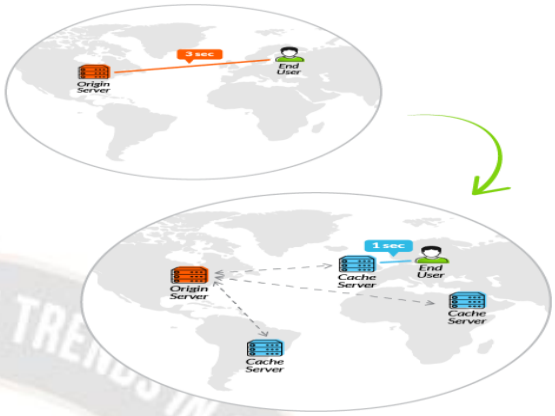


Fig. 2. CDN Network [5]

It not only reduces the time to reach the desired content but also reduces the number of hops the packet made so that load on the intermediate routers will be reduced. Fig. 3, shows a real life example of CDN network domain name.

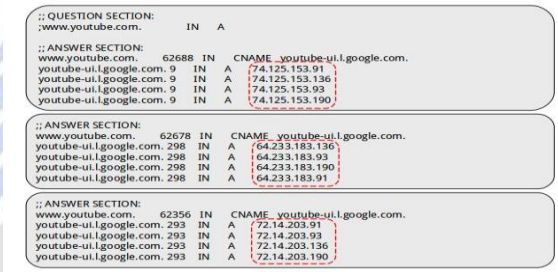


Fig. 3. content distribution network example [5]

C. Botnet

As its name suggests, bot mean compromised systems and net means network. A botnet is a collection of Internet-connected devices that are controlled by the attacker which may include IP phones, printers, PC's, mobile devices, IP cameras, servers, etc. Originally, botnets were designed for IRC channels but attackers exploit vulnerabilities in IRC networks and developed bots to carry out malicious activities [6]. Botnets are used in many illegal internet activities such as click fraud, ad fraud, data theft, DDoS attacks [7], cryptocurrency mining [8], keystroke logging [9], send spam content, etc. Attackers do not target any individual system to attack, they simply send malware to such vulnerable devices that do not have proper security implementation, i.e, outdated security products, such as antivirus software or firewalls [10]. The interesting thing is in most of the cases, victims of botnet are not aware of that they are part of some botnet. Infected systems are controlled remotely by a C&C Server [11] which keeps track of things like which bot is online, which bot has high availability and bandwidth. Command and control server send instructions to bots to perform an attack on behalf of it.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
Article Received: *25 July 2023* ***Revised:*** *12 September 2023* ***Accepted:*** *30 October 2023*

BOTNET ARCHITECTURE

In recent years, attacks of malware have evolved in organized and such a way that it easily evade disruption and detection. Attackers use various approaches to establish their botnet such as client-server model, peer to peer model,hybrid model. They all contain same botnet components but in different ways.

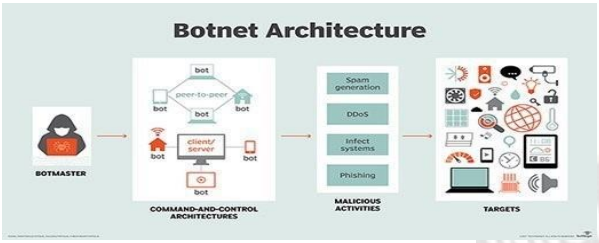


Fig. 4. Botnet architecture [10]

D. Fast Flux Service Networks

Fast Flux Service Networks are used by cyber-criminals to hide the malicious content hosting websites behind the network, which is changing it's IP frequently [12]. Fast Flux Service Networks provide robustness as well as increase lifespan of malicious content hosting websites. Fast Flux Service Network is more like CDN but the difference is that in CDN the administrator have the control to choose which IP addresses should return in response of DNS query, whereas in case of Fast Flux Service Network attacker do not have such privilege as he/she can not predict the availability of infected system, i.e, when a particular system will come online [13]. CDN is used for legal activities, whereas cyber-criminals use Fast Flux Service Network for illegal activities using bots [14]. Multiple IP addresses are mapped with a single malicious website such that after the expiration of every TTL value, new IP addresses are reflected in the DNS response which kept changing from the pool of IP addresses of the botnet, which make hard to track the actual IP addresses and location of the malicious website. Storm Worm [15][16] is the first known malware to use this technique. In case of Storm Worm, TTL value is very low 0 or less than 2 seconds, so that user has to make the request again and again after every 2 seconds to DNS server and every time it will connect with different infected PC.

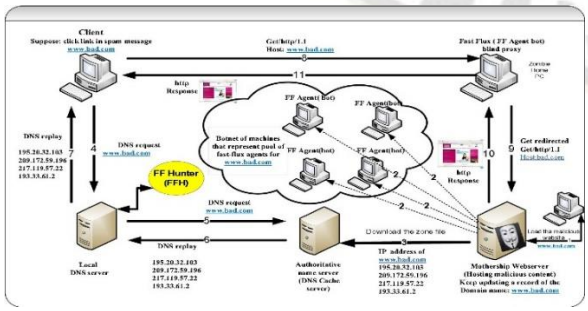


Fig. 5. fast flux service network [17]

For example, Fig.5, explains working of Fast Flux Service Network, first attacker host the malicious website on the mothership who checks which bots are active at that time and gathered there information, i.e, IP addresses. Mothership sends some IP addresses of these bots or fast flux agents to the Authoritative Nameservers, so when a user wants to access the malicious website, local DNS server requests to Authoritative DNS servers to get the IP addresses of a malicious website. When the user gets some IP addresses mapped to that malicious website, he/she contact one of the fast flux agents which then contact mothership for the webpage. As in the case of Fast Flux Service Network, TTL value of DNS resource records is low so the user has to make the request again to Authoritative Nameservers which now give a different set of IP addresses of fast flux agents as returned by the mothership. IP flux and Domain flux are two ways, which are used in two types of Fast Flux Service Networks, i.e, in Single Flux and Double Flux. In case of IP Flux, multiple IP addresses are associated with a single domain name, whereas Domain Flux is a technique used by the malicious botnet to keep operated by frequently changing domain names of botnet owner's mothership (or Command and Control Server) [10]. Bots use an algorithm like DGA to generate several random domain names for their Command and Control Server.

III. PROBLEM STATEMENT

Many approaches have been proposed for the fast flux domain detection using various machine learning algorithms with the help of various features. The main drawback of these approaches is their accuracy in detection FFSN. So, I have proposed an approach which uses relevant features for detection of FFSN using five machine learning algorithms, and compares them to find the best classifier among all the classifiers.

IV. LITERATURE SURVEY

Generally, fast flux domain keep its TTL value very low around 3-10 minutes so that the IP can change very frequently. But by analyzing DNS traffic, I came to know that there are some legitimate sites which also act like Fast Flux service networks like google.com, etc. But there is a difference between the legitimate domain and non-legitimate domain [18]. Whenever we get a response from a legitimate domain, we do not get diverse IP addresses, i.e, physical location of these IP's are not far from each other whereas when we get a response from a non-legitimate domain, IP addresses be very diverse and unique. Also, the number of unique IP addresses for a non-legitimate domain is very high around thousand or even lakhs [19]. In some cases like Storm Worm attack, TTL is very less (2 sec or 0) so that DNS queries were made very frequently.

- T. Holz, C. Gorecki, K. Rieck, and F. C. Freiling, 2008[20], they have presented a study in which they demonstrated the importance of threat that FFSN possesses. They formulate a metric to detect Fast Flux service networks threat effectively and suggested some strategies that can be helpful to mitigate Fast Flux service network.
- Jose Nazario and Thorsten Holz, 2008[21], suggested some heuristics to qualify a domain name as a Fast Flux domain. They have shown the fast flux service network behaviour over the network, such as fast flux domains are mostly

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 October 2023*

dormant during there lifetime. Top level TLDs are mostly used in the fast flux service network as suggested by them which is not seen in previous works. They have shown that it is possible to differentiates between individual botnets using fast flux techniques using DNS data mining

Scott Campbell, Stephen Chan and Jason R.Lee, 2011[22], suggested a different approach which is helpful in distinguishing between CDN and FFSNs to some point.
In this, they measure the total number of IP addresses over some amount of time for a given domain name. To make a base class, the set of the IP addresses which are responded in the first query for a domain name are taken. After that other A records which are recorded from the various different DNS responses are compared to the base class if any of the IP address is different from the base class, the counter increase. Then they calculate the metric to find out the Fast Flux domain names from the normal network.

- Z.Berkay Celik and Sema Oktug, 2013[23], detected fast flux networks on the basis of various DNS features categorized into classes, i.e, DNS answer-based, domain name based, spatial based, network-based, timing based. They used Decision Tree classifier for the classification of their dataset using total 19 features for detection of fast-flux networks. As per their study, spatial based features are best to detect fast flux domains, followed by network and DNS answer based features.
- Xiangzhan Yu, Bo Zhang, Le Kang and Juan Chen, 2012[24], used weighted support vector machine as a classifier for the detection of fast flux domain from the normal network domain dataset. They have used six different features for the classification. They have compared linear classification result with the result of SVM classification. As per their study, SVM is good enough for detecting fast flux domain from the normal network data.
- Xin Hu, Matthew Knysz, Kang G. Shin, 2011[25], created a probe engine to detect fast flux domain, called as DIGGER. For the classification of the dataset, they have used seven features for identification of fast flux domain. Their work focused on the DNS queries which comes from four different geographical locations by creating 240 vantage points.
- A.F.A. Kadir, R.A.R Othman, and N.A. Aziz, 2012[26], proposed to dissect and imagine the conduct of FFSN to encourage FFSN detection. They gather, characterize and screen over 500 spaces and by investigating and imagining the prepared data, they find the new sorts fluxing designated as NS-Name-Flux(NF). The investigation consequences of NF uncovered that FFSN have turned out to be widely advanced and dynamic. This represents that perception is an option and viable information investigation technique for understanding the complex practices of FFSN. For the classification, they have used kNN classifier using 7 features classified in three different classes namely, Benign, Single Flux, and Double Flux.
- R. Perdisci, I. Corona, D. Dagon, and W. Lee, 2009[27], work is based on the recursive DNS traffic traces. They have collected network data on a very large scale so that for more fine analysis, they clustered DNS traffic on the basis of the same Content Distribution Network or Internet Service or some malicious fast-flux service network. For classification, they have used C4.5 decision tree classifier using twelve different features.
- Erik Poll, Harald Vranken, SiccoVerwer, Barry Weymes, 2013[28], developed a proof-of-concept ready to naturally break down what's more, recognize botnet action utilizing pDNS information. They suggested the use of machine learning techniques namely Random Forests, k-Nearest Neighbours(kNN), and Decision Trees.
- Toni G., Darko P., Marko M., Filip V. and Tibor K., 2014[29], demonstrated a method to detect Fast Flux called CROFlux that depends on the detached DNS replication strategy. The exhibited model can fundamentally decrease the quantity of false positive location, and can recognize different suspicious spaces that are utilized for fast flux.
- Dinh-Tu Truong and Guang Cheng, 2016[30], presented a strategy that dissects DNS traffic to remove the length and the expected value, which can recognize a domain name produced by humans or bots. They have used various machine learning algorithms are applied to train predictive models for their detection system. In order to evaluate the adequacy of the proposed method.

All these related work use some features and machine learning algorithm to detect FFSN from the network and tried to keep the false positive and false negative result as low as possible with high accuracy. So in this work, we propose some features which detect FFSN efficiently and use five machine learning algorithms for classification while keeping the false positive and false negative as low as possible with high accuracy which is not achieved in previous proposed approaches.

V. PROPOSED MODEL

Fig. 6, shows the proposed model for the detection of Fast Flux domain names based on various machine learning algorithms using DNS network data. Each step of proposed model is explained in further sections.

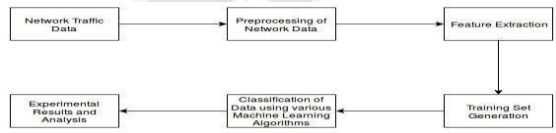


Fig.6. fast flux domain detection model

A. NETWORK TRAFFIC DATA

The first task to accomplish in our work is to collect the network traffic data, which should contain both the benign as well as fast flux domains. Also, the dataset should be retrieved from a trusted source, which can ensure the data correctness and authenticity. We have collected the network traffic dataset from the CAIDA (Center for Applied Internet Data Analysis), which conducts network research and builds research infrastructure to support large-scale data collection [31]. For the verification of benign domains, we use dataset collected from most visited websites on Google [32] and domains from Alexa [33], and for the verification of fast flux domains we use a list of malicious domains from DNS-BH [34], which maintains a list of all malicious domains collected from various

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 October 2023*

sources. Data which we collected is raw DNS traffic captured over the network. In Fig.7, we can see the contents of DNS message. Each packet in network traffic contains all this information, so we need to fetch only required data from it.

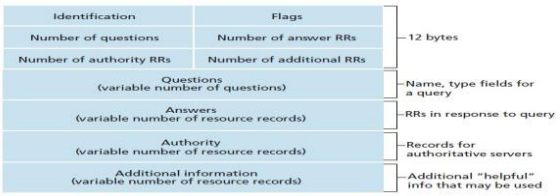


Fig. 7. DNS Message

B. DATA PRE-PROCESSING

Network traffic not only contains DNS traffic but other information as well. Removing this information from the network traffic is not enough as there are some packets in DNS traffic which are useless in this work. Conditions on which data filtration is done are described below:

- Packets other than DNS messages are removed.
- DNS packet which contain response are taken into account, rest are discarded.
- Query type of DNS packet must be A records only, i.e, query must be of type IPv4 address.
- Response type of DNS packet must be A records only, i.e, response must be of type IPv4 address.
- DNS packets which contain empty answer sections are discarded.

TShark:- It is a command line version of Wireshark, which is used for analyzing a packet data [35]. It can be used to analyze the captured network traffic (i.e, pcap file) or to capture live packet data over the network.

Python: - It is an interpreted high-level programming language used for general purpose programming [36]. All the data filtration on network traffic is done using Tshark, which removes unnecessary data packets from the pcap files and then convert pcap file to text file for further processing. Python programming language is used for fetching needed fields from the whole DNS packet. The fields that are fetched for our work are TTL value, query domain name, number of questions, number of answer RRs, number of authority RRs, number of additional RRs, type of query, IP addresses mapped to the domain name, and IP addresses mapped to the NS.

1) Feature Extraction

All the features which we used in our work are listed in Table 1:

After pre-processing of the data, we need to create other features which are not directly collected from the data, like ASN value of IP address, the average distance between the IP address, the number of unique IP addresses in single DNS response A record and the number of unique IP addressing in single DNS response NS records.

WhoIsServer:-Whois [37] is query and response protocol which is used for querying databases that store information about the registered users like domain name, autonomous system number, IP addresses, etc. Fig. 8, shows a general query to get information for the IP address 8.8.8.8, which is the IP address for the Google public DNS server.

```
akshay@akkyAirborne ~ $ whois -h whois.cymru.com "-v 8.8.8.8"
Warning: RIPE flags used with a traditional server.
AS      | IP             | BGP Prefix          | CC | Registry | Allocated  | AS Name
15169   | 8.8.8.8        | 8.8.8.0/24          | US | arin     | 1992-12-01 | GOOGLE - Google LLC, US
```

Fig. 8. whoisquerry to retrieve ASN value

Python language is used for further feature extraction. All the available fields are given as input to python script, which then calculates remaining features for each section of DNS message, i.e, for A records and for NS records.

MySql:- It is a relational database management system.

The pre-processed data is further divided into two sets, one for A records and another one for NS records. By analyzing the features from A records, we can conclude whether a domain is single fast flux domain or not. Similarly, from NS records we can conclude whether a domain is single or double fast flux domain. Multiple database tables are then created for the creation of features from A records and NS records. Fig. 9, shows features which we retrieved and calculated.

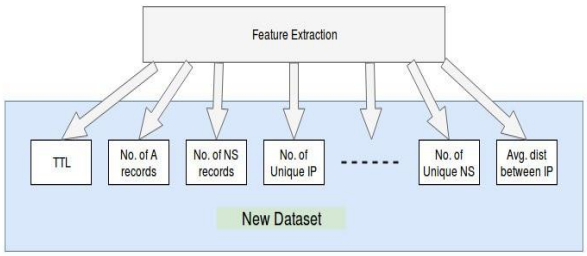


Fig. 9. Feature Extraction

Table 1. Feature description

Variables	Description
domain	Domain Name of DNS query
ttl	TTL value of each packet
ipavg	Average distance between IP addresses of A records
ipcount	Number of IP addresses in a single A records
ipasn	Number of ASN associated with IP addresses

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 October 2023*

	in a single A records
nsavg	Average distance between IP addresses of NS records
nscount	Number of IP addresses in a single NS record
nsasn	Number of ASN associated with IP addresses in a single NS record
unique_ip	Total number of unique IP addresses associated with a single domain name
ns unique ip	Total number of unique nameservers associated with a single domain name
asnip	Total number of ASN's associated with IP addresses mapped to a single domain name
asnns	Total number of ASN's associated with nameservers IP addresses mapped to a single domain name

2) Training Set Generation

Generation of the training set is done using AWK processing language. All the data stored in multiple database tables are then merged to form a single dataset for classification. Labeling of each record is done with the help of AWK language

AWK:- It is a Linux/Unix text processing language that searches and substitutes text or does database sort/validate/index. AWK is also used for pattern matching process. It is a very powerful and simple to use programming language.

The fast flux domain which we get from the dataset must meet these following requirements:

- The TTL value of A record should be less than 10 minutes (i.e, 600 seconds). If it is less than 2 second, then it is surely a Storm Worm attack in which user has to make query repeatedly in contrast to which he gets unique IP addresses every time.
- The number of unique IP addresses in a single record of DNS response should be greater than 5.
- The average distance between the IP addresses returned in DNS response A record should be more than /16 netblocks, i.e, 65,535 addresses.
- The number of the unique ASNs in a single DNS response A record should be more than 2.
- The number of unique NS records in a single DNS response should be more than 4.
- The average distance between the IP addresses returned in DNS response NS record should be more than /16 netblocks, i.e, 65,535 addresses.
- The nameservers must belong to more than 2 distinct ASNs.
- Total number of IP addresses corresponding to a single domain name should be more than 200.
- Total number of nameservers corresponding to a single domain name should be more than 20.
- Total number of unique ASN for A records corresponding to a single domain name should be more than 10.
- Total number of unique ASN for NS records corresponding to a single domain name should be more than 10.

If any record satisfies more than four requirements from the above, then it is labeled as a Fast Flux domain. The step by step process for training set generation is described below:

- Retrieve the data from the MySql database from various tables.
- Merge those tables with the help of AWK language and create a new dataset comprising of all features.
- Label each record using AWK language according to the requirements.
- Convert this data into CSV file.

This training set is used for the classification using various machine learning algorithms. The classifiers used in this work is explained in the next section.

C. Classification UsingVarious Machine Learning Algorithms

Classification of above training dataset is done using various supervised machine learning algorithms such as Decision Tree, Random Forest, Support vector machines, Boosted Tree, and KNN. These supervised machine learning algorithms are described below:

1 Algorithm: Decision Tree
Input: CSV file having labelled dataset
Output: Confusion matrix having classification result
1 Calculate the Entropy of set
2 for A_i >*number of attributes* do
3 for each attribute A_i
4 Split the set based on A_i
5 Calculate entropy after splitting the set

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 October 2023*

6 Calculate information gain
7 Information gain = entropy before splitting - entropy after splitting
8 end for calculate maximum information gain for attribute A_i 10 split the set based on the attribute A_i having maximum information gain 11 repeat this process until all subsets after splitting becomes pure.

2 Algorithm: Random Forest

Input: CSV file having labelled dataset
Output: Confusion matrix having classification result
1 Randomly select "k" features from total "m" features where k <m.
2 Among the "k" features, calculate the node "d" using the best split point.
3 Split the node into daughter nodes using the best split.
4 Repeat 1 to 3 steps until "l" number of nodes has been reached.
Build forest by repeating steps 1 to 4 for "n" number times to create "n" number of trees.

3 Algorithm: Boosted tree
Boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees.

4 Algorithm: SVM

Input: Training dataset, optional kernel function, hyperparameters,
Output: Trained SVM model.

1. Choose kernel function and hyperparameters.
2. identify support vectors, optimize hyperplane, Construct hyperplane to separate classes.
3. Classify new data based on position relative to hyperplane.
4. use kernel function for non-linear separation.
5. Optimize parameters for performance.
6. Assess model performance using metrics.

5 Algorithm: K-Nearest Neighbor

Input: CSV file having labelled dataset
Output: Confusion matrix having classification result 1 Determine the parameter value k.
2 Calculate the distance between the new object that needs to be classified with all objects in the training dataset.
3 Arrange computed distances in the ascending order and identify k nearest neighbors with the new object.
4 Take all the labels of the k neighbors selected above.
5 Based on the k labels that have been taken, the label that holds the majority will be assigned to the new object.

D. METRICES AND TERMINOLOGY

Metrics and terminology which are used in comparison study of results of different classifiers are discussed below:

Accuracy: It is the ratio of number of correct predictions to the total number of predictions. It simply predicts that how often a classifier make predictions right.

$$Accuracy = \frac{Correct\ prediction}{Total\ prediction}$$

- Confusion Matrix: Confusion Matrix or Confusion Table or Error Matrix is a table to visualize the performance of an algorithm. It shows correct and incorrect classification for each class in detailed way [38]. Confusion table is an SFrame consisting of three columns:

- target label: The label of ground truth.
- predicted label: The predicted label.
- count: The number of times target label was predicted as predicted label.
- Precision: The precision score or positive predictive value quantifies the ability of a classifier to not label a negative example as positive. The precision score can be interpreted as the probability that a positive prediction made by the classifier is positive.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

- Recall: Recall or Sensitivity is the ratio of correctly predicted positive observations to the all observations in actual class.

$$Recall = \frac{True\ Positive}{True\ Positive\ +\ False\ Positive}$$

- F1 score: it is a metric which combines both precision and recall via their harmonic mean.

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision\ +\ Recall}$$

- ROC Curve: Receiver Operating Characteristic is a graphical plot that illustrates the performance of a binary classifier system as its prediction threshold is varied. The ROC curve provides nuanced details about the behavior of the classifier. The curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings [39]. The result of the roc curve is a multicolumn SFrame with the following columns:

  - tpr: True positive rate, the number of true positives divided by the number of positives.
  - fpr: False positive rate, the number of false positives divided by the number of negatives.
  - p: Total number of positive values.
  - n: Total number of negative values.
  - class: Reference class for this ROC curve (for multi-class classification).

- AUC: Area under the curve used in classification analysis in order to determine which of the used model predict the classes best.

This explained detailed approach and features used to detect FFSN from network traffic as well as the metrics and terminology used to compare different classifiers.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
Article Received: *25 July 2023* ***Revised:*** *12 September 2023* ***Accepted:*** *30 October 2023*

VI. EXPERIMENTAL RESULTS AND ANALYSIS

All the experiments have been performed on systems having 8 GB RAM and Intel Core i7 processor. The implementation of the proposed model was carried out on Linux platform. Preprocessing of data was carried out using python, MySQL, and AWK language. Further, for classification we have used graphlab library in python. Experimental results consists classification results as well as the comparison of those classifications. Effective generation of the dataset through filtration and feature extraction from the network data will be discussed in further sections [40]. Next, classification using various machine learning classifiers will be discussed. The result of each machine learning classifier is then compared with each other to predict the best classifier.

A. Result of Data Preprocessing and Feature Extraction

As unwanted network traffic (i.e, other than DNS traffic) is present in the dataset, tshark is used for filtration of undesired data as well as for removing DNS packets which are not needed in this work such as DNS query packets, empty answer section DNS packet and other than A type records.

After the filtration of data, filtered data is stored in database tables for further processing and creation of features, which are not directly retrievable from DNS packet [41].

Fig. 10, shows the A records that contains the information such as the Frame Number of the DNS packet, Domain Name in the response packet, IP addresses corresponding to that Domain Name, ASN number for that IP addresses of the Domain Name and the integer value of that IP address.

```
mysql> select * from A_records limit 30;
+--------------+-----------------------+----------+-----------------+--------+---------------------+
| frame_number | domain_name           | ttl_value | ip_addresses    | asn    | ip_to_integer_value |
+--------------+-----------------------+----------+-----------------+--------+---------------------+
|            1 | tinnie.arin.net       |    43200 | 199.212.0.53    | 393225 |          3352559669 |
|            2 | ns1.apnic.net         |     3600 | 202.12.29.25    |   4608 |          3389791513 |
|            3 | ns3.apnic.net         |     3600 | 202.12.28.131   |   4777 |          3389791363 |
|            4 | ns4.apnic.net         |     3600 | 202.12.31.140   |  18366 |          3389792140 |
|            5 | sec1.authdns.ripe.net |     3600 | 193.0.9.3       | 197000 |          3238004995 |
|            6 | apnic1.dnsnode.net    |     7200 | 194.146.106.106 |   8674 |          3264375402 |
|            7 | ns2.dns.br            |   172800 | 200.192.232.53  |  11284 |          3368085557 |
|            8 | ns4.apnic.com         |     7200 | 202.12.31.140   |  18366 |          3389792140 |
|            9 | ns.isc.afilias-nst.info |    86400 | 199.254.63.254  |  12041 |          3355328510 |
|           10 | a.in-addr.cn          |    86400 | 203.119.25.3    |  24151 |          3413580035 |
|           11 | b.in-addr.cn          |    86400 | 203.119.26.3    |  24406 |          3413580291 |
|           12 | c.in-addr.cn          |    86400 | 203.119.27.3    |  24406 |          3413580547 |
|           13 | t.arin.net            |    43200 | 199.253.249.63  |  53535 |          3355310399 |
|           14 | w.arin.net            |    43200 | 72.52.71.2      |   6939 |          1211385602 |
|           15 | u.arin.net            |    43200 | 204.61.216.50   |     42 |          3426605106 |
|           16 | v.arin.net            |    43200 | 63.243.194.2    |   6453 |          1072939522 |
|           17 | x.arin.net            |    43200 | 199.71.0.63     | 393220 |          3343319103 |
|           18 | y.arin.net            |    43200 | 192.42.93.32    |  36618 |          3224001824 |
|           19 | z.arin.net            |    43200 | 199.212.0.63    | 393225 |          3352559679 |
|           20 | r.arin.net            |    43200 | 199.180.180.63  |  53535 |          3350580607 |
|           21 | a.in-addr.cn          |    86400 | 203.119.25.3    |  24151 |          3413580035 |
|           22 | b.in-addr.cn          |    86400 | 203.119.26.3    |  24406 |          3413580291 |
|           23 | c.in-addr.cn          |    86400 | 203.119.27.3    |  24406 |          3413580547 |
|           24 | ns.cstnet.cn          |    10800 | 159.226.8.6     |   7497 |          2682390534 |
|           25 | ns1.cstnet.cn         |    10800 | 159.226.8.28    |   7497 |          2682390556 |
|           26 | r.arin.net            |    43200 | 199.180.180.63  |  53535 |          3350580607 |
|           27 | t.arin.net            |    43200 | 199.253.249.63  |  53535 |          3355310399 |
|           28 | x.arin.net            |    43200 | 199.71.0.63     | 393220 |          3343319103 |
|           29 | y.arin.net            |    43200 | 192.42.93.32    |  36618 |          3224001824 |
|           30 | z.arin.net            |    43200 | 199.212.0.63    | 393225 |          3352559679 |
+--------------+-----------------------+----------+-----------------+--------+---------------------+
```

Fig. 10. A RECORDS

Fig. 11, shows the NS records that contains the information such as the Frame Number of the DNS packet, Domain Name in the response packet, Name Server corresponding to that Domain Name, IP addresses corresponding to that Name Server, ASN number for that IP addresses of the Name Server and the integer value of that IP address [42].

```
mysql> select * from NS_records limit 30;
+--------------+-----------------+-----------------------+-----------------+--------+---------------------+
| frame_number | domain_name     | ns_name               | ns_ip           | asn    | ip_to_integer_value |
+--------------+-----------------+-----------------------+-----------------+--------+---------------------+
|            1 | tinnie.arin.net | v.arin.net            | 63.243.194.2    |   6453 |          1072939522 |
|            1 | tinnie.arin.net | ns2.arin.net          | 199.71.0.100    | 393220 |          3343319140 |
|            1 | tinnie.arin.net | ns1.arin.net          | 199.212.0.100   | 393225 |          3352559724 |
|            1 | tinnie.arin.net | u.arin.net            | 204.61.216.50   |     42 |          3426605106 |
|            2 | ns1.apnic.net   | NULL                  | NULL            |   NULL |                   0 |
|            3 | ns3.apnic.net   | NULL                  | NULL            |   NULL |                   0 |
|            4 | ns4.apnic.net   | NULL                  | NULL            |   NULL |                   0 |
|            5 | sec1.authdns.ripe.net | prt.authdns.ripe.net | 193.0.9.5  | 197000 |          3238004997 |
|            6 | apnic1.dnsnode.net | slave.gbg.netnod.se | 192.36.133.116  |  39871 |          3223618932 |
|            6 | apnic1.dnsnode.net | slave.sth.netnod.se | 192.36.144.116  |   8674 |          3223621748 |
|            7 | ns2.dns.br      | a.dns.br              | 200.160.0.10    |  22548 |          3365928970 |
|            7 | ns2.dns.br      | b.dns.br              | 200.189.41.10   |  10906 |          3367040010 |
|            7 | ns2.dns.br      | c.dns.br              | 200.192.233.10  |  11284 |          3368085770 |
|            7 | ns2.dns.br      | d.dns.br              | 200.219.154.10  |  11644 |          3369835018 |
|            8 | ns4.apnic.com   | NULL                  | NULL            |   NULL |                   0 |
|            9 | ns.isc.afilias-nst.info | ns1.hkg1.afilias-nst.info | 65.22.6.1 | 12041 |          1091962369 |
|            9 | ns.isc.afilias-nst.info | ns1.ams1.afilias-nst.info | 65.22.6.79 | 12041 |          1091962447 |
|            9 | ns.isc.afilias-nst.info | ns1.mia1.afilias-nst.info | 65.22.7.1 | 12041 |          1091962625 |
|            9 | ns.isc.afilias-nst.info | ns1.sea1.afilias-nst.info | 65.22.8.1 | 12041 |          1091962881 |
|            9 | ns.isc.afilias-nst.info | ns1.yyz1.afilias-nst.info | 65.22.9.1 | 12041 |          1091963137 |
|           10 | a.in-addr.cn    | b.in-addr.cn          | 203.119.26.3    |  24406 |          3413580291 |
|           10 | a.in-addr.cn    | c.in-addr.cn          | 203.119.27.3    |  24406 |          3413580547 |
|           11 | b.in-addr.cn    | a.in-addr.cn          | 203.119.25.3    |  24151 |          3413580035 |
|           11 | b.in-addr.cn    | c.in-addr.cn          | 203.119.27.3    |  24406 |          3413580547 |
|           12 | c.in-addr.cn    | a.in-addr.cn          | 203.119.25.3    |  24151 |          3413580035 |
|           12 | c.in-addr.cn    | b.in-addr.cn          | 203.119.26.3    |  24406 |          3413580291 |
|           13 | t.arin.net      | v.arin.net            | 63.243.194.2    |   6453 |          1072939522 |
|           13 | t.arin.net      | ns2.arin.net          | 199.71.0.100    | 393220 |          3343319148 |
|           13 | t.arin.net      | ns1.arin.net          | 199.212.0.100   | 393225 |          3352559724 |
|           13 | t.arin.net      | u.arin.net            | 204.61.216.50   |     42 |          3426605106 |
+--------------+-----------------+-----------------------+-----------------+--------+---------------------+
```

Fig. 11. NS RECORDS

Fig. 12, contains the information like Frame Number of DNS packet, Domain Name in the response packet, the Average distance between IP addresses of the Domain Name in that single DNS response, Number of unique IP addresses in single DNS response, Number of ASNs associated with the IP addresses contained in a single DNS response [43].

```
mysql> select * from avg_dist where ip_count > 3 limit 20 ;
+--------------+-----------------------------+-----------+----------+-----------+
| frame_number | domain_name                 | avg_dist  | ip_count | asn_count |
+--------------+-----------------------------+-----------+----------+-----------+
|         2972 | ns1.hostnoc.net             |         1 |       10 |         1 |
|         2973 | ns2.hostnoc.net             |         6 |       10 |         1 |
|         3899 | NS1.HOSTNOC.NET             |         1 |       10 |         1 |
|         3900 | NS2.HOSTNOC.NET             |         6 |       10 |         1 |
|         6972 | ns1.hostnoc.net             |         1 |       10 |         1 |
|         6973 | ns2.hostnoc.net             |         6 |       10 |         1 |
|         7826 | dns1.registrar-servers.com  |        70 |        4 |         1 |
|         7830 | dns4.registrar-servers.com  |        70 |        4 |         1 |
|         9321 | ns1.hostnoc.net             |         1 |       10 |         1 |
|         9322 | ns2.hostnoc.net             |         6 |       10 |         1 |
|        10785 | vienna1-dns-auth1.bbnplanet.com | 573689748 |        4 |         4 |
|        11041 | ns.ati.tn                   |        69 |        4 |         1 |
|        12611 | ns.ati.tn                   |        69 |        4 |         1 |
|        13244 | ns1.boulnare.com            |        18 |        5 |         1 |
|        13245 | ns2.boulnare.com            |        17 |        5 |         1 |
|        13587 | starlet.deltatelecom.ru     |         1 |        4 |         1 |
|        14310 | NS1.HOSTNOC.NET             |         1 |       10 |         1 |
|        14311 | NS2.HOSTNOC.NET             |         6 |       10 |         1 |
|        14458 | ns1.dnsv2.com               | 414890002 |        4 |         3 |
|        14459 | ns2.dnsv2.com               | 297298599 |        4 |         4 |
+--------------+-----------------------------+-----------+----------+-----------+
```

Fig. 12. Average Distance between IP addresses of A records

If we query for average distance between IP addresses > 65,535 value, we get result like as shown in Fig. 13, containing 1,09,307 out of 1,63,93,313 DNS packets of A records.

```
mysql> select * from avg_dist where avg_dist > 65535;
+--------------+-----------------------+-----------+----------+-----------+
| frame_number | domain_name           | avg_dist  | ip_count | asn_count |
+--------------+-----------------------+-----------+----------+-----------+
|          384 | ns4.dns-diy.com       | 976843045 |        2 |         2 |
|          385 | ns3.dns-diy.com       | 518491090 |        2 |         2 |
|          659 | ns2.bouyguesbox.fr    | 882852921 |        2 |         1 |
|          660 | ns1.bouyguesbox.fr    | 882852793 |        2 |         1 |
|         1082 | nsb.domain-robot.org  | 1123786014 |        2 |         2 |
|         1086 | nsb6.schlundtech.de   | 1123786004 |        2 |         2 |
|         1192 | ns0.ja.net            | 544517760 |        2 |         1 |
|         1333 | nsb.domain-robot.org  | 1123786014 |        2 |         2 |
|         1337 | nsb6.schlundtech.de   | 1123786004 |        2 |         2 |
|         1399 | ns0.ja.net            | 544517760 |        2 |         1 |
|     19049254 | ayw001.cm.ndmc.ne.jp  | 140421865 |        2 |         1 |
|     19049258 | ayw001.cm.ndmc.ne.jp  | 140421865 |        2 |         1 |
|     19049519 | s1.ccu.edu.tw         | 189837159 |        2 |         1 |
|     19049746 | ns2.htudns.com        | 1007444800 |        3 |         2 |
|     19049747 | ns1.htudns.com        | 789594774 |        3 |         2 |
|     19049972 | ns0.ja.net            | 544517760 |        2 |         1 |
|     19049979 | ns0.ja.net            | 544517760 |        2 |         1 |
|     19051334 | dns1.hichina.com      | 443460864 |        3 |         1 |
|     19051335 | dns2.hichina.com      | 443460864 |        3 |         1 |
|     19051846 | dns11.hichina.com     | 443460864 |        3 |         1 |
|     19051847 | dns12.hichina.com     | 443460864 |        3 |         1 |
|     19052244 | dns.wki.it            | 740409600 |        3 |         3 |
|     19052248 | dns.ipsoa.it          | 1110614400 |        2 |         2 |
|     19052255 | ns2.powername.at      | 1123786014 |        2 |         2 |
|     19052258 | ns2.powername.at      | 1123786014 |        2 |         2 |
+--------------+-----------------------+-----------+----------+-----------+
109307 rows in set (1 min 45.35 sec)
```

Fig. 13. Average Distance between IP addresses for A records > 65,535

Fig. 14, contains the information like Frame Number of DNS packet, Domain Name in the response packet, the Average distance between IP addresses of the Name Server in that

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
Article Received: *25 July 2023* ***Revised:*** *12 September 2023* ***Accepted:*** *30 October 2023*

single DNS response, Number of unique IP addresses in single DNS response, Number of ASNs associated with the IPs contained in a single DNS response.

```
mysql> select * from ns_avg_dist limit 20;
+--------------+-----------------------------+----------+----------+-----------+
| frame_number | domain_name                 | avg_dist | ip_count | asn_count |
+--------------+-----------------------------+----------+----------+-----------+
|            1 | tinnie.arin.net             | 588416396 |        4 |         4 |
|            2 | ns1.apnic.net               |     NULL |        0 |         0 |
|            3 | ns3.apnic.net               |     NULL |        0 |         0 |
|            4 | ns4.apnic.net               |     NULL |        0 |         0 |
|            5 | sec1.authdns.ripe.net       |        0 |        1 |         1 |
|            6 | apnic1.dnsnode.net          |     1408 |        2 |         2 |
|            7 | ns2.dns.br                  |   976512 |        4 |         4 |
|            8 | ns4.apnic.com               |     NULL |        0 |         0 |
|            9 | ns.isc.afillas-nst.info     |      154 |        5 |         1 |
|           10 | a.in-addr.cn                |      128 |        2 |         1 |
|           11 | b.in-addr.cn                |      256 |        2 |         2 |
|           12 | c.in-addr.cn                |      128 |        2 |         2 |
|           13 | t.arin.net                  | 588416396 |        4 |         4 |
|           14 | w.arin.net                  | 588416396 |        4 |         4 |
|           15 | u.arin.net                  | 759873401 |        3 |         3 |
|           16 | v.arin.net                  |  27761986 |        3 |         3 |
|           17 | x.arin.net                  | 588416396 |        4 |         4 |
|           18 | y.arin.net                  | 588416396 |        4 |         4 |
|           19 | z.arin.net                  | 588416396 |        4 |         4 |
|           20 | r.arin.net                  | 588416396 |        4 |         4 |
+--------------+-----------------------------+----------+----------+-----------+
```

Fig. 14. Average Distance between IP addresses of NS records

If we query for average distance between IP addresses > 65,535 value, we get a result like as shown in Fig. 15, containing 36,84,400 out of 1,63,93,313 DNS packets of NS records.

```
mysql> select * from ns_avg_dist where avg_dist > 65535;
+--------------+-----------------+----------+----------+-----------+
| frame_number | domain_name     | avg_dist | ip_count | asn_count |
+--------------+-----------------+----------+----------+-----------+
|            1 | tinnie.arin.net | 588416396 |        4 |         4 |
|            7 | ns2.dns.br      |   976512 |        4 |         4 |
|           13 | t.arin.net      | 588416396 |        4 |         4 |
|           14 | w.arin.net      | 588416396 |        4 |         4 |
|           15 | u.arin.net      | 759873401 |        3 |         3 |
|           16 | v.arin.net      |  27761986 |        3 |         3 |
|           17 | x.arin.net      | 588416396 |        4 |         4 |
|           18 | y.arin.net      | 588416396 |        4 |         4 |
|           19 | z.arin.net      | 588416396 |        4 |         4 |
|           20 | r.arin.net      | 588416396 |        4 |         4 |
|           26 | r.arin.net      | 588416396 |        4 |         4 |
|     19052301 | ns1.EAPPS.COM   |  836147432 |        3 |         2 |
|     19052316 | ns1.hc.ru       |   81835155 |        2 |         2 |
|     19052317 | ns2.hc.ru       |   81835155 |        2 |         2 |
|     19052334 | ns5.verizon.net |  347549700 |        4 |         1 |
|     19052335 | ns6.verizon.net |  347549700 |        4 |         1 |
|     19052336 | ns7.verizon.net |  347549700 |        4 |         1 |
|     19052337 | ns8.verizon.net |  347549700 |        4 |         1 |
|     19052345 | ns1.verizon.net |  463399600 |        3 |         1 |
|     19052346 | ns3.verizon.net |  463396869 |        3 |         1 |
|     19052347 | ns2.verizon.net |   17077589 |        3 |         1 |
|     19052348 | ns1.celeste.fr  |   88389393 |        2 |         2 |
|     19052350 | ns3.celeste.fr  |   88389393 |        2 |         2 |
+--------------+-----------------+----------+----------+-----------+
3684400 rows in set (1 min 51.01 sec)
```

Fig. 15. Average Distance between IP addresses for NS records > 65,535

Number of unique IP addresses corresponding to a particular domain name in A records are shown in Fig. 16.

```
mysql> select * from unique_ip limit 20;
+-------------------------------------------------+-----------+
| domain_name                                     | unique_ip |
+-------------------------------------------------+-----------+
| 0.dal.soa.firstam.com                           |         1 |
| 0.ns.inpher.com                                 |         1 |
| 0.ns.orbis.net                                  |         1 |
| 0.ns.tetadomains.com                            |         1 |
| 0.ns.usmac.net                                  |         1 |
| 0.ns.vaengatelecom.ru                           |         1 |
| 0.ns.winlin-net.be                              |         1 |
| 0.rev.cs.ritsumei.ac.jp                         |         1 |
| 0.sna.soa.firstam.com                           |         1 |
| 0000mps.webpreview.dsl.net                      |        56 |
| 01.auth.nym1.appnexus.net                       |         1 |
| 01.auth.nym2.appnexus.net                       |         1 |
| 01.dnsv.jp                                      |         1 |
| 01.ns.dinfotec.com                              |         1 |
| 01.ns.hosting.infrastructure.virtuoso.net.au    |         1 |
| 010.giga.com.pl                                 |         1 |
| 01dns.mgmtnet.se                                |         1 |
| 01VMDNS.dr1.cnrs.fr                             |         1 |
| 02.dnsv.jp                                      |         1 |
| 02.ns.dinfotec.com                              |         1 |
+-------------------------------------------------+-----------+
```

Fig. 16. Number of unique IP addresses in A records

If we query for unique IP addresses > 200 for a particular Domain Name, we get result like shown in Fig. 17.

```
mysql> select * from unique_ip where unique_ip > 200;
+-------------------------------------------------+-----------+
| domain_name                                     | unique_ip |
+-------------------------------------------------+-----------+
| 07f2kuc2szyosycbteg7.littlematchagirl.com.au    |      1175 |
| 2070modern.com                                  |      1850 |
| 3mcot5epaf9fsphdeehn.littlematchagirl.com.au    |      2325 |
| acikdeniz-bireysel.com                          |       237 |
| ahbddp.com                                      |       232 |
| alchenomy.com                                   |       280 |
| aleidangroup.com                                |      1150 |
| alruno.home.pl                                  |       261 |
| bankofamerica.avantiinteractive.us              |      1450 |
| bgwxwjtzjn.786vf7ueyw.madpendesign.com.au       |      1675 |
| blacorindy245.ml                                |      1100 |
| brecobdirectintl.com                            |      3725 |
| bttucd7itf1vg41exelg.maherstcottage.com.au      |      1050 |
| comsumersview.com                               |       238 |
| condosguru.com                                  |       295 |
| coriew.club                                     |      1566 |
| cw0uzqfzmw61gslwho75.littlematchagirl.com.au    |      5125 |
| cxcw5naj81aoyydamzzw.littlematchagirl.com.au    |      2060 |
| cxwthqm5i0iplki1smyd.maherstcottage.com.au      |      1250 |
| yikesprospho.com.gridhosted.co.uk               |      2490 |
| yourfreezunehd.com                              |       226 |
| yvigoc1dulw7jtrufnas.maherstcottage.com.au      |      4340 |
+-------------------------------------------------+-----------+
80 rows in set (0.05 sec)
```

Fig. 17. Number of unique IP addresses for a particular domain name > 200 in A records

Number of unique IP addresses corresponding to a particular domain name in NS records are shown in Fig. 18.

```
mysql> select * from ns_unique_ip limit 20;
+-------------------------------------------------+-----------+
| domain_name                                     | unique_ip |
+-------------------------------------------------+-----------+
| 0.dal.soa.firstam.com                           |         1 |
| 0.ns.inpher.com                                 |         3 |
| 0.ns.orbis.net                                  |         2 |
| 0.ns.tetadomains.com                            |         0 |
| 0.ns.usmac.net                                  |         1 |
| 0.ns.vaengatelecom.ru                           |         1 |
| 0.ns.winlin-net.be                              |         3 |
| 0.rev.cs.ritsumei.ac.jp                         |         1 |
| 0.sna.soa.firstam.com                           |         1 |
| 0000mps.webpreview.dsl.net                      |        21 |
| 01.auth.nym1.appnexus.net                       |         3 |
| 01.auth.nym2.appnexus.net                       |         3 |
| 01.dnsv.jp                                      |         3 |
| 01.ns.dinfotec.com                              |         2 |
| 01.ns.hosting.infrastructure.virtuoso.net.au    |         0 |
| 010.giga.com.pl                                 |         0 |
| 01dns.mgmtnet.se                                |         0 |
| 01VMDNS.dr1.cnrs.fr                             |         2 |
| 02.dnsv.jp                                      |         3 |
| 02.ns.dinfotec.com                              |         2 |
+-------------------------------------------------+-----------+
```

Fig. 18. Number of unique IP addresses in NS records

If we query for unique IP addresses > 20 for a particular Domain Name, we get result like shown in Fig. 19.

```
mysql> select * from ns_unique_ip where unique_ip > 20;
+-------------------------------------------------+-----------+
| domain_name                                     | unique_ip |
+-------------------------------------------------+-----------+
| 0000mps.webpreview.dsl.net                      |        21 |
| 02zps9iimwo9gukluvz8.littlematchagirl.com.au    |        21 |
| 0jaqkuc24kdjvpgdc8va.littlematchagirl.com.au    |        32 |
| 0jaqkuc24kdjvpgdc8va.maherstcottage.com.au      |        22 |
| 1bkeldneetdiq5vkcfax.maherstcottage.com.au      |        21 |
| 3liennaak6.djc2tvnzza.madpendesign.com.au       |        23 |
| 3years.lethanon.net                             |        23 |
| 52flz.com                                       |        39 |
| 91kuyue.com                                     |        23 |
| 9ambgkopn1xawlzj7um8.maherstcottage.com.au      |        21 |
| abbyspanties.com                                |        21 |
| accessrealtyco.com                              |        21 |
| acusticjjw.pl                                   |        23 |
| adfrut.cl                                       |        21 |
| admission.lampangvc.ac.th                       |        21 |
| agroluftbild.de                                 |        21 |
| agwoan.com                                      |        24 |
| ahbddp.com                                      |        21 |
| aikman.tk                                       |        21 |
| airtyrant.com                                   |        24 |
| ako.org                                         |        21 |
| al-hilal.com.pk                                 |        21 |
| alchenomy.com                                   |        22 |
| alruno.home.pl                                  |        21 |
| amazonesecust.com.ref894909veri39.qicadasposcreks.com |        21 |
| zgopxgi9uqlgoioatuuc.littlematchagirl.com.au    |        29 |
| ziraatbankasi.ml                                |        21 |
| zotasinc.com                                    |        22 |
+-------------------------------------------------+-----------+
329 rows in set (0.05 sec)
```

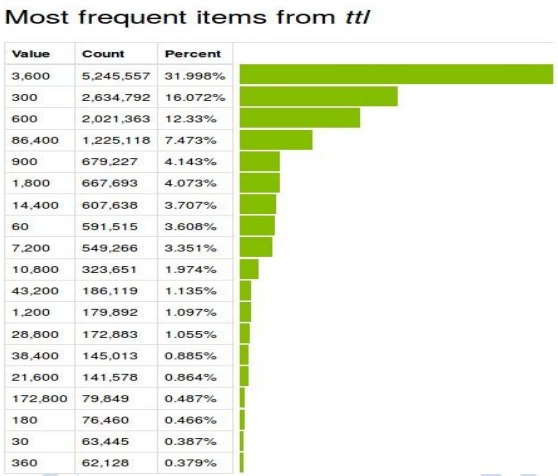
Fig. 19. Number of unique IP addresses for a particular domain name > 20 in NS records

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 October 2023*

B. Analysis Of Training Dataset

On analyzing the dataset, we get an idea about what should be the values of each feature which help us to label the data as benign or fast flux [44]. The threshold value for each feature is described below:

In case of fast flux network, TTL values of DNS resource records is very low, unlike the other benign domains. As we can see in Fig. 20, TTL value 600s is third most used TTL value for the DNS resource records and half of the network data has TTL value less than 600 seconds [45]. As many of the benign domain also has the TTL value.

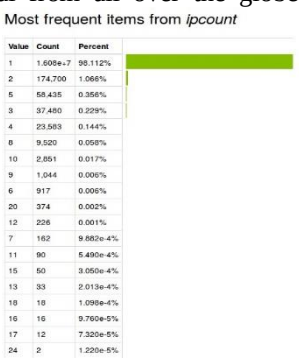


Value	Count	Percent
3,600	5,245,557	31.998%
300	2,634,792	16.072%
600	2,021,363	12.33%
86,400	1,225,118	7.473%
900	679,227	4.143%
1,800	667,693	4.073%
14,400	607,638	3.707%
60	591,515	3.608%
7,200	549,266	3.351%
10,800	323,651	1.974%
43,200	186,119	1.135%
1,200	179,892	1.097%
28,800	172,883	1.055%
38,400	145,013	0.885%
21,600	141,578	0.864%
172,800	79,849	0.487%
180	76,460	0.466%
30	63,445	0.387%
360	62,128	0.379%

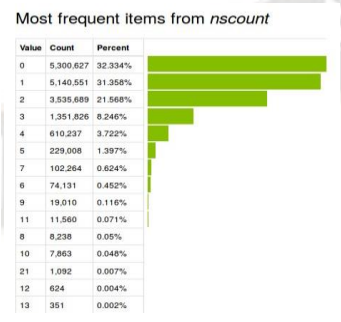
Fig. 20. Frequencies of TTL values

less than 600 seconds, other features are needed to label the dataset. Benign domains generally have less than 5 unique IP addresses reflected in a single DNS resource record but in case of Fast Flux domains, they have equal to or more than 5 unique IP addresses reflected in a single DNS resource record [46].

As we can see in Fig. 21a, most domains have unique IP addresses less than 5. Attackers do not have control over the physical situation of a bot, i.e, whether the machine is running or not. So bot master selects the IP addresses from the bots which are running and functional from all over the globe. Because of that IP addresses reflected in a single DNS resource record could belong to different geographical locations. From the Figure 21b, we can observe that most of the domains have less than 5 nameservers for a domain name in a single DNS resource record. In case of double flux network, nameservers keep changing in NS section of DNS resource record. Because of that number of nameservers also increase in a single DNS resource record for a domain name. Benign domain generally doesn't have more than 4 unique nameservers in a single DNS resource record.



Value	Count	Percent
1	1.605e+7	98.112%
2	174,700	1.068%
5	58,435	0.356%
3	37,480	0.229%
4	23,583	0.144%
8	9,520	0.058%
10	2,851	0.017%
9	1,044	0.006%
6	917	0.006%
20	374	0.002%
12	226	0.001%
7	162	9.882e-4%
11	90	5.490e-4%
15	50	3.050e-4%
13	33	2.013e-4%
18	18	1.098e-4%
16	16	9.760e-5%
17	12	7.320e-5%
24	2	1.220e-5%



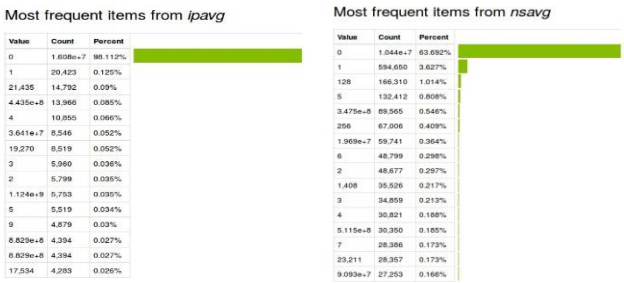
Value	Count	Percent
0	5,300,627	32.334%
1	5,140,551	31.358%
2	3,535,689	21.568%
3	1,351,826	8.246%
4	610,237	3.722%
5	229,008	1.397%
7	102,264	0.624%
6	74,131	0.452%
9	19,010	0.116%
11	11,560	0.071%
8	8,238	0.05%
10	7,863	0.048%
21	1,092	0.007%
12	624	0.004%
13	351	0.002%

(a) For A records (b) For NS record

Fig. 21. Count of unique IP addresses in a Single DNS resource record

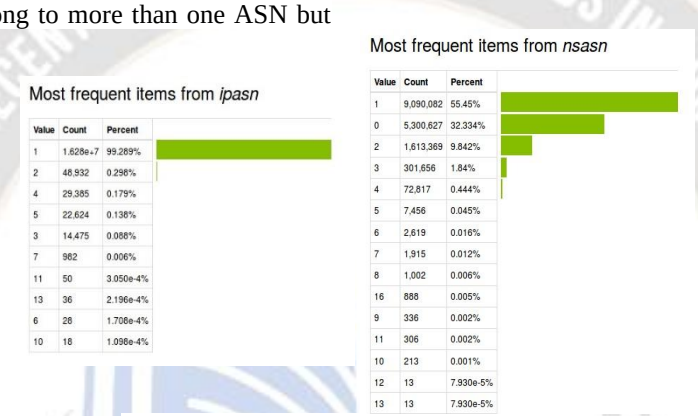
From Fig. 22, we can observe that in most of the cases the average distance between IP addresses is less than 65,535 addresses, i.e, less than /16 netblocks. Fig. 22a, shows that most of the domain doesn't have average IP distance more than 65,535 and Fig. 22b, shows that most of the benign domains have an average distance less than 65,535 addresses between nameservers IP addresses.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 October 2023*



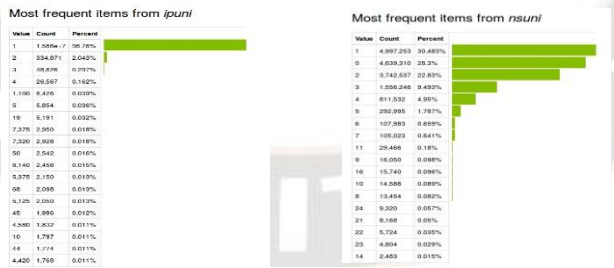
(a) For A Records (b) For NS Records
Fig. 22. Average Distance between IP addresses

In Fig. 23a, we can see that most of IP addresses have only one ASN that means they belong from same Autonomous System. As in the case of CDN domains, there may be a possibility of IP addresses belong to more than one ASN but generally not more than 2. In case of double flux network number of ASN is more for NS records. From Fig. 23b, we can see that most of the nameservers have 2 or less unique ASN associated with IP addresses.



(a) For A Record (b) For NS Record
Fig. 23. Number of unique ASN in a single DNS resource record

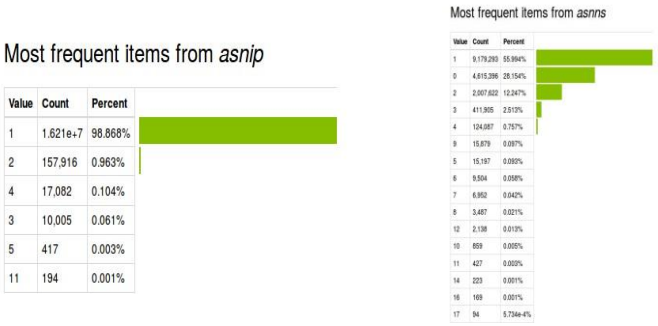
Benign domains do not have many unique IP addresses associated with them, generally. But in case of FFSN, there are many unique IP addresses associated with domain names. We have taken 200 as the threshold value for the total number of unique IP addresses reflected for a single domain name. Fig. 24, shows the number of count of unique IP addresses associated with domain names as well as their nameservers.



(a) For A record (b) For NS record
Fig. 24. Number of unique IP addresses in all DNS resource record

For the identification of double flux domain out of the total network we have taken into account the total number of ASN reflected for the domain name as well as for their nameservers. Fig. 25, shows the total number of ASN for A record and for NS records.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 October 2023*



(a) For a domain name (b) For a Name server

Fig. 25. Total number of ASN

C. Comparison Of Classification Results

In this section, we compared results of each classifiers using some metrics like Accuracy, Precision, Recall, F1-score, and AUC curve. As we can see from Table 2, boosted tree classifier outperformed every other classifier in each comparison metric.

Table 2: Comparison of classifiers based on machine learning metrics

Metrics	Decision Tree	RF	SVM	Boosted Tree	KNN
Accuracy	0.99974	0.99970	0.99746	0.99998	0.99996
Precision	0.99989	0.99996	0.99952	1.00000	0.98915
Recall	0.97135	0.96730	0.71935	0.99776	0.99726
F1-score	0.98541	0.98336	0.83660	0.99888	0.99319
AUC score	0.98907	0.99061	0.87562	0.99999	0.98906

This presented experimental results and analysis for the detection of FFSN. Upon comparison of all classifiers, we observed that, boosted tree classifier gives best accuracy, precision, recall, f1-score and auc-score, with 0% false positive and 0.00002% false negative result.

VII. CONCLUSION AND FUTURE WORK

In this work, we have used machine learning techniques to identify the Fast Flux Service Networks from the normal ones. For this, we have used total 11 features, i.e, ttl, ipcount, ipavg, ipasn, nscount, nsavg, nsasn, unique ip, ns unique ip, asnip, asnns and proposed their threshold values to efficiently identify fast flux domains. We have used 5 different machine learning algorithms, namely, Random Forest, Decision Tree, Boosted Tree, kNN and SVM. Out of these Boosted tree classifier shows the best result with 99.998% Accuracy, 99.999% AUC score, 100% Precision, 99.776% Recall and 99.888% F1-score.

In future, we would like to expand this work by considering other features such as network delay. We would also like to use some other classification techniques for detection of fast-flux service networks.

REFERENCES

[1] Wikipedia. (2018). Round robin DNS. Retrieved from http://en.wikipedia.org/wiki/Round-robin DNS.
[2] Wikipedia. (2018). Content delivery network(CDN). Retrieved from http://en.wikipedia.org/wiki/Content delivery network.
[3] "The Honeynet Project: Know Your Enemy: Fast-Flux Service Networks", 2007. Retrieved from http://www.honeynet.org/papers/ff/fast-flux.pdf.

[4] InfosecInstitute. (2018). ”Fast Flux Working and Detection” Retrieved from http://resources.infosecinstitute.com/fast-flux-networks-working-detection-part-1
[5] Incasula.(2018). ”Cloud-based application delivery platform”. Retrieved from https://www.incapsula.com/cdn-guide/what-is-cdn-how-it-works.html
[6] Woodiss-Field, A., Johnstone, M. N., & Haskell-Dowland, P. (2024). Examination of Traditional Botnet Detection on IoT-Based Bots. Sensors, 24(3), 1027.
[7] S. Kandula, D. Katabi, M. Jacob, and A. Berger, ”Botz-4-sale: Surviving organized ddos attacks that mimic flash crowds,” in 2nd Symposium on Networked Systems Design and Implementation (NSDI), May 2005.
[8] Howtogeek. (2018). Botnet Retrieved https://www.howtogeek.com/183812/htg-explains-what-is-a-botnet.
[9] Technopedia.(2018).Botnet Retrieved from https://www.techopedia.com/definition/384/botnet.
[10] Techtarget. (2018). Botnet Retrieved from https://searchsecurity.techtarget.com/definition/botnet.
[11] A. Ramachandran, N. Feamster, and D. Dagon, ”Revealing botnet membership using dnsbl counter-intelligence,” in USENIX 2nd Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI 06), June 2006.
[12] Yan, X., Xu, Y., Yao, S., & Sun, Y. (2023). A Domain Embedding Model for Botnet Detection Based on Smart Blockchain. IEEE Internet of Things Journal.
[13] Zhou, S. (2015). A survey on fast-flux attacks. Information Security Journal: A Global Perspective, 24(4-6), 79-97.
[14] Wazzan, M., Algazzawi, D., Bamasaq, O., Albeshri, A., & Cheng, L. (2021). Internet of Things botnet detection approaches: Analysis and recommendations for future research. Applied Sciences, 11(12), 5713.
[15] T. Holz, M. Steiner, F. Dahl, E. Biersack, and F. Freiling ”Measurements and Mitigation of Peer-to-Peer-based Botnets: A Case Study on Storm Worm” In Proceedings of the First USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET ’08), 2008.
[16] Jeremy. Storm worm fast fast flux domain. (2008).“superdrugtesting.com.” Retrieved from http://www.sudosecure.net/archives/36.
[17] K. Alieyan, A. Almomani, A. Manasrah, M. M. Kadhum, ”A survey of botnet detection based on DNS”, Neural Computing and Applications, pp. 1-18, 2015.
[18] Zang, X., Cao, J., Zhang, X., Gong, J., & Li, G. (2024). BotDetector: a system for identifying DGA-based botnet with CNN-LSTM. Telecommunication Systems, 85(2), 207-223.
[19] Dai, Y., Huang, T., & Wang, S. (2024). DAmpADF: A framework for DNS amplification attack defense based on Bloom filters and NAmpKeeper. Computers & Security, 139, 103718.
[20] T. Holz, C. Gorecki, K. Rieck, and F. C. Freiling “Measuring and Detecting FastFlux Service Networks” In Proceedings of the 15th Annual Network Distributed System Security Symposium (NDSS), 2008.
[21] J. Nazario, T. Holz. “As the net churns: Fast-flux botnet observations”. International Conference on Malicious and Unwanted Software, MALWARE 2008.
[22] Scott Campbell, Stephen Chan and Jason R.Lee “Detection of Fast Flux Service Networks” 9th Australasian Information Security Conference (AISC 2011), Perth, Australia, January 2011.
[23] Z. Berkay Celik and Serna Oktug, ”Detection of Fast-Flux Networks using various DNS feature sets” Computers and Communications (ISCC), 2013 IEEE Symposium.
[24] Xiangzhan Yu, Bo Zhang, Le Kang and Juan Chen, ”Fast-Flux Botnet Detection Based on Weighted SVM” Information Technology Journal, 2012.
[25] X. Hu, M. Knysz, K. G. Shin, ”Measurement and Analysis of Global IP-Usage Patterns of Fast-Flux Botnets”, Proceedings of the IEEE Conference on Computer Communications (INFOCOM 2011). Shanghai: IEEE, pp. 2633-2641, 2011.
[26] Kadir A. F. A., Othman, R. A. R., and Aziz, N. A., (2012). ”Behavioral analysis and visualization of fast-flux DNS.” Proceedings—2012 European Intelligence and Security Informatics Conference, EISIC 2012, pp. 250–253.
[27] R. Perdisci, I. Corona, D. Dagon, W. Lee, ”Detecting malicious flux service networks through passive analysis of recursive DNS traces”, Computer Security Applications Conference 2009. ACSAC ‘09. Annual, pp. 311-320, 2009.
[28] P. M. da Luz, Botnet Detection Using Passive DNS. PhD thesis, Master Thesis/Pedro Marques da Luz, 2013.
[29] Toni Grznic,˘ DarkoPerhoc˘ Marko Maric,´ Filip Vlasi˘ c,´ Tibor Kulcsar, ”CROFlux—Passive DNS method for detecting fast-flux domains” in Information and Communication Technology Electronics and Microelectronics (MIPRO) 2014 37th International Convention on, IEEE, pp. 1376-1380, 2014.
[30] D.-T. Truong, G. Cheng, ”Detecting domain-flux botnet based on dns traffic features in managed network”, Security and Communication Networks, vol. 9, no. 14, 2016.
[31] CAIDA (2018). Caida datasets. Retrieved from http://data.caida.org/datasets/topology/ark/ipv4/dns-traffic/.
[32] Google (2018). Google top 1000 sites. Retrieved from http://www.google.com/adplanner/static/top1000/.
[33] Alexa (2018), ”The web information company”. Alexa top sites. http://www.alexa.com/topsites.
[34] DNS-BH – Malware Domain Blocklist by Risk Analytics (2018). Retrieved from http://dns-bh.sagadc.org/domains.txt.
[35] TShark (2018). Dump and analyze network traffic. Retrieved from https://www.wireshark.org/docs/man-pages/tshark.html.

[36] Python (2018). Python programming language. Retrieved from https://en.wikipedia.org/wiki/Python (programming language).
[37] WHOIS (2018). Query and Response protocol. Retrieved from https://en.wikipedia.org/wiki/WHOIS.
[38] Graphlab. (2018). "Graph-based, high performance, distributed computation, framework written in C++". Retrieved from https://turi.com/
[39] Siva, G. (2024). Matrix variate receiver operating characteristic curve for binary classification. Statistics, 1-8.
[40] Talukder, M. A., Islam, M. M., Uddin, M. A., Hasan, K. F., Sharmin, S., Alyami, S. A., & Moni, M. A. (2024). Machine learning-based network intrusion detection for big and imbalanced data using oversampling, stacking feature embedding and feature extraction. Journal of Big Data, 11(1), 33.
[41] Fontein, R. L. H. (2023). Investigating DNS Information Flow In Corporate Networks (Master's thesis, University of Twente).
[42] Shafiqul Abidin et al , "Cloud Computing Encryption Image Retrieval Strategy in Cloud Computing Using a Hybrid Optimization Algorithm", Journal of Intelligent & Fuzzy Systems, ISSN: 1875-8967, March 2024
[43] Ansar Isak Sheikh, M I Thariq Hussan, Shafiqul Abidin et al , "Revolutionizing Collaborative Auditing : A Dynamic Blockchain-Based Cloud Storage Framework for Data Updates and Assurance", Journal of Intelligent & Fuzzy Systems, ISSN: 1875-8967, March 2024
[44] Wasim Khan, Shafiqul Abidin et al, " Anomalous node detection in attributed social networks using dual variational with generative adversarial networks", Data Science and Management, ISSN: 2666-7649, November 2023
[45] Vikas Rao Vadi, Shafiqul Abidin, Azimuddin Khan , Mohd Izhar "Enhanced Elman spike neural network fostered blockchain framework espoused intrusion detection for securingInternet of Things network", Transactions on Emerging Telecommunications Technologies, John Wiley, (ISSN:2161-3915), August, 2022
[46] Vikas Rao Vadi, Shafiqul Abidin,Naveen Kumar et al "Wireless Communication Without the Need for Pre-shared Secrets is Consummate via the use of Spread Spectrum Technology", Journal of Nuclear Science and Power Generation Technology (Special Issue) , Volume -10, Issue -9, September, 2021