**PROJECT REPORT ON**

# Identifying Algorithmically Generated Domain using cluster analysis

# Carried Out at



## CENTRE FOR DEVELOPMENT OF ADVANCED COMPUTING

## ELECTRONIC CITY, BANGALORE

**UNDER THE SUPERVISION OF Ms. Swati Singh**

### Submitted By:

### Surya Pratap Verma (220950125099)

### Swapnil Shukla (220950125100)

### Toshik Jambhorkar (220950125102)

### Utsav Yadav (220950125104)

### Vedant Kakde (220950125106)

PG DIPLOMA IN BIGDATA ANALYSIS

C-DAC, BANGALORE

### Candidate’s Declaration

### We hereby certify that the work being presented in the report entitled Identifying Algorithmically Generated Domain using cluster analysis, in partial fulfillment of the requirements for the award of PG Diploma Certificate and submitted in the department of PG-DBDA of the C-DAC Bangalore, is an authentic record of our work carried out during the period, 15th Jan 2023 to 15th March 2023 under the supervision of Ms.Swati Singh, C-DAC Bangalore. The matter presented in the report has not been submitted by us for the award of any degree of this or any other Institute/University.

**(Name and Signature of Candidate)**

### Surya Pratap Verma (220950125099)

### Swapnil Shukla (220950125100)

### Toshik Jambhorkar (220950125102)

### Utsav Yadav (220950125104)

### Vedant Kakde (220950125106)

**Counter Signed by**

**-------------------------**

**ACKNOWLEDGMENT**

This project “Identifying Algorithmically Generated Domain using cluster analysis” was a great learning experience for us and we are submitting this work to Advanced Computing Training School (CDAC ACTS).

We take this opportunity to express our gratitude to all those people who have directly and indirectly helped us during the competition of this project. We pay thanks to Ms.Swati Singh who has given guidance and a light to us during this project. His versatile knowledge about “Identifying Algorithmically Generated Domain using cluster analysis” has eased us in the critical times during the span of this Final Project.

We acknowledge here support of our friends and seniors specially Mr.Deepak Ror,Mr.Abhishek Bharti,Mr.Nalin Pushp,Mr.Satyam Kumar, Mr.Shubham kore,Mr.Sagar Patki,Mr.Anuj Pathekar, and who contributed significantly to one or more steps.

We have tried our best to keep report simple yet technically correct. We hope we succeed in our attempt. We take full responsibility for any remaining sins of omission and commission.

From

The Whole Team.

**CERTIFICATE**

### This is to certify that the work titled Identifying Algorithmically Generated Domain using cluster analysis attack detection method using cluster analysis is carried out by Surya Pratap Verma(220950125099), Swapnil Shukla (220950125100), Toshik Jambhorkar (220950125102), Utsav Yadav (220950125104), Vedant Kakde (220950125106) the bonafide students of Diploma in Big Data Analytics of Centre for Development of Advanced Computing, Electronic City, Bangalore from 15th January 2023-15th March 2023. The Course End Project work is carried out under my direct supervision.

**Ms. Swati Singh**

Name of Supervisor

Cdac-DBDA #68, Electronic City, Bangalore - 560100, India

## ABSTRACT

Domain Name System (DNS) is one of the most important services keeping the Internet communication running. Its main task is to convert domain names to IP addresses and vice versa. DNS tunneling is a method used by malicious users who intend to bypass the firewall to send or receive data and also Attacker can exploit DNS to hide data in DNS tunnel as the organization would not block DNS packets in order to access the Internet.

One of the major problems that the world faces today is cyber attacks. Denial of Service attacks have been one of most frequent attacks. Some of the mitigating techniques are whitelisting / blacklisting IP addresses, rate limiting etc. The major goal of any DoS attack is to bring down the reputation of the victim organisation. So, instead of facing the issues after the attack, it is always advisable to develop a smart detection system to detect and prevent DoS attacks. While there are many ways to detect DoS attacks, applying machine learning techniques to detect and prevent the attacks turns out to be a promising one. Since there is a lot of data available about DoS attacks, machine learning algorithms can detect patterns of these DoS attacks and thus apply these patterns to new requests and classify them as malicious or benign requests.

**TABLE OF CONTENT**

1. Introduction……………………………………………………………………………………..1
2. Literature Survey……………………………………………………………………….……….2
3. Architecture ………………………………………………………………………………….….3
   1. DNS name Hierarchy……………………………………………………………….3
   2. DGA attacks………………………………………………………………………..5
4. Methodologies…………………………………………………………………………………..7
   1. Source Dataset, Algorithm and Technique………………………………………….7
   2. Algorithm in Jupyter Notebook……………………………………………………..8

4.2.1 Jupyter Notebook benefits…………………………………………………….8

4.2.2 Jupyter Notebook components…………………………………………….…9

4.3 Machine Learning…………………………………………………………………10

4.3.1 Random Forest ……………………………………………………………..10

4.3.2 Ways to increase accuracy of RF Model…..………………………………..11

1. Conclusion…………………………………………………………………………………….13
2. References… 14

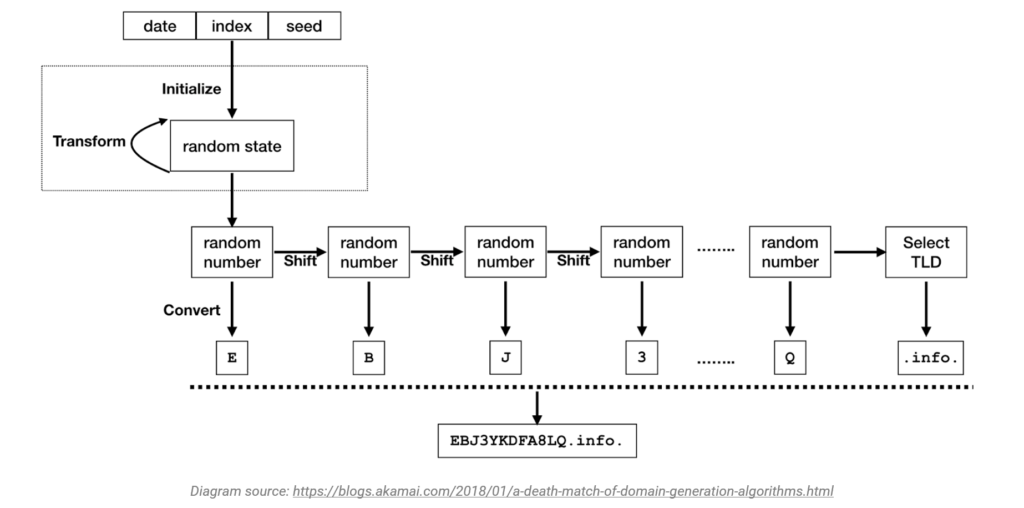
# CHAPTER 1 INTRODUCTION

The number of organisations and devices connected over the internet have been increasing rapidly in recent times. The devices exchange information through web applications. With the increasing number of interactions, the risk of cyber attacks is also increasing. One such popular attack is Denial of Service attack(DoS) DoS attacks result in the loss of time, money (in the form of resources) and the reputation of an organisation. DDoS attacks, in some scenarios are used to divert the attention of network security teams in order to perform other attacks. Most of the attacks are targeted to bringdown the server and can also lead to temporary shutdown of most important services it should offer. To prevent these type of attacks, a smart detection system needs to be integrated in the network security system to categorise the incoming requests. The popular supervised learning technique of classification is used to categorize them. The classification can be either binary or multi-class. A binary class problem generally deals with categorisation as DoS or benign, whereas a multi class problem deals with more than 2 classes. The multi-class problem further deals with the attack classification into the type of DoS attack. Dos attack is an active attack which crashes servers and systems on the network by flooding packets or requests in the network. Since the size of network is large today, there may be a very large no.of users available in the network. Therefore it becomes very hard to identify who are legal users and who are hackers. And also as the technology grows, the techniques for creating Dos attack also becomes increased. Identifying the Dos attack becomes more complex issue since there are various types of Dos attack strategies exist. . Some of the types of the DoS attacks are ICMP flood, SYN flood, IP packet flood etc,. To solve this problem a machine learning

based detector is a good solution.

# CHAPTER 2 LITERATURE SURVEY

Traditionally, malware used to have hard-coded domain names or IP addresses to connect directly with the **Command and Control (C&C or C2)** server. In contrast, DGAs use algorithms to periodically generate a large number of domain names which function as rendezvous points for malware command and control servers. Because the attacker only needs to register one of the thousands of domains the DGA produces to provide the C2 service, all the other domains serve as a distraction to flood the DNS stream with requests for DGA domains making it easy for the attackers to hide and protect the C&C server which will be used to carry out the attack.



**Blacklisting/Whitelisting IP addresses:** This is the simplest way to prevent malicious requests. It is basically creating a method to allow only specific IPs requests or to block requests from specific IPs. This method works well in scenarios where the network is dedicated to serve only a particular organisation. In this case, only IPs of those organisations are whitelisted. But in real time scenarios, this technique might not work very well. It is highly impossible to find out those specific IPs that need to be allowed/ blocked. When the list of IP addresses is generalised to some range, the range might also have many legitimate requests. This will again lead to denial of services to legitimate requests. One malicious user might gain access to one of the IP addresses from a legitimate group and this might result in blacklisting the whole group. So, this technique can be applied to real time scenarios.

# CHAPTER 3 ARCHITECTURE

### DNS name hierarchy

The domain structure of DNS is a tree hierarchy consisting of nodes,

areas, domains,

subdomains, and other elements. The top of the domain structure is the root zone. Set-ting the root zone is located on multiple servers/mirrors placed around the world. This contains information about all servers in the root zone, and is also responsible for the top level domains

(ru, net, org, etc.).The servers

of the root zone process and respond to

requests, giving

information only about the top level domains. The root and top level domains are administered by The Internet Corporation for Assigned Names and Numbers (ICANN). Below the top level domains (TLD), the administration of namespace is delegated to organizations.

FIGURE 1 shows the DNS hierarchy tree. Each node in the tree represents a DNS name. This name can include DNS domains, computers or services. DNS domains can contain both hosts (computers or services) and other domains (subdomains). Each organization is assigned authority for a part of the domain namespace and is responsible for administering DNS domains and computers within this sector of the namespace

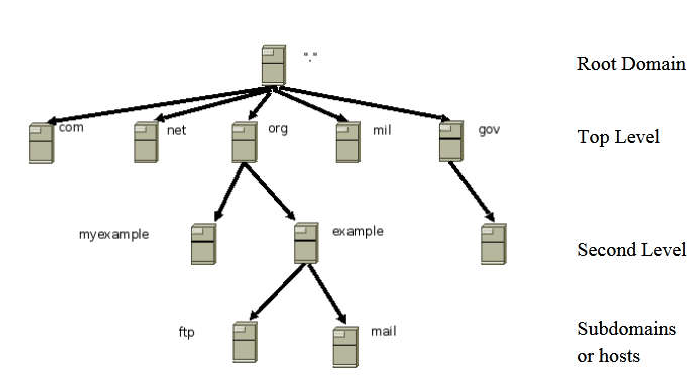


Figure 1 DNS Hierarchy

**Zone** is any part of the tree of the domain name that is placed as a whole on a DNS server. Zone can be called a "zone of responsibility". The purpose of the allocation of the tree in separate

areas the transfer of responsibility to persons or organizations. Each zone authoritative server DNS which stores all the information about the zone responsible.

has at least one for which it is

**Domain** is a named branch or subtree in the tree of DNSstructure, therefore, it is a particular node, including all subordinate nodes. Each node in the DNS hierarchy is separated from its parent by a dot. The domain name begins with a dot (the root do-main) and passes through the

domains of the first, second, and third (if necessary) levels and ends with hostname. Thus,

domain name fully reflects the structure of DNShierarchy. Often, the last dot (the designation of the root domain) in the domain name is omitted.

**Fully Qualified Domain Name (FQDN)** domain name that uniquely identifies the domain name and includes the names of all parent domains in the DNShierarchy, including the root. It is like an analogue of the absolute path in file system. A Root Domain TopLevel Second Level Sub domains or hosts 10 example of this is present in *Figure 2* that introduces the domain name ftp.example.org from *Figure 1*.

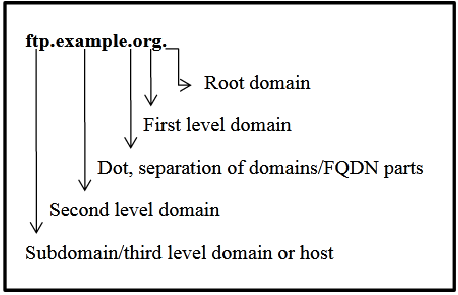


Figure 2. Fully Qualified Domain Name example.

### 

### 3.2 DGA Attacks

### What is a domain generation algorithm (DGA)?

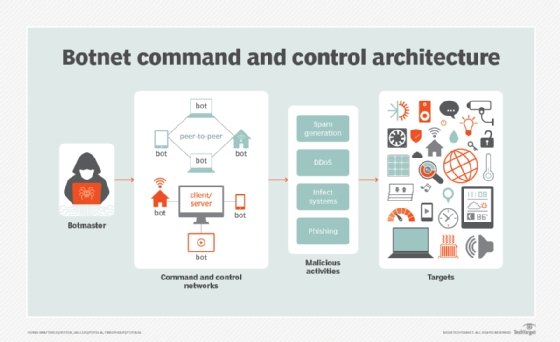
A domain generation algorithm (DGA) is a program that generates a large list of domain names. DGAs provide malware with new domains in order to evade security countermeasures.

Cybercriminals and botnet operators use domain generation algorithms to deliver malware that can generate hundreds of new, random [domains](https://www.techtarget.com/whatis/definition/domain) they can switch between during attacks, making it harder for the victim that is being targeted to block and remove these domains.Changing domain names helps hackers by preventing their servers from being blocklisted or taken down by their targeted victims. The idea is to have an algorithm produce random domain names that the malware can use and quickly switch between. Security software typically blocks and takes down the malicious domains that malware uses, so switching domains quickly enables cybercriminals to continue pursuing the attack.

DGAs are one of the top-known methods that make it harder for malware victims to protect against attacks. They have been used for over 10 years, and some malicious programs still pose a challenge for some to counter. Recent examples of malware attacks that used DGA to create command-and-control servers include Conficker, Zeus and Dyre.

**How does a DGA work?**

DGAs periodically generate a large number of domain names. These domains act as a rendezvous point for malware C&C servers.



This image shows how command-and-control servers operate.

DGAs are pseudo-random generators that construct a random sequence of characters used to form domain names. DGAs can also use words from a dictionary to construct domains. Dictionaries are either hardcoded in malware or taken from an accessible source. DGA generators are normally seed-based and can generate thousands of domain names. The seed is known to both sides, so the same sequences generate on both client and source sides without needing to communicate. This enables the attacker to know beforehand which domain name the malware will use. The attacker then registers that one domain from the sequence to form a communication channel -- or rendezvous point -- for the malware.If a domain is identified as malicious and is taken down, then the domain and C&C server are quickly switched.

For example, if a website owner wants to use the domain name mysite.com and a search on a domain name registrar's site reveals that the desired domain name is not available, a DGA running in the site's background might return suggestions for 50 similar site names that actually are available.

**How does malware use a DGA?**

Security software can quickly block malware that depends on a fixed domain or [Internet Protocol (IP) address](https://www.techtarget.com/whatis/definition/IP-address-Internet-Protocol-Address). So, in response, attackers use DGAs to switch the malware to a new domain at a regular time interval. This is opposed to using a new version of the malware or setting up a new server every time the domain is blocked. The large number of potential rendezvous points makes it difficult for law enforcement to shut down the malware effectively. The additional use of [public key cryptography](https://www.techtarget.com/searchsecurity/definition/asymmetric-cryptography) in the malware's code makes it more challenging to mimic commands from malware controllers.

Botnets are a collection of internet-connected devices that are infected and controlled by malware. Botnet operators have discovered they can use DGAs to hide the operator's C&C server and evade detection by blocklists, signature filters, reputation systems, [intrusion prevention systems](https://www.techtarget.com/searchsecurity/definition/intrusion-prevention), security gateways and other security methods. The scheme, which is called domain fluxing, is similar to hiding a needle (the C&C server) in a haystack (a long list of IP addresses).

# CHAPTER 4 METHODOLOGIES

Despite the fast increasing popularity of cloud services, ensuring the security and availability of data, resources and services remains an ongoing research challenge. Distributed denial of service (DDoS) attacks are not a new threat, It is major security issue and a wide topic of ongoing research interest. In this section we discuss the various DDoS intend and Launch methods that could be used to conduct or facilitate DDoS attacks, as well as reviewing intrusion Detection Methodologies and defence strategies.

**4.1 Source Dataset, Algorithm and Technique:**

A collection of instances is a dataset and when working with machine learning methods, we typically need a few datasets for different purposes. Features have a data type. They may be real or integer-valued or may have a categorical or ordinal value.

It would be can have strings, dates, times, and more complex types, but typically they are reduced to real or categorical values when working with traditional machine learning methods. Source datasets for this project taken form C-DAC Bengluru.

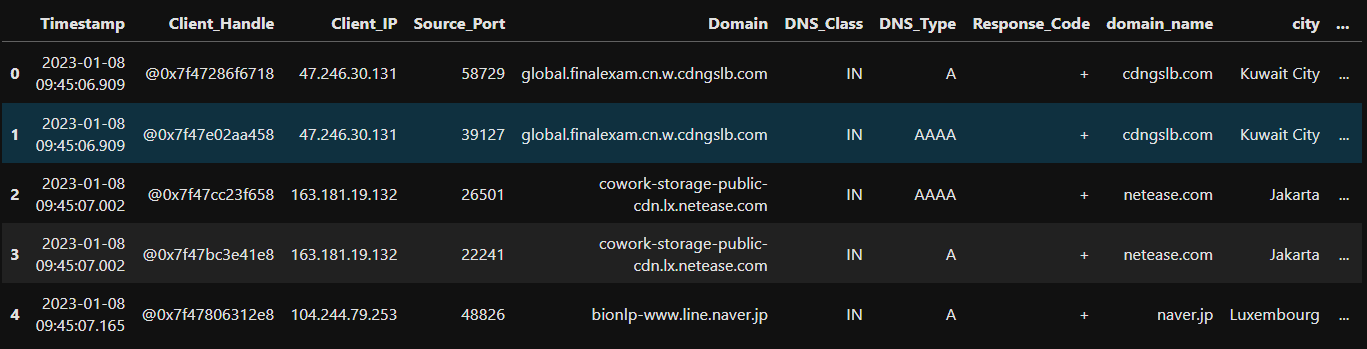


Figure 3.1 example of dataset in csv format

**4.2 Algorithm in Jupyter Notebook:**

Jupyter Notebook was created to make it easier to show one’s programming work, and to let others join in. Jupyter Notebook allows you to combine code, comments, multimedia, and visualizations in an interactive document — called a notebook, naturally — that can be shared, re-used, and re-worked And because Jupyter Notebook runs via a web browser, the notebook itself could be hosted on your local machine or on a remote server.

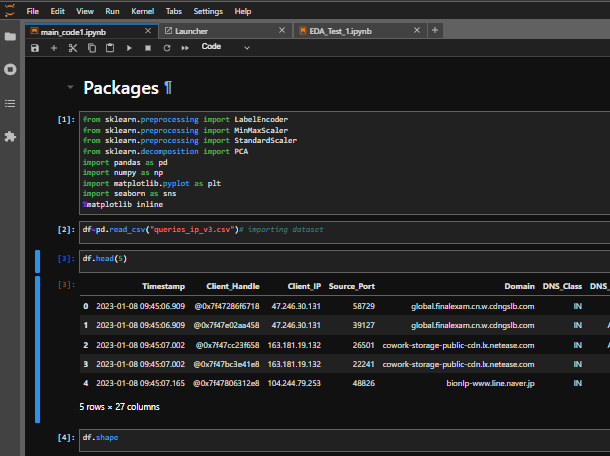


Figure 3.2 shows how the Jupyter Notebook allows to combine code, comments, multimedia, and visualizations in an interactive document

**4.2.1 Jupyter Notebook benefits**

Originally developed for data science applications written in Python, R, and Julia, Jupyter Notebook is useful in all kinds of ways for all kinds of projects:

**• Data visualizations:** Most people have their first exposure to Jupyter Notebook by way of a data visualization, a shared notebook that includes a rendering of some data set as a graphic. Jupyter Notebook lets you author visualizations, but also share them and allow interactive changes to the shared code and data set.

**•Code sharing:** Cloud services like GitHub and Pastebin provide ways to share code, but they’re largely non-interactive. With a Jupyter Notebook, you can view code, execute it, and display the results directly in your web browser.

**•Live interactions with code:** Jupyter Notebook code isn’t static; it can be edited and re-run incrementally in real time, with feedback provided directly in the browser. Notebooks can also embed user controls (e.g., sliders or text input fields) that can be used as input sources for code.

•**Documenting code sample:** If you have a piece of code and you want to explain line-by-line how it works, with live feedback all along the way, you could embed it in a Jupyter Notebook. Best of all, the code will remain fully functional—you can add interactivity along with the explanation, showing and telling at the same time.

**4.2.2 Jupyter Notebook components**

Jupyter Notebooks can include several kinds of ingredients, each organized into discrete blocks:

**•Text and HTML**: Plain text, or text annotated in the Markdown syntax to generate HTML, can be inserted into the document at any point. CSS styling can also be included inline or added to the template used to generate the notebook**.**

**•Code and output:** The code in Jupyter Notebook notebooks is typically Python code, although you may add support in your Jupyter environment for other languages such as R or Julia. The results of executed code appear immediately after the code blocks, and the code blocks can be executed and re-executed in any order you like, as often as you like.

**• Visualizations:** Graphics and charts can be generated from code, by way of modules like Matplotlib, Plotly, or Bokeh. Like output, these visualizations appear inline next to the code that generates them. However, code can also be configured to write them out to external files if needed.

**• Multimedia:** Because Jupyter Notebook is built on web technology, it can display all the types of multimedia supported in a web page. You can include them in a notebook as HTML elements, or you can generate them programmatically by way of the IPython.display module.

**• Data:** Data can be provided in a separate file alongside the .ipynb file that constitutes a Jupyter Notebook notebook, or it can be imported programmatically—for instance, by including code in the notebook to download the data from a public Internet repository or to access it via a database connection.

**4.3 Machine Learning**

Machine learning is a subfield of artificial intelligence that involves the development of algorithms and models that can learn patterns from data, and use that learning to make predictions or decisions without being explicitly programmed. The main goal of machine learning is to enable computers to learn from data and improve their performance over time, without being explicitly programmed for each new task.

In machine learning, a model is trained on a dataset, and the model uses that data to identify patterns and make predictions on new, unseen data. Machine learning algorithms can be categorized into supervised learning, unsupervised learning, and reinforcement learning, depending on the type of data used for training and the goal of the model.

Supervised learning involves training a model on labeled data, where the desired output or outcome is known. The goal of the model is to learn a mapping from input to output, such as predicting whether an email is spam or not.

Unsupervised learning involves training a model on unlabeled data, where the goal is to identify patterns or groupings within the data, such as clustering similar customers based on their purchasing behavior.

Reinforcement learning involves training a model to make decisions in a dynamic environment, where the model receives feedback in the form of rewards or punishments for its actions.

Overall, machine learning has numerous applications in various fields, such as healthcare, finance, marketing, and more, and it has the potential to revolutionize the way we interact with technology and solve complex problems.

**4.3.1 Random Forest:**

Random Forest is a supervised machine learning algorithm that can be used for both classification and regression tasks. It is an ensemble method that combines multiple decision trees, where each tree is trained on a different subset of the data and a random subset of the features.

During the training process, the algorithm builds a large number of decision trees, each of which predicts the target variable for a given input by recursively splitting the data based on the most informative features. The final prediction of the random forest is then made by aggregating the predictions of all the individual trees. This aggregation process can be done in several ways, such as taking the mean (for regression) or the majority vote (for classification) of the predictions.Random Forest is a powerful and versatile algorithm that is widely used in various fields such as finance, medicine, and marketing. It is known for its ability to handle large datasets with high dimensionality and noisy data, as well as its robustness to overfitting.

Random Forest has several advantages that make it a popular choice for many machine learning applications:

* Versatility: Random Forest can be used for both classification and regression tasks, making it suitable for a wide range of problems.
* Robustness: Random Forest is less prone to overfitting than other machine learning algorithms, thanks to its use of multiple decision trees and feature subsets.
* Scalability: Random Forest can handle large datasets with high dimensionality, making it a good choice for big data problems.
* Interpretability: Random Forest provides feature importance scores that can help to identify the most informative features in the data.
* Performance: Random Forest is often competitive with or outperforms other machine learning algorithms in terms of accuracy and speed.

In addition to the advantages mentioned earlier, Random Forest has several other benefits that make it a popular choice for machine learning applications:

* Handle missing values: Random Forest can handle missing values in the data, making it a suitable choice for datasets with incomplete information.
* Reducing overfitting: Random Forest can reduce overfitting by averaging the predictions of multiple decision trees, which helps to prevent the model from memorizing the noise in the data.
* Resistant to outliers: Random Forest is resistant to outliers in the data, thanks to its use of multiple decision trees that are trained on different subsets of the data.
* Easy to use: Random Forest is relatively easy to use compared to other machine learning algorithms, and does not require extensive parameter tuning.
* Non-linear decision boundaries: Random Forest can capture non-linear decision boundaries in the data, making it suitable for problems where the relationship between the input variables and the target variable is complex.

Overall, Random Forest is a robust and versatile algorithm that can handle a wide range of machine learning problems. Its ability to handle missing values, reduce overfitting, and capture non-linear relationships in the data make it a powerful tool for data analysis and modeling.

That being said, the choice of the best machine learning algorithm for a particular project depends on several factors, including the type and size of the data, the desired level of interpretability, the available computational resources, and the specific goals of the project. It's always a good idea to try several different algorithms and compare their performance before choosing the best one for the DGa project.

**4.3.2 Ways to increase accuracy of Random Forest Model**

Some ways to increase accuracy in random forest algorithm:

* Increase the number of trees: Random forest algorithm works by creating an ensemble of decision trees, and increasing the number of trees in the forest can often lead to improved accuracy. However, adding too many trees can also lead to overfitting, so it's important to find the optimal number of trees using cross-validation.
* Tune hyperparameters: Random forest algorithm has several hyperparameters that can be tuned to improve accuracy, such as the maximum depth of the trees, the minimum number of samples required to split an internal node, and the number of features to consider when looking for the best split. Using techniques such as grid search or random search can help find the optimal hyperparameters.
* Use feature selection: Random forest algorithm can sometimes benefit from feature selection, which involves identifying the most important features for the prediction task and only using those features in the model. This can reduce overfitting and improve accuracy.
* Address class imbalance: If the dataset is imbalanced, where one class is significantly more common than the other, random forest algorithm can sometimes struggle to accurately predict the minority class. Techniques such as oversampling or undersampling can be used to balance the classes and improve accuracy.
* Use ensemble methods: Random forest algorithm is already an ensemble method, but combining multiple models can sometimes lead to even better accuracy. Techniques such as bagging or boosting can be used to create an ensemble of multiple random forest models.

Overall, improving accuracy in random forest algorithm involves a combination of tuning hyperparameters, optimizing the number of trees, addressing class imbalance, using feature selection, and possibly combining multiple models.

# CHAPTER 6 CONCLUSION

DGA attacks have been launched against many organizations, resulting in severe impacts.

While researchers have been attempting to tackle DGA attacks, the solutions put forth thus

far have not been effective against DGAS attacks. In this thesis, we enrich the field with a

number of frameworks with different objectives.

The training process and tuning the Machine Learning from the standard data set are the

basis for the generated model and it has an acceptable percentage of classification at the

time of the evaluation of the prototype in a production environment with real information.

The selection of metrics in the intrusion detection problem: false positive rate, false negative

rate, rate classification, ROC curves, allow having a standard of comparison against other

models. The application of techniques with supervised training as SVM model, has large

advantages over the technique based on rules, since the generation of the model is based on

a statistical model that changes its behavior according to the input parameters defined in the

training and based on rules it requires human interaction. In the prototype evaluation was

found a better classification rate for normal and anomalous requests in the training phase, is

directly related to standardization and proper selection of input parameters, allowing output

variables to be generated with minimum percentage of misclassification, generating

reliability in the generated model and the detection of these behaviours

# Chapter 7 REFERENCES

* [1] M. Antonakakis, R. Perdisci, W. Lee, N. Vasiloglou II, and D. Dagon, (2011, August) “Detecting Malware Domains at the Upper DNS Hierarchy”. In USENIX security symposium Vol. 11, pp. 1-16, 2011
* [2] M. Antonakakis, R. Perdisci, Y. Nadji, N. Vasiloglou II, S. Abu-Nimeh, W. Lee, and D. Dagon, “From Throw-Away Traffic to Bots: Detecting the Rise of DGA-Based Malware”. In USENIX security symposium Vol. 12, 2012.
* [3] M. Antonakakis, R. Perdisci, N. Vasiloglou, and W. Lee, Detecting and Tracking the Rise of DGABased Malware. The magazine of USENIX & SAGE, 37(6), 15-24, 2012
* [4] H. Armstrong, (2015, July 05). Machine that learn in the wild. Available: <https://www.nesta.org.uk/sites/default/files/machines_that_learn_in_the_wild.pd>
* [5] P. Arntz, (2016, June 27). Explained: Domain Generating Algorithm. Available: <https://blog.malwarebytes.com/security-world/2016/12/explained-domain-generatingalgorithm/>
* [6] J. Lee, and H. Lee, “GMAD: Graph-based Malware Activity Detection by DNS traffic analysis”. Computer Communications, 49, 33-47, 2014. Behal, Sunny & Kaur, Sukhvirpal & Kumar Ahuja, Dr. Gulshan. (2017). Detection of DDoS Attacks using Weka Tool : A Case Study.of Master of Computer application, SBSSTC Ferozepur-152001, Punjab, India