

A Project Report on

**DARK TRACER EARLY MALWARE DETECTION BASED ON
SPATIOTEMPORAL PATTERNS USING XGBOOST ALGORITHMS**

A Dissertation submitted to JNTU Hyderabad in partial fulfillment of the
academic requirements for the award of the degree.

Bachelor of Technology

In

Computer Science and Engineering

Submitted by

JANANI CHALAPATI

(20H51A05E2)

VENKATA SAINATHA REDDY

(20H51A05M3)

MOHAMMED AWAIS KHAN

(20H51A05A0)

Under the esteemed guidance of

MS.A. Mounika Rajeswari

(Assistant professor)



Department of Computer Science and Engineering

CMR COLLEGE OF ENGINEERING & TECHNOLOGY

(UGC Autonomous)

*Approved by AICTE *Affiliated to JNTUH *NAAC Accredited with A⁺ Grade

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD - 501401.

2020- 2024

CMR COLLEGE OF ENGINEERING & TECHNOLOGY

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD – 501401

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



CERTIFICATE

This is to certify that the Major Project report entitled "**DARK TRACER
EARLY MALWARE DETECTION BASED ON SPATIOTEMPORAL
PATTERNS USING XGBOOST ALGORITHMS**" being submitted by Janani Chalapati (20H51A05E2), Venkata Sainath Reddy(20H51A05M3), Mohammed Awais khan(20H51A05A0) in partial fulfillment for the award of **Bachelor of Technology in Computer Science and Engineering** is a record of bonafide work carried out under my guidance and supervision.

The results embodied in this project report have not been submitted to any other University or Institute for the award of any Degree.

Ms. A. Mounika Rajeswari
Assistant Professor
Dept. of CSE

Dr. Siva Skandha Sanagala
Associate Professor and HOD
Dept. of CSE

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

With great pleasure we want to take this opportunity to express our heartfelt gratitude to all the people who helped in making this project a grand success.

We are grateful to **MS.A. Mounika Rajeswari (Assistant Professor)** Department of Computer Science and Engineering for her valuable technical suggestions and guidance during the execution of this project work.

We would like to thank, **Dr. Siva Skandha Sanagala**, Head of the Department Computer Science and Engineering, CMR College of Engineering and Technology, who is the major driving forces to complete our project work successfully.

We are very grateful to **Dr. Ghanta Devadasu**, Dean-Academics, CMR College of Engineering and Technology, for his constant support and motivation in carrying out the project work successfully.

We are highly indebted to **Major Dr. V A Narayana**, Principal, CMR College of Engineering and Technology, for giving permission to carry out this project in a successful and fruitful way.

We would like to thank the **Teaching & Non- teaching** staff of Department of Computer Science and Engineering for their co-operation

We express our sincere thanks to **Shri. Ch. Gopal Reddy**, Secretary& Correspondent, CMR Group of Institutions, and **Shri Ch Abhinav Reddy**, CEO, CMR Group of Institutions for their continuous care and support.

Finally, we extend thanks to our parents who stood behind us at different stages of this Project. We sincerely acknowledge and thank all those who gave support directly or indirectly in completion of this project work.

Janani Chalapati	-	(20H51A05E2)
Venkata Sainath Reddy	-	(20H51A05M3)
Mohammed Awais	-	(20H51A05A0)

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	LIST OF FIGURES	iii
	ABSTRACT	iv
1	INTRODUCTION	1
	1.1 Problem Statement	2
	1.2 Research Objective	4
	1.3 Project Scope and Limitations	4
2	BACKGROUND WORK	6
	2.1. Automated detection of malware activities using non-negative matrix factorization	7
	2.1.1. Introduction	7
	2.1.2. Merits, Demerits and Challenges	10
	2.1.3. Implementation	11
	2.2. Real-Time Detection of Global Cyberthreat Based on Darknet by Estimating Anomalous Synchronization.	
	2.2.1. Introduction	12
	2.2.2. Merits, Demerits and Challenges	13
	2.2.3. Implementation	14
	2.3. Behavior-based Malware Detection Systems	
	2.3.1. Introduction	16
	2.3.2. Merits, Demerits and Challenges	17
	2.3.3. Implementation	19
3	PROPOSED SYSTEM	21
	3.1. Objective of Proposed Model	22
	3.2. Algorithms Used for Proposed Model	22

	3.3. Designing	30
	3.3.1. Architecture	32
	3.3.2 Sequence Diagram	33
	3.3. Stepwise Implementation and Code	
4	RESULTS AND DISCUSSION	45
	4.1. Performance metrics	47
5	CONCLUSION	49
	5.1 Conclusion and Future Enhancement	50
	REFERENCES	51
	GitHub Link	54
	RESEARCH PAPER AND CERTIFICATION	55

List of Figures**FIGURE**

NO.	TITLE	PAGE NO.
1.1	Problem Statement Image	3
2.1	Overview of Dark NMF	9
2.2	Implementation Image	11
2.3	Architecture of Existing System	15
2.4	Implementation of Behavior-based Malware Detection System	20
3.2.1	KNN Working	23
3.2.2	Decision Tree Architecture	24
3.2.3	Naive Bayes Architecture	25
3.2.4	Transfer Learning Architecture	26
3.2.5	Implementation of XGBoost algorithm	29
3.2.6	Working of XGBoost Algorithm	29
3.2.7	Performance Measure of Algorithms Used	30
3.3.1	Architecture of Proposed System	32
3.3.2	Sequence flow of Proposed System	33
3.3.3	Remote User Flow Chart	34
3.3.4	Service Provider Flow Chart	35
3.3.5	Data Flow Diagram	36
4.1	Login Interface	46
4.2	User Registration Interface	46
4.3	Prediction Interface	47
4.4	Prediction Interface	47
4.5	Predicted Results	48

ABSTRACT

In the contemporary cybersecurity landscape, the proliferation of sophisticated malware poses significant challenges to the integrity and security of network infrastructure. This project proposes a novel approach for detecting malware, specifically the elusive Dark Tracer variants, by leveraging spatiotemporal patterns within network traffic data. The proposed methodology integrates advanced machine learning techniques, with a focus on the XGBoost algorithm, to identify subtle anomalies indicative of malicious behavior.

The core objective of this research is to enhance the accuracy and efficiency of malware detection systems by harnessing the inherent spatial and temporal dynamics of network activities. By analyzing the interconnectedness of network nodes and the temporal flow of data, the model aims to distinguish between benign network behavior and suspicious activities associated with Dark Tracer malware.

Key components of the proposed model include feature extraction, model training, and real-time monitoring. Feature engineering techniques are employed to extract informative features from raw network traffic data, capturing the nuanced characteristics of spatiotemporal patterns. The XGBoost algorithm is utilized for its robustness, scalability, and capability to handle complex classification tasks. Through iterative training on labeled datasets comprising both benign and malicious network traffic, the model learns to discern subtle indicators of malware presence.

The effectiveness of the proposed approach is evaluated using rigorous performance metrics, including accuracy, precision, recall, and F1-score. Real-world deployment of the model within existing cybersecurity frameworks enables continuous monitoring of network traffic for potential Dark Tracer infections. Prompt detection of suspicious activities facilitates swift response mechanisms, such as quarantine and mitigation strategies, to prevent further propagation and damage caused by malware.

CHAPTER 1

INTRODUCTION

CHAPTER 1

INTRODUCTION

1.1. Problem Statement

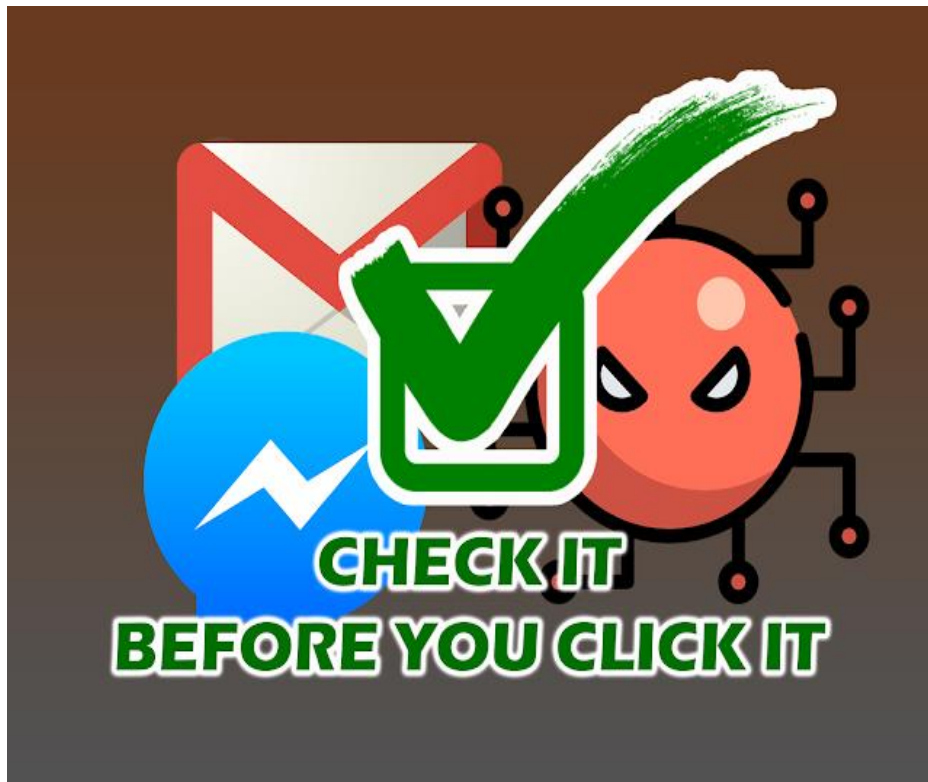
The increasing sophistication of malware, particularly exemplified by the elusive Dark Tracer variants, presents a formidable challenge to the security of network infrastructure. Conventional methods of malware detection, relying heavily on signature-based or rule-based heuristics, often falter in identifying these advanced threats. Dark Tracer variants are adept at stealthily infiltrating networks, exploiting vulnerabilities, and evading detection by conventional means. Their polymorphic nature and ability to swiftly adapt to evolving security measures render traditional detection approaches ineffective.

A key obstacle in detecting Dark Tracer malware lies in the intricate spatiotemporal patterns they exhibit within network traffic. These patterns encapsulate the dynamic relationships between network nodes and the temporal flow of data, which serve as crucial indicators of malicious activity. However, existing detection systems struggle to effectively discern these subtle anomalies amidst the vast expanse of network traffic data.

The challenge at hand is twofold: first, to develop a malware detection system capable of accurately identifying Dark Tracer variants based on their distinct spatiotemporal signatures; and second, to ensure the system's efficiency in real-time monitoring and response. Achieving this requires a paradigm shift in malware detection methodologies, moving away from traditional, static approaches towards dynamic, adaptive techniques that can effectively analyze and interpret the complex interplay of spatiotemporal patterns within network traffic.

Moreover, the solution must address concerns of scalability and adaptability, as network environments continue to evolve in complexity and scale. The detection system must be capable of seamlessly integrating into existing cybersecurity frameworks, providing continuous surveillance of network traffic while remaining resilient against emerging threats.

In essence, the problem at hand is to develop a sophisticated malware detection system that can accurately identify Dark Tracer variants based on spatiotemporal patterns within network traffic data. By addressing this challenge, the proposed solution aims to bolster the resilience of network infrastructure and safeguard critical data assets against the ever-evolving landscape of cyber threats.



1.2. Research Objective

- **Developing an Advanced Detection Model:** Design and develop an advanced malware detection model capable of accurately identifying Dark Tracer variants based on spatiotemporal patterns within network traffic data.
- **Leveraging Machine Learning Techniques:** Utilize machine learning techniques, with a focus on the XGBoost algorithm, to analyze and interpret complex spatiotemporal patterns inherent in network traffic data for effective malware detection.
- **Enhancing Feature Engineering:** Conduct extensive feature engineering to extract informative features from raw network traffic data, capturing the nuanced characteristics of Dark Tracer malware behavior and enhancing the model's detection capabilities.
- **Enabling Real-Time Monitoring:** Design the detection model to enable real-time monitoring of network traffic, allowing for prompt detection and response to potential Dark Tracer infections, thus bolstering network security.
- **Evaluating Performance:** Conduct rigorous performance evaluation to assess the efficacy of the detection model in terms of accuracy, precision, recall, and other relevant metrics, thus providing insights into its effectiveness in detecting Dark Tracer malware.
- **Integrating with Cybersecurity Frameworks:** Integrate the detection model into existing cybersecurity frameworks to complement existing security measures and enhance overall network defense capabilities against sophisticated cyber threats.

1.3. Project Scope and Limitations

Scope:

The scope of the "Dark Tracer Malware Detection" project encompasses the following areas:

- **Development of Detection Model:** The project focuses on developing a sophisticated malware detection model capable of accurately identifying Dark Tracer variants based on spatiotemporal patterns within network traffic data.

- **Utilization of Machine Learning:** The detection model leverages advanced machine learning techniques, particularly the XGBoost algorithm, to analyze and interpret.
- **Feature Engineering:** Extensive feature engineering is conducted to extract informative features from raw network traffic data, capturing the nuanced characteristics of Dark Tracer malware behavior.
- **Real-Time Monitoring:** The detection model is designed for real-time monitoring of network traffic, enabling prompt detection and response to potential Dark Tracer infections.
- **Performance Evaluation:** Rigorous performance evaluation is conducted to assess the efficacy of the detection model in terms of accuracy, precision, recall, and other relevant metrics.
- **Integration with Cybersecurity Frameworks:** The detection model is integrated into existing cybersecurity frameworks to complement existing security measures and enhance overall network defence capabilities.

Limitations:

Despite its scope, the "Dark Tracer Malware Detection" project has certain limitations that need to be acknowledged:

- **Dependency on Data Quality:** The effectiveness of the detection model is contingent upon the quality and relevance of the training data. Limited or biased datasets may impact the model's ability to generalize to new and unseen malware variants.
- **Inherent Uncertainty:** As with any machine learning-based approach, there is inherent uncertainty associated with the detection model's predictions.
- **Computational Resources:** The computational resources required for training and deploying the detection model, particularly in real-time monitoring scenarios, may pose scalability challenges, especially in large-scale network environments.
- **Generalization to New Threats:** While the detection model may perform well against known Dark Tracer variants, its ability to generalize to new and emerging malware threats is limited.

CHAPTER 2

BACKGROUND

WORK

CHAPTER 2

BACKGROUND WORK

2.1. AUTOMATED DETECTION OF MALWARE ACTIVITIES USING NON-NEGATIVE MATRIX FACTORIZATION:

2.1.1. Introduction

Malware that exploits vulnerabilities in cyberspace to launch attacks is evolving and becoming more sophisticated. In particular, with the popularization of Internet of Things (IoT) devices in the recent years, malware targeting these devices have diversified and increased. Many vulnerable IoT devices are infected with IoT malware and exploited for large-scale distributed denial of service attacks, causing serious damage. Moreover, unlike conventional malware, IoT malware performs a higher infection spread speed with their aggressive scanning, e.g., parallel scans to multiple ports. The emergence of diverse and sophisticated IoT malware has further complicated cyber threats.

Malware, which has a self-propagating mechanism via a network, indiscriminately searches for more vulnerable targets through infected victims and spreads the infection. To prevent damage caused by such indiscriminate scanning attacks, it is important to detect a potential malware activity accurately at an early stage before the infection of the malware spreads.

However, accurately distinguishing potential threats from a myriad of indiscriminate scanning and probing is like finding a needle in a haystack. In this study, we make efforts to tackle the challenge of identifying potential malware activities from a myriad of indiscriminate scanning attacks quickly and accurately with few false positives and no false negatives.

A darknet, which passively monitors with an unreachable dark IP address block, is an effective system for observing indiscriminate global cyber threats via the Internet. Please note that here the darknet is not an anonymous network such as Tor. A darknet sensor can be easily installed on a large scale and can capture a large amount of initiation packets of indiscriminate scanning and probing. Although a darknet sensor does not responds, the intention of packets can be roughly grasped by observing the destination ports of each packet.

As conventional research, many studies have tried automatic detection of malware activities, focusing on change points of network-probe campaigns in which the amount of darknet traffic increases rapidly. However, in recent years, malware activities are often inadequate with these changepoint detection methods, such as multiple cyber threats being intricately entwined or being active continuously. Hence, a new approach is required. Devices infected with similar malware that share a scan module tend to scan in a similar spatiotemporal pattern to search for new infection targets.

Such a tendency is also observed on the darknet. Here we define source hosts or destination ports scanned in a similar spatiotemporal pattern as synchronized. By focusing on the synchronization of spatiotemporal features in darknet traffic, the advantage of this approach is expected to eliminate unsynchronized noise traffic, such as misconfigured packets. Furthermore, even if there are no clear spikes in traffic volume in malware activities, they can be detected if spatiotemporal features are highly synchronized.

In our earlier work, an attempt was made to detect potential malware activities by determining the source hosts that are highly synchronized with the spatiotemporal patterns in darknet traffic. It employs the nonnegative matrix factorization (NMF) to decompose darknet traffic into latent temporal and spatial patterns to determine the spatiotemporal features that present anomalous synchronization. Then, it detects anomalous spatial features with high synchronization on the decomposed spatial patterns. The previous study demonstrated an applicability of NMF for detecting new malware activities.

However, it needs modifications for use in actual operations in real-time. Moreover, its performance should be evaluated using long-term traffic data observed in an actual network to demonstrate its effectiveness and reliability. In this study, we propose Dark-NMF, which has been developed into a practically operational method in near real-time by improving impractical parts of the preceding method. DarkNMF is a darknet analysis engine using NMF, and it detects anomaly synchronized malware activities automatically.

For the evaluation, we prepared a human-labeled ground truth of malware activities in a certain period and compared detection results of Dark-NMF and existing methods, Change Finder and GLASSO. Previously, only the source host feature space was used as a spatial feature, but we also treat the destination port feature space as a spatial feature to improve the detection rate of malware activities. As a result, Dark-NMF detects all malware activities that should be detected in the ground truth without false negatives; however, compared to the existing methods, more false positives are obtained. In malware activity detection, although fewer false positives are desired, a system without false negatives is highly valued.

Further, Dark-NMF has several advantages over GLASSO, such as low calculation cost, no past data required for detection. Dark-NMF can be operated immediately in near real time, and we provide a guideline on selecting an algorithm suitable for practical operation. Thus, intelligence information on periods, port numbers, and source IP addresses of detected threats can be supplied to organizations managing the security incident response, allowing them to take prompt measures such as detailed malware activities analysis and cause investigation.

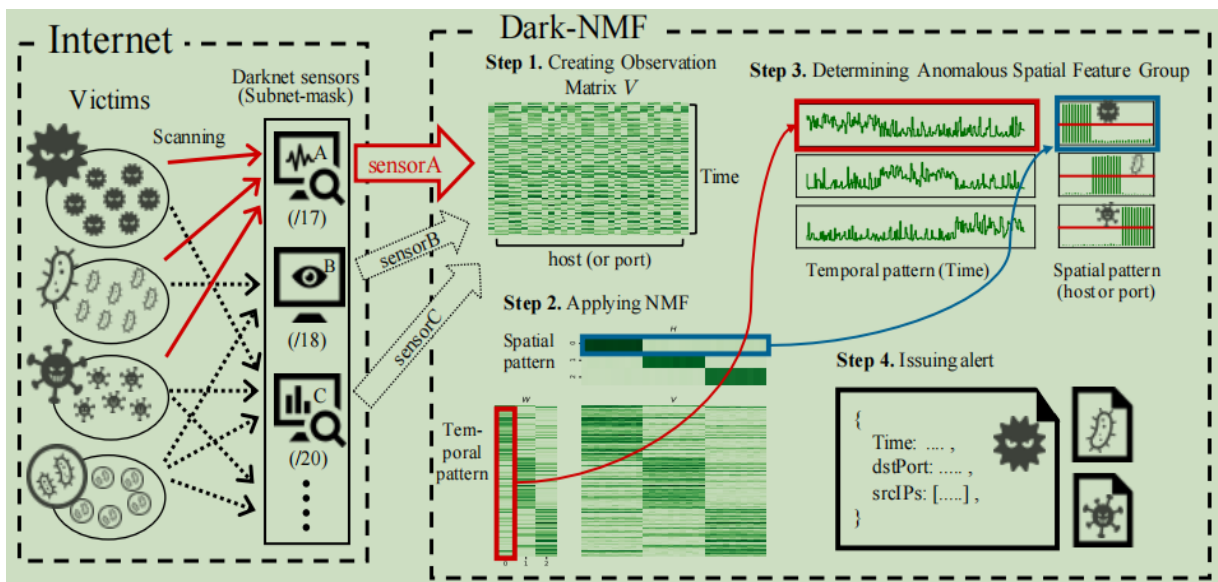


Fig 2.1: Overview of dark NMF

2.1.1.2. Merits, Demerits and Challenges

Merits:

- **Feature Extraction:** NMF can effectively extract latent features and patterns from high-dimensional data, allowing for the identification of signature behaviors associated with malware activities.
- **Dimensionality Reduction:** By reducing the dimensionality of data while preserving essential information, NMF enables more efficient analysis and detection of malware.
- **Interpretability:** The decomposed components obtained from NMF can provide insights into the underlying structure of malware activities, facilitating interpretability and aiding cybersecurity analysts in understanding potential threats.

Demerits:

- **Overfitting:** Like many machine learning techniques, NMF is susceptible to overfitting, where the model captures noise or irrelevant patterns from the data, leading to decreased performance on unseen samples.
- **Complexity:** Implementing NMF for malware detection may require expertise in both cybersecurity and machine learning, making it challenging for organizations with limited resources or specialized skill sets.
- **Limited Context:** NMF focuses primarily on extracting patterns from the raw data without considering contextual information or domain-specific knowledge, which may limit its effectiveness in certain scenarios.

Challenges:

- **Adversarial Evasion:** Malware authors continuously employ evasion techniques to evade detection by security systems. Adapting NMF-based detection methods to counter adversarial evasion is a significant challenge in cybersecurity.
- **Data Imbalance:** Imbalanced datasets, where the number of malware samples is significantly smaller than benign samples, can pose challenges for NMF-based detection systems, potentially leading to biased models and reduced detection accuracy.
- **Scalability:** Scaling NMF-based detection systems to handle the increasing volume and complexity of malware samples requires efficient algorithms and infrastructure capable of processing large datasets in real-time.

2.1.3 Implementation:

- **Data Collection:** Collect a diverse dataset of malware samples and benign software for training and evaluation purposes, ensuring proper labeling and annotation.
- **Feature Engineering:** Extract relevant features from the collected data, such as system calls, file attributes, or network traffic patterns, to represent the behavior of malware and benign software.
- **Nonnegative Matrix Factorization:** Apply NMF to the feature matrix representing the dataset, decomposing it into latent components that capture the underlying structure of malware activities.
- **Model Training:** Train a classifier, such as a support vector machine (SVM) or a neural network, using the decomposed components obtained from NMF as input features, and labels indicating the presence or absence of malware.
- **Evaluation:** Evaluate the performance of the trained model using metrics such as accuracy, precision, recall, and F1-score on a separate test dataset to assess its effectiveness in detecting malware activities.
- **Deployment:** Deploy the trained model in real-world cybersecurity systems for automated detection of malware activities, integrating it with existing security infrastructure for comprehensive threat mitigation.

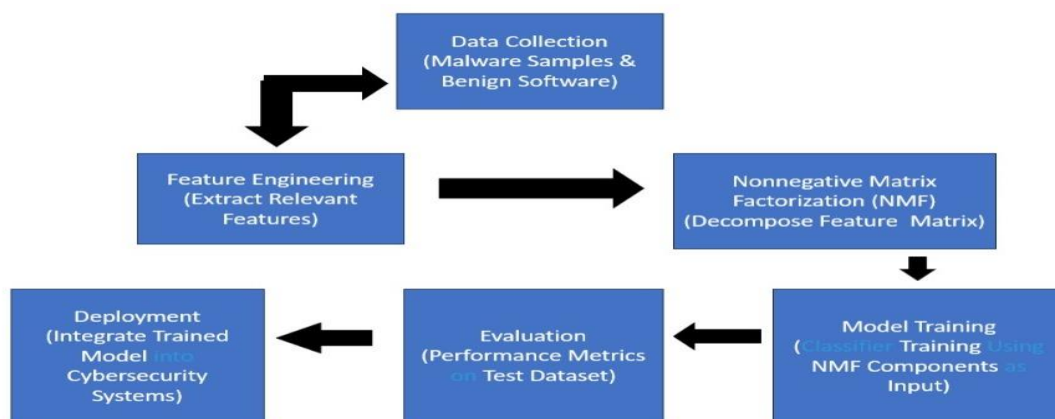


Fig 2.2: Implementation Image

2.2 Real-Time Detection of Global Cyberthreat Based on Darknet by Estimating Anomalous Synchronization Using Graphical Lasso:

2.2.1. Introduction:

Network security threats (cyberthreats) by malware such as worms, bots, and automated exploit tools send Internet-wide scans to a large number of unspecified hosts on the Internet and attempt to exploit vulnerabilities, attack and intrude into targeted systems. When the malware succeeds in infecting vulnerable hosts, the infected hosts will search for next vulnerable targets one after another, and the infection of the malware will spread worldwide.

To minimize the impact of these threats, it is important to understand and determine the behavior of malware on the Internet quickly and precisely. Network security operators or analysts may also monitor the cyberthreats using heuristic and rule-based systems, but these are costly, and human errors may happen. Accordingly, a method that can catch cyberthreats automatically, quickly and precisely is needed. Passively monitoring unused or dark address space (a darknet) is one promising method of investigating these threats [2], [3], because there are no legitimate hosts there. Other than misconfiguration, packets destined for the darknet are almost always malicious [4]. In general, it is suitable for monitoring explosions, not small events [5].

We consider the following approach to resolve the above difficulty. There is no synchronization or cooperation between the packet transmission time patterns of unrelated source hosts. On the other hand, a single host tends to perform similar behavior at each darknet address [3], hence we assume that when source hosts infected with similar malware (such as IoT malware), the source hosts are synchronizing with each other in the darknet. Also, the infected hosts that form a botnet (such as IoT botnet) have the characteristic of acting synchronously when they receive commands from a command and control (C&C) server [6].

2.2.2. Merits, Demerits and Challenges

Merits:

- **Real-Time Detection:** By employing Graphical Lasso to estimate anomalous synchronization patterns, this approach enables the real-time detection of potential cyber threats, allowing for swift response and mitigation measures.
- **Utilization of Darknet Data:** Leveraging data from darknet sources provides insights into potentially malicious activities that may go undetected by conventional cybersecurity measures, thereby enhancing threat detection capabilities.
- **Anomalous Synchronization Analysis:** Focusing on anomalous synchronization patterns allows for the identification of coordinated activities indicative of cyber threats, such as distributed denial-of-service (DDoS) attacks or botnet activities, enhancing the accuracy of threat detection.

Demerits:

- **Data Privacy and Legality:** Accessing and analyzing data from darknet sources may raise concerns regarding data privacy and legality, as these sources often host illicit activities. Ensuring compliance with legal and ethical standards while collecting and analyzing such data is essential.
- **False Positives:** Like any detection method, there is a risk of generating false positives, where legitimate activities are flagged as potential threats. Mitigating false positives is crucial to avoid unnecessary alarm and resource allocation.
- **Scalability:** Analyzing large volumes of darknet data in real-time can pose scalability challenges, requiring robust infrastructure and computational resources to handle the workload efficiently.

Challenges:

- **Data Quality:** Darknet data may be noisy and heterogeneous, making it challenging to extract meaningful insights. Addressing issues related to data quality and preprocessing is crucial for accurate threat detection.
- **Adversarial Techniques:** Cyber adversaries continuously evolve their tactics to evade detection. Developing algorithms resilient to adversarial techniques is essential to maintain the effectiveness of threat detection systems.
- **Interpretability:** Interpreting anomalous synchronization patterns and translating them into actionable insights for cybersecurity professionals can be challenging. Enhancing the interpretability of detection results is essential for effective decision-making.

Implementation:

- **Data Collection:** Collect darknet data from sources such as underground forums, marketplaces, and communication channels, ensuring compliance with legal and ethical standards.
- **Preprocessing:** Clean and preprocess the collected data to remove noise, standardize formats, and prepare it for analysis.
- **Graphical Lasso Estimation:** Apply Graphical Lasso to estimate the synchronization patterns among entities within the darknet data, identifying anomalous patterns indicative of potential cyber threats.
- **Real-Time Monitoring:** Implement a real-time monitoring system that continuously analyzes incoming data streams, flagging suspicious activities based on detected anomalous synchronization patterns.

- **Evaluation and Iteration:** Evaluate the performance of the detection system using metrics such as precision, recall, and false positive rate, and iteratively refine the algorithm to improve its effectiveness and adaptability to emerging threats.

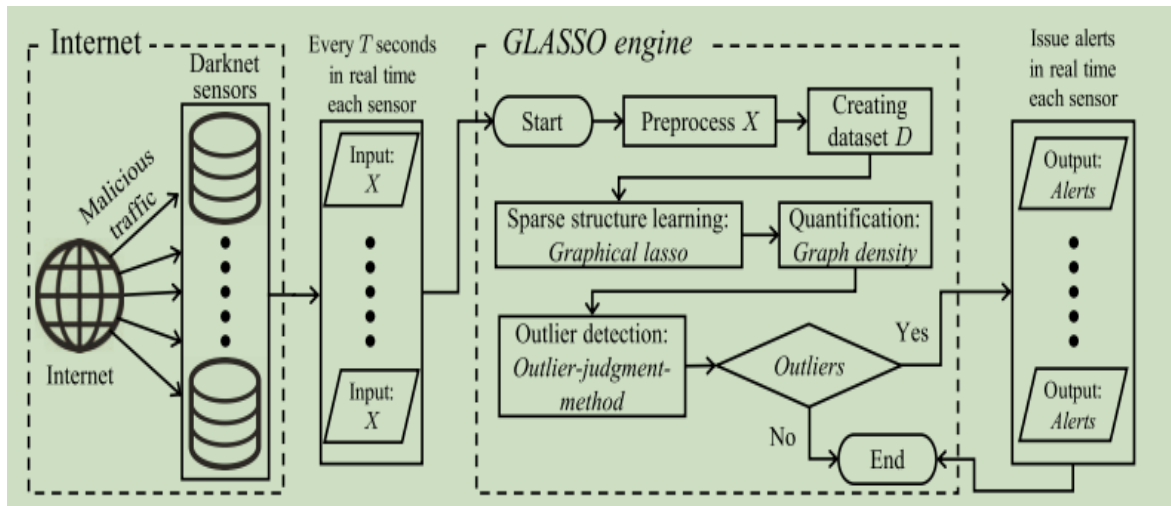


Fig 2.3: The architecture of existing system

2.3 Behavior-based Malware Detection Systems:

2.3.1. Introduction

Behaviour-based malware detection systems take a more dynamic approach to identifying threats by analysing the behaviour of software in real-time. Instead of relying on predefined signatures, these systems monitor and analyse various activities such as file accesses, network communications, system calls, and process executions. By establishing baselines of normal behaviour for different types of software and users, behaviour-based systems can detect deviations or anomalies that may indicate malicious activity.

Nowadays, there is a tremendous increase in both the amount and severity of cyber-related attacks. In general, different malware variants are the main reason for cyber-attacks. Malware is any kind of software which is designed to exploit computer and network systems' vulnerabilities to perform malicious activities and gain financial benefits. Each malicious code variant and its family are designed for different purposes. While some malware variants steal sensitive data, others initiate distributed denial of service (DDoS) attacks and allow remote code execution [1]. During sophisticated attacks, more than one malware type and family are used.

Sandboxing Techniques: Sandboxing involves executing suspicious files or code in isolated environments, known as sandboxes, to observe their behavior without risking damage to the host system. By containing potentially harmful activities within a controlled environment, sandboxing allows analysts to study malware behavior and identify malicious actions.

Heuristic Approaches: Heuristic techniques are used to identify potentially malicious behavior based on predefined rules or patterns. Unlike signature-based detection, which relies on exact matches, heuristic analysis looks for indicators or characteristics commonly associated with malware. These indicators may include suspicious file properties, abnormal system activities, or known attack vectors.

2.3.2. Merits, Demerits and Challenges

Merits:

- **The Dynamic Detection:** Behavior-based malware detection systems offer a dynamic approach to threat identification by analyzing the real-time behavior of software.

- **Adaptive Baselines:** These systems establish baselines of normal behavior for different types of software and users, allowing them to detect deviations or anomalies indicative of malicious activity. By continuously updating these baselines based on observed behavior, behavior-based systems can improve their accuracy over time and adapt to changes in the environment.
- **Effectiveness Against Unknown Threats:** Behavior-based detection is particularly effective at identifying previously unknown threats and zero-day attacks. Unlike signature-based methods, which rely on predefined signatures, behavior-based systems can detect and respond to new malware variants and attack techniques in real-time.
- **Reduced False Positives:** Behavior-based detection tends to produce fewer false positives compared to signature-based methods. By focusing on the behavior of software rather than specific signatures, these systems can better differentiate between malicious and benign activities, reducing the likelihood of false alarms.

Demerits:

- **Complexity:** Implementing behavior-based malware detection systems can be more complex compared to traditional signature-based methods. These systems require sophisticated algorithms and analytics to monitor and analyze various activities such as file accesses, network communications, and system calls in real-time.
- **False Positives:** While behavior-based detection tends to produce fewer false positives compared to signature-based methods, it is still possible for legitimate software behavior to deviate from expected norms, resulting in false alarms. Tuning the system to minimize false positives without compromising detection accuracy can be challenging.

- **Resource Intensive:** Behavior-based detection systems may require significant computational resources to monitor and analyze the behavior of software in real-time. This can include processing power, memory, and storage to handle the large volume of data generated by monitoring activities such as file accesses, network communications, and system calls.
- **Training and Tuning:** Behavior-based detection systems require careful training and tuning to establish accurate baselines of normal behavior for different types of software and users. This process can be time-consuming and may require access to diverse datasets representing various behaviors and usage patterns.

Challenges:

- **Response to Evasion Techniques:** Malware authors continuously develop evasion techniques to bypass behavior-based detection systems. These techniques may include delaying malicious actions, disguising malware behavior as legitimate, or exploiting vulnerabilities in the detection system itself. To counter these evasion tactics, behavior-based systems must constantly evolve and adapt their detection mechanisms to detect and mitigate emerging threats effectively.
- **Scalability:** As the volume of data generated by monitoring software behavior increases, behavior-based detection systems must be able to scale to handle the additional workload. Ensuring scalability while maintaining detection accuracy and minimizing resource consumption is a significant challenge for these systems.

2.3.3. Implementation:

- **Behaviour Monitoring Setup:** This phase involves deploying monitoring mechanisms to observe and analyse the behaviour of software and users in real-time. These mechanisms capture various activities such as file accesses, network communications, system calls, and process executions. For example, a security tool installed on endpoints may utilize kernel-level monitoring to track system events and capture relevant behavioural data.
- **Baseline Establishment:** The system establishes baselines of normal behavior for different types of software and users based on observed activities. It collects and analyzes historical data to identify typical behavior patterns and sets thresholds for deviation detection. For instance, it may learn that a web browser typically accesses specific domains for updates and downloads, establishing a baseline for normal network activity.
- **The Anomaly Detection:** Using the established baselines, the system continuously monitors for deviations or anomalies indicative of malicious activity. It employs anomaly detection algorithms to analyze behavioral data in real-time and identify suspicious patterns or outliers. For example, if a program suddenly exhibits behavior inconsistent with its established baseline, such as attempting to access sensitive system files or communicating with known malicious domains, the system may raise an alert.
- **Response Mechanism:** Upon detecting anomalous behavior, the system initiates response actions to mitigate the potential threat. These actions may include quarantining the suspicious program, blocking network communication associated with the activity, or notifying administrators for further investigation. For example, if a user's account starts exhibiting unusual behavior, such as accessing sensitive files outside of normal working hours, the system may temporarily disable the account and trigger a security incident response process.

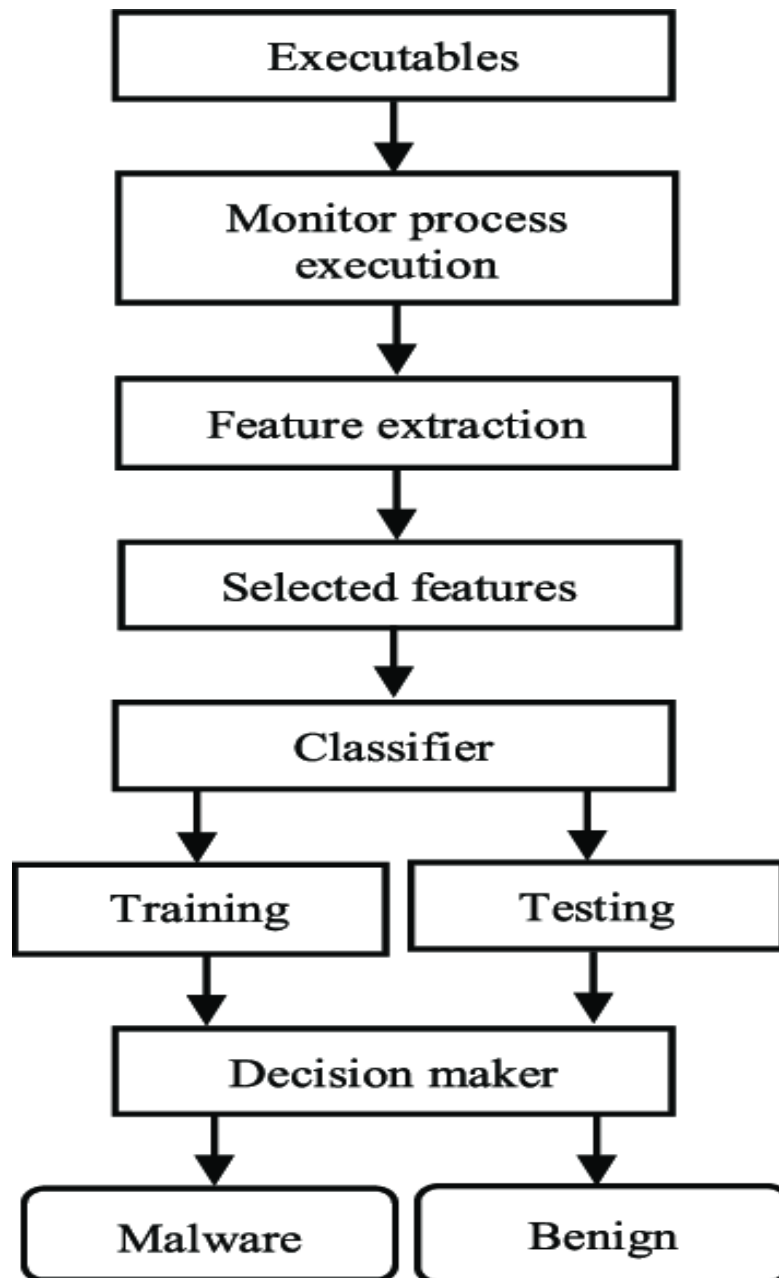


Fig 2.4: Implementation of Behavior-based Malware Detection System

CHAPTER 3

PROPOSED

SYSTEM

CHAPTER 3

3.1. Objective of Proposed Model:

The objective of the proposed model in the "Dark Tracer Malware Detection" project is to develop an advanced detection system capable of accurately identifying Dark Tracer variants by leveraging spatiotemporal patterns within network traffic data. The primary aim is to enhance detection accuracy, ensuring that the model effectively distinguishes between benign network behavior and activities indicative of Dark Tracer malware presence.

By analysing the interconnectedness of network nodes and the temporal flow of data, the model aims to capture subtle anomalies that signify malicious behavior. To achieve this, the XGBoost algorithm is employed due to its efficiency, scalability, and ability to handle complex classification tasks. The model is designed for real-time monitoring, enabling prompt detection and response to potential Dark Tracer infections to mitigate further propagation and damage.

Additionally, scalability and adaptability are key considerations, ensuring that the model remains effective in diverse network environments and against evolving malware threats. Integration with existing cybersecurity frameworks facilitates seamless deployment and operationalization, enhancing overall network defense capabilities. Ultimately, the proposed model aims to bolster the resilience of network infrastructure and safeguard critical data assets against the persistent threat posed by Dark Tracer malware.

3.2. Algorithms Used for Proposed Model:

For the "Dark Tracer Malware Detection" project, the following algorithms are considered for implementation:

3.2.1. k-Nearest Neighbors (kNN):

- In the project, kNN can be used to classify network traffic instances based on their feature vectors. Each instance represents a data point in a high-dimensional feature space, where features are extracted from network traffic data.

- During the training phase, kNN builds a database of labeled instances, where each instance is associated with a class label (benign or malicious). Feature vectors of training instances are stored in memory.
- When a new network traffic instance is encountered during the testing phase, kNN calculates the distances (e.g., Euclidean distance) between the feature vector of the new instance and the feature vectors of the k nearest neighbors in the training set.
- **Feature Vector Representation:** Network traffic data is transformed into feature vectors representing various network activity characteristics.
- **Distance Metric Definition:** A suitable distance metric (e.g., Euclidean distance) is defined to quantify similarity between feature vectors.
- **Training Phase:** kNN builds a database of labeled instances, storing their feature vectors and class labels.
- **Testing Phase:** When a new network traffic instance is encountered, kNN calculates distances to k nearest neighbors in the training set.
- **Classification:** The class label of the new instance is determined by a majority vote among its k nearest neighbors.
- **Hyperparameter Tuning:** Parameters like k and distance metric are tuned via cross-validation to optimize model performance.
- **Handling Imbalanced Data:** Techniques such as oversampling or weighted distances address class imbalance, enhancing model accuracy.

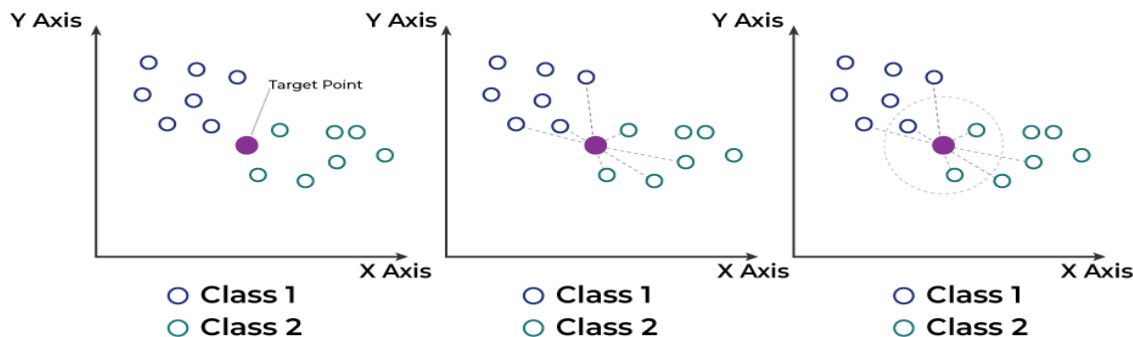


Fig 3.2.1: KNN working

Decision Tree:

- Decision trees are used to learn decision rules from the features extracted from network traffic data.
- During the training phase, the decision tree algorithm recursively partitions the feature space based on feature values to create a tree-like structure.
- At each node, the algorithm selects the feature that best splits the data into homogeneous subsets with respect to the class labels.
- The decision tree continues to split the data until a stopping criterion is met, such as reaching a maximum depth or minimum number of samples in each leaf node.
- During the testing phase, network traffic instances traverse the decision tree from the root node to a leaf node, following the decision rules learned during training. The class label assigned to the leaf node determines the classification outcome.

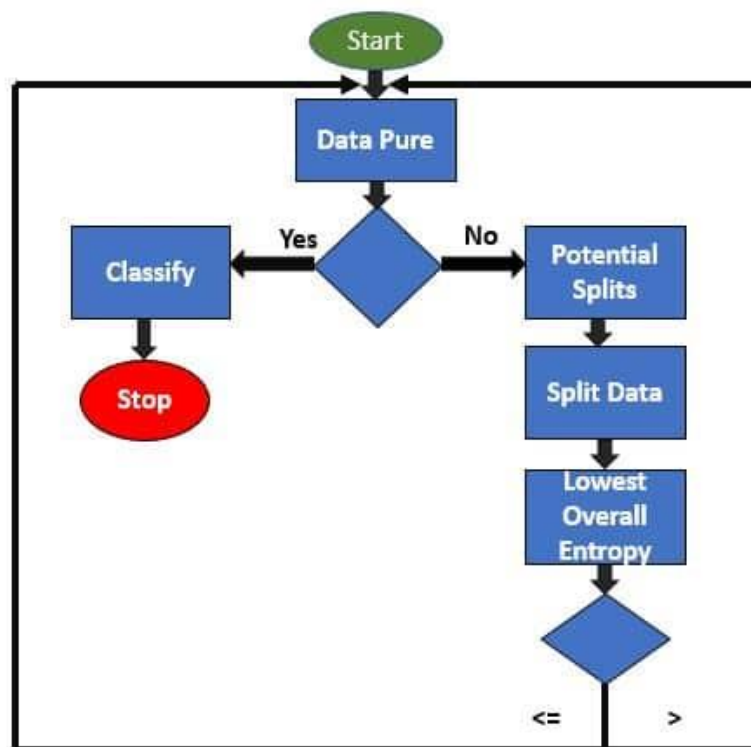
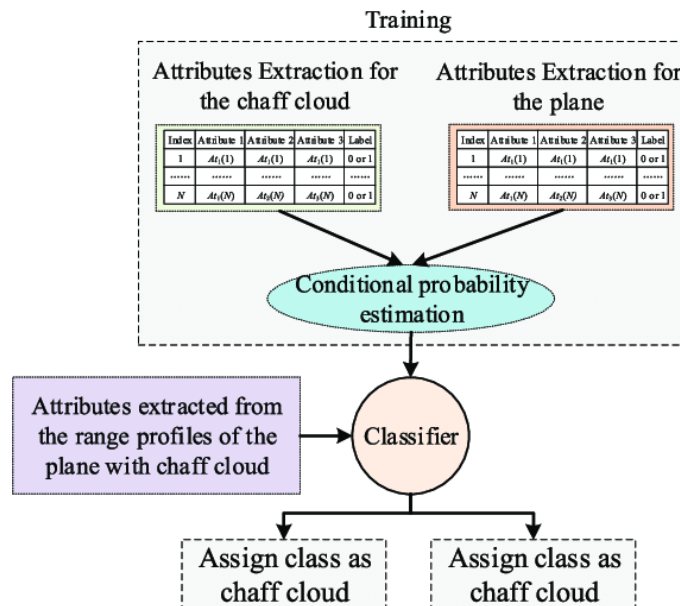


Fig 3.2.2: Decision Tree Architecture

Naive Bayes:

- Naive Bayes is a probabilistic classifier based on Bayes' theorem with the assumption of feature independence given the class.
- Despite its simplifying assumption, naive Bayes often performs well in practice, especially with high-dimensional and sparse data typical in malware detection tasks.
- During the training phase, naive Bayes estimates the class conditional probabilities of each feature given the class labels (benign or malicious).
- During the testing phase, the algorithm computes the posterior probability of each class for a given network traffic instance using Bayes' theorem and assigns the instance to the class with the highest probability.
- Naive Bayes can be particularly effective when there are strong dependencies between features and the class labels are well-separated in the feature space.

**Fig 3.2.3: Naive Bayes Architecture**

3.2.2. Logistic Regression:

- Logistic regression is a linear classification algorithm that models the probability of an instance belonging to a particular class (e.g., benign or malicious) using a logistic function.
- During the training phase, logistic regression estimates the parameters of the logistic function by maximizing the likelihood of the observed data given the model parameters.
- Logistic regression outputs probabilities rather than discrete class labels, allowing for probabilistic interpretation of classification decisions.
- The decision boundary learned by logistic regression is linear, which may limit its ability to capture complex relationships between features. However, polynomial or interaction terms can be incorporated to capture nonlinearity if necessary.

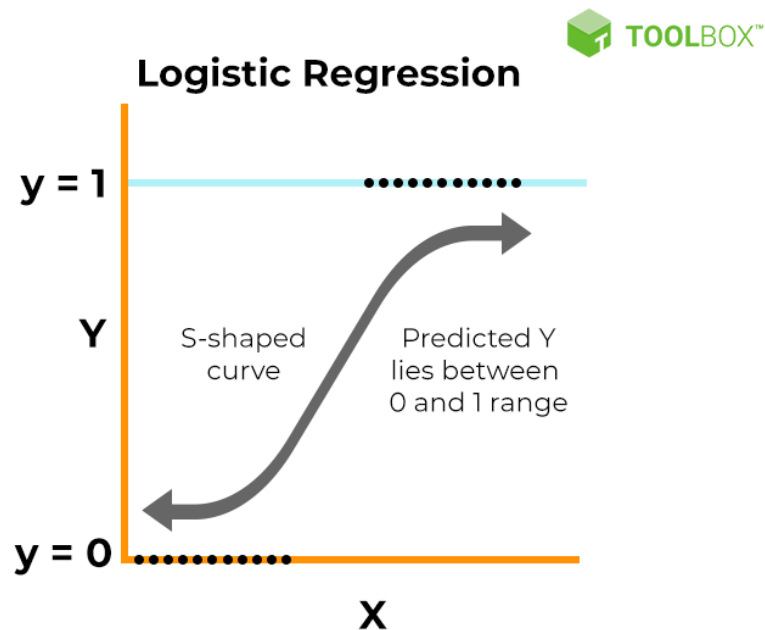


Fig 3.2.4: Transfer Learning Architecture

3.2.3. XGBoost (Extreme Gradient Boosting)

- XGBoost is an ensemble learning algorithm that builds a collection of weak learners (decision trees) to create a strong predictive model.
- During the training phase, XGBoost sequentially builds decision trees, with each subsequent tree aiming to correct the errors of the previous trees. The algorithm optimizes a loss function, such as binary cross-entropy, to minimize the prediction errors.
- XGBoost incorporates regularization techniques to prevent overfitting, such as tree pruning and shrinkage, which help generalize the model to unseen data.
- During the testing phase, network traffic instances are fed into the ensemble of decision trees, and their predictions are aggregated to produce the final classification outcome.

Feature Engineering:

- Before utilizing XGBoost, feature engineering is performed to extract relevant features from raw network traffic data. These features could include packet size, protocol type, source and destination IP addresses, port numbers, timestamps, and more.
- Feature engineering ensures that the input data is properly formatted and contains informative features that can help distinguish between benign and malicious network traffic.

Data Preparation:

- The labeled dataset, consisting of instances of both benign and malicious network traffic, is split into training and testing sets.
- It's essential to maintain class balance or apply appropriate techniques to handle class imbalance, ensuring that the classifier learns from representative samples of both classes.

Model Training:

- XGBoost is then employed to train a classification model using the labeled training data. During training, XGBoost sequentially builds an ensemble of decision trees, with each tree aiming to correct the errors of the previous trees.
- The algorithm optimizes a predefined loss function (e.g., binary cross-entropy) by iteratively fitting decision trees to minimize prediction errors.

Hyperparameter Tuning:

- Hyperparameters of the XGBoost classifier, such as tree depth, learning rate, number of trees in the ensemble, and regularization parameters, are tuned to optimize model performance.
- Techniques like grid search or random search combined with cross-validation are commonly used to search the hyperparameter space and find the optimal configuration.

Model Evaluation:

- The trained XGBoost model is evaluated using the labeled testing data to assess its performance metrics such as accuracy, precision, recall, and F1-score.
- Performance evaluation helps gauge the effectiveness of the model in accurately classifying instances of both benign and malicious network traffic.

Real-Time Monitoring and Deployment:

- Once trained and evaluated, the XGBoost model can be deployed for real-time monitoring of network traffic.
- Network traffic instances are fed into the model, and predictions are generated in real-time to identify potential instances of Dark Tracer malware.

- Prompt detection of suspicious activities facilitates timely response mechanisms, such as quarantine or mitigation strategies, to prevent further propagation and damage caused by malware.
- By leveraging XGBoost in the project, the detection model can effectively analyze and interpret the complex spatiotemporal patterns present in network traffic data, enhancing the accuracy and efficiency of Dark Tracer malware detection.

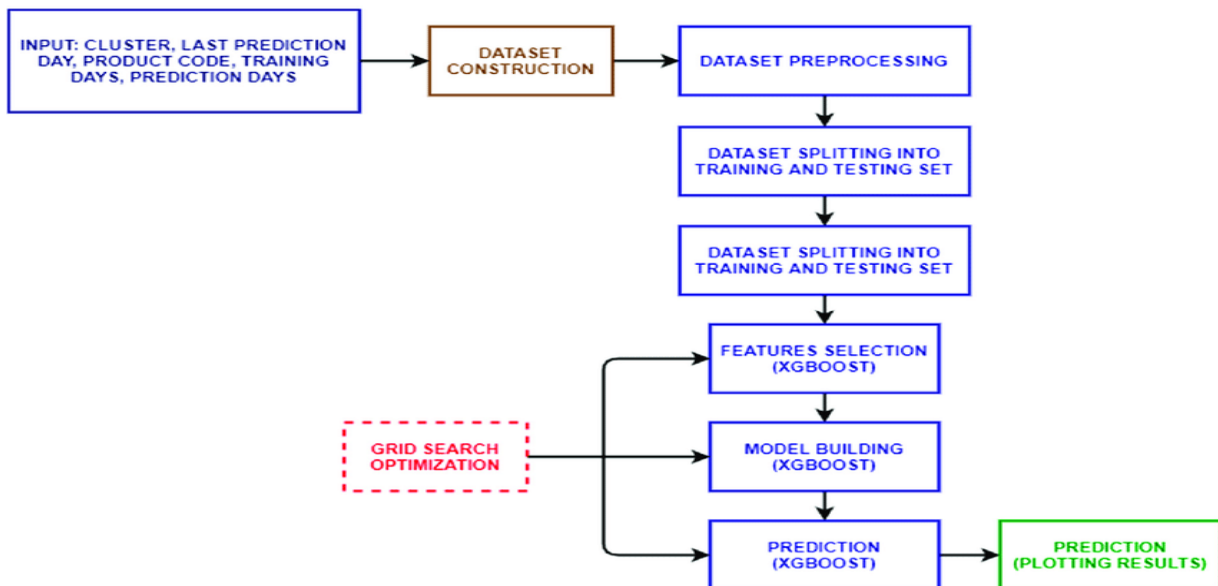


Fig 3.2.5: IMPLEMENTATION OF XGBOOST ALGORITHM

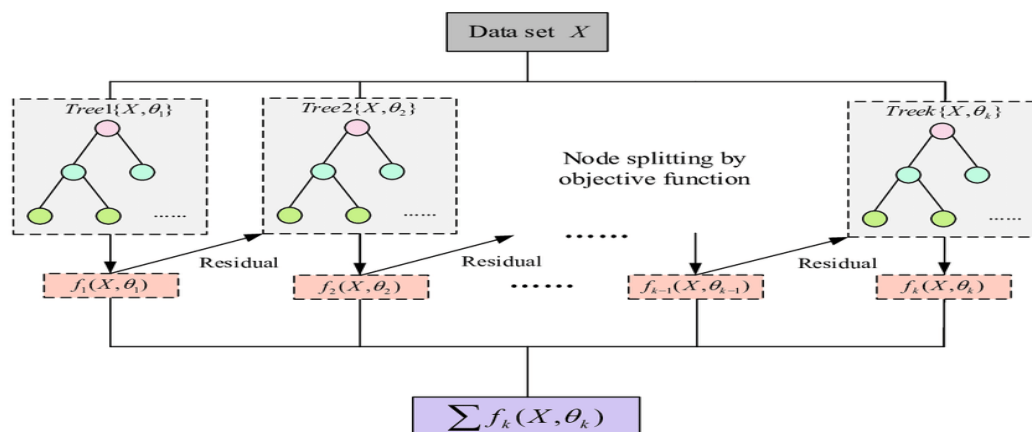


Fig 3.2.6: WORKING OF XGBOOST ALGORITHM

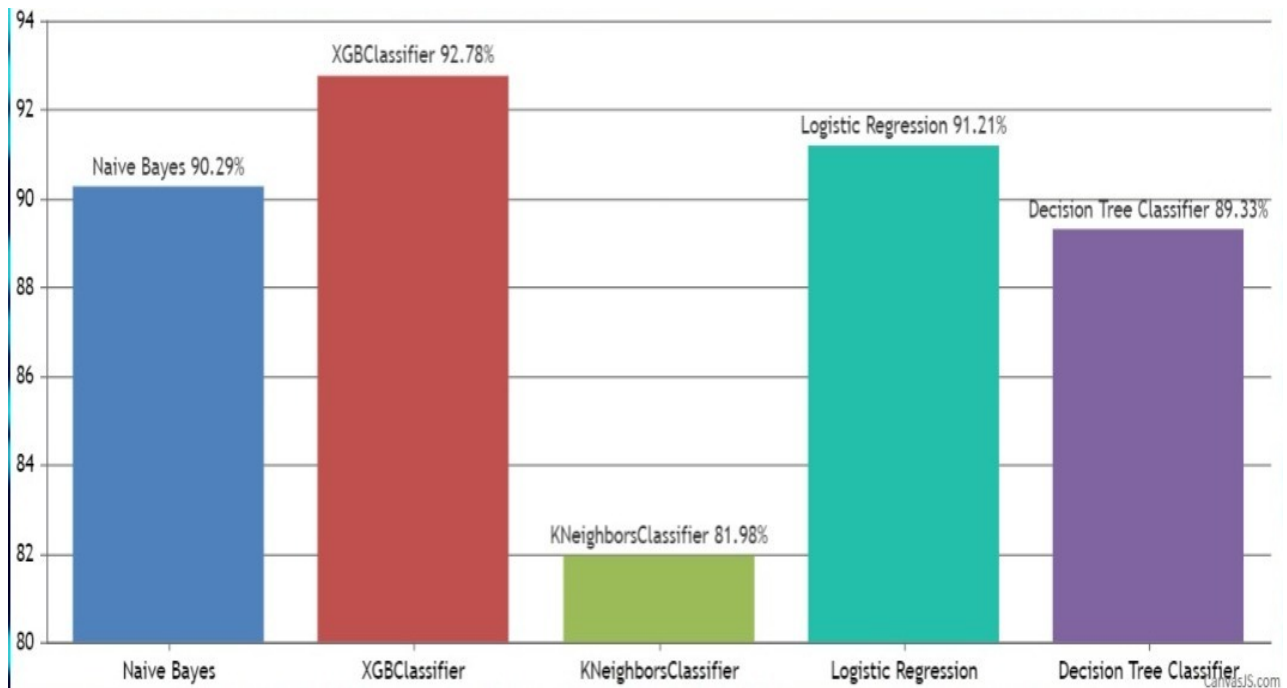


Fig 3.2.7: PERFORMANCE MEASURE OF ALGORITHMS USED

3.3. Design:

3.3.1. Architecture:

Registration/Login System:

- Remote users first register their details with the service provider.
- Upon registration, users can log in using their credentials.
- Data Preparation:

The system utilizes both training and testing datasets for building and evaluating its models.

Model Training:

- The training dataset is used to train machine learning or deep learning models for malware detection.
- This involves feeding labeled data (malware vs. legitimate) into the model to learn patterns and characteristics of malware.

Model Testing:

- The testing dataset is used to evaluate the trained models' performance.
- The models make predictions on this dataset, and their accuracy is assessed based on the known labels.

Visualization of Accuracy:

- The system generates a bar chart displaying the accuracy of the trained models on both the training and testing datasets.
- This visualization helps to assess the performance of the models in distinguishing between malware and legitimate software.

Viewing Accuracy Results:

- Users can access detailed accuracy results of the trained models, likely in the form of numerical values or metrics.
- These results provide insights into how well the models are performing and their effectiveness in detecting malware.

Malware Activity Prediction:

- The system provides details on the malware activity predicted by the models.
- This could include information such as the type of malware detected and its characteristics.

Malware Detection Type Ratio:

- Users can view the ratio of different types of malware detected by the system.
- This information helps in understanding the distribution and prevalence of various malware types.

Download Predicted Datasets:

- Users have the option to download datasets containing predictions made by the models.
- These datasets may include information such as predicted labels (malware or legitimate) for specific files or URLs.

Viewing Detection and Ratio Results:

- Detailed results of malware detection and the ratio of different types of malware are available for users to view.
- This allows users to understand the performance of the system and the prevalence of different malware types.

Remote User Management:

- The system provides functionality to view all remote users registered with the service provider.
- This could include details such as user IDs, registration dates, and activity logs.

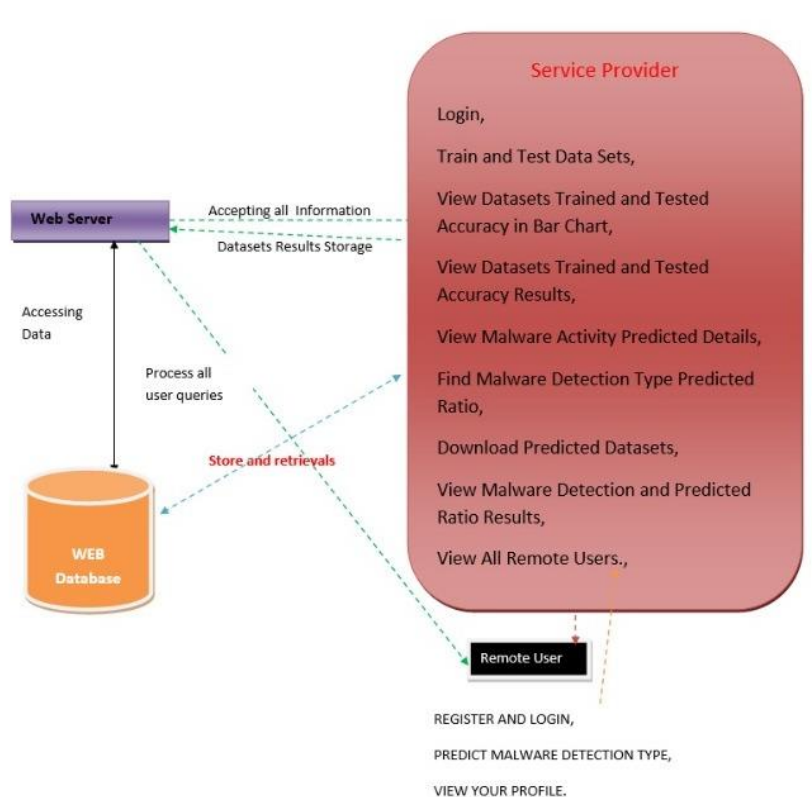


Fig 3.3.1: Architecture of the Proposed System

3.3.2. Sequence Diagram:

A **sequence diagram** is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called **event diagrams**, **event scenarios**, and timing diagrams.

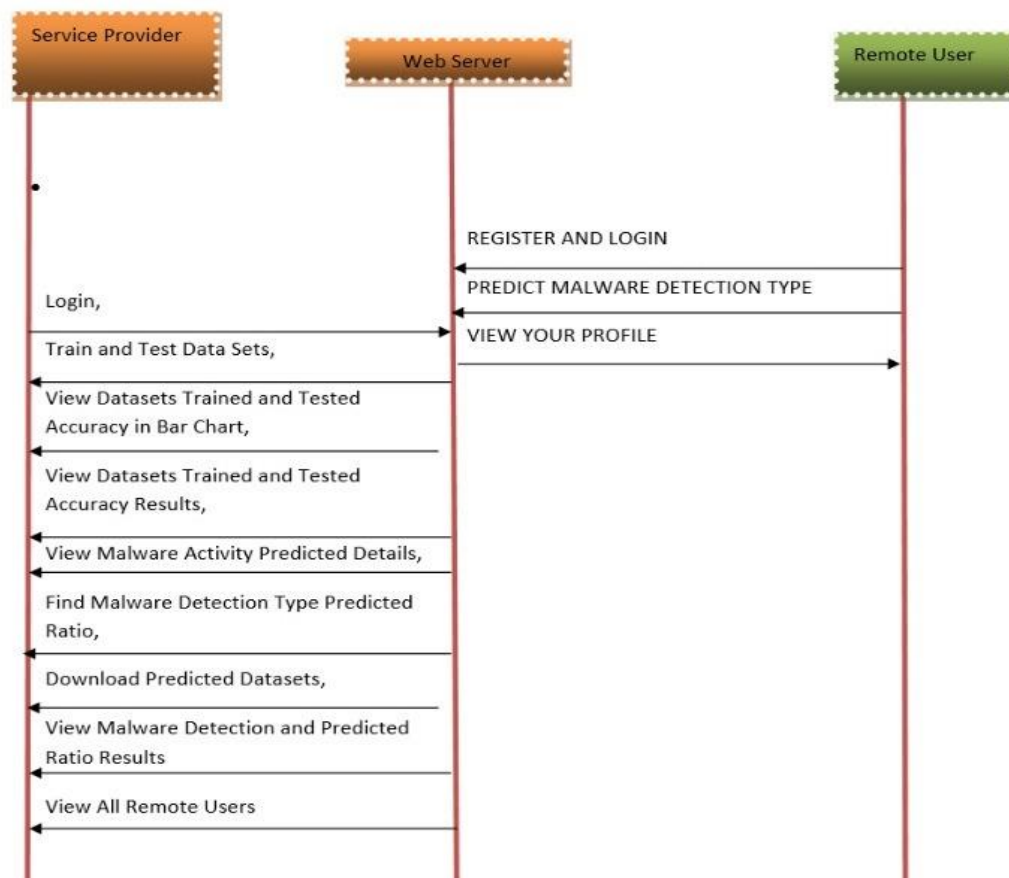


Fig 3.3.2: Sequence flow of Proposed System

➤ **Flow Chart : Remote User**

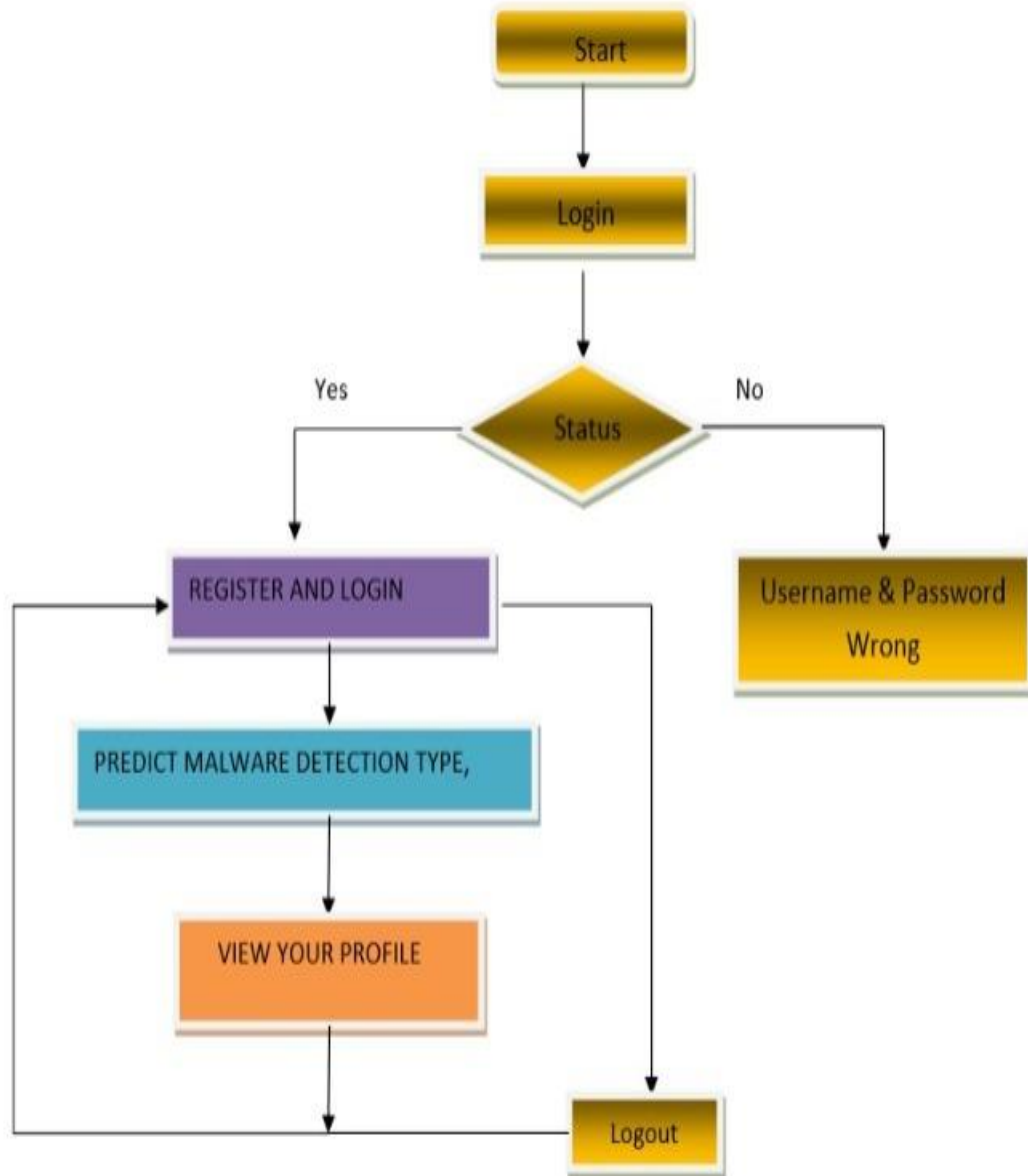
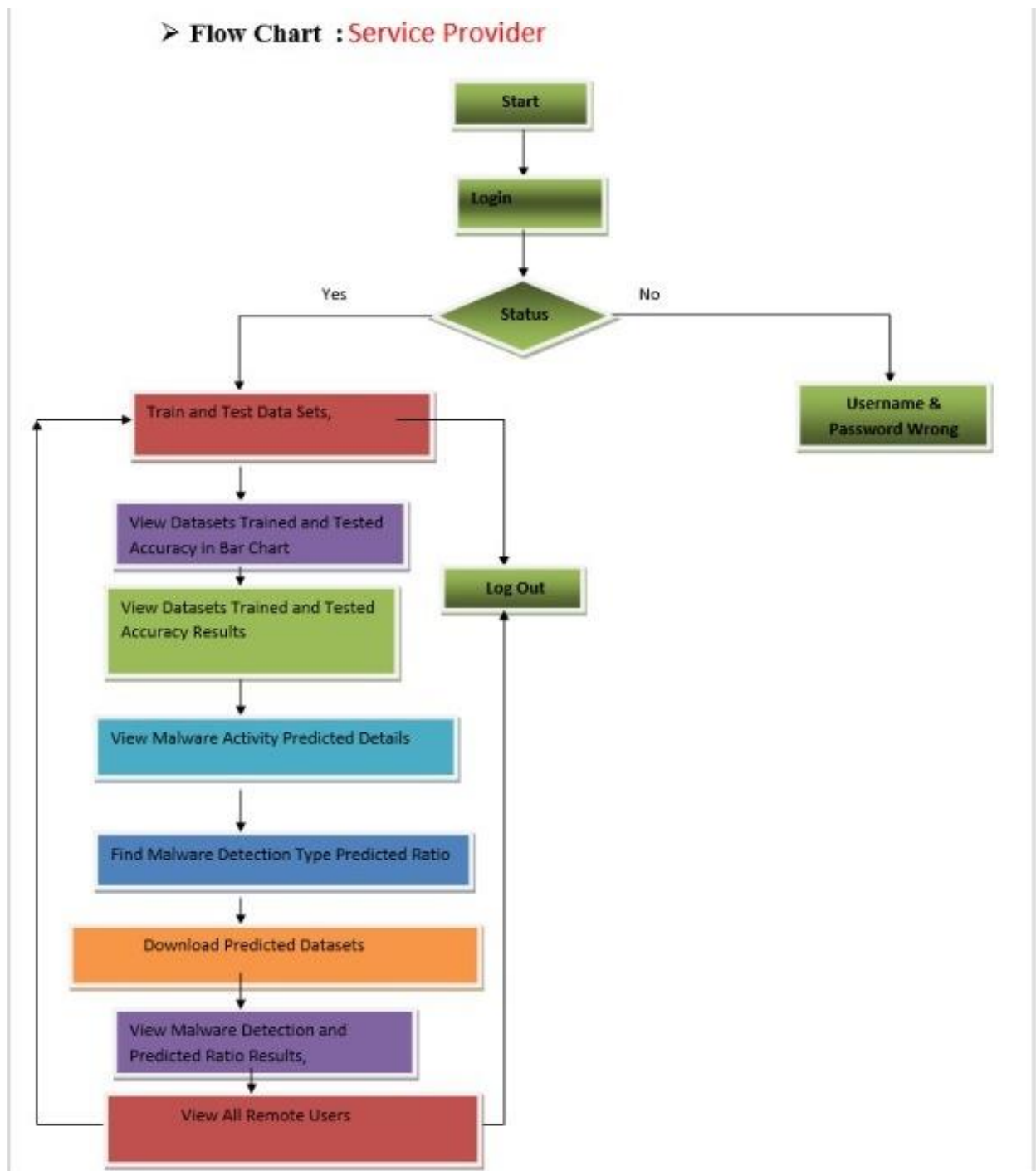


Fig 3.3.3: Remote User Flow Chart

**Fig 3.3.4: Service Provider Flow Chart**

DATA FLOW DIAGRAM

Data flow diagrams illustrate how data is processed by a system in terms of inputs and outputs. Data flow diagrams can be used to provide a clear representation of any business function. The technique starts with an overall picture of the business and continues by analyzing each of the functional areas of interest. This analysis can be carried out in precisely the level of detail required.

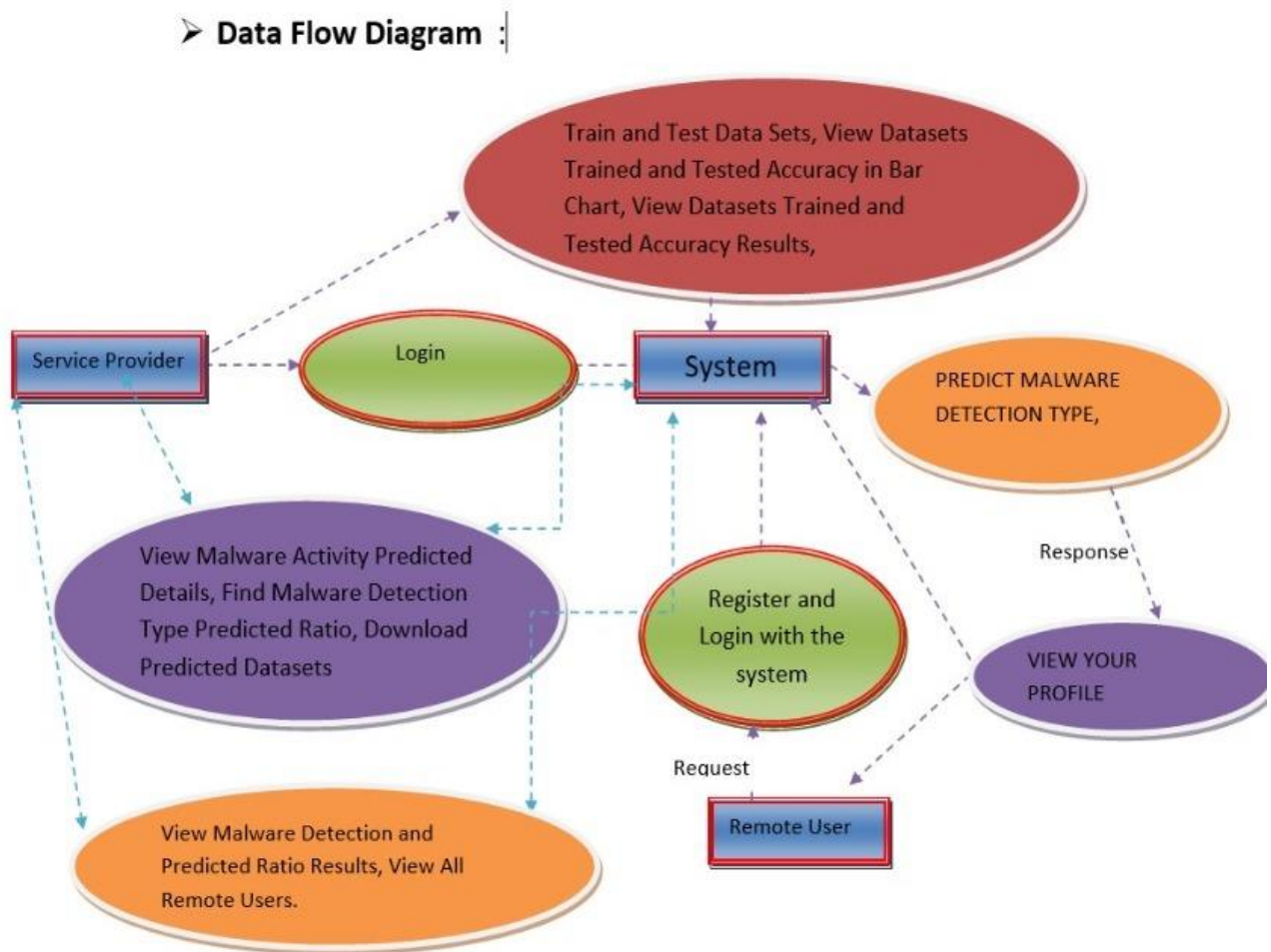


Fig 3.3.5: Data Flow Diagram

Manage.py:

```
#!/usr/bin/env python

"""Django's command-line utility for administrative tasks."""

import os

import sys

def main():

    """Run administrative tasks."""

    os.environ.setdefault('DJANGO_SETTINGS_MODULE',
'dark_tracer_early_detection.settings')

    try:

        from django.core.management import execute_from_command_line

    except ImportError as exc:

        raise ImportError(

            "Couldn't import Django. Are you sure it's installed and "

            "available on your PYTHONPATH environment variable? Did you "

            "forget to activate a virtual environment?"

        ) from exc

    execute_from_command_line(sys.argv)

if __name__ == '__main__':

    main()
```

Views.py:

- Views.py defines the the views in the browser such as admin/user according to the login.
- After logging in the pages are rendered dynamically according to privileges.

CODE:

```
from django.db.models import Count, Avg
from django.shortcuts import render, redirect
```

```
from django.db.models import Count
from django.db.models import Q
import datetime
import xlwt
from django.http import HttpResponse
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import re
from sklearn.ensemble import VotingClassifier

import warnings
warnings.filterwarnings("ignore")
plt.style.use('ggplot')
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.metrics import accuracy_score
from sklearn.metrics import f1_score

# Create your views here.
from Remote_User.models import
ClientRegister_Model,detection_type,detection_ratio,detection_accuracy

def serviceproviderlogin(request):
    if request.method == "POST":
        admin = request.POST.get('username')
        password = request.POST.get('password')
        if admin == "Admin" and password == "Admin":
            detection_accuracy.objects.all().delete()
            return redirect('View_Remote_Users')

    return render(request,'SProvider/serviceproviderlogin.html')

def Find_Malware_Detection_Type_Ratio(request):
    detection_ratio.objects.all().delete()
    ratio = ""
    kword = 'malware'
    print(kword)
```

```
obj = detection_type.objects.all().filter(Q(Prediction=kword))
obj1 = detection_type.objects.all()
count = obj.count();
count1 = obj1.count();
ratio = (count / count1) * 100
if ratio != 0:
    detection_ratio.objects.create(names=kword, ratio=ratio)

ratio1 = ""
keyword1 = 'legitimate'
print(keyword1)
obj1 = detection_type.objects.all().filter(Q(Prediction=keyword1))
obj11 = detection_type.objects.all()
count1 = obj1.count();
count11 = obj11.count();
ratio1 = (count1 / count11) * 100
if ratio1 != 0:
    detection_ratio.objects.create(names=keyword1, ratio=ratio1)

obj = detection_ratio.objects.all()
return render(request, 'SProvider/Find_Malware_Detection_Type_Ratio.html', {'objs': obj})

def View_Remote_Users(request):
    obj=ClientRegister_Model.objects.all()
    return render(request,'SProvider/View_Remote_Users.html',{'objects':obj})

def charts(request,chart_type):
    chart1 = detection_ratio.objects.values('names').annotate(dcount=Avg('ratio'))
    return render(request,"SProvider/charts.html", {'form':chart1, 'chart_type':chart_type})

def charts1(request,chart_type):
    chart1 = detection_accuracy.objects.values('names').annotate(dcount=Avg('ratio'))
    return render(request,"SProvider/charts1.html", {'form':chart1, 'chart_type':chart_type})

def Predict_Malware_Detection_Type_Details(request):

    obj =detection_type.objects.all()
    return render(request, 'SProvider/Predict_Malware_Detection_Type_Details.html',
{'list_objects': obj})
```

```
def likeschart(request,like_chart):
    charts =detection_accuracy.objects.values('names').annotate(dcount=Avg('ratio'))
    return render(request,"SProvider/likeschart.html", {'form':charts, 'like_chart':like_chart})

def Download_Predicted_DataSets(request):

    response = HttpResponse(content_type='application/ms-excel')
    # decide file name
    response['Content-Disposition'] = 'attachment; filename="Predicted_Datasets.xls"'
    # creating workbook
    wb = xlwt.Workbook(encoding='utf-8')
    # adding sheet
    ws = wb.add_sheet("sheet1")
    # Sheet header, first row
    row_num = 0
    font_style = xlwt.XFStyle()
    # headers are bold
    font_style.font.bold = True
    # writer = csv.writer(response)
    obj = detection_type.objects.all()
    data = obj # dummy method to fetch data.
    for my_row in data:
        row_num = row_num + 1
        ws.write(row_num, 0, my_row.url, font_style)
        ws.write(row_num, 1, my_row.length_url, font_style)
        ws.write(row_num, 2, my_row.length_hostname, font_style)
        ws.write(row_num, 3, my_row.https_token, font_style)
        ws.write(row_num, 4, my_row.page_rank, font_style)
        ws.write(row_num, 5, my_row.Prediction, font_style)
    wb.save(response)
    return response
def train_model(request):
    detection_accuracy.objects.all().delete()
    df = pd.read_csv('Malware_Datasets.csv', encoding='latin-1')

    def apply_results(status):
        if (status == "legitimate"):
            return 0 # legitimate
        elif (status == "malware"):
            return 1 # malware
    df['Results'] = df['status'].apply(apply_results)
```

```
# cv = CountVectorizer()
X = df['url']
y = df['Results']
cv = CountVectorizer(lowercase=False, strip_accents='unicode', ngram_range=(1, 1))
X = cv.fit_transform(X)
models = []
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20)
X_train.shape, X_test.shape, y_train.shape
print("Naive Bayes")
from sklearn.naive_bayes import MultinomialNB
NB = MultinomialNB()
NB.fit(X_train, y_train)
predict_nb = NB.predict(X_test)
naivebayes = accuracy_score(y_test, predict_nb) * 100
print(naivebayes)
print(confusion_matrix(y_test, predict_nb))
print(classification_report(y_test, predict_nb))
models.append(('naive_bayes', NB))
detection_accuracy.objects.create(names="Naive Bayes", ratio=naivebayes)
# SVM Model
print("SVM")
from sklearn import svm
lin_clf = svm.LinearSVC()
lin_clf.fit(X_train, y_train)
predict_svm = lin_clf.predict(X_test)
svm_acc = accuracy_score(y_test, predict_svm) * 100
print(svm_acc)
print("CLASSIFICATION REPORT")
print(classification_report(y_test, predict_svm))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test, predict_svm))
models.append(('svm', lin_clf))
detection_accuracy.objects.create(names="SVM", ratio=svm_acc)
print("Logistic Regression")
from sklearn.linear_model import LogisticRegression
reg = LogisticRegression(random_state=0, solver='lbfgs').fit(X_train, y_train)
y_pred = reg.predict(X_test)
print("ACCURACY")
print(accuracy_score(y_test, y_pred) * 100)
print("CLASSIFICATION REPORT")
print(classification_report(y_test, y_pred))
```



```
print("CONFUSION MATRIX")
    print(confusion_matrix(y_test, y_pred))
    models.append(('logistic', reg))
    detection_accuracy.objects.create(names="Logistic Regression",
ratio=accuracy_score(y_test, y_pred) * 100)
    from sklearn.tree import DecisionTreeClassifier
    print("Decision Tree Classifier")
    dtc = DecisionTreeClassifier()
    dtc.fit(X_train, y_train)
    dtcpredict = dtc.predict(X_test)
    print("ACCURACY")
    print(accuracy_score(y_test, dtcpredict) * 100)
    print("CLASSIFICATION REPORT")
    print(classification_report(y_test, dtcpredict))
    print("CONFUSION MATRIX")
    print(confusion_matrix(y_test, dtcpredict))
    models.append(('DecisionTreeClassifier', dtc))
    detection_accuracy.objects.create(names="Decision Tree Classifier",
ratio=accuracy_score(y_test, dtcpredict) * 100)
    print("KNeighborsClassifier")
    from sklearn.neighbors import KNeighborsClassifier
    kn = KNeighborsClassifier()
    kn.fit(X_train, y_train)
    knpredict = kn.predict(X_test)
    print("ACCURACY")
    print(accuracy_score(y_test, knpredict) * 100)
    print("CLASSIFICATION REPORT")
    print(classification_report(y_test, knpredict))
    print("CONFUSION MATRIX")
    print(confusion_matrix(y_test, knpredict))
    models.append(('KNeighborsClassifier', kn))
    detection_accuracy.objects.create(names="KNeighborsClassifier",
ratio=accuracy_score(y_test, knpredict) * 100)
    classifier = VotingClassifier(models)
    classifier.fit(X_train, y_train)
    y_pred = classifier.predict(X_test)

predicts = 'Labeled_Data.csv'
# df['predict_nb'] = predict_text
df.to_csv(predicts, index=False)
df.to_markdown
```

```
obj = detection_accuracy.objects.all()
return render (request,'SProvider/train_model.html', {'objs': obj})
```

Html Codes:

- The html codes provide a structure for the dynamic rendering and these are the pages seen on the browser.

Code:

```
{% extends 'SProvider/design1.html' %}
{% block researchblock %}
<link rel="icon" href="images/icon.png" type="image/x-icon" />
  <link href="https://fonts.googleapis.com/css?family=Lobster" rel="stylesheet">
  <link href="https://fonts.googleapis.com/css?family=Righteous" rel="stylesheet">
<link href="https://fonts.googleapis.com/css?family=Fredoka+One" rel="stylesheet">
  <style>
    body {background-color:#000000;}
    .container-fluid {padding:50px;}
    .container{background-color:white;padding:50px; }
    #title{ font-family: 'Fredoka One', cursive;
  }
    .text-uppercase{
      font-family: 'Righteous', cursive;
    }
    .tweettext{
      border: 2px solid yellowgreen;
      width: 1104px;
      height: 442px;
      overflow: scroll;
      background-color:;
    }
    .style1 {
      color: #FF0000;
      font-weight: bold;
    }
    .style9 {color: #FFFF00}
  </style>
<body>
<div class="container-fluid">
  <div class="container">
    <div class="row">
      <div class="col-md-5">
        <form role="form" method="POST" >
          {% csrf_token %}
          <fieldset>
```

Of Malware Activity Details !!! </p>

```

    <hr>
    <div class="tweettext">
        <table border="5" bordercolor="#FF00FF">

            <tr>
                <td bgcolor="#FF0000"><span class="style8 style6">url</span></td>
                <td bgcolor="#FF0000"><span class="style8 style6">length_url</span></td>
                <td bgcolor="#FF0000"><span class="style8 style6">length_hostname</span></td>
                <td bgcolor="#FF0000"><span class="style8 style6">https_token</span></td>
                <td bgcolor="#FF0000"><span class="style8 style6">page_rank</span></td>
                <td bgcolor="#FF0000"><span class="style8 style6">Prediction </span></td>
            </tr>

            { % for object in list_objects % }
            <tr>
                <td bgcolor="#FFFFFF" style="color:red; font-size:20px; font-family:fantasy" ><div
                    align="center">{ { object.url } }</div></td>
                <td bgcolor="#FFFFFF" style="color:red; font-size:20px; font-family:fantasy" ><div
                    align="center">{ { object.length_url } }</div></td>
                <td bgcolor="#FFFFFF" style="color:red; font-size:20px; font-family:fantasy" ><div
                    align="center">{ { object.length_hostname } }</div></td>
                <td bgcolor="#FFFFFF" style="color:red; font-size:20px; font-family:fantasy" ><div
                    align="center">{ { object.https_token } }</div></td>
                <td bgcolor="#FFFFFF" style="color:red; font-size:20px; font-family:fantasy" ><div
                    align="center">{ { object.page_rank } }</div></td>
                <td bgcolor="#FFFFFF" style="color:red; font-size:20px; font-family:fantasy" ><div
                    align="center">{ { object.Prediction } }</div></td>
            </tr>

            { % endfor % }

        </table>
    </div>
</fieldset>
</form>
</div>
<div class="col-md-2">
    <!--null-->
</div>
</div>
</div>
</div>
{ % endblock % }
<tr>

```

CHAPTER 4

RESULTS AND DISCUSSION

CHAPTER 4

RESULTS AND DISCUSSION

4.1 OUTPUT SCREENS:

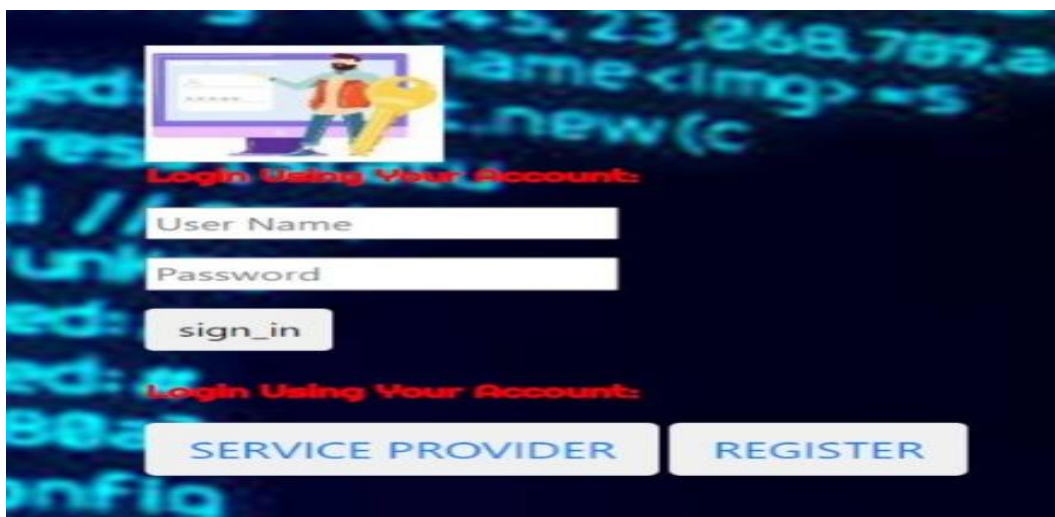


Fig 4.1: LOGIN INTERFACE

- The above is the user login interface where the user should login using their credentials



Fig 4.2: USER REGISTRATION INTERFACE

- The above image is the user registration page where the user first register with their valid details.

Enter url

http://shadetretechnology.com/V4/validation/a111aedc8ae390eabcf a130e041a10a4

Enter length_url

77

Enter length_hostname

23

Enter https_token

1

Enter page_rank

2

Predict

Detection and Prediction Type :

Fig 4.3: PREDICTION INTERFACE

- The above is the prediction interface where the user should paste their link to identify whether it is malicious or not.

Dark-TRACER Early Detection Framework for Malware Activity Based on Anomalous Spatiotemporal Patterns Using XGBoost

Train and Test Data Sets View Datasets Trained and Tested Accuracy in Bar Chart View Datasets Trained and Tested Accuracy Results View Malware Activity Predicted Details Find Malware Detection Type Predicted Ratio

Download Predicted Datasets View Malware Detection and Predicted Ratio Results View All Remote Users Logout

VIEW ALL REMOTE USERS !!!

USER NAME	EMAIL	Gender	Address	Mob No	Country	State	City
Harish	Harish123@gmail.com	Male	#892,4th Cross,Rajajinagar	9535866270	India	Karnataka	Bangalore
Manjunath	tmksmanju13@gmail.com	Male	#892,4th Cross,Malleswaram	9535866270	India	Karnataka	Bangalore
tmksmanju	tmksmanju13@gmail.com	Male	#892,4th Cross,Rajajinagar	9535866270	India	Karnataka	Bangalore
janani	janani@gmail.com	Female	kandlakoya	9966339966	India	Telangana	Hyderabad
praveen	praveen123@gmail.com	Male	cmr	9391041988	india	telangana	hyderabad
mounika	mounika123@gmail.com	Female	cmr	7075650702	india	telangana	hyderabad

Fig 4.4: PREDICTION INTERFACE

- The above screen shows all the users registered to predict their link.

View Dark TRACER Early Detection Of Malware Activity Details !!!

	length_url	length_hostname	https_token	page_rank	Prediction
https://support- m.secureupdate.duilawyeryork.com/ap/89e6a3b4b063b8d/? nd=_update&dispatch=89e6a3b4b063b8d1ba.locale=_	126	50	0	0	malware
http://www.mutuo.it	19	12	1	1	legitimate
http://wave.progressfilm.co.uk/time3/?logon=myposte	51	23	1	4	malware
appleid.com.secureupdate.duilawyeryork.com/ap/bb14d7ff1fcbf29? nd=_update&dispatch=bb14d7ff1fcbf29bb&locale=_us	126	50	1	0	malware
http://yummy-cummy-in-my-tummy.tumblr.com	41	34	1	8	legitimate
http://wave.progressfilm.co.uk/time3/?logon=myposte	51	23	1	4	malware
http://appleid.apple.com-app.es/	32	24	1	0	malware
http://www.crestonwood.com/router.php	37	19	1	4	legitimate
http://www.crestonwood.com/router.php	37	19	1	4	legitimate
http://www.crestonwood.com/router.php	37	19	1	4	legitimate
technology.com/V4/validation/a111aedc8ae390eabcfa130e041a10a4	77	23	1	2	malware

Fig 4.5: PREDICTED RESULTS

- The above is the predicted results screen where admin have the access to check the links which are predicted by users

CHAPTER 5

CONCLUSION

CONCLUSION

- our project has demonstrated the efficacy of multiple machine learning algorithms, including XGBoost, logistic regression, random forest, and others, in the realm of cyber threat detection within dark net traffic.
- By synthesizing the strengths of these diverse approaches and integrating them into our comprehensive system, DarkTRACER, we have achieved a potent solution for the early identification of malware-induced scanning attacks.
- Through rigorous data preprocessing, feature engineering, and model tuning, we have cultivated a robust framework capable of discerning subtle patterns indicative of cyber threats, thereby contributing to the preservation of internet security.
- Looking ahead, our focus remains on continual refinement and adaptation, leveraging insights from real-world deployments to stay ahead of evolving cyber threats and safeguard the integrity of our digital infrastructure.

FUTURE SCOPE

- **Advanced Feature Engineering:** Find new ways to understand cyber threats by creating more detailed characteristics from dark net traffic data.
- **Incorporation of Deep Learning Models:** Use advanced deep learning technology like CNNs and RNNs to make the system better at detecting complex cyber threats.
- **Real-time Threat Intelligence Integration:** Add features to keep the system updated with the latest information about cyber threats in real-time, making it more effective at identifying new threats.
- **Continuous Model Refinement:** Make the system smarter over time by constantly improving its ability to detect cyber threats using techniques like online learning and ensemble methods.
- **Scalability and Deployment Optimization:** Ensure the system can handle large amounts of data efficiently and can be easily deployed in different organizations and network environments.

REFERENCES

REFERENCES

- [1] C. Han, J. Takeuchi, T. Takahashi, and D. Inoue, "Automated detection of malware activities using nonnegative matrix factorization," in Proc. IEEE Int. Conf. Trust, Secure. Privacy Comput. Commun. (TrustCom), Oct. 2021.
- [2] C. Han, J. Shimamura, T. Takahashi, D. Inoue, J. Takeuchi, and K. Nakao, "Real-time detection of global cyberthreat based on darknet by estimating anomalous synchronization using graphical lasso," IEICE Trans. Inf. Syst., vol. 103, no. 10, pp. 2113_2124, Oct. 2020.
- [3] H. Kanehara, Y. Murakami, J. Shimamura, T. Takahashi, D. Inoue, and N. Murata, "Real-time botnet detection using nonnegative tucker decomposition," in Proc. 34th ACM/SIGAPP Symp. Appl. Comput., Apr. 2019, pp. 1337_1344.
- [4] H. Grifoen and C. Doerr, "Examining Mirai's battle over the Internet of Things," in Proc. ACM SIGSAC Conf. Comput. Commun. Secur., Oct. 2020, pp. 743_756.
- [5] D. Cohen, Y. Mirsky, M. Kamp, T. Martin, Y. Elovici, R. Puzis, and A. Shabtai, "DANTE: A framework for mining and monitoring darknet traffic," in Proc. 25th Eur. Symp. Res. Comput. Secur. (ESORICS). Springer, 2020, pp. 88_109.
- [6] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, and Y. and Zhou, "Understanding the Mirai botnet," in Proc. 26th USENIX Secur. Symp., 2017, pp. 1093_1110.
- [7] FOX-IT. (2016). Recent Vulnerability in Eir D1000 Router Used to Spread Updated Version of Mirai DDoS Bot. [Online]. Available: [https://blog.foxit.com/2016/11/28/recent-vulnerability-in-eir-d1000-ro%uter-used-to-spread-updated-version-of-miraiddos-bot/](https://blog.foxit.com/2016/11/28/recent-vulnerability-in-eir-d1000-router-used-to-spread-updated-version-of-miraiddos-bot/)

- [8] Anandkumar, P. Jain, Y. Shi, and U. N. Niranjan, "Tensor vs. matrix methods: Robust tensor decomposition under block sparse perturbations," in Proc. 19th Int. Conf. Artif. Intell. Statist., (AISTATS), vol. 51, 2016, pp. 268_276.
- [9] Liu, T. Suzuki, and M. Sugiyama, "Support consistency of direct sparsechange learning in Markov networks," in Proc. 29th AAAI Conf. Artif. Intell., 2015, pp. 2785_2791.
- [10] Durumeric, D. Adrian, A. Mirian, M. Bailey, and J.A. Halderman, "A search engine backed by internetwide scanning," in Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur., 2015, pp. 542_553.
- [11] Gibberd and J. D. B. Nelson, "High dimensional changepoint detection with a dynamic graphical lasso," in Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP), May 2014, pp. 2684_2688.
- [12] Zhao, C. F. Caiafa, D. P. Mandic, L. Zhang, T. Ball, A. Schulze-Bonhage, and A. Cichocki, "Multilinear subspace regression: An orthogonal tensor decomposition approach," in Proc. 25th Annu. Conf. Neural Inf. Process. Syst., 2011, pp. 1269_1277.
- [13] Phan and A. Cichocki, "Tensor decompositions for feature extraction and classification of high dimensional datasets," IEICE Nonlinear Theory Appl., vol. 1, no. 1, pp. 37_68, 2010.
- [14] Caiafa and A. Cichocki, "Generalizing the column_row matrix decomposition to multi-way arrays," Linear Algebra its Appl., vol. 433, no. 3, pp. 557_573, Sep. 2010.
- [15] Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," IEEE Comput., vol. 42, no. 8, pp. 30_37, Aug. 2009

GITHUB LINK:

- [https://github.com/VenkataSaiNathaReddyVaddi/Dark tracer Early Detection-using-spatiotemporal-patterns](https://github.com/VenkataSaiNathaReddyVaddi/Dark_tracer_Early_Detection-using-spatiotemporal-patterns)

DOI LINK:

- <https://doi.org/10.22214/ijraset.2024.59304>

RESEARCH PAPER AND CERTIFICATION



IJRASET

International Journal For Research in
Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 12 **Issue:** III **Month of publication:** March 2024

DOI: <https://doi.org/10.22214/ijraset.2024.59337>

www.ijraset.com

Call: ☎ 08813907089

E-mail ID: ijraset@gmail.com



Dark Tracer Early Malware Detection Based on Spatiotemporal Patterns Using Xgboost Algorithms

Ms. A. Mounika Rajeswari¹, Janani Chalapati², V. Venkata Sai Natha Reddy³, Mohammed Awais Khan⁴

UG Student, Department of Computer Science & Engineering, CMR College of Engineering & Technology, Hyderabad, India

Abstract: Cyber assaults are on the rise throughout the world, therefore it's important to spot patterns so we can respond appropriately. Due to the lack of genuine communication on the darknet, an underused area for IP addresses, it is very easy to observe and analyse random cyber assaults. Similar spatiotemporal patterns are commonly seen in malware's indiscriminate scanning efforts, which are used to propagate infestations. These tendencies are also detected on the darknet. Our main emphasis is on abnormal spatiotemporal examples seen in darknet traffic information to handle the issue of early malware movement discovery. In our earlier research, we suggested algorithms that use three separate machine learning techniques to automatically predict and identify real-time aberrant spatiotemporal patterns of darknet traffic. In this exploration, we coordinated all of the beforehand suggested approaches into a unified framework called Dark-TRACER and tested its detection capabilities for various malware behaviours using quantitative tests. We used data collected from our large-scale darknet sensors, which cover the period from October 2018 to October 2020, to analyse darknet activity at subnet sizes of up to /17. The findings show that the approaches' shortcomings operate together, and the suggested framework has a 100% recall rate overall. On top of that, unlike trustworthy third-party security research organisations, Dark-TRACER finds malware activities an average of 153.6 days before they are publicised. Lastly, we calculated how much it would cost to employ human analysts to put the suggested system into action, and we proved that it would take around seven and a half hours for two analysts to carry out all the routine tasks required to run the framework.

Keywords: Anomalous synchronization estimation, darknet, malware activity, spatiotemporal pattern.

I. INTRODUCTION

The escalating frequency and complexity of cyber attacks pose significant challenges to internet security, necessitating the identification and mitigation of malware-induced scanning assaults. To address this, our project focuses on detecting patterns of cyber attacks globally and promptly identifying malware-induced indiscriminate scanning attacks before they propagate extensively. Leveraging dark net analysis, we exploit the distinct signal-to-noise ratio of non-targeted scanning communications to detect cyber threats effectively.

Despite the abundance of legitimate communication on typical networks, the use of "dark nets" enables the identification of suspicious activities, as non-targeted scanning communications stand out amidst genuine traffic. This approach facilitates the detection of cyber threats by highlighting anomalous patterns in communication. However, the exponential growth of traffic on the dark web presents challenges in distinguishing between benign and malicious activities, underscoring the need for advanced detection techniques.

Our research emphasizes the importance of synchronised spatiotemporal patterns in identifying malware activity. By analysing dark net traffic data, we employ machine learning approaches such as graphical tether, nonnegative matrix factorization (NMF), and nonnegative Tucker decomposition (NTD) to gauge synchronisation and detect potential threats. These techniques enable early identification of malware activity, even in cases of small-scale infection activity.

The integration of these methodologies culminates in DarkTRACER, a comprehensive system capable of detecting cyber threats with high accuracy. Through rigorous testing, DarkTRACER exhibits remarkable recall rates, identifying threats an average of 153.6 days before public disclosure. Moreover, its efficiency allows for practical deployment in real-world scenarios, supporting organisations such as Security Operation Centres (SOCs) and Computer Security Incident Response Teams (CSIRTs) in safeguarding against cyber threats worldwide.

In conclusion, our project underscores the critical importance of early detection and mitigation of cyber threats. By leveraging advanced techniques and machine learning algorithms, we have developed DarkTRACER, a sophisticated system capable of identifying and responding to cyber threats proactively. This research represents a significant advancement in the field of cybersecurity, providing organisations with the tools and insights needed to protect against evolving cyber threats effectively.

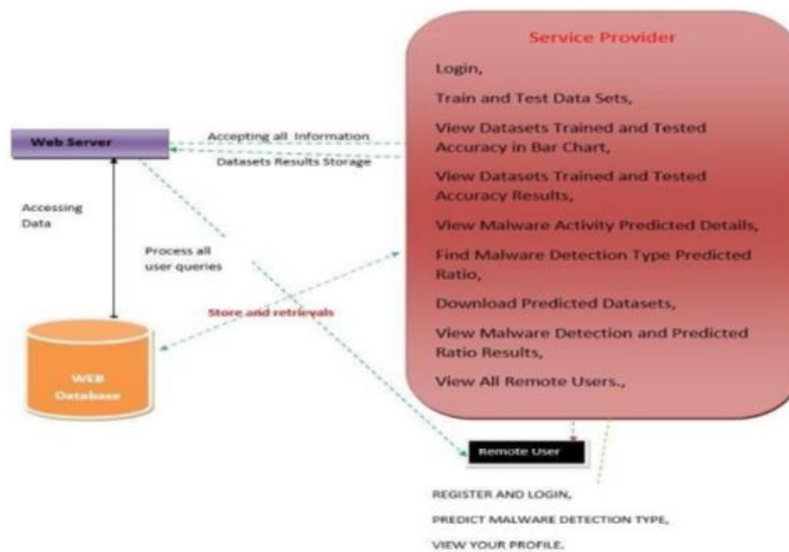


Fig 1: Architecture

II. RELATED WORK

- 1) *Cybersecurity Threat Landscape Analysis*: The increasing frequency and complexity of cyber attacks pose significant challenges to internet security. The sheer volume of random cyber assaults reported in recent years underscores the urgency in identifying and addressing these threats to safeguard online infrastructure
- 2) *Leveraging Dark Nets for Threat Detection*: Recognizing the difficulty in identifying malware scanning attacks amidst legitimate network traffic, researchers turned to "dark nets," unutilized IP address areas, for potential solutions. By distinguishing between genuine communication and non-targeted scanning activities in these environments, researchers aimed to enhance signal-to-noise ratios and improve threat detection capabilities.
- 3) *Spatiotemporal Analysis for Malware Detection*: Studies have shown that malware-induced indiscriminate scanning attacks often exhibit synchronized spatiotemporal patterns. By analyzing the synchronicity of network traffic across various hosts and ports, researchers sought to develop early detection methods for identifying potential malware activities, even in instances of small-scale infection.
- 4) *Machine Learning Approaches for Detection*: Previous research explored the use of machine learning techniques such as graphical tether, nonnegative matrix factorization (NMF), and nonnegative Tucker decomposition (NTD) to gauge synchronization in spatiotemporal models derived from dark net traffic data. These methods aimed to differentiate between normal and aberrant synchronization patterns indicative of malicious activity.
- 5) *Development of DarkTRACER*: To address the limitations of existing approaches, researchers integrated multiple methods into a comprehensive system named DarkTRACER. By unifying common elements and modularizing previous approaches, DarkTRACER aimed to enhance the robustness and effectiveness of malware detection, particularly in early identification.
- 6) *Evaluation Studies*: DarkTRACER underwent rigorous evaluation to assess its performance and feasibility. Studies focused on quantitative disclosure and early identification judgment using real-world malware datasets. Results demonstrated high recall rates and significant lead time in threat identification compared to public disclosures.

III. METHODS AND EXPERIMENTAL DETAILS

- 1) **Decision Tree Classifier**: A decision tree classifier is a machine learning algorithm that divides a dataset into smaller subsets based on the importance of input features. This partitioning process is performed iteratively; The goal of each partition is to increase the homogeneity of the resulting subset with a different target (e.g., class list). By creating a tree model of the decision node, each decision node represents the specific and corresponding decision, the decision tree moves the tree from roots to leaves, facilitating the splitting of new events. In this project, a decision tree classifier is used to identify cyber attack patterns by identifying key features associated with identifying the occurring activities of malware. This allows the system to detect and respond to emerging threats in a timely manner.



- 2) **Gradient Boosting:** Gradient boosting is an ensemble learning technique that creates a robust prediction model by combining multiple weak learners (usually decision trees). With each iteration, gradient boosting focuses on the mistakes made by the previous model, learns from them, and improves overall model performance. By optimizing the predictive model, gradient boosting improves the ability to capture complex patterns and relationships in data, increasing prediction accuracy. This project uses gradient boosting technology to increase the accuracy of malware detection by creating a robust classification model that can identify subtle patterns in information that indicate a network threat.
- 3) **K-Neighbor (KNN):** K-Neighbor Neighbor (KNN) is a non-parametric classification algorithm used for pattern recognition and classification functions. This algorithm works by classifying new data according to the classes of most of its nearest neighbors at a given location. KNN model building should not include any information; instead it stores all data points and calculates the distance between them to identify neighbors. In this project, KNN can play a role in classifying dark web data by comparing its similarity with neighboring data. KNN helps detect network threats by detecting nearest neighbors identifying malicious activity on the network.
- 4) **Logistic Regression Classifier:** Logistic regression classifier is a method used to classify binary functions that have only two possible values for different purposes. The algorithm models the probability of a binary outcome based on one or more predictors, using a logistic function to constrain the predicted value between 0 and 1. Logistic regression estimates the coefficients of the predictor variables that represent the effect of each variable on the probability of the outcome. This project will use a logistic regression classifier to predict the probability of network activity associated with malicious behavior, thus helping in the early detection and mitigation of network threats.
- 5) **Random Forest:** Random Forest is a learning technique that creates multiple decision trees and makes the final classification by combining their predictions. Each decision tree in a random forest is trained on a random subset of the training data and a random subset of features, reducing the risk of overfitting and improving generalization. The final prediction of a random forest is determined by summing the predictions of all trees, usually by majority vote. This project used Random Forest to identify network attack patterns in dark web data to increase the robustness and accuracy of malware detection.
- 6) **Naive Bayes:** Naive Bayes is a distribution method based on Bayes theorem and with the assumption of independence. Although the assumption is simple, Naive Bayes is used specifically for classification of texts and is known for its simplicity and performance. The algorithm calculates the probability of each class given input and selects the class with the highest probability based on the list of predicted classes. In this project, Naive Bayes can be used to detect inconsistencies in traffic from malicious messages and helps detect cyber threats at an early stage using probability testing of the algorithm.
- 7) **Support Vector Machine (SVM):** Support Vector Machine (SVM) is a powerful supervised learning algorithm used for classification and regression. The working principle of SVM is to find a general plane in high-dimensional space that can separate different groups of data points. This algorithm aims to separate the support vectors (i.e. the data points closest to the decision boundary) and thus improve the performance of the model. In this project SVM played an important role in detecting complex patterns in cyber attacks by effectively separating the attack-related information content from the network under normal conditions.
- 8) **XGBoost:** XGBoost is an optimization of gradient boosting, known for its scalability and efficiency in processing big data. Similar to gradient boosting, XGBoost sequentially generates multiple weak learners (usually decision trees) and combines their predictions to make the final classification. However, XGBoost includes various optimizations such as parallelization and tree pruning to increase training speed and model performance. In this project, XGBoost was used to improve the ability to detect and mitigate cyber threats by improving classification models and preserving subtle patterns in information that indicate malware activity.

Model Type	Accuracy
Naive Bayes	89.85126859142608
Logistic Regression	90.5949256342957
Decision Tree Classifier	88.4514435695538
KNeighborsClassifier	82.28346456692913
XGBClassifier	92.001312335958

Table: Metrics

IV. IMPLEMENTATION AND BLOCK DIAGRAM

A. Data Collection and Exploration

- 1) Gather dark net traffic data, including scanning activities and communication patterns.
- 2) Explore dataset characteristics, identify missing values, and assess feature relevance.

B. Data Preprocessing

- 1) Clean data by handling missing values and outliers.
- 2) Encode variables and standardize numerical features.
- 3) Split data into training and testing sets.

C. Model Selection

- 1) Choose XGBoost for its ability to handle complex data relationships.
- 2) Define target variable and train XGBoost model on training data.

D. Hyperparameter Tuning:

- 1) Fine-tune XGBoost parameters to optimize performance and prevent overfitting.

E. Training the Model

- 1) Train XGBoost model to learn patterns indicative of cyber threats in dark net traffic.

F. Evaluation

- 1) Evaluate model performance using metrics like accuracy and recall on testing data.

G. Interpretability

- 1) Analyze feature importance to identify key indicators of malicious behavior.

H. Deployment

- 1) Deploy model for real-time detection of cyber threats in dark net traffic.
- 2) Continuously refine model based on performance feedback.

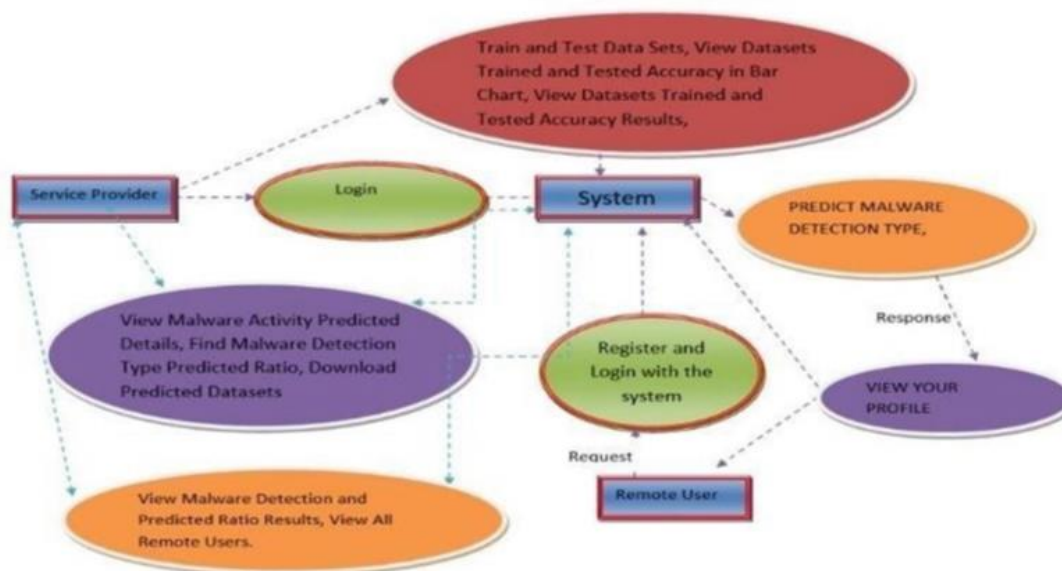


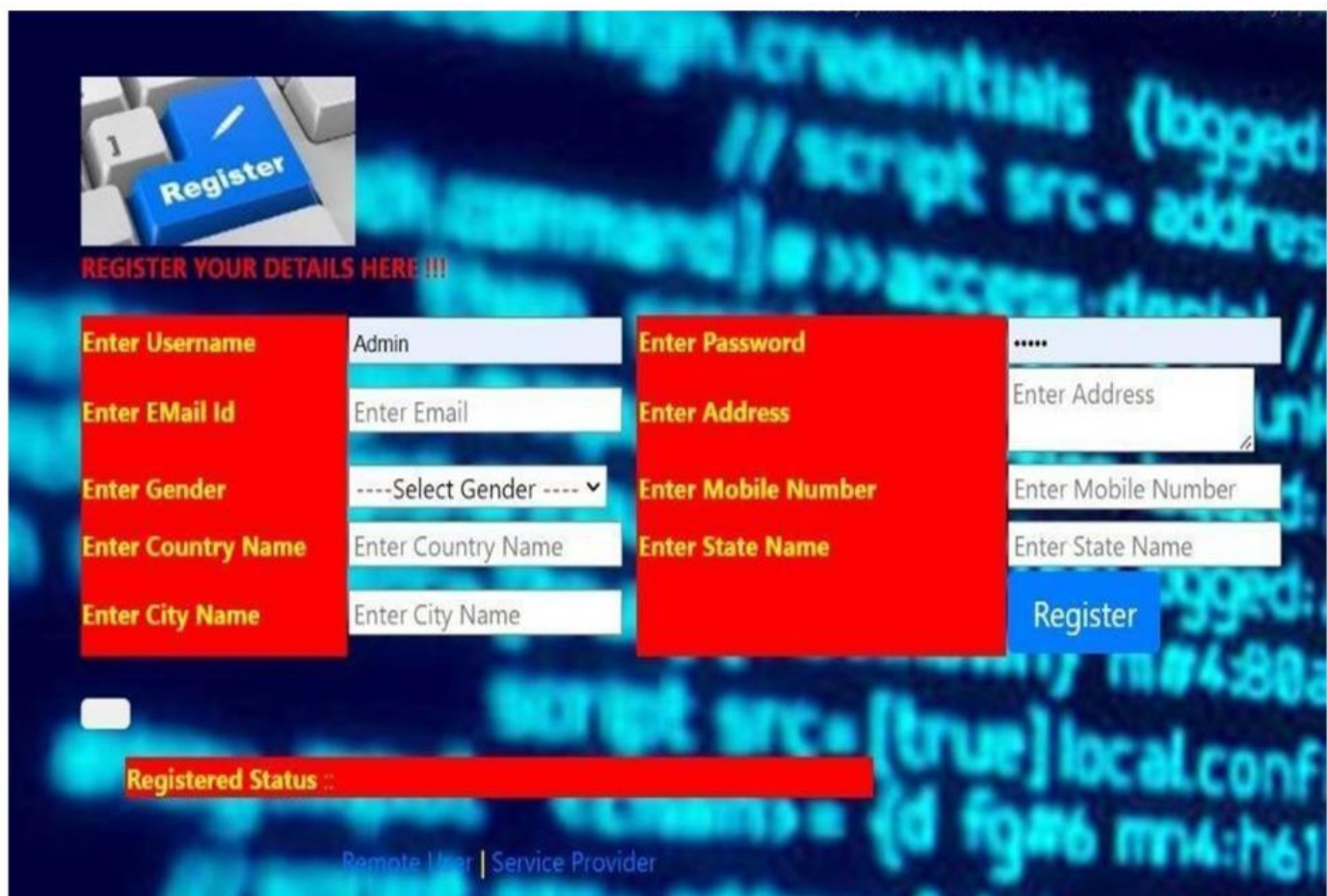
Fig 2: Block diagram

V. INTERFACES



The login interface features a dark blue background with faint, glowing green code snippets. At the top left, there is an illustration of a person in a red shirt and yellow pants standing next to a computer monitor. Below this, the text "Login Using Your Account:" is displayed in red. There are two input fields: "User Name" and "Password". A "sign_in" button is located below the password field. Below the sign_in button, the text "Login Using Your Account:" is repeated in red. At the bottom, there are two buttons: "SERVICE PROVIDER" and "REGISTER".

Fig 3: Login interface



The register interface has a dark blue background with faint, glowing green code snippets. At the top left, there is an illustration of a blue keyboard key with the word "Register" on it. Below this, the text "REGISTER YOUR DETAILS HERE !!!" is displayed in red. The form is divided into two main sections. The left section has a red background and contains the following labels and input fields: "Enter Username" (with "Admin" entered), "Enter EMAIL Id" (with "Enter Email" entered), "Enter Gender" (with a dropdown menu showing "----Select Gender ----"), "Enter Country Name" (with "Enter Country Name" entered), and "Enter City Name" (with "Enter City Name" entered). The right section has a red background and contains the following labels and input fields: "Enter Password" (with "****" entered), "Enter Address" (with "Enter Address" entered), "Enter Mobile Number" (with "Enter Mobile Number" entered), "Enter State Name" (with "Enter State Name" entered), and a "Register" button. At the bottom left, there is a "Registered Status ::" label. At the bottom center, there is a "Remote User | Service Provider" label.

Fig 4: Register Interface

Enter url

Enter length_url

Enter length_hostname

Enter https_token

Enter page_rank

Predict

http://shadetreetechnology.com/V4/validation/a111aedc8ae390eabcf a130e041a10a4

77

23

1

2

Detection and Prediction Type :

Fig 5: Prediction Interface

Dark-TRACER Early Detection Framework for Malware Activity Based on Anomalous Spatiotemporal Patterns Using XGBoost

[Train and Test Data Sets](#)
[View Datasets Trained and Tested Accuracy in Bar Chart](#)
[View Datasets Trained and Tested Accuracy Results](#)
[View Malware Activity Predicted Details](#)
[Find Malware Detection Type Predicted Ratio](#)

[Download Predicted Datasets](#)
[View Malware Detection and Predicted Ratio Results](#)
[View All Remote Users](#)
[Logout](#)

VIEW ALL REMOTE USERS !!!

USER NAME	EMAIL	Gender	Address	Mob No	Country	State	City
Harish	Harish123@gmail.com	Male	#892,4th Cross,Rajajinagar	9535866270	India	Karnataka	Bangalore
Manjunath	tmksmanju13@gmail.com	Male	#892,4th Cross,Malleshwaram	9535866270	India	Karnataka	Bangalore
tmksmanju	tmksmanju13@gmail.com	Male	#892,4th Cross,Rajajinagar	9535866270	India	Karnataka	Bangalore
janani	janani@gmail.com	Female	kandlakoya	9966339966	India	Telangana	Hyderabad
praveen	praveen123@gmail.com	Male	cmr	9391041988	india	telengana	hyderabad
mounika	mounika123@gmail.com	Female	cmr	7075650702	india	telengana	hyderabad

Fig 6: User Details

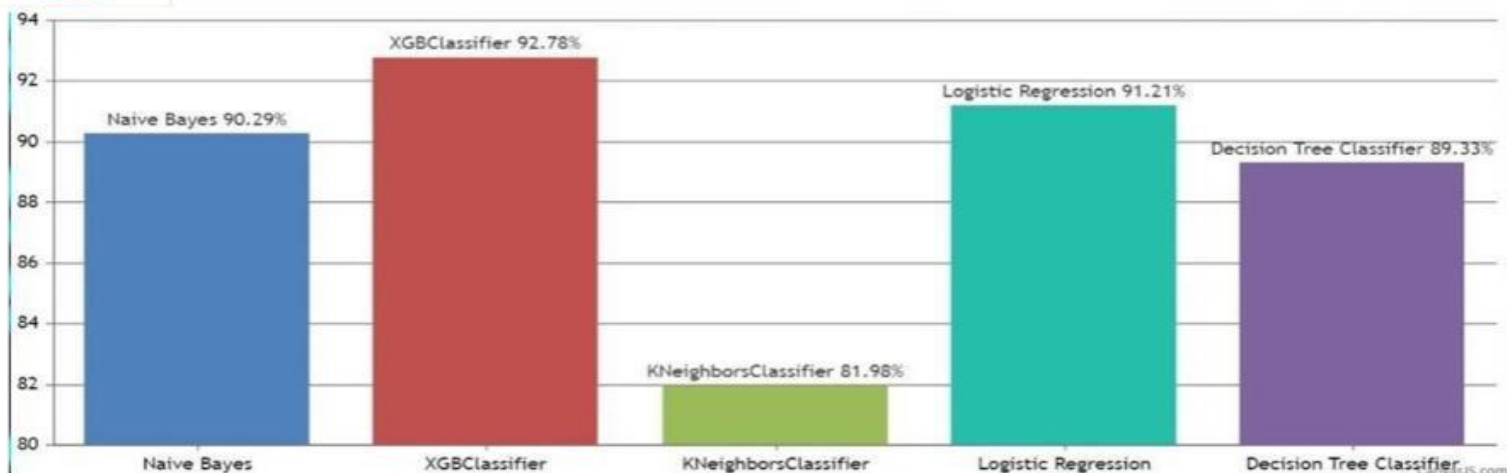


Fig 7: Comparison Bar Chart

VI. RESULTS AND DISCUSSION

We utilize a suite of powerful algorithms including XGBoost, Support Vector Regression (SVR), and Decision Trees to assess the authenticity of each link. Through rigorous analysis, we determine the genuineness of each link, providing a transparent evaluation process. On our website, you'll find the authenticity of links displayed in the form of percentages, derived from the comprehensive examination conducted by these algorithms. This ensures that users can trust the links they encounter, fostering a safe and reliable online environment.

[View Dark TRACER Early Detection Of Malware Activity Details !!!](#)

	length_url	length_hostname	https_token	page_rank	Prediction
https://support-m.secureupdate.duilawyeryork.com/ap/89e6a3b4b063b8d/?id=_update&dispatch=89e6a3b4b063b8d1ba&locale=_	126	50	0	0	malware
http://www.mutuo.it	19	12	1	1	legitimate
http://wave.progressfilm.co.uk/time3/?logon=myposte	51	23	1	4	malware
appleid.com.secureupdate.duilawyeryork.com/ap/bb14d7ff1fcbf29?nd=_update&dispatch=bb14d7ff1fcbf29bba&locale=_us	126	50	1	0	malware
http://yummy-cummy-in-my-tummy.tumblr.com	41	34	1	8	legitimate
http://wave.progressfilm.co.uk/time3/?logon=myposte	51	23	1	4	malware
http://appleid.apple.com-app.es/	32	24	1	0	malware
http://www.crestonwood.com/router.php	37	19	1	4	legitimate
http://www.crestonwood.com/router.php	37	19	1	4	legitimate
http://www.crestonwood.com/router.php	37	19	1	4	legitimate
technology.com/V4/validation/a111aedc8ae390eabcfaf130e041a10a4	77	23	1	2	malware

Fig 8: Predicted Results

VII. CONCLUSION

Our project has demonstrated the efficacy of multiple machine learning algorithms, including XGBoost, logistic regression, random forest, and others, in the realm of cyber threat detection within dark net traffic. By synthesizing the strengths of these diverse approaches and integrating them into our comprehensive system, DarkTRACER, we have achieved a potent solution for the early identification of malware-induced scanning attacks. Through rigorous data preprocessing, feature engineering, and model tuning, we have cultivated a robust framework capable of discerning subtle patterns indicative of cyber threats, thereby contributing to the preservation of internet security. Looking ahead, our focus remains on continual refinement and adaptation, leveraging insights from real-world deployments to stay ahead of evolving cyber threats and safeguard the integrity of our digital infrastructure.



REFERENCES

- [1] Lavecchia, "Deep learning in drug discovery: opportunities, challenges and future prospects," *Drug Discovery Today*, 2019.
- [2] Karimi, D. Wu, Z. Wang, and Y. Shen, "DeepAffinity: interpretable deep learning of compound–protein affinity through unified recurrent and convolutional neural networks," *Bioinformatics*, vol. 35, no. 18, pp. 3329–3338, 2019.
- [3] Tan, O. F. O'zgu' l, B. Bardak, I. Eksio' lu, and S. Sabuncuo' glu, "Drug response prediction by ensemble learning and drug-induced gene expression signatures," *Genomics*, vol. 111, no. 5, pp. 1078–1088, 2019.
- [4] Gonczarek, J. M. Tomczak, S. Zareba, J. Kaczmar, P. Dabrowski, and M. J. Walczak, "Interaction prediction in structure-based virtual screening using deep learning," *Computers in Biology and Medicine*, vol. 100, pp. 253–258, 2018.
- [5] O'ztku' rk, A. O'zgu' r, and E. Ozkirimli, "DeepDTA: deep drug– target binding affinity prediction," *Bioinformatics*, vol. 34, no. 17, pp. i821–i829, 2018.
- [6] T. Nguyen and D.-H. Le, "A matrix completion method for drug response prediction in personalized medicine," in *Proceedings of the International Symposium on Information and Communication Technology*, 2018, pp. 410–415.
- [7] H. Le and V.-H. Pham, "Drug response prediction by globally capturing drug and cell line information in a heterogeneous network," *Journal of Molecular Biology*, vol. 430, no. 18, pp. 2993–3004, 2018.
- [8] H. Le and D. Nguyen-Ngoc, "Multi-task regression learning for prediction of response against a panel of anti-cancer drugs in personalized medicine," in *Proceedings of the International Conference on Multimedia Analysis and Pattern Recognition (MAPR)*. IEEE, 2018, pp. 1–5. [12] K. Matlock, C. De Niz, R. Rahman, S. Ghosh, and R. Pal, "Investigation of model stacking for drug sensitivity prediction," *BMC Bioinformatics*, vol. 19, no. 3, p. 71, 2018.
- [9] Turki and Z. Wei, "A link prediction approach to cancer drug sensitivity prediction," *BMC Systems Biology*, vol. 11, no. 5, p. 94, 2017.
- [10] Azuaje, "Computational models for predicting drug responses in cancer research," *Briefings in Bioinformatics*, vol. 18, no. 5, pp. 820–829, 2017.
- [11] I. I. Baskin, D. Winkler, and I. V. Tetko, "A renaissance of neural networks in drug discovery," *Expert Opinion on Drug Discovery*, vol. 11, no. 8, pp. 785–795, 2016.



10.22214/IJRASET



45.98



IMPACT FACTOR:
7.129




IMPACT FACTOR:
7.429



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089  (24*7 Support on Whatsapp)



ISSN No. : 2321-9653

IJRASET

**International Journal for Research in Applied
Science & Engineering Technology**

IJRASET is indexed with Crossref for DOI-DOI : 10.22214

Website : www.ijraset.com, E-mail : ijraset@gmail.com

Certificate

It is here by certified that the paper ID : IJRASET59304, entitled
*Dark Tracer Early Malware Detection Based on Spatiotemporal Patterns Using
Xgboost Algorithms*

by
A Mounika Rajeswari

after review is found suitable and has been published in
Volume 12, Issue III, March 2024
in

By 

Editor in Chief, IJRASET

International Journal for Research in Applied Science &
Engineering Technology
(International Peer Reviewed and Refereed Journal)
Good luck for your future endeavors



ISRA Journal Impact
Factor: 7.429



45.98
INDEX COPERNICUS



THOMSON REUTERS
Researcher ID: N-9081-2016



TOGETHER WE REACH THE GOAL
SJIF 7.429



ISSN No. : 2321-9653

IJRASET

**International Journal for Research in Applied
Science & Engineering Technology**

IJRASET is indexed with Crossref for DOI-DOI : 10.22214

Website : www.ijraset.com, E-mail : ijraset@gmail.com

Certificate

It is here by certified that the paper ID : IJRASET59304, entitled
*Dark Tracer Early Malware Detection Based on Spatiotemporal Patterns Using
Xgboost Algorithms*

by
Janani Chalapati

after review is found suitable and has been published in
Volume 12, Issue III, March 2024
in

By 

Editor in Chief, IJRASET

International Journal for Research in Applied Science &
Engineering Technology
(International Peer Reviewed and Refereed Journal)
Good luck for your future endeavors



ISRA Journal Impact
Factor: 7.429



45.98
INDEX COPERNICUS



THOMSON REUTERS
Researcher ID: N-9081-2016



TOGETHER WE REACH THE GOAL
SJIF 7.429



ISSN No. : 2321-9653

IJRASET

**International Journal for Research in Applied
Science & Engineering Technology**

IJRASET is indexed with Crossref for DOI-DOI : 10.22214

Website : www.ijraset.com, E-mail : ijraset@gmail.com

Certificate

It is here by certified that the paper ID : IJRASET59304, entitled
*Dark Tracer Early Malware Detection Based on Spatiotemporal Patterns Using
Xgboost Algorithms*

by
V.Venkata Sai Natha Reddya

after review is found suitable and has been published in
Volume 12, Issue III, March 2024
in

By 

Editor in Chief, IJRASET

International Journal for Research in Applied Science &
Engineering Technology
(International Peer Reviewed and Refereed Journal)
Good luck for your future endeavors



ISRA Journal Impact
Factor: 7.429



45.98
INDEX COPERNICUS



THOMSON REUTERS
Researcher ID: N-9581-2016



10.22214/IJRASET
doi
crossref
TOGETHER WE REACH THE GOAL
SJIF 7.429



ISSN No. : 2321-9653

IJRASET

**International Journal for Research in Applied
Science & Engineering Technology**

IJRASET is indexed with Crossref for DOI-DOI : 10.22214

Website : www.ijraset.com, E-mail : ijraset@gmail.com

Certificate

It is here by certified that the paper ID : IJRASET59304, entitled
*Dark Tracer Early Malware Detection Based on Spatiotemporal Patterns Using
Xgboost Algorithms*

by
Mohammed Awais Khan

after review is found suitable and has been published in
Volume 12, Issue III, March 2024
in

By 

Editor in Chief, IJRASET

International Journal for Research in Applied Science &
Engineering Technology
(International Peer Reviewed and Refereed Journal)
Good luck for your future endeavors



ISRA Journal Impact
Factor: 7.429



45.98
INDEX COPERNICUS



THOMSON REUTERS
Researcher ID: N-9581-2016



10.22214/IJRASET
doi
crossref
TOGETHER WE REACH THE GOAL
SJIF 7.429

