THE DEVELOPMENT AND IMPLEMENTATION OF ANOMALY-BASED MAN-IN-THE-MIDDLE INTRUSION DETECTION SYSTEM WITH AN IMPROVED ENSEMBLE MODELLING SCHEME IN DOMAIN NAME SERVER AND EDGE COMPUTING

by

RAMAHLAPANE LERATO MOILA

THESIS

Submitted in fulfilment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTER SCIENCE

In the

FACULTY OF SCIENCE AND AGRICULTURE

(School of Mathematical and Computer Sciences)

at the

UNIVERSITY OF LIMPOPO

Supervisor: Professor M. Velempini

2025

Declaration

I declare that the thesis entitled "***The Development and Implementation of Anomaly-Based Man-in-the-Middle Intrusion Detection System with an Improved Ensemble Modelling Scheme in Domain Name Server Edge Computing***" hereby submitted to the University of Limpopo, for the degree of Ph.D (Computer Science) has not previously been submitted by me for a degree at this or other university; that it is my work in design and in, and that all material contained herein has been duly acknowledged.

02 APRIL 2025

_____ _____

Surname, Initials (title) Date

Acknowledgements

I would like to begin by expressing my heartfelt gratitude to God whose strength, wisdom, and guidance enabled me to complete this thesis. Without His support, this accomplishment would not have been achievable. I would also like to express my deepest gratitude to all those who have supported me throughout this journey. My supervisor, Prof Mthulisi Velempini, deserves special recognition for their invaluable guidance, support, and encouragement during this research. Their knowledge and insights were essential to the completion of this work.

I am grateful to the University of Limpopo for providing the necessary resources and facilities to conduct my research. Special thanks to the Conerge group for their consistent support and assistance. I further acknowledge the financial support from the University of Limpopo and the MICSETA bursary, which made this research possible. Finally, I would like to thank my family and friends for their unwavering support and encouragement. Their patience and understanding were instrumental in helping me stay focused and motivated. I also thank everyone who contributed to this research in any way. Your support and contributions are deeply appreciated.

Dedication

This thesis is dedicated to my beloved baby sister, Lethabo Nkadimeng, for her constant love and support. Her unwavering assistance with my three boys, Katlego, Retshepile, and Thatego Moila, was invaluable. She sacrificed her time and life to ensure my success. I also dedicate this thesis to God, who was always there to pick me up and encourage me to carry on, especially during the most challenging times. To my entire family, this is a significant milestone as I become the first daughter to receive a PhD degree. Let this achievement serve as a reminder that with God's guidance, anything is possible, even beyond our wildest dreams.

Abstract

Technology is evolving at a rapid pace and its role in adding value to businesses around the world has come sharply into focus. Due to this rapid growth of devices, the centralised cloud is now experiencing significant difficulties in protecting large volumes of digital data. It has also become expensive to manage and maintain data accuracy. Mobile Edge Computing has become a promising solution with innovative data management, cost effectiveness, reliability, and uninterrupted connectivity. While the technology has transformed how data is handled and processed, it remains susceptible to security attacks such as Man-in-the-Middle (MitM) attacks. These attacks can cause severe consequences, as the attacker can intercept communications between any two parties without their knowledge, compromising and disrupting sensitive data, card credentials, and passwords.

This study aims to develop an anomaly-based intrusion detection scheme using ensemble modelling to combat MitM attacks. The scheme is designed to address false positives and improve accuracy. The proposed Ensemble Cuckoo was trained on Kaggle platform using Python as a programming language. We used the Cuckoo Search Algorithm to optimise the ensemble model (random forest). The scheme was compared to the Support Vector Machine (SVM) and the Local Outlier Factor (LOF) algorithms. To evaluate the effectiveness of the proposed Ensemble Cuckoo, this study utilised the F1-score, Precision, Recall and Accuracy metrics.

The simulation results indicate that the proposed Ensemble Cuckoo outperformed the algorithms it was compared against, achieving detection accuracy of 99.9%, showing a good improvement in terms of minimising false positives. The results were validated using Bayesian Dynamic Stackelberg Game Theory, which simulates the interactions between the defender and the attacker. Despite its effectiveness, the study acknowledges certain limitations, including the need for refinement in real-time processing and challenges related to scaling in large, and distributed networks. Future research could focus on extending the application of the proposed Ensemble Cuckoo, paving the way for broader adoption and deployment in real-world scenarios.

Keywords: *Cuckoo Search Algorithm, Man-in-the-Middle Attacks, Mobile Edge Computing, detection accuracy, ensemble modelling, support vector machine, local outlier factor, Bayesian dynamic Stackelberg game theory.*

CONTENTS

Declaration i
Acknowledgements ii
Dedication iii
Abstract iv
List of Figures viii
List of Tables ix
List of Abbreviations x
Chapter 1 1
Introduction 1
1.1 Introduction 1
1.2 Research Problem 3
1.3 Research Significance 4
1.4 Research Aim 4
1.5 Research Questions 4
1.6 Research Objectives 5
1.7 Literature Review 5
1.8 Hypothesis 6
1.9 Methodology 6
1.10 Research Contributions 8
1.11 Publication Contributions 8
1.12 Thesis Overview 8
Chapter 2 10
Literature Review 10
2.1 Introduction 10
2.1.1 Related Work 11
2.2 Background and Overview 11
2.2.1 Key Concepts and Definitions 12
2.2.2 Challenges and Limitations 12
2.3 Man-in-the-Middle attacks 13
2.4 Severe cases of MitM attacks on businesses 15
2.5 Related work 16
2.5.1 Artificial Intelligence Techniques 16
2.5.2 Ensemble techniques 18

2.5.3 Federated Learning 20
2.5.4 Deep Learning techniques 21
5.5 Machine Learning Techniques 24
2.5.2 Block-chain methods 31
2.5.3 Other Methods 32
2.5.4 Game theory-related works 40
2.6 Chapter Summary 42
Chapter 3 44
Methodology 44
3.1 Introduction 44
3.2 Research Design 45
3.3 Research tools 45
3.4 Ensemble solution and ML techniques 46
3.4.1 Feature Selection 46
3.4.2 Dataset Partitioning 46
3.4 Dataset 47
3.5 MitM Attack prevention techniques 48
3.6 Network Architecture 51
3.7 Detecting the presence of an attack 52
3.8 Intrusion Detection Scheme 52
3.9 Cuckoo Search Algorithm 53
3.9.1 CSA Principles 53
3.9.2 Initialisation 54
3.10 Proposed Ensemble Cuckoo Scheme 55
3.11 Chapter Summary 58
Chapter 4 59
Data and Results Analysis 59
4.1 Introduction 59
4.2 Network Topology 59
4.3 Simulation Parameters 61
4.4 Dataset 65
4.6 Results Discussion 74
4.6.1 Discussion of the Results 74
4.6.3 Results and Related Work 75
4.6.4 Research Implications 76
4.7 Chapter Summary 76

Chapter 5 77
Adaptive Game Theoretic Model 77
5.1 Introduction 77
5.2 Background 77
5.3 Stackelberg Game Theory 78
5.3.1 Players: 78
5.3.2 Types of attackers 78
5.3.4 Probability distribution over types 78
5.4 The defender's strategy is determined by: 79
5.5 Stackelberg Equilibrium 79
5.6 Dynamic Stackelberg game 79
5.7 Dynamic Stackelberg game formulation 80
5.7.1 Dynamics state: 80
5.7.2 Defenders' problem: 80
5.7.3 Attacker's problem: 80
5.8 Bayesian Dynamic Stackelberg Game 80
5.9 State Variables and Dynamics 81
5.10 Bayesian beliefs 81
5.11 Defender's objective function 81
5.12 Defender's optimisation problem 82
5.14 Experimental Results 84
5.15 Chapter Summary 88
Chapter 6 89
Conclusion 89
6.1 Introduction 89
6.2 Algorithms utilised 89
6.3 Summary of the results 89
6.4.1 Research Outputs 92
6.4.2 Filling Research Gaps 92
6.4.3 Relation to Existing Theory 93
6.4.4 Practical Applications 93
6.5 Limitations 94
6.6 Recommendations 95
6.7 Chapter Summary 96
Bibliography 97
Appendix A 109

Appendix B 113
Appendix C 117

List of Figures

FIGURE 2. 1 DEPICTS NETWORK FLOW WITH AND WITHOUT MITM14
FIGURE 3. 1 MAN-IN-THE-MIDDLE ATTACK DATASET [97].47.....46
FIGURE 3. 2 MAN-IN-THE-MIDDLE ATTACK INTERCEPTION.49.....48
FIGURE 3. 3 MITM ATTACKS AND HOW THEY ARE ADDRESSED.50 49
FIGURE 3. 4 MOBILE EDGE NETWORK WITH A DOMAIN NAME SYSTEM.51.....50
FIGURE 3. 5 NETWORK PARAMETERS DETECTING THE PRESENCE OF AN ATTACK.52.....51
FIGURE 3. 6 PROPOSED INTRUSION DETECTION SYSTEM.56.....55
FIGURE 4. 1 INTERCONNECTION OF NODES, MEC SERVERS, AND DNS SERVER.60.....59
FIGURE 4. 2 NORMAL NODES VERSUS ATTACKER NODES IN RED COLOUR.60 59
FIGURE 4. 3 DISTRIBUTION OF THE NETWORK TRAFFIC.62..... 61
FIGURE 4. 4 CUCKOO SEARCH CLASS.62.....61
FIGURE 4. 5 PRECISION, RECALL, F1-SCORE AND ACCURACY METRICS RESPECTIVELY.64.....63
FIGURE 4. 6 DATASET USED WAS CAPTURED AND THEIR DEVICES [97].66.....65

FIGURE 4. 7 DATA CLASSIFICATION66.....67
FIGURE 4. 8 EVALUATION METRICS OF THE PROPOSED ENSEMBLE CUCKOO SCHEME.68.....67
FIGURE 4. 9 CONFUSION MATRIX69.....68
FIGURE 4. 10 THE PRECISION METRICS AMONG THE FOUR ALGORITHMS.71.....70
FIGURE 4. 11 THE RECALL METRIC PERFORMANCE OF THE PROPOSED ENSEMBLE CUCKOO SCHEME.72.....71
FIGURE 4. 12 THE F1-SCORE RESULTS OF THE PROPOSED ENSEMBLE CUCKOO SCHEME.7372
FIGURE 4. 13 ACCURACY RESULTS OF THE PROPOSED ENSEMBLE CUCKOO SCHEME AND OTHER SCHEMES.74.....73
FIGURE 5. 1 THE PRECISION AND RECALL METRICS PERFORMANCE.86.....85
FIGURE 5. 2 F1-SCORE AND ACCURACY PERFORMANCE METRICS.87.....86
FIGURE 5. 3 CROSS VALIDATION AND CONVERGENCE OF BELIEFS OVER TIME.87.....86
FIGURE 5. 4 PAYOFF METRIC AND THE ENHANCED ATTACKER STRATEGY.88.....87

List of Tables

TABLE 1.1 SIMULATION PARAMETERS7
TABLE 3.2 DETERMINES STEPS TAKEN BY ALGORITHM.57.....57
TABLE 4.1 THE SIMULATION PARAMETERS.61.....60

TABLE 4.2 EXPERIMENTAL RESULTS FOR THE ALGORITHMS.63.....62
TABLE 4.3 KITSUNE DATASET RESULTS67.....66
TABLE 4.4 THE PERFORMANCE METRICS FOR THE ALGORITHMS.70.....69
TABLE 5.1 BAYESIAN STACKELBERG GAME THEORY.82.....81
TABLE 5.2 EXPERIMENTAL RESULTS.85.....84

List of Abbreviations

ARP – Address Resolution Protocol

APT – Advanced Persistent Threats

AI – Artificial Intelligence

ADS – Artificial Detection System

CSA – Cuckoo Search Algorithm

CNN – Convolutional Neural Networks

DL – Deep Learning

DTCN – Dynamic Temporal Convolutional Network

DNS – Domain Name Server

DoS – Denial of Service

DC – Decision Tree

DNN – Deep Neural Network

DDoS – Distributed Denial of Service

EP – End Point

FML – Federated Machine Learning

FGS – Forward Greedy Selection

GPU – Graphics Processing Unit

HTTP – Hypertext Transfer Protocol

IoT – Internet of Things

IETF – Internet Engineering Task Force

IIoTs – Industrial Internet of Things

ICS – Industrial Control Systems

IDS – Intrusion Detection Scheme

IP – Internet Protocol

K-NN – K-Nearest Neighbour

LAN – Local Area Network

LOF – Local Outlier Factor

LSTM – Long Short-Term Memory

MLA – Machine Learning Algorithms

MEC – Mobile Edge Computing

MitM – Man-in-the-Middle

MOECSA – Multi-Objective Enhanced Capuchin Search Algorithm

MDP – Markov Decision Process

NIDS – Network Intrusion Detection System

OSI – Open System Interconnection

PCA – Principal Component Analysis

QoS – Quality of Service

RF – Random Forest

RNN – Recurrent Neural Networks

SARSA – State-Action-Reward-State-Action

SSL – Secure Socket Layer

SaaS – Software-as-a-Service

SVM – Support Vector Machine

TLS – Transport Layer Security

TCE – Train Communication Ethernet

TPU – Tensor Processing Unit

UDP – User Datagram Protocol

WSN – Wireless Sensor Network

Chapter 1

Introduction

1.1 Introduction

Technology is evolving at a rapid pace and its role in adding value to businesses around the world has become sharply in focus [1]. Internet of Things (IoTs) is an innovative technology that has transformed how individuals consult for their health services, how agriculture is conducted today, and how the home has become a smart home [2]. All the information such as organisations store customer data, employee information, business documents, projects files, marketing materials, and operational data in the cloud; individuals store personal files, emails and contacts, backups, social media content, and health data; while the global community utilises cloud storage for global databases, open source repositories, academic research, media libraries, and public archives is stored on centralised cloud to allow individuals to access their data anywhere, anytime. Due to the rapid growth of the devices, the centralised cloud is now experiencing difficulties in protecting these volumes of digital data; it has also become expensive to manage and maintain the data accurately, also facing attacks such as cloud misconfiguration, Denial of Service (DoS), insider threats, minimised infrastructure visibility and unauthorised use of cloud services [3].

A solution to offload the centralised cloud such that instead of relying solely on a single, distant data center, some of the data processing and storage should be handled by local servers or edge devices was introduced in 1990 by Akamai, and a new technology called Mobile Edge Computing (MEC) has emerged as a promising solution, with innovative management of the data, lower connectivity costs, reliable and uninterrupted connection [4]. MEC technology has changed how digital data is currently handled and processed and, most importantly, it has brought the data closer to the originating source for processing. Although MEC technology has all these advantages, it has also introduced new security challenges including data privacy concerns, increased attack surfaces, and the need for robust encryption methods that require new approaches to ensure that the network and devices are protected.

The use of Local Area Networks (LANs) within the centralised cloud environments and corporate data centres was considered beneficial for security purposes, as it enabled organisations to secure their data with multiple layers of security defences spanning

both virtual and physical environments. However, with the adoption of MEC, which involves distributing computational resources away from the secure core, this security benefit may no longer hold [5]. Security issues on edge computing have become a major concern; cybercriminals are targeting to compromise confidentiality, integrity, and availability of data. The edge devices have become the target due to the position and functions they provide such as data transmission, monitoring, filtering, and storing data that passes from one network to another [6].

Cybercriminals aim to disrupt the normal operation of a network; in the event of unauthorised access attacks they target weak passwords, previously compromised devices or networks, and insider threats. Additionally, attackers construct botnets and use them to direct false traffic at the targeted network, devices and servers. This study focuses on the Man-in-the-Middle (MitM) attack [7], where a conversation between two or more communication devices is compromised through eavesdropping or impersonation.

The attack is commonly practised but difficult to detect if the attacker is only observing and actively hiding their activities [7]. The objective of this study is to determine a cost-effective technique for the detection and minimising false positives on the edge level. For an intrusion detection system (IDS) to identify attacks in a dynamic network, it is overwhelming and time-consuming, making the IDS unreliable because of the high false alarm rate. This proclivity increases the burden on the security analysts and slowly allows harmful attacks to go undetected because the system is overwhelmed and takes time to respond to attacks [8].

Machine Learning Algorithms (MLA) have gained more attention recently, and the MLA approaches are deployed to generate predictive models for monitoring network attacks. This study proposes to design and implement an anomaly-based IDS against the MitM attacks using ensemble modelling to improve the detection rate and minimise false positives. Furthermore, attacks currently reach the edge network through the utilisation of the Domain Name Server (DNS). DNS was developed in 1983 by [9], as a scalable way to map hostnames to their Internet Protocol (IP) addresses.

Just as the protocols designed for the early internet, DNS was designed without considering the security feature. They normally use plaintext messages implying that it is easy for a third party to read an ongoing conversation. The attacker positions

themselves between the end-user and the DNS server to alter a response from the DNS with false IP addresses, effectively redirecting the end-user where they will be susceptible to security attacks. Not much has been done to mitigate this kind of continuous issues that haven't been thoroughly addressed, except by protecting the network path utilised by the DNS responses [10].

1.2 Research Problem

The rapid growth of edge networks and the critical role of DNS in modern infrastructure have made these spaces vulnerable and easy targets for cyberattacks. MitM attacks pose a real threat by allowing attackers to intercept and manipulate ongoing communication between two parties without their knowledge. This leads to severe consequences, including data breaches, loss of sensitive information, and financial losses. Though many IDS systems exist, the current methods do not effectively detect MitM attacks due to several compromises, such as the difficulty in distinguishing between legitimate network traffic and malicious activity.

Network traffic is increasingly encrypted using Secure Sockets Layer / Transport Layer Security (SSL/TLS), while IDS systems inspect network traffic by checking the packet headers or payloads. In tandem, encryption conceals the payload data. Thus, if the MitM attack manipulates data in an encrypted form, then the IDS may not be able to detect it unless the traffic can be decrypted. This is not possible. Sophisticated attack methods focus on signature-based detection, which depends on known attack patterns, making it ineffective against advanced threats such as Zero-Day attacks and challenges in analysing real-time traffic. While anomaly-based detection can be adaptable, it is prone to high false positive rates, degrading the capability of the detection system, particularly in a dynamic environment [11].

The existing solutions for MitM attacks do not effectively secure edge networks. Most of the security schemes are tailored for traditional cloud computing environments and are not adequate for edge networks [12]. To address this challenge, there is a growing demand to fill the gap in current solutions by developing a robust anomaly-based MitM detection scheme using ensemble modelling, tailored specifically for edge networks and DNS, to minimise the false positives and enhance the detection rate.

The study in [13], proposed a commitment-based scheme for securing Bluetooth connections by having users draw figures on an Android screen and further proposed

an IDS based on Snort that uses firewall and netcut to block Internet Protocol (IP) addresses of malicious traffic to detect MitM attacks. However, the use of traditional IDS and firewalls is not sufficient in preventing MitM attacks due to their dynamic and complex nature. Using MLA to detect attacks could improve the learning process and ensure that the IDS becomes effective.

1.3 Research Significance

This study was motivated by the rise of devices susceptible to security attacks where the MitM attacks exploit the weaknesses in the system which lacks the authenticity mechanisms. We realised that the MitM attack can also facilitate other attacks including Denial of Service (DoS), and session hijacking. As such, this study strives to understand these vulnerabilities, develop and implement a robust detection scheme using an optimised random forest with a Cuckoo Search Algorithm (CSA) for the assurance of confidentiality, integrity, and reliability within the MEC networks.

1.4 Research Aim

Given the lack of security standards on the edge network, this study aims to develop and implement an effective anomaly-based detection scheme using ensemble modelling against the MitM attacks, tailored to detect specific attack vectors and behaviours in edge networks and DNS. This is designed to address scalability issues, and assess its feasibility and suitability for deployment in real-world scenarios.

1.5 Research Questions

- ✓ How can a robust scheme be constructed to detect and minimise MitM attacks in edge networks and DNS?
- ✓ How does the proposed Ensemble Cuckoo scheme improve the detection rate performance and minimise the false positives and negatives of the MitM attack?
- ✓ What are the challenges and limitations of deploying the proposed Ensemble Cuckoo scheme in edge networks and DNS, and how can these challenges be mitigated?
- ✓ How does the proposed Ensemble Cuckoo scheme meet the security requirements and standards for edge networks and DNS, and what are the implications for its adoption and deployment?

1.6 Research Objectives

To achieve the aforementioned research objectives and address the research questions, this study is founded on the following objectives:

- ✓ To develop and design an effective anomaly-based IDS scheme for detecting and minimising of MitM attacks.
- ✓ To evaluate the effectiveness of the proposed Ensemble Cuckoo scheme in improving the accuracy and efficiency of the scheme to detect MitM attacks.
- ✓ To identify and analyse the challenges and limitations of deploying the proposed Ensemble Cuckoo scheme in edge networks, and DNS and propose solutions to overcome these challenges.
- ✓ To evaluate the proposed Ensemble Cuckoo scheme against the security requirements and standards for edge networks and DNS and assess its feasibility and suitability for implementation in real-world scenarios.

1.7 Literature Review

The study in [14], proposed a deep learning-based MitM-IDS algorithm to address the MitM attacks, monitoring attackers based on signature-ID templates. The key strength of this model is its ability to detect malicious behaviour in the shortest time possible. However, while it is effective in detection, the scheme's reliance on signature-ID validation may limit its adaptability to novel attack patterns. To differentiate between legitimate and malicious nodes in the edge network, [15] introduced a blockchain-based certificate signcryption mechanism built upon the elliptic curve discrete logarithm problem. Despite demonstrating low signcryption overhead, the scheme still faces efficiency challenges, indicating room for improvement in practical deployment.

Building on the idea of signcryption, [16] proposed a heterogeneous online/offline signcryption to enhance network scalability. The study aimed to assign monitoring nodes to enable quick response and attack detection. This monitoring tool checks latency, response time, and data, but the approach does not fully eliminate MitM attacks, suggesting that while mitigation is possible, complete eradication remains a challenge.

In [17], an efficient TLS-based authentication mechanism was proposed for web applications to prevent attackers from impersonating legitimate DNS servers. Although the mechanism minimised communication overheads, its direct application to DNS

servers makes them vulnerable on the edge. This limitation highlights the need for additional measures to secure edge devices.

Lastly, [18] proposed DNSCrypt to enhance customer edge security amidst high DNS latencies. The scheme evaluates communication between edge devices and DNS servers based on latency, assuming that higher latency indicates an attack. While the simulation results are promising, the scheme's reliance on latency as an indicator of attacks needs further validation due to various factors that can contribute to latency.

1.8 Hypothesis

The ensemble modeling approach significantly improves the true positive rate and minimises the false positive rate of the proposed Ensemble Cuckoo scheme while enhancing the computational efficiency as compared to the standalone anomaly detection method.

1.9 Methodology

The methodology focuses on developing a robust ensemble model to enhance the detection and classification of network intrusions. By combining multiple machine learning algorithms, the ensemble model aims to leverage the strengths of each individual algorithm while mitigating their weaknesses. This approach ensures improved accuracy, stability, and generalisation capabilities, making the model more effective in real-world scenarios.

We designed our ensemble model by carefully selecting and integrating multiple classifiers. Each classifier contributes to the final decision through techniques such as bagging and boosting. This ensemble approach enables the model to perform well across diverse network environments and various types of attacks.

Table 1.1 illustrates the simulation parameters used to carry out our study. The study uses the Kaggle platform as an experimental tool because it has free access to high-quality datasets; it has integrated tools offering a cloud-based Jupiter notebook environment with free access to Graphics Processing Unit (GPUs) and Tensor Processing Units (TPUs). Kaggle has pre-installed libraries like TensorFlow, PyTorch, and Scikit-learn which saves time and effort. The study uses Python as a programming language due to its simple syntax, versatility, and flexibility. Python is also open-source, free to use and accessible to anyone.

To evaluate the proposed Ensemble Cuckoo scheme, metrics such as false positive, accuracy, precision, recall, and f1-score are utilised. The Proposed Ensemble Cuckoo scheme is trained using a Random Forest (RF) optimised using CSA and compared against the Support Vector Machine (SVM) and Local Outlier Factor (LOF). The reason why the proposed Ensemble Cuckoo scheme is compared with SVM and LOF is that SVM is robust to outliers and noise in the data, particularly when using the soft margin approach, which is relevant when a scheme is designed to operate in a noisy environment. The LOF algorithm is specifically designed for anomaly-based or outlier detection and our scheme involves the detection of anomalies hence the LOF is a relevant and strong comparator.

Table 1.1 Simulation Parameters

Parameters	Values
Simulation tool	Kaggle platform
Programming language	Python
Metrics	False positive, Accuracy, Precision, Recall, F1-score
Algorithms	CSA, RF, SVM, LOF
Framework	Bayesian inference, dynamic Stackelberg game theory
Dataset	Generated dataset and MitM attack dataset
Network	Mobile Edge Network
Server	Domain Name Server
Train and Test split	70% training and 30% testing
Cross validation	5 folds and 10 folds
Number of nodes	80

To validate the simulation results obtained we use the Bayesian inference which provides a natural framework for quantifying uncertainty in the model's prediction and parameters. The Dynamic Stackelberg Game Theory is a defender-attacker allowing the analysis of strategic decisions in an evolving environment. The Stackelberg game theory provides an optimal strategy design that benefits the defender (detection scheme) because the defender can secure a first-mover advantage, making the approach particularly useful in validating strategies that involve timing and commitment [19].

The study has used two datasets, one generated from a simulated network and other downloaded from Kaggle MitM attack dataset. We use Scapy, a powerful Python tool

that allows for the manipulation of creating malicious packets for various purposes. The Dataset is split into 70% training and 30% testing our proposed Ensemble Cuckoo scheme. The simulation results were cross-validated using 5 folds and 10 folds. On the simulated network, we consider the number of nodes to be 80.

1.10 Research Contributions

The study contributes to the body of knowledge by presenting an innovative, multi-layered security protocol designed for edge networks. This protocol incorporates multiple security measures at various levels to provide comprehensive protection against threats. By implementing layers such as authentication, encryption, intrusion detection, and access control, the protocol ensures that even if one layer is breached, other layers continue to secure the network. Which is evaluated against the latest ISO/IEC 27001 and IETF security guidelines, demonstrating its feasibility and suitability for deployment in real-world applications. The study culminates in providing practical guidance for organisations and developers to secure their systems.

1.11 Publication Contributions

- ✓ Optimising the cuckoo search algorithm for improved quality of service in cognitive radio networks. 2023 International Conference on Artificial Intelligence, Big Data, Computing and Data Communication (icABCD). DOI:10.1109/icABCD59051.2023.10220569
- ✓ An Optimized Machine Learning Model for the Detection of Man-in-the-Middle attacks in Mobile Edge Computing. 2024 IST-Africa Conference (IST-Africa). DOI:10.23919/IST-Africa63983.2024.10569231
- ✓ A Robust Bayesian Dynamic Stackelberg Game Theory Detection Scheme for Man-in-the-Middle attack in Mobile Edge Computing Networks. 2025 The Indonesia Journal of Computer Science (IJCS).

1.12 Thesis Overview

In Chapter 1, the study's context is introduced, along with the identification of research objectives and questions and an explanation of the study's significance. Chapter 2 reviews the existing literature and lays the foundation, identifying existing gaps and approaches used to address MitM attacks and scalability issues in edge networks. Chapter 3 explores the methodological choices, tools for development and implementation, and algorithms for experimental simulations. Chapter 4 focuses on

the developed Ensemble Cuckoo scheme and its implementation, training, and evaluation. Chapter 5, the Bayesian Dynamic Stackelberg Game Theory is applied to validate the performance of the proposed Ensemble Cuckoo scheme, and finally, Chapter 6 concludes the study, providing recommendations for future research.

Chapter 2

Literature Review

2.1 Introduction

An anomaly-based detection is a process of detecting patterns or behaviour in data that significantly deviate from what is considered normal. Anomalous data can indicate critical underlying issues like infrastructure failure, security threats, architectural optimisation opportunities, and improvement of marketing strategies [20]. Detecting outliers can be quite challenging due to the rarity of anomalies and the complexity and dynamism of normal behaviour patterns [21]. From a business standpoint, it is crucial to accurately identify genuine anomalies rather than mistaking false positives or data noise for significant issues. This precision ensures that resources are effectively utilised and decisions are based on informed and reliable insights.

The related work contributes to the field by offering a clear and concise overview of the current state of research on anomaly-based IDS for MitM attacks using ensemble modeling in edge networks, which can serve as a valuable resource for other researchers and practitioners, it critically helps in making informed decisions by presenting a nuanced evaluation of existing studies and methodologies. This review serves as a guide for future research by identifying gaps and proposing new approaches for investigation, particularly in the context of edge networks. Further, it enhances understanding of the challenges and opportunities associated with using ensemble modeling for anomaly-based IDS in edge networks by integrating and synthesising diverse perspectives and results.

This review section focuses on anomalies specifically related to MitM attacks within the edge networks. It covers various forms of MitM attacks, including eavesdropping, session hijacking, and data manipulation. We focus on anomaly-based detection techniques, with a particular emphasis on ensemble modeling approaches. We explore different ensemble techniques, including bagging, boosting, and stacking, and their application in detecting MitM attacks. The primary application context is the edge networks, including IoT environments, and other decentralised network architectures. The review traces and examines how anomaly-based IDS can be effectively implemented in these settings to assess security.

The review is limited to peer-reviewed journal articles, peer-reviewed conference proceedings, and credible industry reports published in the last decade. This ensures that the review is based on the most current and relevant research. While there are many techniques for anomaly detection, this review specifically focuses on ensemble modeling. Ensemble modeling is utilised because it combines multiple individual models to improve the overall performance and accuracy of anomaly detection, providing a more robust solution compared to other single-method approaches. Other methods, such as statistical or machine learning-based approaches, are only discussed in the context of their integration with ensemble models.

2.1.1 Related Work

The primary goal of this literature review is to explore and evaluate the current state of research on anomaly-based IDS for MitM attacks using ensemble modeling in edge networks. By systematically analysing and synthesizing existing studies, the review aims to:

Identify gaps – highlight areas where current research on anomaly-based IDS for MitM attacks in edge networks is lacking or where further investigation is needed.

Summarise key results – provide a comprehensive summary of the most significant results and trends related to the use of ensemble modeling for detecting MitM attacks in edge networks.

Evaluate methodologies – assess the strengths and weaknesses of various research methodologies used in the studies reviewed, particularly focusing on ensemble modeling techniques.

Propose future directions – suggest potential directions for future research based on the identified gaps and trends, with an emphasis on improving the effectiveness and efficiency of anomaly-based IDS in edge networks.

2.2 Background and Overview

Anomaly detection research has evolved significantly over the decades, adapting to the growing complexity and volume of data. Initial approaches focused on statistical methods, such as control charts and hypothesis testing, to identify outliers in small datasets [15][18]. With the emergence of machine learning, techniques like clustering, nearest neighbour, and SVM have become increasingly popular for anomaly detection

[22]. These methods can handle larger datasets and more complex patterns, but the explosion of Big Data has introduced new challenges. Researchers developed algorithms to handle high-dimensional data and large-scale datasets [23]. Techniques like principal component analysis and isolation forests were introduced to improve detection accuracy [23].

From early 2010 till the present, deep learning has revolutionised anomaly detection, enabling the analysis of complex data types such as images, videos, and time-series data. Techniques such as auto-encoders, Convolutional Neural Networks (CNN), and Recurrent Neural Networks (RNNs) are now widely used [24]. The current research focuses on integrating anomaly detection with other Artificial Intelligence (AI) techniques, such as Random Forest (RF). While there is a growing emphasis on real-time anomaly detection [25], this evolution reflects the increasing complexity and diversity of data, driving continuous innovation in anomaly detection techniques.

2.2.1 Key Concepts and Definitions

An outlier is a data point that substantially varies from the rest of the dataset. It can be due to variability in the data or an indication of an error. For example, in a dataset of people's ages, a value of 150 years would be considered an outlier. Novelty detection refers to identifying new or previously unseen data points that differ from the known data. This is often used in scenarios where the system needs to recognise new patterns or behaviours that were not present during training [26]. For instance, a fraud detection system might identify a new type of fraudulent transaction as a novelty.

An anomaly is a data point or pattern that does not conform to the expected behaviour. Anomalies can indicate critical incidents, such as fraud, network intrusions, or equipment failure. Anomaly detection aims to identify these unusual patterns in a bid to take appropriate actions. For example, a sudden spike in network traffic could be an anomaly indicating a potential cyberattack.

2.2.2 Challenges and Limitations

While anomaly detection is powerful, it comes with several challenges and limitations. *Data quality* – poor data quality, including missing values, inconsistent formats, and duplicate entries, can significantly impact the accuracy of anomaly detection models [27]. As a challenge the ongoing difficulties faced during the process of anomaly detection due to poor data quality. This includes the need for continuous efforts to

clean and process data. As the limitation this highlights weaknesses in the anomaly detection models when faced with poor data quality Hence ensuring high-quality data is crucial for reliable results.

Imbalance data – anomalies are usually infrequent compared to normal data, resulting in imbalanced datasets. This imbalance can cause traditional MLAs to perform poorly, as they may be biased towards the majority class [28].

Defining anomalies - determining what constitutes an anomaly can be subjective and context-dependent. Different applications may have varying definitions of what is considered abnormal, making it difficult to develop a one-size-fits-all solution that fits all scenarios [29].

High false positive rate - anomaly detection systems frequently suffer from high false positive rates, where normal data points are incorrectly identified as anomalies. This issue can result in unnecessary investigations and wasted resources [29].

Scalability - as datasets grow and become more complex, the computational cost of anomaly detection increases. Efficiently processing large-scale data while maintaining accuracy is a significant challenge [29].

Evolving data patterns - Data patterns can change over time, requiring anomaly detection models to adapt continuously. Static models may become outdated and less effective in detecting new types of anomalies [29].

Interpretability - Understanding why a specific data point is identified as an anomaly can be difficult, especially with complex models like DL. Improving the interpretability of these models is essential for practical applications [29].

2.3 Man-in-the-Middle attacks

In this section, we examine the security threats and vulnerabilities emerging from the edge network. The rapid growth of IoT devices and sensors transmitting vast amounts of data to centralised cloud systems or corporate data centers faces several challenges. Processing a large amount of data incurs delays, and can be expensive, and the technical challenges of data movement mean that storage and privacy are compromised every day [30]. A recommended approach to address these challenges is edge computing. Edge Computing involves processing data closer to the source I.e., IoT devices and sensors rather than sending it to centralised cloud systems or

data centers. This reduces latency, lowers costs, and enhances privacy and security by minimising the amount of data that needs to be transmitted and stored centrally.

MitM is a cyberattack where the attacker secretly interrupts the normal communication taking place between two or more devices that they believe are communicating with each other [30]. MitM attack at the edge of the network involves intercepting and manipulating data as it travels between Internet of Things (IoTs) devices and the central cloud or data center. In edge environments, where data processing occurs closer to the source, attackers can exploit vulnerabilities such as rogue access points, Domain Name System (DNS) spoofing, or Address Resolution Protocol (ARP) spoofing to insert themselves into the communication channels. This allows them to eavesdrop on sensitive information, modify data, or inject malicious content without the knowledge of either party involved.

MitM attacks intercept any ongoing communication to eavesdrop, interrupt, and control. Technology has advanced and changed how businesses operate, experiencing an increase in online shopping, surfing the internet, making transactions online and booking health services online as depicted in Figure 2.1.

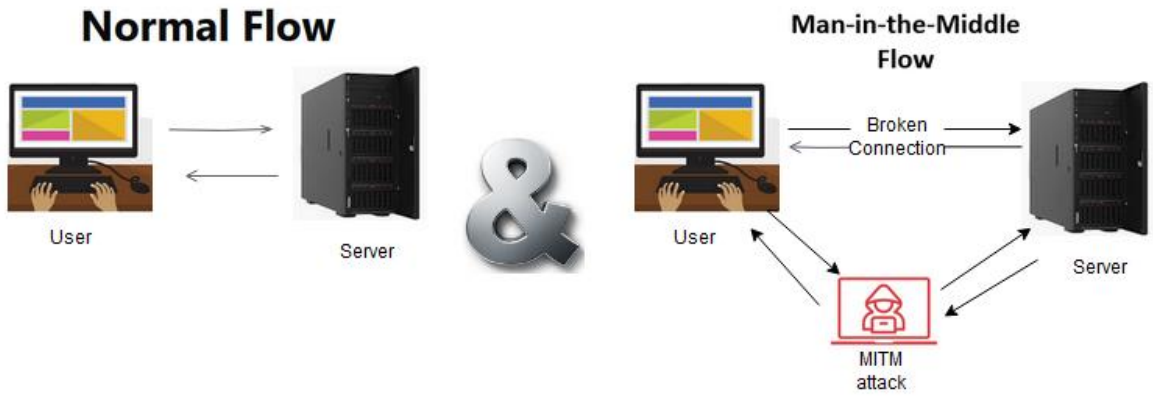

Figure 2.1 Depicts network flow with and without MitM attack

Cybercriminals are always on the lookout, targeting systems that have been previously compromised, insufficient security patches, lack of vulnerability scans, insufficient account protection, and incomplete network coverage [31]. In a traditional MitM attack, cybercriminals intercept and relay messages, steal, or modify data without the knowledge of the devices communicating.

The attacker can install a packet analyser to monitor the network traffic for insecure communications. When the user logs into a website, the attacker then retrieves their

information and directs them to a fake website that mimics the real website [32]. On the fake website, the attacker collects the users' valuable information which they can now use on the real website. The attacker can modify data transmission to prevent either party from recognising the modified data malware, linking or inserting the data into the communication channel. MitM attacks can cause severe disasters for small businesses such as data breaches, heavy fines, and reputational harm and losses.

The Dutch registrar website, known as DigiNotar, was breached in 2011, enabling the attacker to gain access to at least 500 certificates for websites such as Google, Skype, and so forth. The access meant that the attacker had legal access to these websites hence they launched the MitM attack to steal users' data tricking users into entering their passwords on fraudulent mirror websites while thinking it was legitimate [1].

A recent MitM attack identified by CrowdStrike was a Trickbot module called shaDII. The module targets all systems that are infected and installs a Socket Secure Layer (SSL) certificate, which enables the attacker to gain access. The attack was launched by two known cyber groups LUNAR SPIDER and WIZARD SPIDER. Hence it is still important that algorithms using ML be developed to minimise the high launching of MitM attacks [33].

2.4 Severe cases of MitM attacks on businesses

There are several applications that individuals are currently using and are affected by security flaws such as Absa Homeowner, Discovery, DStv Now, Standard Bank App, Standard Bank mobile banking, Ster-kinekor or theatres, and Takealot.com. These MitM attacks exploit confidential customer information which can lead to identity theft, financial losses, and privacy violations [34].

The attack disrupts normal operations by causing downtime, leading to a loss of productivity and revenue. In 2017, a confirmed data breach at Equifax compromised the personal information of 143 million Americans. In response, Equifax created a website, EquifaxSecurity2017.com, allowing users to check if they were affected by the breach. However, the issue arose when thousands of other websites shared the same SSL certificate used by Equifax's site. This vulnerability allowed attackers to host fake websites, redirect users, or intercept their data through methods like DNS manipulation and SSL spoofing were used [35].

In 2014, Lenovo issued Personal Computers (PCs) with the adware Super Fish Visual Search. This adware allowed attackers to monitor web activity and login details while users browsed on Chrome or Internet Explorer. By modifying SSL certificates, attackers could inject their advertisements onto encrypted web pages, compromising security. Both consumers and businesses are vulnerable to MitM attacks [36]. Consumers face the risk of having their information tracked down, stolen, and exploited if they fall victim to fraudulent Wi-Fi networks, fake websites, or compromised email accounts. Users of any website or program that requires login credentials or stores financial data are prime targets for such attacks.

Businesses have user-interactive websites and software applications that keep a lot of client data at risk. Businesses must devote resources to detect and defend against MitM attacks in addition to the operational slowdowns caused by mitigating or responding to one, as well as the recovery process of dealing with liability issues and re-establishing brand credibility [37]. Furthermore, MitM attacks have severe consequences for both businesses and individuals, including data theft, financial losses, reputational damage, disruption of operations, and legal and regulatory consequences. It is essential to take appropriate measures to minimise the occurrence of MitM attacks, such as using secure communication protocols, implementing intrusion detection and prevention systems, and educating users about the risks of MitM attacks.

2.5 Related work

2.5.1 Artificial Intelligence Techniques

The study in [38] presents a survey of AI techniques for securing IoT services in edge computing environments. The authors discuss the challenges of securing IoT services at the network edge and propose an AI-based approach to address them. Various AI techniques, including Machine Learning, Deep Learning, and Reinforcement Learning, are used to enhance security. The study further reviews existing AI-based security solutions for edge computing, such as Intrusion Detection, anomaly detection, and threat intelligence.

Each solution's strengths, weaknesses, and applications are analysed. The authors also discuss technical challenges like efficient data processing, scalable model training, and effective threat modeling. Furthermore, the study provides a

comprehensive overview of AI-based security solutions for IoT services in edge computing and highlights their potential to enhance IoT system security as shown in [39]. The study provides a comprehensive survey of AI techniques for securing IoT services in edge computing, but it could be strengthened by providing more empirical evidence to support the effectiveness of the proposed AI-based security solutions.

The study in presents a comprehensive literature review of cybersecurity attacks in IoT environments using Artificial Intelligence (AI) methods. The authors address the unique security challenges of IoT networks, such as many devices, and a lack of centralised security measures. The authors review various AI-based detection techniques, including ML, DL, and rule-based systems, providing a detailed analysis of their effectiveness, scalability, and limitations.

These researchers further discussed key technical challenges in developing AI-based detection systems for IoT, such as the lack of labeled data, the need for real-time detection, and resource-efficient models. The study offers insightful discussion on the future directions in AI-based detection systems for IoT, including the development of explainable AI models, the integration of multiple detection techniques, and the application of blockchain and other evolving technologies. The study further provides a valuable overview of the current state-of-the-art in AI-based detection systems for IoT cybersecurity attacks and highlights the potential of these systems for enhancing the security of IoT networks. The study reviews AI-based detection techniques for IoT cybersecurity, addressing challenges and future directions, but lacks practical examples and real-world validations.

Distributed Edge nodes are more susceptible to attacks due to their limited computational and storage resources compared to other endpoint devices like computers and cell phones. The study in [40] provides a comprehensive overview of how AI can enhance the security of IoT services in edge computing. The proposed approaches are designed to address the unique challenges and requirements of securing IoT devices deployed on edge networks with limited resources.

While using AI to secure IoT devices, there are still gaps identified including adaptability to a dynamic environment hence the development of the scheme should investigate factors to ensure that the proposed Ensemble Cuckoo scheme addresses evolving security threats to ensure the robustness and adaptability of the AI models in

handling a dynamic environment which has become a problem calling for ongoing monitoring and updating.

2.5.2 Ensemble techniques

The study in [41] developed an anomaly-based IDS using an ensemble ML approach. The scheme was designed to improve IDS by detecting and classifying malicious activities through analysing network traffic. Due to the advancement of cyberattacks, the traditional IDS used here lacks robustness and does not consider the complexity caused by increasing traffic. To address this challenge, the current study suggests focusing on making the network scalable to minimise complexity problems. Furthermore, the IDS could be incorporated using a CSA to effectively improve its detection rate over MitM attacks.

A crucial part of how Internet applications function is the DNS. It offers a method of translating Domain names into the relevant IP addresses. DNS spoofing, also referred to as DNS cache poisoning, is an attack in which altered DNS records are used to redirect web traffic to a malicious or unreliable resource, potentially leading to data theft or exposure to harmful content.

The study in [42] presents a novel intrusion detection approach for IoT environments using ML ensemble techniques. The authors address the challenges of intrusion detection in IoT, involving large amounts of data from multiple sources, by proposing an ensemble-based approach. The approach uses multiple MLAs to learn normal behaviour patterns and detect anomalies. The study details the feature selection, training, and anomaly classification processes of the ensemble-based framework depicted in [43]. Simulation results highlight the effectiveness of the approach in detecting intrusions within IoT environments. This study contributes to the field of intrusion detection in IoT and highlights the potential of ML ensemble techniques for improving detection accuracy. The study presents a novel ML ensemble approach for intrusion detection in IoT environments, but it would benefit from real-world testing to validate its effectiveness beyond simulation results.

The study in [44] proposed an innovative method for detecting intrusions in IoT networks using ensemble modelling with edge computing. It reviews the limitations of traditional signature-based IDSs and introduces anomaly-based IDSs that identify new attacks by recognising deviations from normal behaviour. The proposed system

integrates various anomaly detection algorithms to enhance accuracy and robustness. Edge computing is used to enhance performance and efficiency by processing data closer to the source minimises the need for large data transfers to a central server. The proposed system has the potential to address the limitations of traditional signature-based intrusion detection systems and enhance the security of IoT networks.

The study in [45] proposes a new ensemble-based IDS for IoT, which consists of a feature selection algorithm, clustering algorithm, and classification algorithm. The feature selection algorithm identifies the most important features from the input data, while the clustering algorithm organises the data into groups. The classification algorithm then applies multiple models to each cluster and combines the results to generate the final decision. The proposed system was evaluated using two public datasets, and the results showed that it outperformed several existing IDS models in terms of accuracy, precision, and recall. The study also reflects the limitation in scalability and efficiency due to the resource-constrained IoT environments.

The study in [46] proposed an ensemble tree-based model for intrusion detection in IIoTs networks, applicable to smart homes and industrial control systems. The model has two stages: decision tree models for feature selection and classification, and an ensemble model combining multiple decision trees for improved accuracy. Evaluated on IIoTs network traffic, the model achieves high accuracy in detecting attacks like DoS, probing, and infiltration, demonstrating its effectiveness for IIoTs security. The study shows that there is a limitation in the proposed ensemble tree-based model such as the computational complexity, which may limit its scalability and efficiency due to the resource-constrained IIoTs environments.

The study in [47] explores the use of anomaly-based ensemble models for detecting intrusions across multiple domains. The authors trace the importance of detecting cyber threats in multiple domains and present the use of ensemble models to improve the accuracy and reliability of IDSs. The paper reviews existing anomaly-based ensemble models and discusses their performance on different datasets. The authors also propose a new ensemble model that combines multiple anomaly detection methods to enhance the identification of anomalies across various domains. The study

concludes with a discussion for future research directions in the field of anomaly-based ensemble models for intrusion detection.

The application of the IoTs in a variety of situations has given it prominence. The authors in [48] suggested a binary classification approach, created with a machine learning ensemble technique, to filter and block malicious traffic, to prevent attackers from accessing the IoT network and its peripherals. The simulation results demonstrate the proposed model's effectiveness against online attacks, making it appropriate for use in crucial IoT applications. The challenge observed from the study is the high rate of false alarms and false positives which occur when the IDS identifies legitimate traffic and classifies it as malicious, hence a proper method must be developed to address and minimise the false alarm rates.

The study in [49] proposes an IDS that uses an ensemble of multiple MLAs to improve the accuracy and robustness of the intrusion detection process. The authors describe the architecture of the proposed system, which includes data pre-processing, feature extraction, and feature selection modules, as well as the ensemble learning framework. The study provides details on the different MLAs used in the ensemble, including decision trees, SVM, and neural networks. The authors also describe the feature selection process, which involves selecting the key features from the telemetry data to enhance the accuracy of the IDS.

The study concludes with an evaluation of the proposed system, comparing its performance with other state-of-the-art IDSs. The authors demonstrate the effectiveness of their approach in detecting various types of attacks, including DoS and DDoS attacks. The study provides a comprehensive overview of the proposed ensemble learning-powered IDS for sensor monitoring data in IoT networks, highlighting its potential for improving the security of IoT networks

2.5.3 Federated Learning

The study in [50] focuses on the challenges and benefits of utilising Federated Machine Learning (FML) for anomaly detection in IoT and IIoTs environments. It proposes a security and privacy-aware Artificial Detection System (ADS) based on FML to address the growing concerns about data confidentiality in IoT and IIoTs applications. FML enables models to be trained across multiple devices without the need to share raw data, thereby maintaining privacy and enhancing security. The

review highlights a limitation of the original study, which was tested in a small environment. The performance of the FML-based ADS in a dynamic environment with factors like latency and network degradation remains unclear.

To address these challenges, the study proposes incorporating homomorphic encryption and differential privacy. These techniques can protect data transmitted in the presence of malicious activities and ensure data integrity even as the volume of data generated by IoT devices increases. The study emphasises the potential of FML for secure and efficient anomaly detection in IoT and IIoTs systems but highlights the need for further research to address its limitations in dynamic environments. The study effectively highlights the potential of FML for secure anomaly detection in IoT and IIoTs environments but falls short in dynamic, real-world conditions.

2.5.4 Deep Learning techniques

The study in [51] developed a deep learning-powered system for detecting DNS spoofing attacks. DNS spoofing is a risk that redirects traffic to bogus server locations by copying the real ones. Users are sent to malicious websites because they are unaware of these attacks, which causes the leakage of sensitive and private information. The proposed Ensemble Cuckoo scheme leverages the power of Deep Learning, including Convolutional Neural Networks (CNN) or Recurrent Neural Networks (RNN), to analyse the traffic patterns of DNS. However, it remains a challenge to detect sophisticated and evolving spoofing techniques, and this issue often leads to a higher false alarm rate, which burdens the server with unnecessary traffic, creating a loophole for serious attacks to bypass the detection mechanism. Therefore, there is still a need to develop scalable, robust, and adaptive models.

System administrators can employ a Network Intrusion Detection System (NIDS) to detect network security breaches within their organisations. However, developing a flexible and effective NIDS capable of handling unexpected and unpredictable attacks presents several challenges. The study in [52] lacks improvement in detection accuracy and robustness of detecting attacks in a complex environment. Therefore, the study in [51] proposed a deep learning-driven approach for developing an efficient and adaptable NIDS. The outcome of the study showed that the proposed model achieved high accuracy, precision, recall, and F-measure values, indicating its effectiveness in detecting and mitigating MitM attacks in edge networks and DNS.

These metrics collectively demonstrate the model's robustness and reliability identifying malicious activities accurately.

The study in [53] explores the application of DL for detecting anomalies in edge computing environments. The authors address the unique challenges associated with handling data at the network edge and propose a DL-based strategy to overcome these obstacles. Convolutional Neural Networks (CNNs) are employed in the proposed Ensemble Cuckoo scheme to identify anomalies by detecting deviations from the system's typical behaviour patterns. The study comprehensively covers the training process, feature extraction, and anomaly classification aspects of the CNN-based system.

Furthermore, the authors present simulation research results that demonstrate the effectiveness of the suggested method in identifying irregularities within fog computing architectures. This study provides a valuable contribution to the field of anomaly detection in fog computing, highlighting the capability of using DL techniques to improve the accuracy of detection. The study provides a thorough and insightful exploration of using DL for anomaly detection in edge computing, but it would benefit from real-world validation to confirm the effectiveness of the proposed CNN-based method beyond simulation results.

This study in [54] proposes an ensemble DL-based IDS for the IoT using Lambda architecture. The authors first discuss the challenges of developing an effective IDS for IoT, including the large-scale and heterogeneous nature of IoT devices and the need for real-time detection. To address the challenges, authors in [55] proposed an ensemble DL-based IDS that combines multiple DL models to improve the accuracy and effectiveness of intrusion detection. The system is designed using Lambda architecture, which enables the processing of both real-time and batch data streams.

The study describes the architecture of the proposed system, including the data processing and storage layers, the batch layer for offline processing, and the serving layer for real-time detection. The authors also provide details on the DL models used in the system, including CNNs and LSTM networks. The study concludes with an evaluation of the proposed system, including its performance in terms of accuracy and efficiency in identifying and categorising different types of attacks. The authors compare the results of their system with other state-of-the-art IDSs and highlight the

advantages of their approach, including its ability to handle large-scale and heterogeneous IoT networks.

Furthermore, the study provides a comprehensive overview of the proposed ensemble DL-based IDS for IoT using Lambda architecture, highlighting its potential for improving the security of IoT networks. This study presents an ensemble learning-based IDS for sensor telemetry data in IoT networks. The authors highlight the challenges of securing IoT networks and the need for effective IDSs to protect against attacks.

The study in [56] proposes an anomaly-based IDS for IoT networks using a DL model. The authors explain the need for effective IDS in IoT networks due to the increasing number of security threats and the vulnerabilities of IoT devices. The proposed IDS involves the collection of IoT network traffic data, which is pre-processed and used to train a DL model to detect anomalies in the data. The authors provide a detailed review of the different DL models that can be utilised for anomaly detection, such as Auto-encoder, Long Short-Term Memory (LSTM), and CNN.

They discuss the challenges of using DL models for anomaly detection in IoT networks, including the requirement for a lot of labeled data and the need for efficient models that can be deployed on resource-constrained IoT devices. The proposed approach is evaluated using a dataset of IoT network traffic, and the results demonstrate that the proposed approach is effective in detecting anomalies with high accuracy and low false positives.

The authors outline the shortcomings of the proposed approach and make recommendations, such as the use of transfer 5036 learning to improve the generalisation of the model and the use of FL to address privacy concerns in IoT networks. Furthermore, the study presents a valuable contribution to the field of anomaly-based IDS in IoT networks and highlights the potential of DL models for enhancing the security of IoT networks.

The literature review in [57] explores the current state of IDS for edge networks using DL. It covers key concepts of edge computing and the challenges of implementing IDS in these networks. The authors review DL algorithms like CNNs, RNNs, and LSTM networks for IDS. They discuss feature extraction techniques such as wavelet transformation and statistical analysis, and the challenges in selecting optimal

features. The review highlights the need for diverse and realistic datasets for training and testing IDS. It also compares various IDS for edge networks, noting their advantages and limitations. The study provides a comprehensive review of DL algorithms for IDS in edge networks, effectively highlighting key challenges and the need for diverse datasets, but it would benefit from practical implementation examples to validate its results.

This study in [58] proposes a deep intelligent attack detection framework for fog-based IoT systems. The framework utilises a deep neural network (DNN) for detecting attacks on fog-based IoT systems. The authors describe the architecture of the proposed framework, which includes three key components: data pre-processing, feature extraction, and classification. The framework is trained using a dataset of attack scenarios on fog-based IoT systems and is evaluated using various performance metrics.

The results demonstrate that the framework outperforms existing methods in terms of accuracy, precision, and recall. The authors conclude with a discussion of the advantages and limitations of the proposed framework and suggest potential future research directions. The growth of the IoT has increased the number of interconnected devices, and in tandem, the risk of cyberattacks. To counter these threats, various IDS have been developed. One approach is to use ensemble-based IDS, which combines multiple IDS models to improve the detection accuracy.

The study in [59] proposes an adaptative deep ensemble anomaly-based IDS and the purpose of the scheme is to improve the security of the IoT by detecting and mitigating malicious activities. The current study proposes a further consideration for the adaptative deep ensemble for real-time detection where investigations should explore more lightweight deep learning architectures, hardware acceleration, and edge computing methods to improve the system’s responsiveness.

5.5 Machine Learning Techniques

The demand for computer networks has grown significantly in the world. The study in [51] proposed a methodical strategy for utilising Forward Greedy Selection (FGS) and Random Forest (RF) algorithms to identify User Datagram Protocol (UDP) packet headers involved in MitM attacks within networks. Since there are very limited studies in the literature that focus on UDP, there is a dire need to explore features such as

insufficient labeled datasets for training, scalability, and the lack of schemes proposed to deal with MitM attacks on UDP. It is important to diligently advance the use of ML for attack detection on UDP packet headers.

This study in [60] provides a comprehensive evaluation of IDS in the IoT domain. The authors examined the techniques employed for IDS in IoT, including rule-based, anomaly-based, and ML-based approaches. They also highlight the importance of a deployment strategy for IDS in IoT and provide an overview of various deployment models such as centralised, distributed, and hybrid models. The study further discusses the validation strategies used for IDS in IoT, such as simulation-based and testbed-based validation, and the challenges associated with validating IDS in IoT environments. The authors then review the different kinds of attacks that can target IoT systems, such as DoS, MitM, and DDoS, and how IDS can be employed to identify and mitigate these attacks.

The study also discusses the importance of public datasets for evaluating and benchmarking IDS in IoT and highlights some of the existing datasets such as KDD Cup 1999, UNSW-NB15, and IoT-23. Finally, the authors highlight some of the challenges and open issues in IDS for IoT, such as the need for more accurate and efficient IDS algorithms, the integration of IDS with other security mechanisms, and the development of IDS for emerging IoT technologies.

The study in [61] focuses on the challenges and solutions related to intrusion detection in IoT networks. The study emphasises the importance of effective data engineering techniques, such as feature selection, scaling, and dimensionality reduction, for improving intrusion detection performance. Various MLAs, including SVMs, decision trees, RFs, and DL models, are used to detect and classify intrusions. The studies evaluate the performance of the proposed systems using real-world IoT traffic datasets and demonstrate their effectiveness in detecting and classifying intrusions with high accuracy and low false positives. The study highlights the importance of data engineering techniques and demonstrates the high accuracy of various MLAs for intrusion detection in IoT networks, but it could further explore the scalability and adaptability of these methods in diverse IoT environments.

The study in [62] presents a comprehensive review of IDS using machine and DL techniques for IoT environments. The study addresses the unique challenges posed

by IoT networks, such as their heterogeneity, scale, and dynamic nature. The study proposed an IDS as a solution to these challenges and reviewed various Machine learning and deep learning techniques, encompassing supervised learning, unsupervised learning, and deep learning methods. The review provides a detailed analysis of the advantages and drawbacks of these techniques. The study provides a thorough review of IDS using machine and deep learning techniques for IoT environments which address the unique challenges of IoT networks, but it could benefit from practical insights into the implementation and real-world performance of these techniques.

The study in [63] explored using ML to enhance IoT security by detecting attacks. It highlights the challenges of IoT networks, such as their diversity and scale, and proposes a method involving data collection, pre-processing, and training models (like neural networks and decision trees) to classify data as normal or malicious. Key features for classification include network traffic, device behaviour, and environmental factors. The study further discusses various ML techniques (supervised, unsupervised, RL) and addresses challenges like the need for large labeled datasets, real-time detection, and resource-efficient models. The study explores the use of ML to enhance IoT security by detecting attacks, but it would benefit from empirical validation and practical implementation examples to support its proposed methods.

The study in [64] presents a case study using a neural network to detect attacks in an IoT network, demonstrating high accuracy and low false positives. This showcases the effectiveness of their proposed approach. The study significantly contributes to IoT attack detection using ML, highlighting the potential of these techniques to enhance IoT network security.

The study in [65] explores the application of ML techniques for detecting network security threats in edge computing systems. The authors describe the security challenges in edge computing, which incorporate the large number of edge devices, the distributed architecture of edge networks, and the need for real-time threat detection. They propose the use of ML for detecting network security threats in edge computing systems, it entails gathering data on network traffic, feature extraction, and the training of ML models, including neural networks and decision trees, to determine whether the data is malicious or normal. The study addresses the application of ML

techniques for detecting network security threats in edge computing systems, but it would benefit from real-world validation and practical implementation examples to support its proposed methods.

The study in [66] discusses various features that could be used for classification, including network traffic characteristics, application behaviour, and device features. Ultimately, this study provides a thorough analysis of the strengths and weaknesses of various ML techniques that can be used for detection. They also discuss the challenges of using ML for network security in edge computing, such as the demand for real-time detection and resource-efficient models.

The case study in this instance shows the effectiveness of the proposed approach, which involves using CNN to detect network security threats in an edge computing system. The results show that the proposed approach is effective in detecting network security threats with high accuracy and low false positives. The study is a valuable contribution to the field of network security in edge computing systems and highlights the potential of ML techniques for enhancing the security of edge networks. The study provides a thorough analysis of ML techniques for network security in edge computing, demonstrating the effectiveness of CNNs, but it would benefit from real-world validation to support its results.

The study in [67] observed the challenges discovered in [41] and proposed a scheme known as the Hybrid Intelligent IDS integrated using ML and meta-heuristic algorithm for applications in IoT-based healthcare. The study aimed to provide robust and reliable security for the healthcare system which has become vulnerable to security attacks. This study proposes that sensors utilised on the network infrastructure should have high quality and meet the desired requirements.

A license for online access to a specific cloud application is known as a Software-as-a-Service (SaaS). However, due to the Internet's periodic availability, which presents opportunities for numerous attacks, these services are occasionally delayed or completely interrupted. The aim of the study in [68] proposes a scheme that provides robust and reliable security for centralised cloud applications that are vulnerable to security attacks. Their scheme includes the utilisation of MLA to improve the detection rate and mitigate the attacks by focusing on real-time analysis and response.

The limitations of this study are similar to those mentioned in [69], such as limited scalability, high false positive rate, inadequate real-time detection, insufficient adaptability, and resource intensiveness, hence there is still a need to fill this gap by proposing continuous monitoring and evaluation of the network's performance to identify malicious activities and to consider mitigation measures using ensemble modeling to detect any activities deviating away from normal. While ensemble modelling is a powerful approach for detecting deviations from normal network activities, its certainty is not only the solution. Ensemble modelling combines the strengths of multiple machine learning algorithms to improve accuracy and robustness, making it effective for identifying complex patterns and anomalies.

The study in [70] reviews anomaly detection techniques for IoT security using ML and data mining. It discusses IoT security challenges and various anomaly detection methods, including statistical-based, clustering-based, and classification-based techniques. The authors explore MLAs like Decision Trees, Random Forest, SVM, and DNN for anomaly detection, emphasizing the importance of feature selection with techniques like PCA and ICA. The study highlights the advantages and limitations of different anomaly detection systems for IoT security. The study provides a review of anomaly detection techniques for IoT security and highlights the importance of feature selection, but it could benefit from real-world validation of the proposed methods.

The study in [71] proposes a novel IDS for IoT devices using a DT based on ML and DL. It addresses the challenges of traditional IDS in IoT due to limited resources. The system combines feature extraction techniques like PCA and ICA with a decision tree algorithm to identify network traffic as either normal or anomalous. Compared to other IDS, it achieves high accuracy and reduces false positives and negatives. The system is noted for its low computational complexity, scalability, and adaptability to various IoT devices, providing an effective approach to enhance IoT security however, the study would benefit from further empirical testing to validate its scalability and adaptability in diverse IoT environments.

The study in [72] proposed a novel approach to intrusion detection in IoT environments using an ML ensemble. The proposed system consists of three stages: pre-processing, feature extraction, and classification. The pre-processing stage involves data cleaning and transformation. The feature extraction stage uses Principal

Component Analysis (PCA) to reduce the dimensionality of the data, and the classification stage involves using an ensemble of MLAs to detect intrusions. The proposed system is evaluated using the UNSW-NB15 dataset, and the results confirm that the proposed system outperforms existing approaches in terms of accuracy and detection rate however, the study does not address its applicability in real-world which may confirm its effectiveness beyond the UNSW-NB15 dataset.

The study in [73] discusses a survey of recent research on the use of ML in the context of edge computing for enhancing security and privacy provisioning. The authors highlight the increasing need for such solutions as more and more devices and services are implemented at the network's edge. It further discusses various approaches to using ML for edge security and privacy, including anomaly detection, intrusion detection, and predictive modelling. The authors also provide a classification framework for these approaches based on the types of data sources, features, and algorithms used. The study goes on to describe some of the challenges and opportunities associated with ML-assisted edge security and privacy provisioning. This includes issues associated to data privacy, scalability, and the need for robust and reliable ML models.

The authors further reviewed some recent research in this area, including case studies and experiments that demonstrate the effectiveness of ML for edge security and privacy. Finally, they identify some of the open research questions and future directions for this field. The study provides a comprehensive overview for the current state of research on ML-supported security and privacy management for edge computing, highlighting its potential for improving the security and privacy of edge systems. The study lacks empirical validation of ML techniques in real-world edge computing environments. Hence, conducting practical experiments and case studies to validate the effectiveness and scalability of ML techniques in diverse, real-world edge computing scenarios may be beneficiary.

The study in [74] reviews some of the recent research in the area of cyberattack detection in IoT devices, including case studies and experiments that demonstrate the effectiveness of machine learning-based methods. The authors also provide a critical analysis of the strengths and weaknesses of these studies, highlighting the need for more comprehensive evaluations and benchmarks. The authors identify some of the

key challenges and prospects for ML-based methods for cyberattack detection in IoT, such as developing more efficient and scalable algorithms, addressing issues related to data privacy and security, and integrating these methods into real-world IoT systems. These essential highlights their potential for improving the security and resilience of IoT systems in the face of growing cyber threats.

Compared to the traditional supply chain, the Fourth Industrial Revolution (Industry 4.0) is transforming the future of supply chains by enhancing their agility and efficiency. However, due to its diverse and ever-changing nature, as well as the fact that non-professional users often manage their information systems when it comes to security matters, data communication between partners in the Industry 4.0 supply chain can be vulnerable to a variety of attackers who exploit security breaches in both the internal and external environment of the partners.

Attackers can breach the data connection between authorised parties in the Industry 4.0 Supply Chain, endangering both the continuity and supply of services to all partners. Consequently, the study in [59] proposed an efficient TLS-based authentication mechanism, that minimises the occurrence of MitM attacks in web applications since the attackers have various ways to compromise the data communication between legitimate parties in the industry.

This study in [75] discusses the challenges and unresolved challenges related to the application of ML in securing IoT devices against Advanced Persistent Threats (APTs). The authors initially highlight the growing security threats posed by APTs, which require sophisticated and adaptive security measures for detection and prevention. They then provide an overview of the state-of-the-art ML techniques used for IoT security, including supervised and unsupervised learning, DL, and ensemble methods. The authors also discuss the challenges of implementing ML for IoT security, such as data privacy and confidentiality, resource constraints, and the need for real-time processing.

The study goes on to discuss various approaches for addressing these challenges, including edge computing, FL, and the use of lightweight MLAs. The authors also identify several open issues and challenges related to the use of ML for IoT security, such as the need for more robust and scalable models, the development of explainable and interpretable algorithms, and the integration of ML with other security

mechanisms. The study concludes by discussing the future objective and potential for ML in IoT security, including the use of advanced techniques such as adversarial ML and RL. The study provides a comprehensive summary of the challenges and open issues focused on the application of ML in IoT security, emphasizing the necessity for further research to create effective and robust security measures against APTs.

2.5.2 Block-chain methods

The study in [76] examines the challenges and benefits of using a classification and threat scheme for analysing security and privacy risks in ML for data analytics. The study identifies a gap in the original study related to the data's origin and accountability. It outlines incorporating block chain-based techniques to address these issues. Block-chain could improve the security, transparency, and traceability of data shared across networks.

Federated Learning (FL) provides a centralised administration for attack correlation, enhancing security measures. The study in [63] highlights the privacy concerns associated with traditional IDSs that require data sharing. FL provides a solution by enabling collaboration without disclosing personal information. To minimise communication overheads, it further recommends incorporating parameters like data processing, model compression, and dimension reduction into the FL-based IDS model. The study addresses the challenge of data heterogeneity in edge computing environments and emphasises the need for a robust scheme to handle data from various devices. The study also highlights the security concerns of edge devices that may not be fully integrated into the network's security framework. It is crucial to emphasise the importance of edge devices having security measures to protect against potential compromises.

The emergence of edge computing enables the implementation of cutting-edge technologies like virtual reality and augmented reality. Therefore, providing users with faster network services while maintaining data transfer secrecy and authentication contributes to new horizons. The study in [77] proposed a certificateless signcryption mechanism based on Blockchain technology for edge networks. The proposed Ensemble Cuckoo scheme enables edge networks to allow devices to communicate effectively and securely without relying on conventional certificate-based methods. Although the scheme shows promising results, several drawbacks have been

observed, such as data quality, availability, and privacy concerns. The scheme must ensure proper data anonymisation, encryption, and secure data handling techniques to protect user privacy.

IoT technology is crucial to many aspects of life and business. However, the increased use of IoT has brought diverse security issues that compromise data privacy and slow down the adoption of the technology in critical areas like the smart grid and intelligent transportation systems. To address this challenge, several methods have been developed to identify and circumvent IoT cyber threats. One such technique is anomaly detection, which establishes the limits of acceptable (normal) behaviour. The study outlines several challenges, including computational requirements, training data availability, and adaptability to evolving attacks.

The study in [78] provides a comprehensive review of IDSs for IoT environments. It addresses the unique challenges posed by IoT networks, such as data sparsity, model complexity, and energy efficiency. The authors propose transfer learning, federated learning, and edge computing as solutions to address these challenges. Additionally, the study discusses future directions for IDS in IoT, including integrating multiple techniques, developing explainable AI models, and utilising blockchain and emerging technologies. Furthermore, the study provides a comprehensive overview of the current state-of-the-art in IDS for IoT and highlights the potential of machine and deep learning techniques to enhance IoT network security. The study provides a thorough review of IDSs for IoT environments, proposing innovative solutions and future directions, but it would benefit from empirical validation of the suggested approaches.

2.5.3 Other Methods

The study in [79] focuses on the challenges and benefits of migrating data management and security tools from cloud computing to edge networks for IoT and IIoTs applications. Traditional cloud-based approaches face limitations due to high data volumes, resource constraints, and security concerns. Edge computing offers a decentralised solution by processing data closer to the source, minimising latency, and improving efficiency. The study identifies a gap in the original study related to data privacy and security. It suggests incorporating techniques like data encryption, secure communication protocols, and differential privacy to protect sensitive data. The research highlights the importance of considering privacy and security when

implementing edge computing solutions for IoT and IIoTs applications. By addressing these concerns, organisations can leverage the benefits of decentralised data processing while securing sensitive information.

However, other methodologies can also be considered to enhance network security. On the internet, the DNS is essential for establishing connections between services and users. In various applications after its initial design, DNS has been expanded to keep up with the demands and challenges of the modern world. There are numerous difficulties. Changes to ensure the anonymity of DNS queries were made in response to revelations of DNS traffic eavesdropping. Attempts to spoof DNS communications prompted modifications to support DNS integrity. Challenges that were identified by the authors in [80] are security threats, scalability, performance, and privacy on which they also provided guidelines to address these challenges. Furthermore, a thorough strategy is necessary to handle the difficulties presented by current DNS monitoring and examining DNS traffic and configurations regularly as protocols designed to improve network security and performance.

The study recognises, analyses, controls, and optimises the traditional physical system using pervasive IoT applications. Several IoT applications have recently had security vulnerabilities, which put physical systems at higher risk. There are several security challenges outlined in this study such as maximised network complexity, limitation of resources, and concern about the security and privacy. The study in [81] proposes an Edge Computing-Based security for IoT devices to minimise the drawbacks brought in by traditional security mechanisms, including the inability to manage the large volumes of data generated by IoT devices, and their vulnerability to security attacks because of the lack of computing power and limited storage capacity.

The study improves the IoT devices' security. However, it is still not clear how the study addresses the issue of increased network complexity. To address this issue, the current study proposes the use of hardware-based security measures, including a trusted execution environment, and securely separated regions to ensure security for IoT applications. It is important to calculate the strengths and drawbacks of the proposed Ensemble Cuckoo scheme designs and ensure that implementation

involves appropriate measures that would improve the network security and privacy concerns.

MitM poses serious attacks, particularly on Industrial Control Systems (ICS). One method used to help identify MitM attacks is the use of honeypots. The study in [82] proposed a Markov Decision Process (MDP) incorporated with State-Action-Reward-State-Action (SARSA). The study was compared to the traditional IDSs and it has achieved a higher accuracy and convergence speed. The study did not consider scalability, which could be difficult to manage and maintain in a large and complex environment.

There is therefore an urgent need to implement appropriate measures that ensure efficiency and good security, thereby mitigating the risks. The rapid improvement of technology and widespread usage of internet networks around the world are to blame for the sharp rise in cybercrime. The lesson of 2019 is that no company, no matter how big or small, is safe from a cyberattack. Cyber-attacks are more advanced and challenging to identify. As a result, security needs to be updated frequently.

The Industrial Internet of Things (IIoTs), which is an open and networked system, has transformed the smart manufacturing environment. Consequently, cyberattacks on smart manufacturing facilities are now more likely to cause physical harm. The majority of cyberattacks on smart factories use malware. Therefore, it is essential to have a system that effectively detects malware in IIoT environments for smart factories by monitoring and analysing network data for malware attacks. The study in [83], proposed an IIoTs Malware detection using Edge Computing and Deep Learning for cybersecurity in smart factories to detect and mitigate the malware attacks in the IIoTs environment.

The problem identified in this study is the unavailability of labeled IIoTs malware datasets. To overcome this challenge, data should be created based on features such as scalability and adaptability to improve the detection rate. One of the most practical and easily implemented Ad-hoc techniques for data transfer is the Wireless Sensor Network (WSN). While WSN provides significant flexibility, it has numerous flaws. MitM, black holes, and other forms of attacks are all made possible by the decentralised architecture of WSN.

The study in [84], presented a MitM-IDS (MitM-IDS) paradigm for attack detection, isolation, and node reconfiguration involves using an Intrusion Detection System (IDS) to proactively prepare nodes for potential threats. The IDS method helps identify attacks, isolate compromised nodes, and reconfigure the network to enhance security and prevent further damage. The study did not consider energy consumption and it is not clear what would happen if the number of nodes increases in terms of network scalability, which could greatly influence the detection scheme. In computer security, the End-point (EP) MitM attack is a well-known danger. The exchange of information between endpoints is the target of this attack.

The confidentiality and integrity of data flow are affected by the attacker's ability to eavesdrop on the conversation between two targets and execute either active or passive monitoring. Researchers have created strategies to counteract this form of attack. The study in [85], proposed a detection EP MitM attack based on Address Resolution Protocol (ARP) analysis. The study observed that real-time detection and mitigation strategies were not considered. A new approach is therefore needed to investigate the feasibility of collaborative detection approaches, by focusing on these areas and developing more effective and reliable approaches for detecting MitM attacks on end-points.

A severe network attack poses a growing threat to modern intelligent Train Communication Ethernet (TCE). Among the most overwhelming attacks are the MitM attacks, which are challenging to identify using standard techniques. Based on these attacks, the study in [86] proposed a dynamic temporal convolutional network-based IDS (DyTCN-IDS) to detect temporal attacks. While their scheme performed better than existing ones, several factors impacting its performance were not considered, such as robustness against adversarial attacks, real-time detection, and network-specific attacks.

This study aims to address these issues by investigating specific attacks targeting the control system of the network protocols and developing a detection mechanism. Traditional cloud computing encounters difficulties in addressing the real-time demands of IoT services due to significant network latency from the exponential growth of IoT data. Edge Computing (EC) addresses this by moving data processing

to edge nodes, significantly enhancing Quality of Service (QoS) for low-latency IoT applications.

The study in [87] provides a comprehensive overview of MitM attacks, their effects, and potential countermeasures. The study highlights the commonality of MitM attacks as a precursor to other attacks like DoS, DNS spoofing, and port stealing. The study proposes a scheme to prevent MitM attacks. The limitations discovered in this study are the real-world implementation, highlighting challenges that affect SSP security. There is a need to develop and propose countermeasures to improve mitigations identified through the MitM attacks to secure simple pairing.

The study in [88] presents an IDS specifically designed to detect temporal attacks in train communication Ethernet networks. Temporal attacks involve patterns of behaviour that change over time, making them difficult to detect using traditional static methods. The proposed IDS utilises a Dynamic Temporal Convolutional Network (DTCN) to address this challenge. The DTCN-based approach includes a feature extractor that captures the temporal behaviour patterns of the network and a classifier that detects anomalies based on deviations from these patterns. Simulation results show the effectiveness of the proposed IDS in accurately detecting temporal attacks in train communication networks. This research contributes to the field of intrusion detection in dynamic network environments and highlights the potential of DTCN-based approaches to improve detection accuracy.

The study in [89] proposes an anomaly-based IDS for the IoT using a CNN and a Multi-Objective Enhanced Capuchin Search Algorithm (MOECSA). The authors describe the need for effective IDS in IoT because of the numerous devices and the increasing number of security threats. The proposed IDS involves the collection of IoT data, which is pre-processed and then used to train a CNN to detect anomalies in the data. The MOECSA algorithm is used to optimise the hyper-parameters of the CNN and improve its accuracy.

The authors discuss the key features that can be used for anomaly detection, including device behaviour, network traffic, and environmental factors, and provide a comprehensive analysis of various CNN architectures and hyper-parameter tuning techniques that can be used for detection. They also discuss the challenges of using

CNN for anomaly detection in IoT, including the requirement for a lot of labeled data and the requirement for resource-efficient models.

The proposed approach is evaluated utilising an IoT traffic dataset, and the results show that the proposed approach is efficient in identifying anomalies with high accuracy and low false positives. The study makes a valuable contribution to the field of anomaly-based intrusion detection in IoT and highlights the potential of CNN and MOECSA techniques for improving IoT network security.

The study in [90] proposes an IDS for Train Communication Ethernet (TCE) networks using a Dynamic Temporal Convolutional Network (DTCN) to detect temporal attacks. The authors explain the importance of securing TCE networks due to the critical nature of train communication and the potential for cyberattacks to disrupt train operations, ensuring robust security measures is essential. The proposed IDS involves the collection of TCE network traffic data, which is pre-processed and used to train a DTCN model to detect temporal anomalies in the data.

The authors provide a detailed analysis of the DTCN model and explain how it could be used to detect temporal attacks, such as port scans and Denial-of-Service (DoS) attacks. They also compare the proposed approach with other ML-based IDS and traditional IDS to demonstrate that the proposed approach outperforms others in terms of accuracy and the rate of false-positives.

The proposed approach is evaluated using a dataset of TCE network traffic, and the results demonstrate that this approach is effective in detecting temporal attacks with high accuracy and low false positives. The authors address the shortcomings of the proposed approach and recommend future research directions, such as the use of transfer learning to improve generalisation of the model and the deployment of IDS on resource-constrained TCE devices. The study presents a valuable contribution to the field of IDS for TCE networks and highlights the potential of DTCN models for enhancing the security of TCE networks.

The study in [91] proposes an anomaly-based IDS for IoT networks with improved data engineering to address the issues of data sparsity and heterogeneity in IoT networks. The authors explain the need for effective IDS in IoT networks due to the increasing number of security threats and the vulnerabilities of IoT devices. The proposed IDS involves the collection of IoT network traffic data, which is pre-processed using

improved data engineering techniques, such as feature selection and normalisation, to address the challenges of data sparsity and heterogeneity.

The authors analyse various MLAs that can be used for anomaly detection, such as k-nearest Neighbour (k-NN), SVM, and Decision Tree (DT). They also compare the proposed approach with other ML-based IDS and demonstrate that this approach outperforms other approaches in terms of accuracy and false-positive rate, the proposed approach is evaluated using a dataset of IoT network traffic, and the results demonstrate its effectiveness in detecting anomalies with high accuracy and low false positives.

The authors further outline the shortcomings of the proposed approach and make recommendations for future research, such as the use of DL algorithms to improve detection performance and the integration of the IDS with other security mechanisms to enhance the overall security of IoT networks. It also presents an important contribution to the field of anomaly-based IDS in IoT networks and highlights the potential of improved data engineering techniques for enhancing the performance of IDS.

The study in [92] provides a comprehensive review of IDS in IoT, highlighting the various techniques, deployment strategies, validation strategies, attacks, public datasets, and challenges associated with IDS in this domain. This study presents a systematic literature review of using mobile computing for security analysis of IoT devices. The authors highlight the importance of security in IoT devices since, these devices are becoming more ubiquitous and integrated into daily life.

The study then describes the methodology used for the literature review, which involved searching various academic databases and selecting relevant studies based on specific inclusion and exclusion criteria. The selected articles were then analysed to identify common themes. The authors identify four main themes in the literature: (1) mobile-based IoT security assessment, (2) security risks and vulnerabilities of IoT devices, (3) mobile-based detection and securing against IoT security threats, and (4) mobile-based mitigation and recovery of IoT security incidents.

The study further provides a depth analysis of the results related to each theme, highlighting the main approaches, techniques, and challenges identified in the literature. The authors further highlight the constraints of the current research study

and provide recommendations for future research in this field. Furthermore, the study provides a comprehensive review of the current state of research on using mobile computing for security analysis of IoT devices. The authors highlight the potential benefits of this approach, as well as the challenges and limitations.

The study in [93] proposed a transfer learning-based threat model called state action learning transition (SALT) for identifying attacks in smart home systems. The authors highlight the increasing risk of cyberattacks in smart home systems and the need for effective threat detection models to mitigate these threats. SALT uses transfer learning to enhance accuracy and efficiency of the threat detection process. The model is trained on a large dataset of attack scenarios and their corresponding features and then fine-tuned on a smaller dataset of smart home data. This enables the model to learn from a wide range of attack scenarios and apply this knowledge to detect new and unforeseen attacks in smart home systems.

The study provides details on the architecture of the SALT model, including the feature extraction and selection modules, as well as the transfer learning framework. The authors also describe the different attack scenarios used in the training and evaluation of the model, including replay attacks, packet sniffing attacks, and password attacks. The study concludes with an evaluation of the SALT model, comparing its performance with other state-of-the-art attack detection models. The authors demonstrate the effectiveness of their approach in detecting various types of attacks, with high accuracy and low false positive rates. Furthermore, the study provides a comprehensive overview of the SALT transfer learning-based threat model for attack detection in smart home systems, highlighting its potential for improving the security of these systems.

The study in [94] proposed a multi-level random forest model-based IDS for IoT networks using a fuzzy inference system. The proposed IDS comprises three levels of random forest models, with each level performing an increasingly specific analysis of the network traffic data. The fuzzy inference system is used to combine the outputs of the random forest models and make the final intrusion decision. The experimental results demonstrate that the proposed IDS outperforms other state-of-the-art methods in terms of accuracy and false alarm rate. The proposed IDS can effectively detect

different types of attacks, including brute-force attacks, DDoS attacks, and SQL injection attacks, in an IoT network.

The study in [95] proposes an enhancement to an IoT hybrid intrusion detection system based on fog-to-cloud computing. The proposed system includes three components: (1) a local detection module for detecting known attacks and anomalies at the fog level, (2) a cloud detection module for detecting unknown and complex attacks at the cloud level, and (3) a fuzzy rule-based system for decision-making. The system uses data mining techniques to analyse the data collected by sensors and the data generated by the system itself. The proposed system is evaluated using the KDDCup99 dataset, and their simulation results show that the system achieves a high accuracy rate in detecting various types of attacks with a low false alarm rate. The proposed system can be used in various IoT applications, such as smart cities and industrial control systems, to enhance the security and reliability of these systems.

The study in [96] examines the challenges and solutions linked to intrusion detection in IoT networks. The study proposed a two-stage approach to address these challenges. The first stage involves pre-processing IoT traffic data using data engineering techniques like feature selection, scaling, and dimensionality reduction. The second stage utilises an MLA, such as an SVM, to detect and classify intrusions. This approach aims to enhance the accuracy and efficiency of intrusion detection in IoT environments. The study proposes a promising two-stage approach for enhancing intrusion detection in IoT networks, but it would benefit from empirical validation to demonstrate its effectiveness in real-world scenarios.

2.5.4 Game theory-related works

The study in [97] focuses on the issue of MitM attacks, which are prevalent in networks and can lead to significant information leakage and financial loss. The study aims to develop a defence strategy against MitM attacks using game theory, specifically focusing on scenarios where defenders are self-interested and not cooperative. The authors modeled the MitM attack-defence scenario as a simultaneous-move game and adopted Nash equilibrium as the solution. The study proposed a practical adaptive algorithm for both defenders and attackers to learn and converge toward unique Nash equilibrium through repeated interactions. The simulation results of the study indicate that their proposed algorithms could effectively converge with the Nash equilibrium

strategy. This indicates that the defenders and attackers can adapt their strategies over time to reach an optimal balance thereby minimising the total loss from MitM attacks.

While the study provides a robust theoretical framework and promising simulation results, it further reflects limitations such as real-world applicability because the study's assumptions about the rationality and self-interest of defenders may not fully capture the complexities of real-world network environment. The study needs to be tested in larger, more complex network settings to evaluate its scalability and effectiveness. Furthermore, the study lacks a detailed discussion on the practical implementation of the proposed defence strategies in real-world systems.

The study is designed to counter the MitM attacks using game theory focusing on defense strategies, while anomaly-based systems detect attacks by identifying deviations from normal behavior. Combining these approaches could enhance security. The adaptive algorithms from the study can improve anomaly-based systems' ability to adapt to evolving attacks. Additionally, the ensemble modeling in anomaly-based systems, which can benefit from game-theoretic insights leading to more robust detection mechanisms. This integration could result in a more effective and adaptive security solution for MitM attacks in DNS and edge computing environments.

A study in [98] examines how first-mover advantage impacts security investment and free-riding behaviour in interdependent security games. The study integrates networks and Stackelberg game models to analyse scenarios where defenders and attackers interact. The authors found that the first-mover advantage can lead to under-investment in security due to free-riding among defenders. The study further highlighted the importance of considering both inter-dependency and sequential moves in security strategies to achieve optimal outcomes.

While the study provides valuable insights into security investment and free-riding behavior, it has some limitations. The assumption that all participants act rationally may not hold in real-world scenarios. Additionally, the model may oversimplify the complexities of actual networks where various factors influence security decisions, and the study lacks a detailed discussion on practical implementation in diverse and dynamic environments. Addressing these limitations could enhance the applicability and effectiveness of the proposed strategies.

A study in [99] addresses the challenges of MitM attacks, which can lead to significant financial and security issues. Unlike the conventional approaches that focus on detection and prevention, this study aims to minimise the overall system loss from inevitable MitM attacks. The authors modeled the interaction between attackers and defenders as a Stackelberg security game and adopted the Strong Stackelberg Equilibrium (SSE) as the defender's strategy. They developed a novel method to minimise the search space for computing the optimal defence strategy given the infinite strategy space. The simulation results of the study showed that the proposed game-theoretic defence strategy significantly outperformed non-strategic defence strategies in minimising total losses from MitM attacks.

While the study presents a robust theoretical framework and promising results, it further reflects some limitations including the assumptions about the rationality of attackers and defenders which may not fully capture real-world complexities. Additionally, the practical implementation of the proposed strategies in diverse and dynamic network environments is not thoroughly discussed which affects the applicability of the proposed Ensemble Cuckoo scheme in a real-world.

2.6 Chapter Summary

Based on the literature review, it can be concluded that there is significant interest in developing an effective anomaly-based IDS for edge networks and DNS. An Anomaly-based IDS offers several advantages including detection of unknown threats, comprehensive monitoring, minimal human intervention, enhanced security, and proactive defence. However, these benefits cannot be fully realised if persistent issues such as the high false positive rates, resource intensity, difficulty in defining normal behaviour, adaptability to new threats and scalability are not addressed.

Ensemble modelling has been widely used in related studies with the combination of other algorithms to enhance detection accuracy. The application of ML algorithms is a promising solution which explains why there is still a need for the development of new integrated methods to address the conundrum above. The novelty of our study is to develop an anomaly-based detection scheme and use the CSA to optimise the ensemble model offering benefits such as improved accuracy in exploring the search space. The envisaged anomaly-based MitM IDS detection finding offers optimal solutions since the CS algorithm provides faster convergence to minimise the time

required to train the model, global search capability, and adaptability. The CS algorithm is efficient in terms of computational resources, making it suitable for large-scale tasks.

Chapter 3

Methodology

3.1 Introduction

This study uses an experimental-based design approach to provide a comprehensive analysis of anomaly-based MitM IDS development. The approach involves the measurement and analysis of the effectiveness of the IDS using statistical metrics, such as accuracy, precision, recall, and F-measure values. The study also encompasses a detailed examination of the techniques and processes used in developing and deploying the IDS. The Chapter further focuses on the techniques used to develop and deploy the anomaly-based MitM IDS. The Anomaly-based MitM IDS is developed to accurately detect MitM attacks in Edge Networks and DNS environments, with a focus on minimising false positives and negatives.

We use the CSA to optimise the parameter selection process within the ensemble model RF to adapt to emerging threats, thereby ensuring data integrity and network security. By simulating the laying of eggs by cuckoos in randomly chosen nests and applying the principle of survival of the fittest to retrain the best solutions, the algorithm iteratively improves the model parameters. Detecting MitM attacks presents significant challenges due to encrypted traffic, advanced attack methods, and the need for immediate response to threats without impacting network performance.

Our scheme employs ensemble modelling and CS to detect anomalies signaling MitM attacks. It uses adaptive models that can adjust their parameters in real-time as new data is encountered, improving their accuracy over time in detecting anomalies. The significance of this work in the field of cybersecurity is profound. Developing a detection scheme tailored for edge networks and DNS addresses the unique vulnerabilities present in these critical infrastructures. This study enhances cybersecurity by introducing an adaptive detection mechanism. This mechanism can cope with a dynamic nature and network traffic with constantly changing patterns. Additionally, it aims to improve the detection accuracy.

This research is particularly important for stakeholders in cybersecurity, as it provides a cutting-edge solution to protect critical network infrastructure. Enhancing the detection and response to MitM attacks ensures the security and trustworthiness of communications, which is vital for businesses, governments, and individuals relying

on digital networks. The advancements presented here aim to set new standards for cybersecurity measures, directly contributing to the overall resilience of our digital society against sophisticated cyber threats.

3.2 Research Design

To evaluate the effectiveness of the proposed Ensemble Cuckoo scheme using ensemble modelling, we improve the detection accuracy, response time, minimising MitM attacks, and adaptability of new emerging threats by using the CSA and we compare the proposed Ensemble Cuckoo scheme's effectiveness with existing IDS solutions identified from the literature review such as [10]. This clarifies the improvements and advantages of our approach. We use statistical tools such as Decision trees and anomaly-based tools such as the Random Forest to train our scheme further and take these outputs as input to the random forest algorithm. We evaluate our proposed Ensemble Cuckoo scheme using metrics such as accuracy, precision, recall, f1-score, ROC Curve, and AUC. We use hypothesis testing to determine any significant difference between the detection rates of different IDS configurations.

3.3 Research tools

We use a Kitsune Network Attack Dataset freely available on the Kaggle platform [100]. Figure 3.1 depicts the network topologies used for data collection and the corresponding attack vectors where the attacks were executed. Network traffic was captured at point 1 and point x at the router. Initially, clean network traffic was recorded for the first 1 million packets, followed by the execution of cyber-attacks shown in figure 3.1. The dataset set has been used by various studies for classification purposes, and we use it for our detection scheme. The dataset includes a wide range of attacks, including MitM, Denial of Service (DoS), and botnet attacks which help in training robust IDSs.

Ensemble modeling optimised with the CSA was chosen for detecting anomaly-based MitM attacks due to its superior robustness and accuracy compared to other methods. Ensemble modeling leverages multiple models to create a more reliable prediction system mitigating the weaknesses of individual models and enhancing overall performance. The CSA inspired by the brood parasitism of cuckoo birds excels in solving complex optimisation problems by efficiently fine-tuning the ensemble model's

parameters, thereby improving detection rates and minimising false positives and negatives. While other methods like Local Outlier Factor (LOF), Support Vector Machine (SVM), and Random Forest (RF) were considered, they each had limitations: LOF may struggle with high-dimensional data, SVM can be computationally intensive with large datasets, and RF can overfit in noisy data scenarios.

Additionally, heuristic-based detection and statistical anomaly detection methods were reviewed, but their limitations in handling zero-day attacks and requiring extensive tuning respectively made them less ideal. Machine learning-based anomaly detection was also considered but posed significant challenges in integration and real-time performance due to high computational demands. By optimising ensemble models with the CSA, this study aims to combine the strengths of various detection techniques, ensuring high detection accuracy and efficiency, and providing a more reliable solution for identifying MitM attacks within edge networks and DNS environments.

3.4 Ensemble solution and ML techniques

The proposed ensemble solution utilises the RF as a homogeneous ensemble, where it creates an ensemble of decision trees which are trained on various subsets of the data and features to enhance accuracy and robustness. The RF technique was chosen for its complementary strengths in handling various aspects of the data and improving the overall detection performance.

3.4.1 Feature Selection

The model is trained using a dataset with 115 features. To prevent overfitting and enhance the model's generalisation capability, feature selection techniques such as Principal Component Analysis (PCA) and Recursive Feature Elimination (RFE) were employed. These techniques help in identifying the most relevant features, minimising the dimensionality of the data, and improving the model's performance.

3.4.2 Dataset Partitioning

The dataset consists of 2.5 million instances. To ensure a fair evaluation of the model, the dataset is partitioned into training and testing sets. The partitioning is done as follows:

Training Set: 70% of the dataset (1.75 million instances) is used for training the model.

Testing Set: 30% of the dataset (750,000 instances) is used for testing and validating the model's performance.

This partitioning strategy ensures that the model is trained on a substantial amount of data while also being tested on a separate set to evaluate its accuracy, precision, recall, and F-measure values.

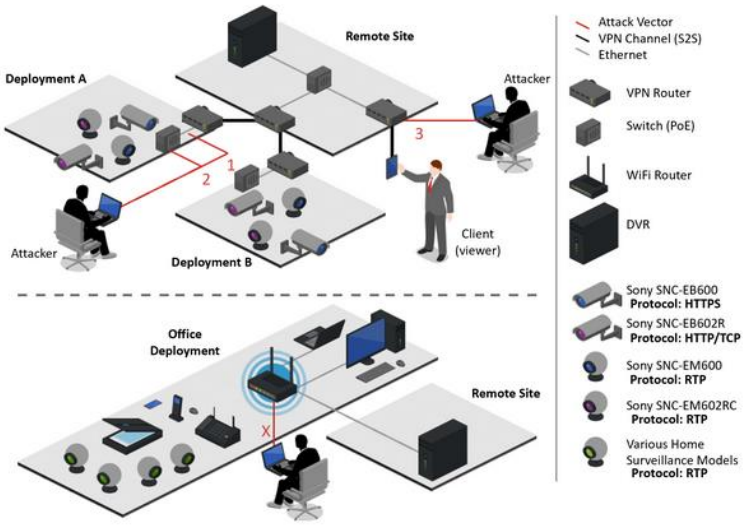

Figure 3. 1 Man-in-the-Middle attack dataset [100]

3.4 Dataset

The Dataset contains millions of network packets, providing a substantial amount of data. The datasets come pre-processed in CSV format, ready for ML applications, along with the corresponding label vector. The kitsune network attack dataset includes 115 features extracted using the afterimage feature extractor. These features provide a statistical snapshot of the network traffic in the context of the current packet traversing the network. The feature includes packet size, packet timing, source IP address, destination IP address, protocol, packet length, flags, flow duration, bytes transferred, inter-arrival time, entropy, count of unique ports, and many more derived from the Afterimage feature extractor.

The statistical overview provides insights into network performance by analysing metrics such as degree distribution, clustering coefficient, betweenness centrality, and average path length and is capable of efficiently processing millions of network traffic streams in real-time to monitor and secure the enterprise network against potential threats. The dataset had an imbalance and to address this issue, we dropped the last row to ensure data consistency and integrity. The kitsune network attack dataset can be effectively used for anomaly-based intrusion detection, which is a critical aspect of network security. Anomaly-based detection systems function by establishing a baseline of normal network behaviour. They then monitor for deviations from this standard, which could signal potential security threats.

3.5 MitM Attack prevention techniques

This study focuses on various types of MitM attacks and how attackers utilise them to disrupt the normal operations of individual systems and businesses. MitM attacks can be launched on any encryption within the Open System Interconnection (OSI) layers, including the Application, Presentation, and Transport layers. Security protocols such as SSL / TLS, are applied to secure communication channels between two parties [98]. However, attackers often find ways to bypass or manipulate these security protocols, leading to intercepting what is perceived as “secure.”

Specifically, attackers may utilise techniques like hypertext transfer protocol secure (HTTPS) spoofing, where they create a fake HTTPS website by spoofing the address of a legitimate website. Subsequently, they send a link for this fake website to victim users, who unknowingly visit the fake site. Secondly, SSL hijacking is a type of MitM attack that occurs when an attacker hijacks a user’s legitimate session, impersonating the user. In this scenario, the server remains unaware that the individual conducting the transaction is not the intended user, and the stripping is where the attacker downgrades the security of a website’s connection, allowing them to access the communications between a client and the connected server.

In this scenario, the user continues communication, believing it to be secure, while the server also perceives a secure connection with a legitimate user. Furthermore, the attacker gains access to a secure connection with the server and an open connection with the user. This allows the attacker to intercept everything exchanged between the user and the server. There is also IP, ARP, and DNS spoofing and many more attacks

of MitM. Figure 3.2 illustrates a connection between a user and the server, where an attacker intercepts their connection, fooling both the user and server to think their connection is real.

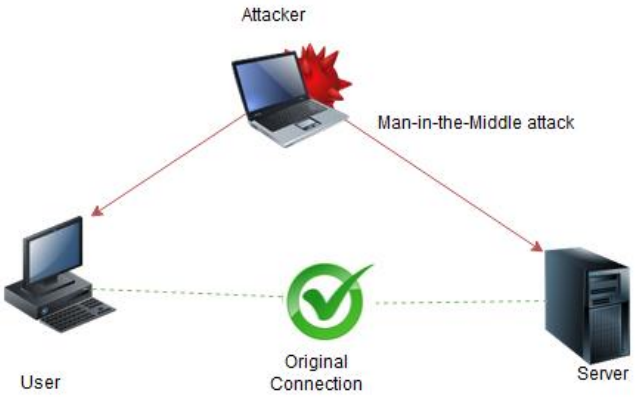

Figure 3. 2 Man-in-the-Middle attack interception

- ✓ The challenges faced with MitM attacks are that it is difficult to detect due to the attacker intercepting and modifying real-time traffic, making it hard to identify anomalies. Even data sent as encryption makes it difficult to detect the presence of anomalies. MitM attacks are also dynamic due to attackers changing their tactics, complicating the development of effective countermeasures.
- ✓ Manipulating DNS resolution redirects traffic to malicious servers or intercepts and modifies communication. Techniques include DNS cache poisoning, DNS tunneling, and DNS hijacking.
- ✓ Exploiting vulnerabilities in edge network infrastructure for unauthorised access, data exfiltration, or attacks against other targets. Techniques include port scanning, edge device exploitation, routers, cameras, or IoT devices.

Figure 3.3 illustrates the types of MitM attacks investigated in this study, how the attacks take place, the impact these attacks have on individual users and organisations, and possible mitigation strategies. Most companies currently work remotely hence it is crucial to design an IDS that ensures security for individual users and organisations around the world.

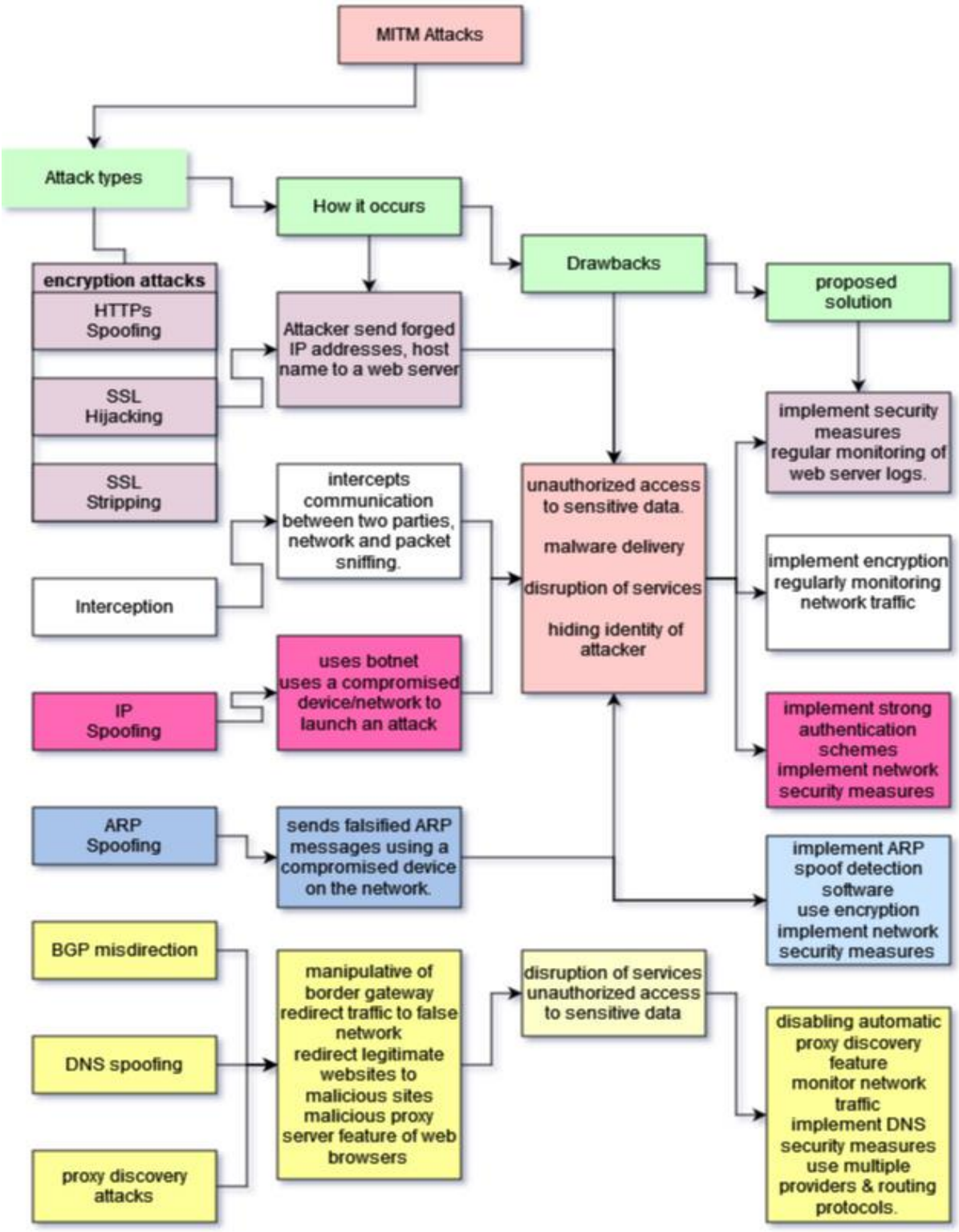

Figure 3.3 MitM attacks and how they are addressed

3.6 Network Architecture

Figure 3.4 illustrates the network features of MEC servers distributed at the network edge, connected to a centralised DNS server for host-name resolution and MEC servers for data processing and storage, enhancing edge computing capabilities while maintaining centralised control and resource management. This architecture optimises latency-sensitive activities by offloading computation to MEC servers while leveraging centralised resources for scalability and efficient management of network resources.

In this network setup, each subnet is equipped with an IDS sensor. The IDS sensor monitors network traffic and detects any suspicious or malicious activity within its assigned subnet. All IDS sensors are connected to a central management server and overseen by an administrator. A potential attack scenario is illustrated in Figure 3.4, where an attacker inserts themselves between the MEC server and a user, disrupting the original connection and assuming a false identity in the process. Additionally, another attacker targets the DNS service. This attack highlights the importance of IDS sensors in detecting unauthorised access attempts and malicious activities within the network, thereby aiding in the timely identification and mitigation of security threats.

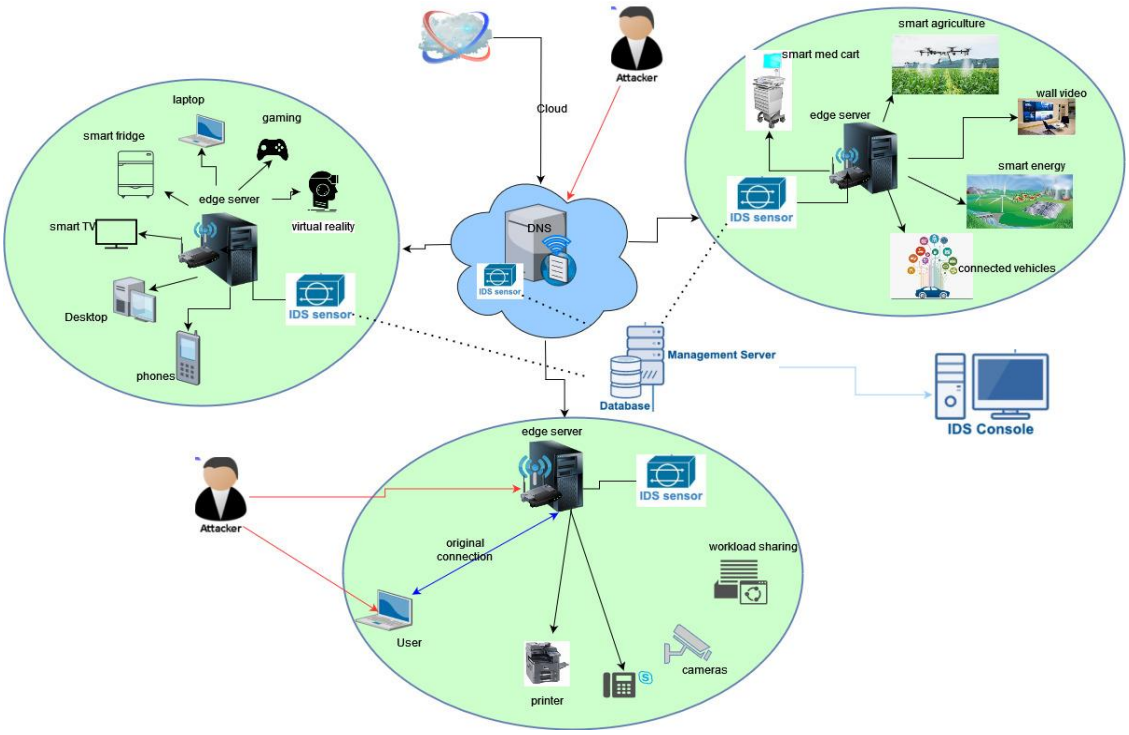

Figure 3. 4 Mobile Edge Network with a Domain Name System

3.7 Detecting the presence of an attack

Figure 3.5 depicts values and network parameters that assist in detecting whether network traffic is legitimate or there are anomalies within the network.

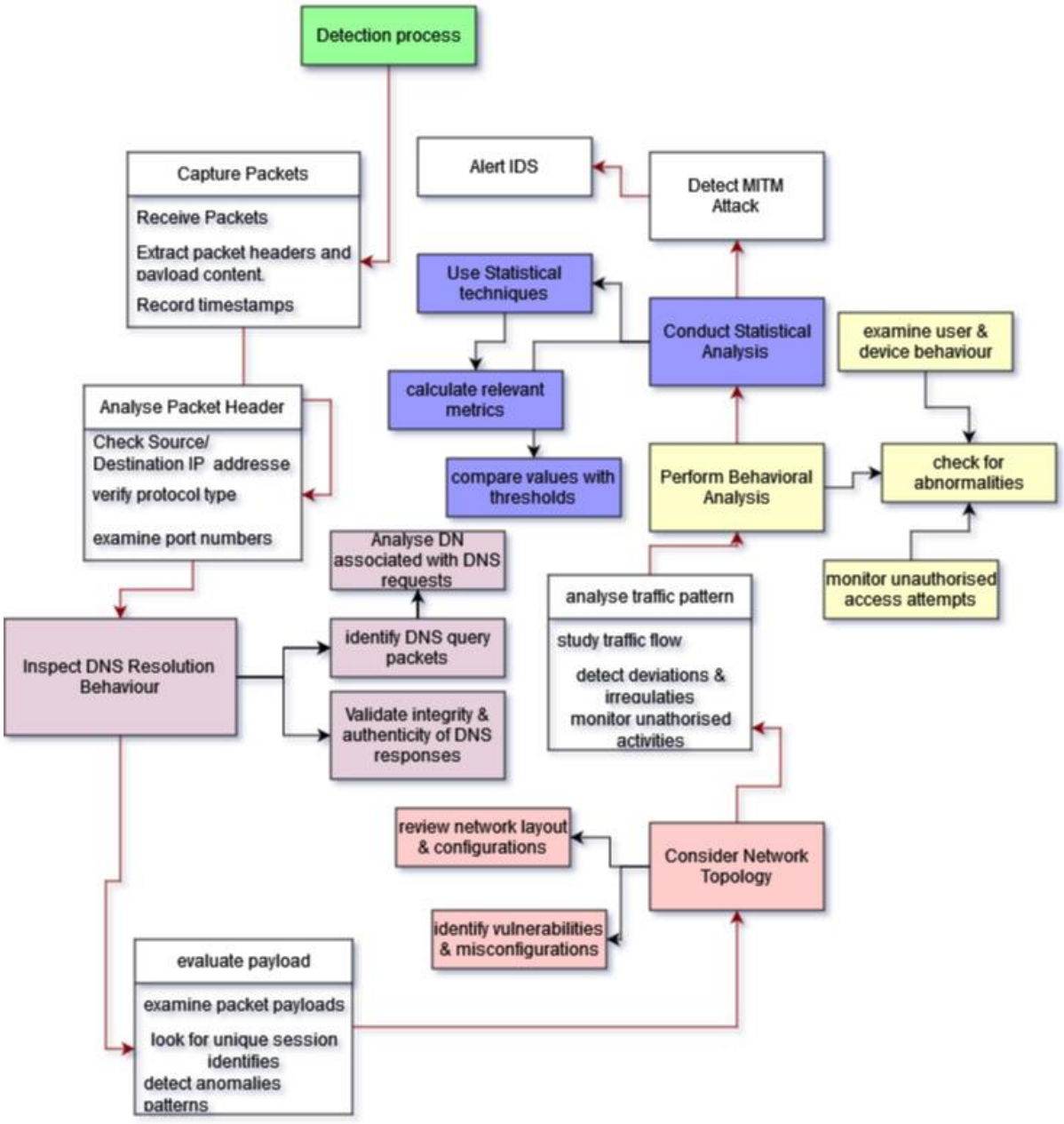

Figure 3. 5 Network Parameters detecting the presence of an attack

3.8 Intrusion Detection Scheme

IDS is crucial and utilised in application services to secure networks, but they have several challenges that should be considered when designing an effective scheme because of the high volume of alerts. The system sometimes generates unnecessary alerts many of which are false positives and negatives. Inspecting every alert can be

time-consuming and resource-intensive, allowing serious attacks to go undetected. Most of the IDS in literature have not examined insider threats but focus on detecting external because identifying and mitigating threats originating from within the organisation can be difficult as these involve legitimate users exploiting their access. The complexity of the attacks, encrypted traffic, performance and scalability, zero-day attacks and evasion techniques are also key areas on interest. It is indeed very challenging to deal with these challenges due to the evolution of technology and adaptive attacks [101].

3.9 Cuckoo Search Algorithm

The CS algorithm, developed by Yang and Deb in 2009, is a robust metaheuristics optimisation technique based on the brood parasitism behaviour observed in cuckoo species. CSA has been widely utilised by researchers for a variety of applications across fields such as function optimisation, engineering optimisation, machine learning, feature selection, image processing, and forecasting [102]. The uniqueness of the CSA makes it applicable in many applications where optimisation is key concern. It can find local minima and global solutions for complex dimensional problems. Furthermore, the algorithm has been adapted and contribute significantly to this study by improving the detection rate and minimising the false positives and negatives.

3.9.1 CSA Principles

The CSA is based on three main principles that can be applied to optimise detecting the MitM attacks in this study:

- ✓ Levy flights – It uses levy flights for global optimisation, a random walk where the step sizes having a heavy-tailed probability distribution. In the context of MitM attack detection, this could help in exploring vast search space of possible solutions efficiently, leading to the discovery of optimal or near-optimal sets of rules and patterns for identifying such attacks.
- ✓ Egg laying – cuckoo lay their eggs in the nests of other birds. If the host bird discovers the cuckoo's eggs, it either throws them away or abandons its nest. The CSA uses this principle to decide the radius or the distance within which new solutions (eggs) are laid around a current solution (nest). For detecting MitM attacks, this means that once a potential detection strategy (solution) is

identified, the CSA can explore nearby strategies to refine and improve detection capabilities.

- ✓ Discovery and abandoning worse nests – a fraction of worse nests are abandoned at each generation, and new nests are built randomly. When applied to MitM attack detection, this principle ensures that ineffective detection strategies are discarded, and new strategies are continuously explored. This helps in adapting to evolving attack patterns and maintaining the effectiveness of the detection system over time.

3.9.2 Initialisation

A network dataset is downloaded with features from the network traffic data, the thresholds for anomaly detection, step size α, discovering probability $P\alpha$and the number of iterations used as an end criterion.

Levy flight is a type of random walk, characterised by step lengths that follow a heavy-tailed probability distribution and the mathematical representation for the step lengths 's' can be drawn from a levy distribution:

$$s \frac{\lambda\gamma(\lambda)sin(\pi\lambda/2)}{2} \frac{1}{u^{1+\lambda}}, (u > 0) \quad (1)$$

where the step length of the length 's' for a Levy flight is used in the CSA to generate new solutions. In the CSA, the step length 's' of a Levy flight is crucial for generating new solutions, as it dictates the distance of the next move within the search space. The step length is influenced by the constant 'λ', typically between 1 and 3, which affects the algorithm's exploratory capabilities, and is scaled by a random number 'u', drawn from a uniform distribution, in conjunction with the gamma function '$\gamma(\lambda)$' and the sine function '$sin(\pi\lambda/2)$', to fit the Levy distribution.

Levy flight integrated with CSA: each iteration, a solution ($x_i(t)$) is updated to a new solution (x_i^{t+1}}) based on the Levy flight process:

$$x_i^{t+1} = x_i^t + \alpha \oplus Le'vy(\lambda) \quad (2)$$

where $\alpha > 0$ is the development measure that must be aligned with the sizes of the problem of interest, often, this can use $\alpha = 1$.

The Levy flight provides a random walk, with the irregular step length drawn from a distribution denoted by equation 2, where λ denotes:

$$Levyu = t^{-}\lambda \quad \text{where } \lambda \in (0,3) \tag{3}$$

CSA is the optimisation technique that mimics the parasitic behaviour of cuckoos to find optimal solutions, using random walks and levy flights for efficient exploration and exploitation of the search space. The max_generations indicate the possible max number of solutions that might be considered.

Table 3.1 Cuckoo Search Algorithm

1.	$Begin$
2.	$Define\ the\ objective\ function\ f(x), x\ being\ the\ solution\ vector.$
3.	$Initialise\ a\ population\ of\ n\ host\ nests(solutions).$
4.	$While\left(t < max_{generations}\right)$
5.	$For\ each\ nest\ i \in the\ population$
6.	$Generate\ a\ new\ solutions_i\ using\ Levy\ flight\ distribution.$
7.	$Evaluate\ its\ quality\ f(s_i).$
8.	$If\left(f(s_i) < f(i)\right), replace\ solution\ i\ with\ the\ new\ solutions_i.$
9.	$End\ For$
10.	$Find\ the\ worst\ nest\ with\ the\ highest\ f(x), remove\ it\ with\ probability\ P_a.$
11.	$Replace\ it\ with\ a\ new\ randomly\ generated\ solution.$
12.	$Apply\ a\ detection\ scheme\ identify\ non-improving\ solutions.$
13.	$If\ a\ non-improving\ solution\ is\ detected, replace\ it\ with\ a\ new\ solution.$
14.	$Rank\ the\ solutions \wedge find\ the\ current\ best.$
15.	$If\ termination\ criteria\ a\ R\ met, exit\ loop.$
16.	$End\ While$
17.	$Out\ put\ the\ best\ solution\ found.$
18.	End

3.10 Proposed Ensemble Cuckoo Scheme

The primary objective of our detection scheme is to thoroughly monitor network traffic and user activities. We are dedicated to minimising false alarms and enhancing the detection rate, particularly for zero-day attacks. To achieve this, we propose a multifaceted IDS that incorporates AI-powered threat intelligence. We employ a statistical tool to thoroughly analyse the training data to understand its characteristics and quality. This process ensures that the data is suitable for building accurate and reliable models. We establish a baseline for the statistical tool, we define a set of normal behaviour metrics against which network anomalies are detected. Additionally,

we will utilise the CSA to enhance the IDS's performance. The system is further trained using the RFA, which is renowned for its effectiveness in classification tasks. The RFA leverages ensemble learning by combining multiple decision trees through bagging, which minimises overfitting and enhances robustness, making it highly effective in classification tasks and handling missing values.

Figure 3.6 illustrates the deployment of the proposed IDS within a network. The scheme operates by receiving or collecting datasets and real-time data from various network nodes. For internal threats, the focus is on behavioural analysis and anomaly detection, which involves scrutinising for malicious activities or deviations from normal patterns. The scheme is trained to learn behaviours associated with insider attacks and APTs. A key feature of this system is the incorporation of real-time processing. This ensures immediate feedback from the RF algorithm, which in turn provides a feedback loop. This loop is crucial for updating the scheme's knowledge base regarding missed attacks, thereby continuously refining the detection capabilities.

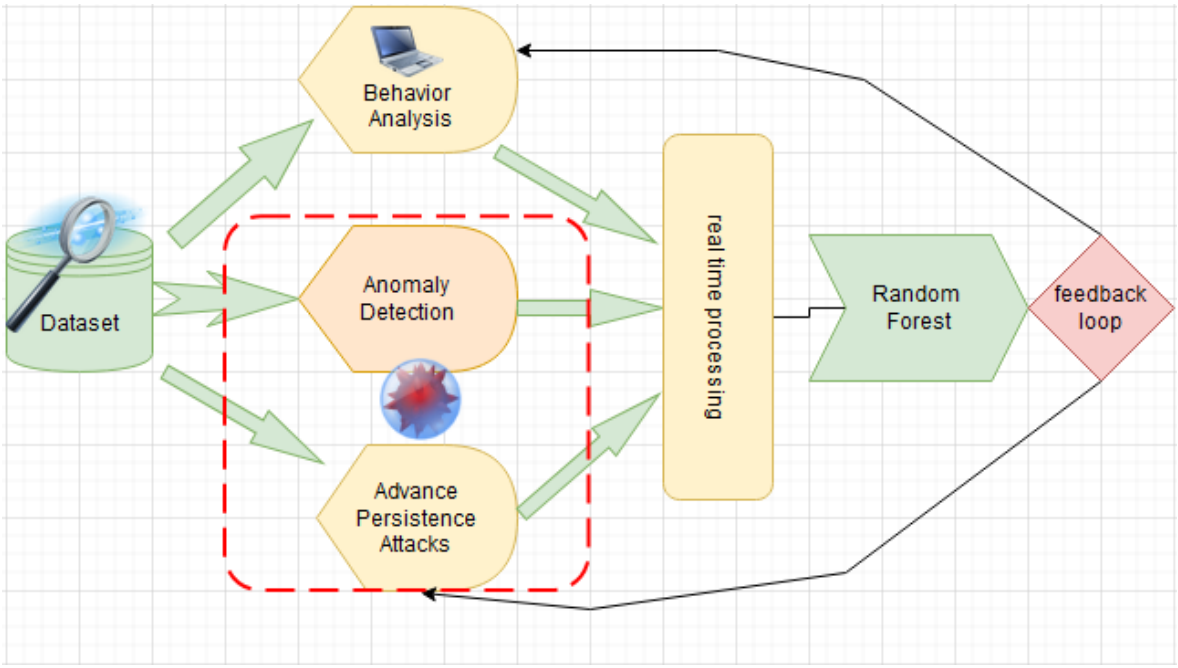

Figure 3. 6 Proposed Intrusion Detection System

The proposed IDS scheme is designed to autonomously learn and define 'normal' behaviour through behavioural analysis using statistical tools. Anomaly detection tools identify deviations, and specialised tools address APTs. The outputs from these analyses feed onto an ensemble learning model, enabling the scheme to self-learn and adapt. The use of ensemble modelling enhances the detection rate and

robustness, reducing prediction variance and error susceptibility. This approach aims to lower false positives and effectively manage various threats, providing a flexible and reliable IDS. The ensemble model's collective intelligence ensures a more efficient and accurate detection system.

The algorithm in Table 3.2 determines what our scheme can do and how, we shall have an input as the dataset, further classified based on inside attacks anomaly-based APTs. We also include a real-time processing from which the RF shall receive input to train, test and cross-validate the proposed Ensemble Cuckoo scheme and provide feedback to its categories to ensure they stay up to date. Then RF will trigger and alarm for suspect activities.

Table 3.1 Determining steps taken by algorithm

1.	$Begin$
2.	$Input: Dataset$
3.	$Output: improved\ detection\ rate, minimised\ f\ else\ positives$
4.	$collect_{data}(\ ): /simulate\ data\ collection\ \ network\ traffic \wedge logs$
5.	$Data = [\ \ ]$
6.	$Return\ data$
7.	
8.	$preprocess_{data}(data)$:
9.	$Features = [(d['packet\_size'], d['duration']),\ for\ d\ in\ data$
10.	$Scaler = standardScaler(\ \ )$
11.	$Normalised_{features} = scaler.fit_{transform}(features)$
12.	$Return\ normalised_{features}$
13.	
14.	$train\ \ (data)$:
15.	$Model = Random\ Forest(randomstate = n)$
16.	$Model.fit(data)$
17.	$Return\ model$
18.	
19.	$detect_{anomalies}(model, data)$:
20.	$Anomalies = model.predict(data)$
21.	$Return\ anomalies$
22.	

23. $generate_{alerts}(anomalies)$:
24. $Alerts =$
25. $return\ alerts$
26.
27. $Raw_{data} = collect_{data}(\ \ )$
28. $Preprocessed_{data} = preprocess_{data}(raw_{data})$
29. $Anomaly_{model} = train\ \ (preprocessed_{data})$
30.
31. $New_{data} = collect_{data}(\ \ )$
32. $New\ \ = preprocess_{data}(new_{data})$
33. $Anomalies = detect_{anomalies}(anomaly_{model}, new\ \ )$
34.
35. $Alerts = generate_{alerts}(anomalies)$
36. $For\ alert \in alerts$:
37. $Print(alert)$

3.11 Chapter Summary

In this chapter, we focused on tools that assist in carrying out our objective by discussing the research design, tools used for data collection, the MitM attacks and their significance in this study. The chapter also dwelt on network architecture, and how attacks are detected. We discussed the IDS and its challenges identified from related works. We outlined the proposed Ensemble Cuckoo scheme architecture and its significance to our study and finally presented our proposed algorithm. In Chapter 4, we specifically focus on the simulation parameters, simulation environment, and experimental results and compare our results with existing algorithms from related works.

Chapter 4

Data and Results Analysis

4.1 Introduction

In the Chapter 3, we explored the methodologies used to generate the comprehensive dataset that forms the foundation for this study. Chapter 4 focuses on elaborating and analysing this dataset, unfolding patterns and insights that directly address our core research questions. Leveraging the capabilities of the ensemble model and the CSA, presented in Chapter 3, we aim to achieve a detection scheme that not only identifies threats but also adapts to the emerging sophisticated strategies utilised by attackers.

The edge network reflects unique and complex characteristics, but it lacks standardised security measures that make it susceptible to exploitation by attackers. However, it benefits users with high-speed data processing and localised decision-making capabilities. Through a series of controlled experiments, we explore the effectiveness of the proposed detection scheme against a range of intrusion scenarios. The results provide valuable insights into the robustness of edge networks and contribute to the broader discourse on DNS security. As we navigate through the complexities of these experiments, we lay the groundwork for a more secure and robust internet infrastructure, capable of withstanding the attacks of modern cyber threats.

4.2 Network Topology

The network is composed of two subnets, each designed for specific functions and equipped with a dedicated set of nodes. Every subnet is paired with a MEC server that supplies essential computational resources, processing power and storage —to its respective nodes. Within the subnets, the nodes form a local network enabling direct inter-node communication. Moreover, each node is linked to its associated MEC server and granting access to the centralised computing services of the MEC infrastructure.

The MEC servers, one per subnet, are interconnected with a central DNS server. This server acts as the central entity for DNS, resolving domain names to Internet Protocol (IP) addresses for all nodes across both subnets. The network's structural design, including the connections among nodes and servers, is depicted in Figure 4.1.

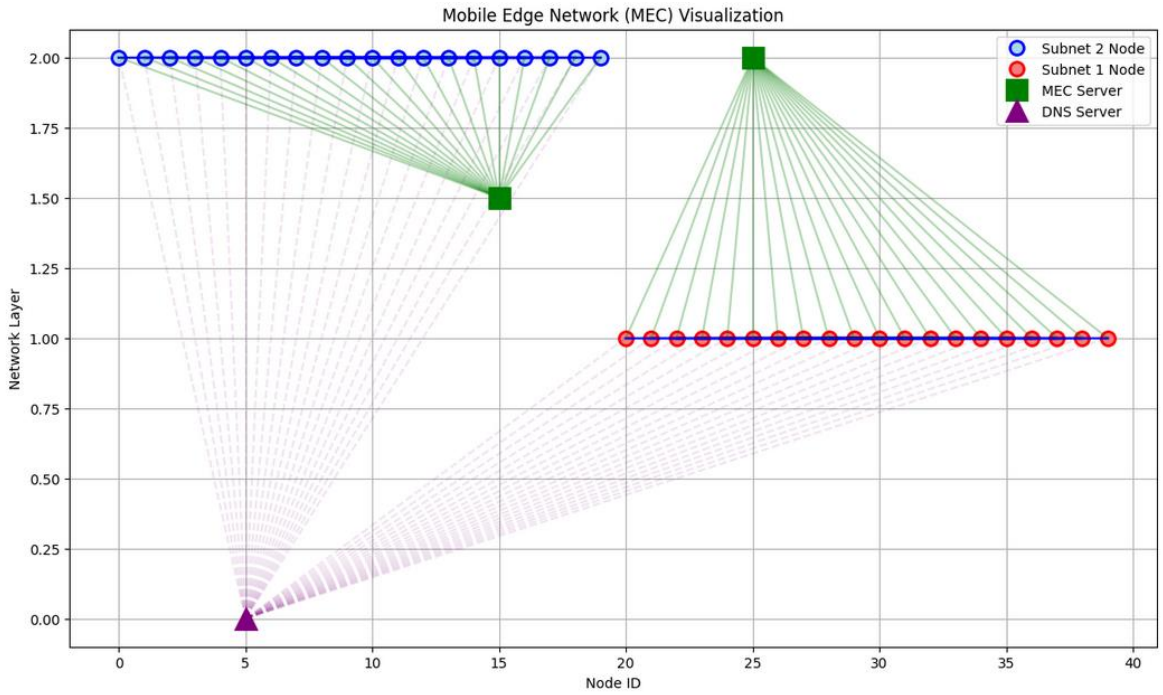

Figure 4. 1 Interconnection of Nodes, MEC Servers, and DNS Server

The network topology illustrated in Figure 4.2 shows a randomly selected subnet and how the nodes and attacker nodes are situated within the subnet. This is the topology of 100 mobile nodes, where each node has the capability to transmit data to nodes within the same subnet.

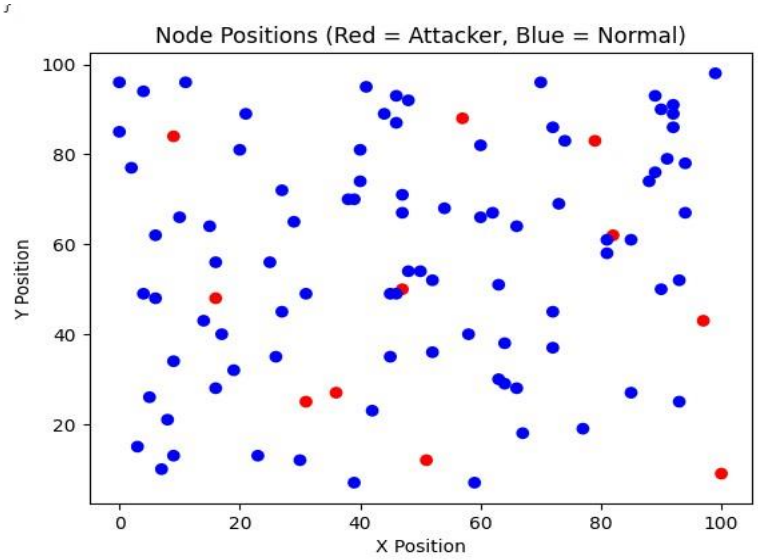

Figure 4. 2 Normal nodes in blue versus attacker nodes in red colour

4.3 Simulation Parameters

In this study we use two publicly available MitM attack datasets, one downloaded from Kaggle and the other dataset generated from a simulated environment. We have implemented the Proposed Ensemble Cuckoo Scheme in our simulation where it is expected to analyse all the incoming traffic from the devices within the network. The attack node periodically attacks any randomly chosen nodes from any subnet.

Table 4.1 outlines the simulation parameters, where the simulation time depends on the number of iterations defined on the network. The nodes in the network are mobile and we use Transmission Control Protocol (TCP) for reliability purposes and evaluate the proposed Ensemble Cuckoo scheme with metrics such as Accuracy, Precision, Recall, and F1-score. We later model the interactions between the defender and the attacker by using the Stackelberg Game Theory.

Table 4.1 The Simulation Parameters.

Simulation Parameters	Simulation Values
Simulation Time	Depends on the number of iterations
Time Step	Updates after every iteration
Evaluation Criteria	Detection accuracy, false positives / negatives
Capturing data	Wireshark
Network Topology	Mesh topology
Node Placement	Random
Node Types	Normal nodes, attack nodes
Node, Behaviour Models	Mobility,
Communication Protocols	Transport Control Protocol (TCP)
Routing Algorithms	Random Forest / Cuckoo Search Algorithm
Attack Intensify	MitM (ARP spoofing) attacks
Performance Metrics	Precision, Recall, F1-score,

Figure 4.3 illustrates the distribution of network traffic after simulating for 15 minutes across 10 iterations. It displays the proportion of normal traffic, attack traffic captured, and the number of packets dropped due to the attack. Specifically, it reveals that 52.1% of the data was utilised for training our scheme, 34.0% of the traffic was identified as attacks, and 13,9% of the packets were dropped. The training instances represent the portion of data labeled as normal traffic to help the model understand typical network behavior. The attack traffic percentage shows the instances used to

train the detection scheme to recognise malicious activities. Lastly, the dropped packets indicate the network disruption caused by the attack, highlighting its impact on performance.

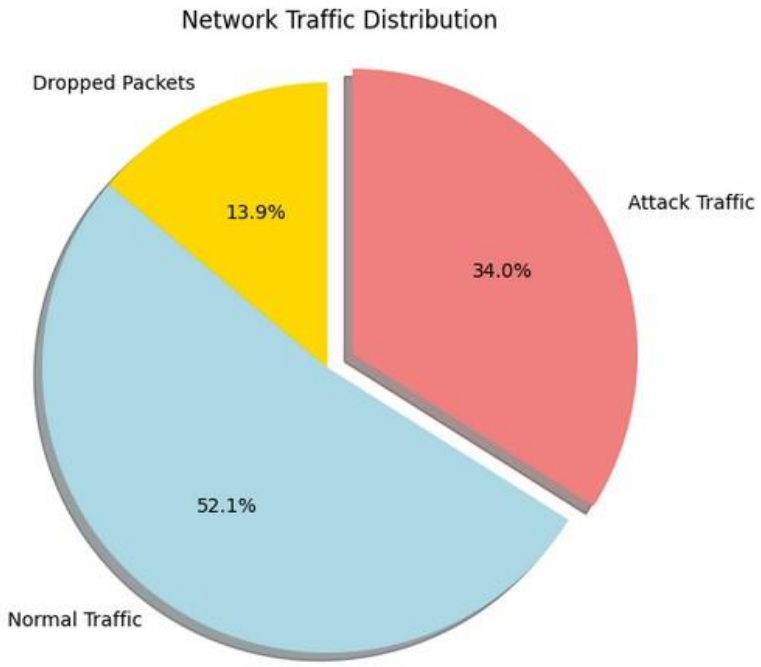

Figure 4. 3 Distribution of the network traffic

Figure 4.4 shows the CS class defined where we have the objective function to be optimised. The function evaluates how good a solution is, the grid of parameters to explore, defines the search space for the algorithm and the number of particles, which is the cuckoo eggs or nests used in the search. This determines the population size in each iteration.

```
# Define the Cuckoo Search class
class CuckooSearch:
    def __init__(self, obj_function, param_grid, X_train, y_train, n_iter=100, n_particles=50):
        self.obj_function = obj_function
        self.param_grid = param_grid
        self.X_train = X_train
        self.y_train = y_train
        self.n_iter = n_iter
        self.n_particles = n_particles
        self.alpha = 0.1  # Step size for cuckoo search
```

Figure 4. 4 Cuckoo Search Class

The proposed detection scheme is compared with the RFA, Isolation Forest (IF), and SVM. RFA uses an ensemble of DTs, which helps in capturing complex patterns and

interactions in the data. The IF and SVM were specifically chosen because they are anomaly-based algorithms. The attack nodes use a sniffing function using Scapy's sniff. In pre-processing, the data used has no missing values. We have removed duplicates because they affect the reliability and the efficiency of data analysis, and we have encoded all the columns that are non-values since we use categorical data. Encoding helps to capture relationships between categorical variables and the target variable, improving the predictive power of the model. The dataset was split into 70% for training and 30% for testing.

The metrics used in this study to evaluate the performance of the proposed Ensemble Cuckoo scheme include Precision, which measures the number of true positives predicted by the algorithms, recall which measures the actual positive samples correctly predicted and other metrics shown in Table 4.2.

Table 4.2 Experimental results for the Algorithms

Scheme	**Precision**	**Recall**	**F1-score**	**Accuracy**
Ensemble Cuckoo	0.79	0.78	0.76	0.79
LOF	0.60	0.55	0.51	0.63
SVM	0.67	0.63	0.61	0.69
RF	0.72	0.75	0.75	0.77

Figure 4.5 shows the Precision and Recall simulation results between the Proposed Ensemble Cuckoo Scheme, LOF, SVM and RF schemes compared. For Precision, we observed on the graph that the Proposed Ensemble Cuckoo Scheme outperformed the other schemes due to the CS algorithm utilised. The algorithm is a nature-inspired meta-heuristic algorithm and has been effectively applied to optimised various MLAs. In this study, the algorithm was used to tune the hyper-parameters of random forests, for instance, the number of trees, maximum depth, and minimum samples split to improve their performance.

The algorithm significantly enhanced the ability of the Proposed Ensemble Cuckoo Scheme to identify relevant features and improve the detection capability of LOF, SVM and RFA. Figure 4.5 shows that the Proposed Ensemble Cuckoo Scheme detected 79% of normal traffic correctly. The Proposed Ensemble Cuckoo Scheme also performed better in terms of Precision than the other schemes because both the RF

and CS algorithms are scalable to large datasets emphasizing that the Proposed Ensemble Cuckoo scheme is capable of managing high-dimensional data and large volumes of data efficiently.

The LOF scheme reflects some weaknesses when applied in a real-time environment such as the sensitivity of parameters, experiences scalability challenges, and makes it less suitable for dynamic environments where data is continuously streaming in. On the other hand, using the SVM in a real-time environment is time-consuming, and resource intensive. This makes SVM experience limitations in updating the model in real-time as new data arrives. These weaknesses highlight the challenges of using LOF and SVM in real-time environments, where computational efficiency, scalability, and quick adaptability are crucial. The CSA improved the detection rate of the proposed Ensemble Cuckoo scheme.

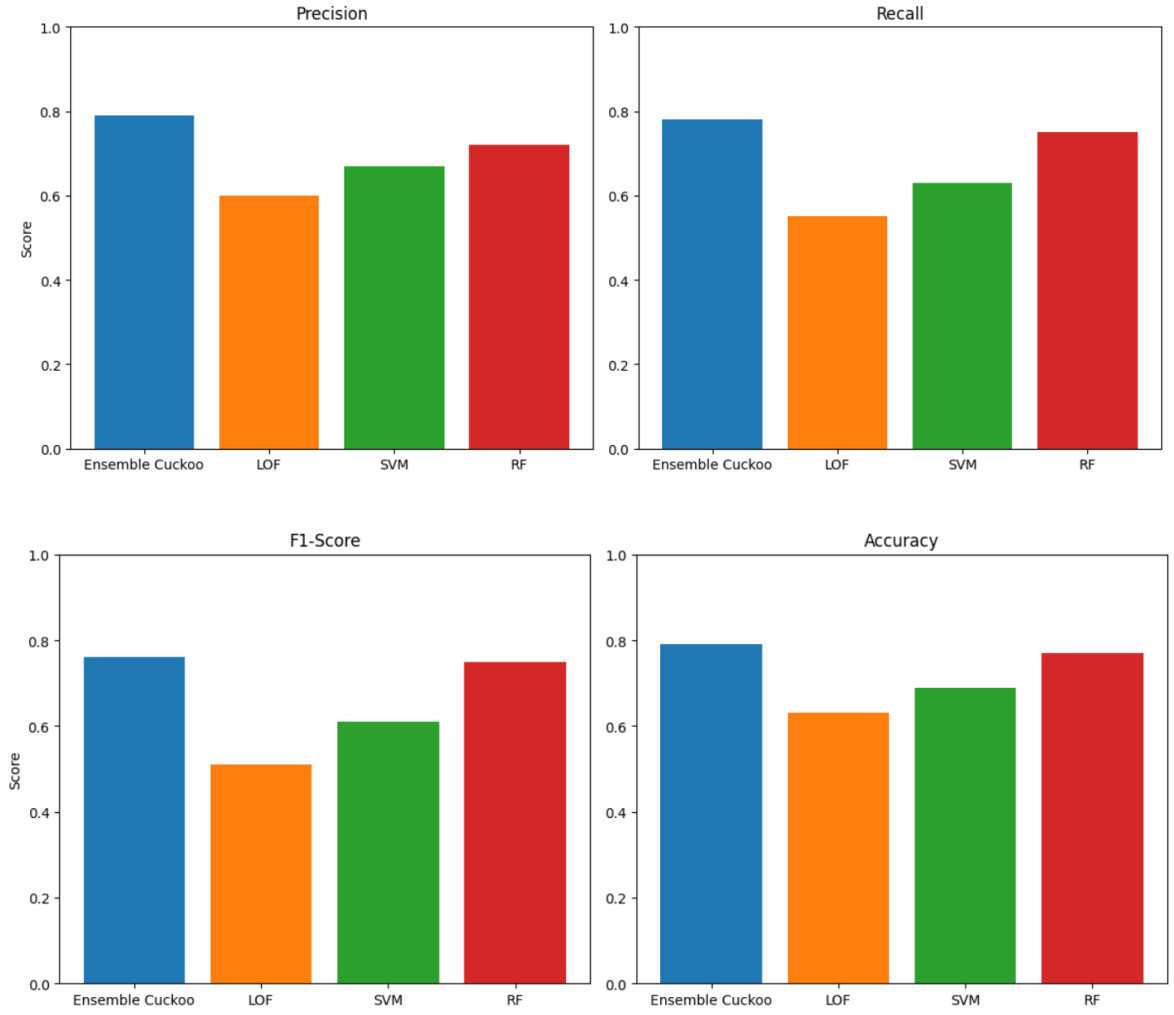

Figure 4. 5 Precision, recall, F1-score and accuracy metrics, respectively

Figure 4.5 further shows the F1-score and Accuracy results. From Figure 4.5, we see that the proposed Ensemble Cuckoo scheme outperformed the LOF, SVM, and RF schemes. Accuracy is the ratio of correct predictions to the total number of predictions; it provides insights into the performance of the algorithms and the state of the false positives. LOF model reflects several challenges that exert an impact on its performance, such as storing pairwise distances between all data points, which can consume significant memory due to the amount of data employed in this study. In this study, we utilised a large dataset with up to 115 features; hence, training SVM became computationally expensive.

The proposed Ensemble Cuckoo scheme performs better because integrating various models minimises the variance of the predictions, making the proposed Ensemble Cuckoo scheme more robust to noise and outliers. The Proposed Ensemble Cuckoo Scheme used CSA, which assisted in stabilising the training process, making the Proposed Ensemble Cuckoo Scheme less sensitive to small changes in the data, which is a positive attribute. This means that the proposed Ensemble Cuckoo scheme assisted by CSA is more stable and robust thus, the model's performance is not easily affected by minor variations, leading to more reliable and consistent outcomes.

4.4 Dataset

In this section, the Kitsune Network dataset is utilised to train, test, and cross-validate the Proposed Ensemble Cuckoo Scheme together with the other anomaly-based models. The dataset was downloaded from the Kaggle platform. The same dataset has been utilised in several studies for classification purposes and anomaly detection [16][39]. We pre-processed the dataset and ensured there were no missing values and NAN values. Figure 4.5 illustrates all the devices used in the network for data capture, as well as the associated attack vector.

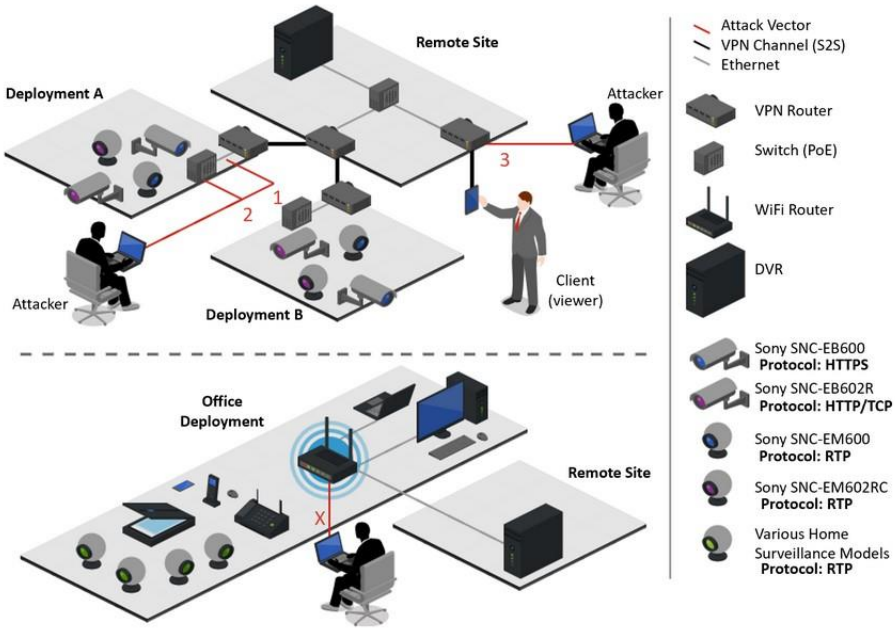

Figure 4. 6 Dataset [97]

The dataset comprises 115 features, and the instance quantity within the dataset is 2 504 255 million instances. The structure of the dataset is shown in Figure 4.7, showing the number of normal packets and malicious packets.

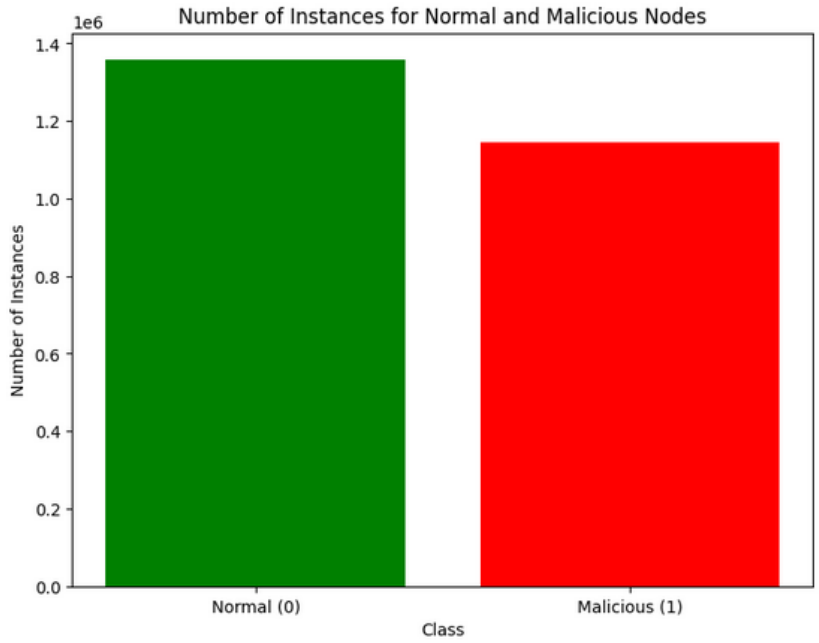

Figure 4. 7 Dataset classification

The experiments were initially done without regarding feature selection to establish a baseline performance and Table 4.3 records the simulation results of the proposed Ensemble Cuckoo scheme, LOF, SVM, and RF algorithms. From Table 4.3 we

observe that the proposed Ensemble Cuckoo scheme outperformed the LOF, SVM, and RF schemes due to its flexibility and robust nature because of the Cuckoo Search algorithm.

Table 4.3 Kitsune Dataset results

Scheme	Precision	Recall	F1-Score	Accuracy
Ensemble Cuckoo	0.979	0.980	0.980	0.981
LOF	0.866	0.891	0.892	0.892
SVM	0.907	0.891	0.905	0.919
RF	0.971	0.970	0.969	0.977

In Figure 4.8, the evaluation results show that the proposed Ensemble Cuckoo scheme and the RF outperformed the other schemes across all metrics- Precision, Recall, F1-Score, and Accuracy- demonstrating their reliability and effectiveness. Both schemes achieve high scores, indicating a strong balance between detecting true positives and minimising false positives. Whereas the LOF method performs the poorest across all metrics, it is less suitable for the given application or dataset. LOF can sometimes identify entire clusters as outliers if they are significantly less dense than the rest of the data, even if individual points within those clusters are not outliers.

The SVM shows moderate performance, primarily utilised for anomaly detection but can be computationally expensive, especially with large datasets, resulting in slower training and prediction times. Operating in a dynamic environment might require SVM to be scalable, which is challenging because it struggles to handle large datasets due to their quadratic complexity in the number of samples. While SVM surpasses LOF, the latter does not perform well on any datasets where anomalies are not separated from normal data, resulting in higher false positive rates.

LOF requires computing the local density for each data point, which can be computationally intensive and slow for larger datasets. Additionally, it still falls short of the performance achieved by the proposed Ensemble Cuckoo scheme and RF, indicating that LOF may benefit from further optimisation. The analysis highlights the robustness of the proposed Ensemble Cuckoo scheme and RF while pointing to potential areas for improvement in LOF and SVM.

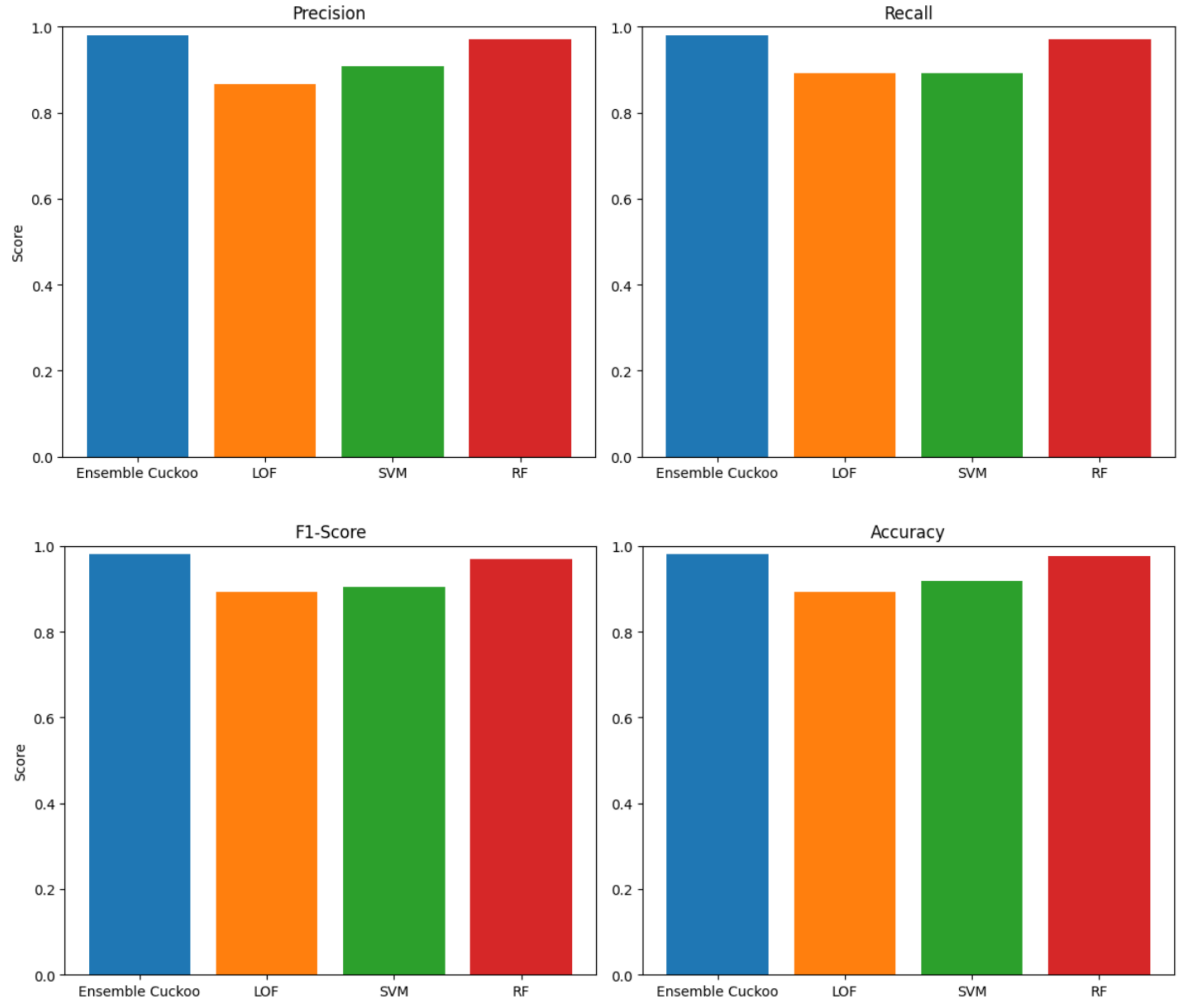

Figure 4. 8 Evaluation metrics of the proposed Ensemble Cuckoo scheme

The confusion matrix shows the instances that were correctly identified and the misclassified instances by the proposed Ensemble Cuckoo scheme. The matrix assists in evaluating the effectiveness of the model's performance. From Figure 4.9, true negatives are the instances where the model correctly predicted the malicious instances, recording 390 000 instances. It further shows the true positives, where the normal instances were correctly predicted as normal instances, which recorded 340 000 instances. The false negatives and positives are recorded as 4 800 and 17 000 respectively.

The confusion matrix indicates a high number of false positives and negatives, despite the overall better performance of the model. Consequently, the number of misclassified instances is quite significant, posing a new challenge faced by the Proposed Ensemble Cuckoo Scheme in differentiating between normal and malicious traffic over extended periods. The dataset used has 115 features and includes too

many features most of which may be irrelevant and can dilute the scheme's ability to focus on important features, thereby minimising classification accuracy.

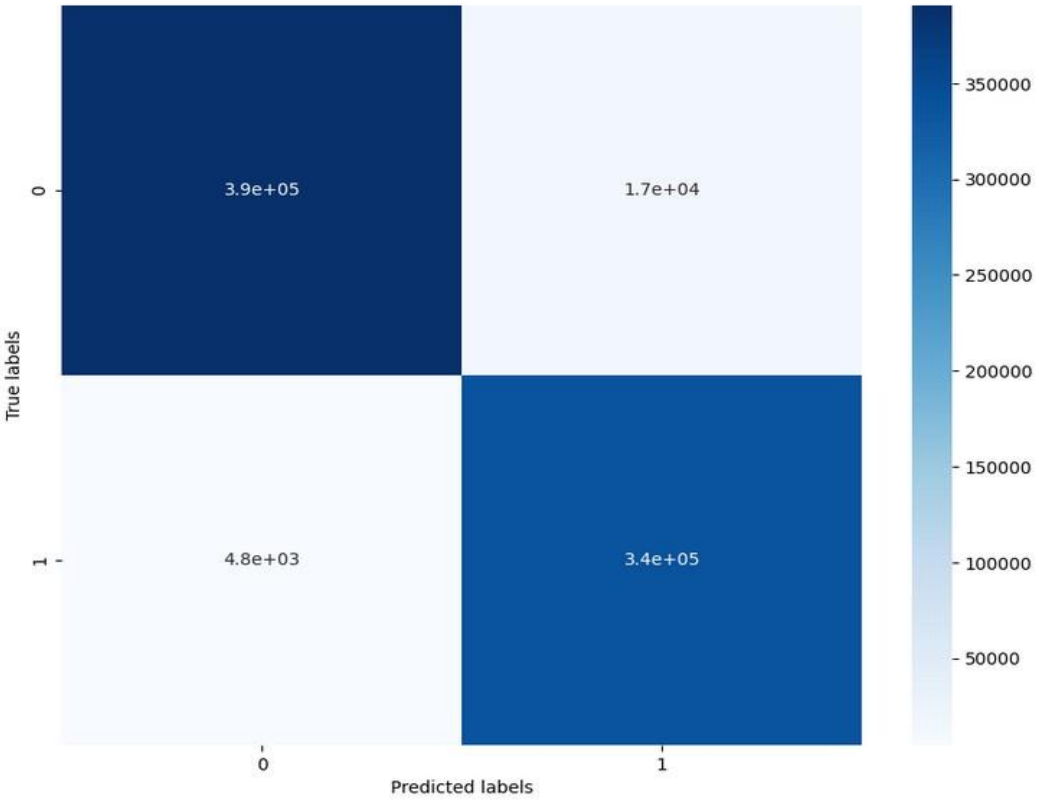

Figure 4. 9 Confusion matrix for the proposed Ensemble Cuckoo scheme

Too many features can cause over-fitting, leading to poor performance on new data and making it difficult to interpret the scheme's predictions. To address this possibility of over-fitting, the study employed feature selection techniques, such as PCA and RFE, to identify the most relevant features and mitigate the risk of overfitting.

Feature selection enhances metrics such as accuracy, along with precision, and recall. For this study, the number of iterations was set at 5 for the proposed Ensemble Cuckoo scheme. Table 4.4 presents the simulation results, clearly demonstrating that the proposed Ensemble Cuckoo scheme outperforms other comparable schemes.

Table 4.4: The performance metrics for the algorithms

Iteration	Random Forest			
	Precision	**Recall**	**F1-score**	**Accuracy**
1	0.995797	0.995711	0.995754	0.995660
2	0.994084	0.994247	0.994163	0.994178
3	0.994986	0.994929	0.994958	0.994972
4	0.995502	0.995501	0.995502	0.995501
5	0.995768	0.995740	0.995754	0.995766
Ensemble Cuckoo				
	Precision	**Recall**	**F1-Score**	**Accuracy**
1	0.998718	0.998718	0.998718	0.998720
2	0.997752	0.997742	0.997747	0.997752
3	0.998705	0.998732	0.998718	0.998720
4	0.998191	0.998161	0.998176	0.998180
5	0.998010	0.998003	0.998003	0.998010
Local Outlier Factor				
	Precision	**Recall**	**F1-Score**	**Accuracy**
1	0.992229	0.992326	0.992277	0.992293
2	0.993549	0.993639	0.993592	0.992229
3	0.993249	0.993295	0.993271	0.993278
4	0.992229	0.992326	0.992277	0.992293
5	0.992296	0.992490	0.992387	0.992399
Support Vector Machine				
	Precision	**Recall**	**F1-Score**	**Accuracy**
1	0.979280	0.978721	0.978947	0.978980
2	0.984140	0.984244	0.984191	0.984239
3	0.983927	0.983888	0.983907	0.983943
4	0.987214	0.987465	0.987334	0.983927
5	0.987214	0.987465	0.987334	0.987372

Figure 4.10 presents a line graph comparing the Precision metric across the proposed Ensemble Cuckoo scheme, LOF, RFA, and SVM. The proposed Ensemble Cuckoo scheme consistently outperforms the other schemes. Its high precision score indicates that the scheme effectively minimises misclassification, ensuring that malicious network is not incorrectly identified as normal packets. However, based on the confusion matrix, various instances were misclassified. This discrepancy suggests that while the scheme is generally effective, certain features may contribute to

misclassification. To address this, the study will further employ feature selection to select only the most important features, enhancing the scheme's ability to correctly classify network traffic.

The confusion matrix and the results may seem at variance because the high precision score reflects the accuracy of correctly identified positive instances (malicious traffic), but it does not account for the overall distribution of false positives and negatives. Therefore, by employing feature selection, we aim to improve the model's robustness and minimise the likelihood of misclassification, thereby aligning the confusion matrix results with the overall performance metrics. As shown in Figure 4.10, the SVM scheme demonstrates the lowest performance. This can be attributed to SVM's inefficiency with larger datasets, where the training time increases significantly due to the complexity of the dynamic environment.

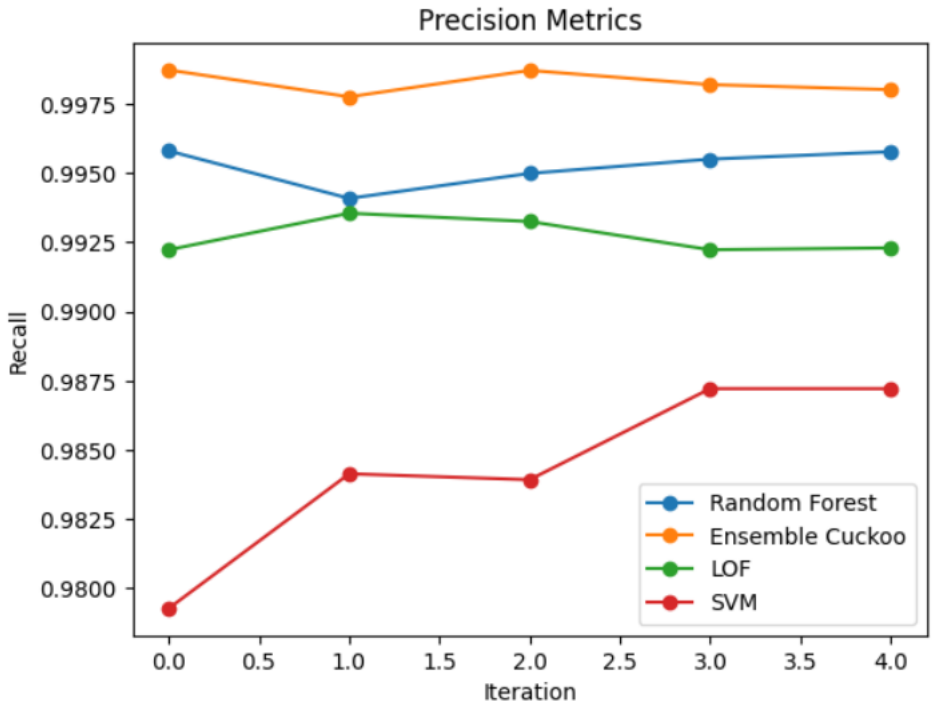

Figure 4. 10 The precision metrics among the four algorithms

Figure 4.11 illustrates the Recall metric results for the proposed Ensemble Cuckoo scheme, LOF, SVM and RFA. From Figure 4.11, we observe that the RF scheme fluctuates slightly around 0.9975, maintaining a consistently high recall value across all iterations. The recall for the proposed Ensemble Cuckoo scheme is consistently the highest among the other models, approximating 0.9980 throughout the iterations.

This indicates that the proposed Ensemble Cuckoo scheme can identify relevant instances compared to the other schemes. The LOF, represented by the green line, fluctuates around 0.925, showing relative stability over the iterations. Focusing on the SVM, we see that it starts around 0.9800 in the first iteration and gradually increases to approximately 0.9875 by the fifth iteration. Although the SVM scheme has the lowest performance, it shows a consistent upward trend, indicating improvement over time.

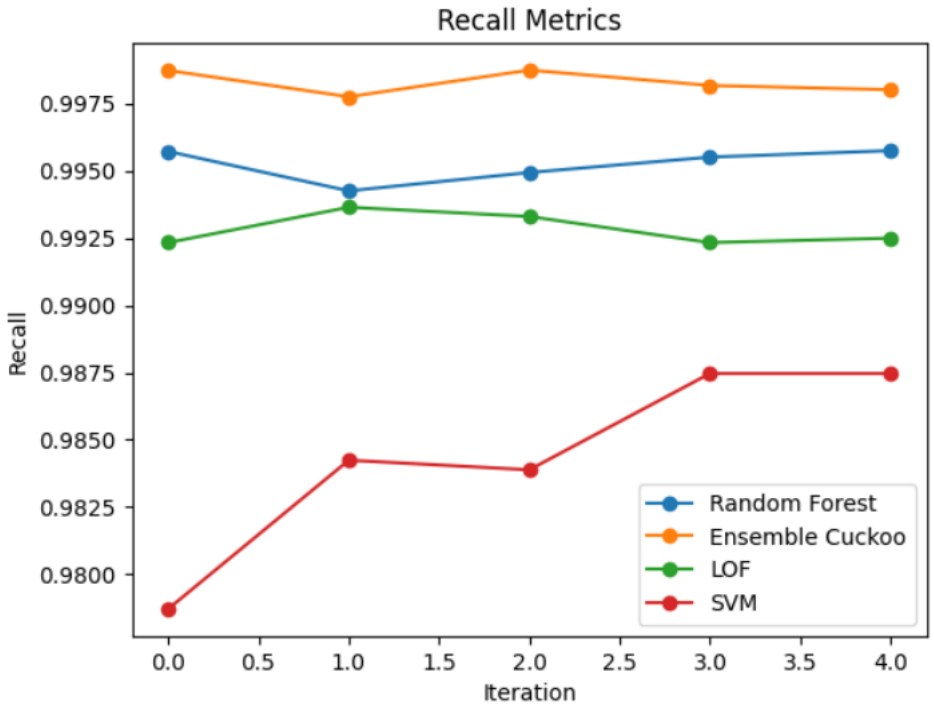

Figure 4. 11 The Recall metric performance of the Ensemble Cuckoo scheme

The proposed Ensemble Cuckoo scheme demonstrates the highest and most consistent performance, indicating a good balance between Precision and Recall. In comparison, both RFA and LOF show high F1-scores, with RF slightly outperforming the LOF model. Meanwhile, SVM initially shows the lowest F1-scores but improves slightly over the iterations, suggesting that it may benefit from further finetuning or additional iterations as shown in Figure 4.12.

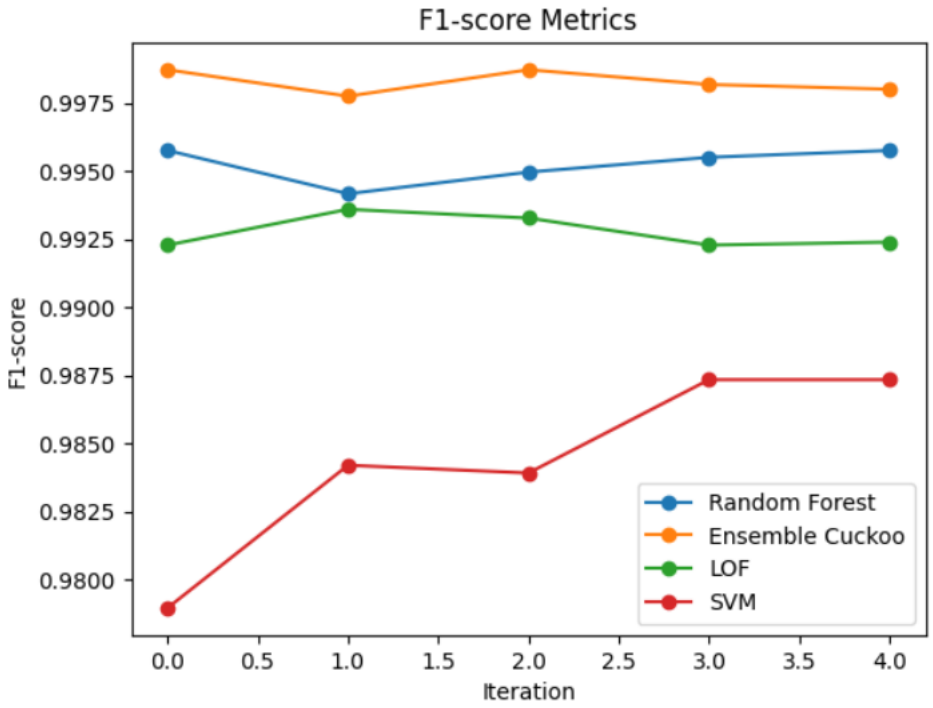

Figure 4. 12 The f1-score results of the models

Figure 4.13 illustrates the accuracy results of the four algorithms over multiple iterations. The proposed Ensemble Cuckoo scheme, represented by the orange line, consistently demonstrates the highest performance, maintaining an accuracy of approximately 0.9975. In contrast, the blue line shows minor fluctuations but generally maintains a high accuracy of around 0.9950.

As shown in Figure 4.13, all schemes perform well across all the iterations, with the proposed model outperforming the others.

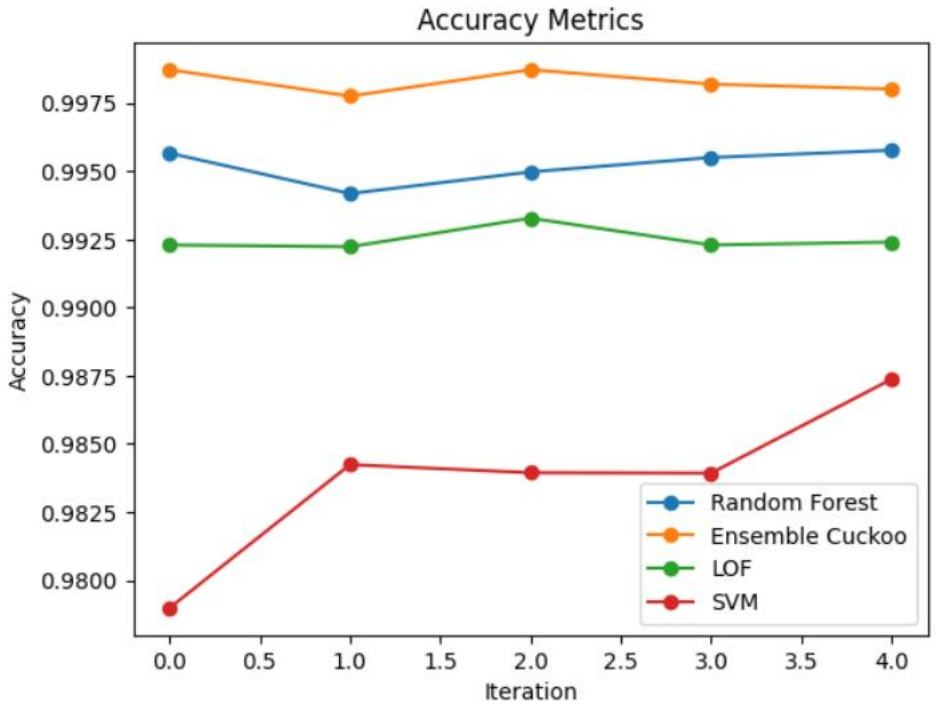

Figure 4. 13 Accuracy results of the Ensemble Cuckoo and other schemes

4.6 Results Discussion

4.6.1 Discussion of the Results

The simulation results indicate that the proposed Ensemble Cuckoo scheme, utilising the CSA, consistently outperforms the LOF, SVM, and RF algorithms across all key metrics—Precision, Recall, F1-Score, and Accuracy. This superior performance is attributed to the scheme's flexibility and robustness, which enable it to achieve high scores while effectively balancing true positive detection with minimising false positives. In comparison, while the RF algorithm also performs well, the LOF algorithm shows the poorest results due to its tendency to misclassify entire clusters as outliers. The SVM algorithm, although better than LOF, struggles with large datasets because of its computational complexity. The confusion matrix highlights a significant number of false positives and negatives, underscoring the challenge of distinguishing between normal and malicious traffic over time. To mitigate potential overfitting and enhance classification accuracy, feature selection techniques were employed, further improving the model's performance. Thus, the proposed Ensemble Cuckoo scheme demonstrates the highest and most consistent performance, proving its effectiveness and reliability in detecting MitM attacks in dynamic environments.

4.6.2 Interpretation of the Results

The simulation results show that the proposed Ensemble Cuckoo scheme, leveraging the CSA, consistently outperforms LOF, SVM, and RF algorithms across all metrics—Precision, Recall, F1-Score, and Accuracy—due to its flexibility and robustness. While RF also performs well, LOF's poor results stem from its tendency to misclassify entire clusters as outliers, and SVM struggles with large datasets due to its computational complexity. The confusion matrix reveals significant false positives and negatives, highlighting the challenge of differentiating between normal and malicious traffic over time. Despite not having access to high-performance computing, the tools used were adequate, and the data was sufficient and fit for use.

Feature selection techniques were employed to address potential over-fitting and improve classification accuracy, further enhancing the model's performance. Therefore, the proposed Ensemble Cuckoo scheme demonstrates the highest and most consistent performance, indicating its effectiveness and reliability in detecting MitM attacks in dynamic environments.

4.6.3 Results and Related Work

The results of the proposed Ensemble Cuckoo scheme, leveraging the CSA, show significant improvements over traditional algorithms like LOF, SVM, and RF in detecting MitM attacks. In comparison to previous works, such as those reviewed in the literature, the proposed Ensemble Cuckoo scheme's superior performance is evident. For instance, traditional methods like LOF often misclassify entire clusters as outliers, and SVM struggles with large datasets due to computational complexity. These limitations are consistent with results in other studies, where LOF and SVM have shown similar weaknesses [101]. Moreover, while RF has been noted for its robustness and good performance in various applications, it still falls short compared to the proposed Ensemble Cuckoo scheme, which achieves higher accuracy and a better balance between true positive detection and minimising false positives.

Previous literature also highlights the challenges of detecting MitM attacks due to their stealthy nature and the sophisticated methods employed by attackers. The proposed Ensemble Cuckoo scheme addresses these challenges effectively, even without high-performance computing resources, by employing adequate tools and sufficient data, along with feature selection techniques to mitigate over-fitting and enhance

classification accuracy. Thus, the proposed Ensemble Cuckoo scheme demonstrates the highest and most consistent performance, aligning with and surpassing the results found in previous studies, indicating its effectiveness and reliability in dynamic environments [102][103].

4.6.4 Research Implications

The research demonstrates that the proposed Ensemble Cuckoo scheme, leveraging CSA, is highly effective and reliable in detecting MitM attacks, outperforming traditional algorithms. This implies that the scheme can significantly enhance cybersecurity measures in dynamic environments. Additionally, the use of adequate tools and sufficient data, even without high-performance computing, highlights the practicality and accessibility of the proposed Ensemble Cuckoo scheme for real-world applications.

4.7 Chapter Summary

In this chapter, we focused on implementing the proposed Ensemble Cuckoo scheme and comparing it with other methods. The experimental simulations showed that the proposed model consistently outperformed the other methods, demonstrating its superior accuracy and robustness in various scenarios. This indicates that the proposed model is more effective at correctly identifying and classifying data, leading to more reliable and trustworthy results in practical applications. The detection scheme was trained using the ensemble models, optimised through CSA for hyper-parameter tuning. By combining multiple schemes, the ensemble achieved higher predictive accuracy than individual schemes. While the Ensemble Cuckoo was optimised, the process can be time-consuming, resulting in longer training times.

The LOF model has been observed to perform better than the SVM model, particularly due to its ability to handle the intensive computation required for large datasets. We conducted a comprehensive set of experiments to evaluate the performance of various ML models on our dataset. The Ensemble Cuckoo and RF exhibited stable performance across iterations, highlighting their reliability and consistency. The minimal fluctuations in their performance metrics suggest that the schemes are less sensitive to data variations and are more resistant to potential over-fitting.

Chapter 5

Adaptive Game Theoretic Model

5.1 Introduction

The continuous dynamic conflict between cyber attackers and defenders (detection system) has led to the development and generation of innovative intrusion detection methods. MitM attacks, represent a major threat to network security, they enable attackers to intercept and manipulate sensitive information. Traditional detection schemes often relied on signature-based approaches, which found it difficult to keep up with the evolving nature of MitM attacks. Recently, game theory has emerged as a promising framework for modelling the interactions between attackers and defenders, enabling proactive and adaptive defence mechanisms.

This study examines the application of game theory by developing a game-theoretical model that incorporates our detection scheme to identify and respond to anomalies in a dynamic system. Hence, by framing the problem as a game, we can analyse the strategic interactions between attackers and defenders, identifying optimal detection and response mechanisms. Our approach has the potential to enhance the robustness and resilience of network security, ensuring the integrity of sensitive information in the increasingly sophisticated attacks.

Game theory is a framework that provides useful mathematical concepts for modelling cybersecurity and identifying optimal defence strategies. In its origin, it analyses the strategic decisions made between two or more rational actors with potentially competing interests. Just like any other normal game, there must be players, and rules to comply with, objectives, mechanics, penalties and resources. The objective of cyber professionals is to defend and build; whereas cyber attackers confront and destroy, hence the motive is never the same. It can be concluded that this is a non-cooperating game because when new technologies emerge, both defender and attacker use them to further their agenda.

5.2 Background

In this study, we focus on the Stackelberg model, named after the German economist Heinrich Freiherr von Stackelberg, originating in the field of economics. The model was introduced then in 1934 in Stackelberg's book [100]. In this model, there are two players -a leader and a follower. They compete on quantity, with the leader moving

first and establishing a quantity that the follower then responds to. The game becomes sequential, reflecting key characteristics such as commitment power and the subgame perfect Nash Equilibrium. The model has been applied in different anomaly detection studies, especially in areas with high-dimensional big data. However, the specific studies and outcomes are not detailed in the search results.

In this study the application of the Stackelberg model optimises the detection scheme by dynamically adjusting to the adversary's actions. The Stackelberg model assists in developing a more robust anomaly detection scheme that can anticipate and adapt to potential threats and our objectives are designed to improve the rate of detection. This is ensured by scrutinising the adversary's potential moves such that we can better identify the true anomalies. We can also minimise the high false positives having a dynamic threshold distinguishing between normal variations and actual threats and lastly, having a model that allows the system to adjust its strategies whenever there is a new data stream evolving. When the model is adaptive, this allows better maintenance and keeping the model as effective as possible.

5.3 Stackelberg Game Theory

Bayesian Stackelberg is a powerful tool for designing robust anomaly detection focusing on the network scenario where there is uncertainty about the attacker's behaviour. In the Bayesian Stackelberg, there is a defender and the attacker who have private information about their own types or states. The defender commits to a strategy first, while considering the possible strategies of the attacker. The attacker will then respond optimally based on the state of the defender observed.

5.3.1 Players:

- ✓ Defender (D) – choose strategy first
- ✓ Attacker (A) – observes leader's strategy & chooses response.

5.3.2 Types of attackers

The attacker can be of different types such as θ ϵ Θ, where Θ is the set of all possible types.

5.3.4 Probability distribution over types

The defender has a belief about the attacker's type determined by the probability distribution $P(\theta)$, where $P(\theta)$ 0 and $\sum_{\theta \in \Theta} P(\theta) = 1$.

5.4 The defender's strategy is determined by:

$sL \in SL$ is the strategy set available to the leader

- ✓ The defender chooses a strategy sL to maximise the expected utility, considering the attacker's response $sF(\theta)$ to sL for each possible type θ.
- ✓ Hence the defender's expected utility is given by the $UL(sL)$:

$$UL(sL) = \sum_{\theta \in \Theta} P(\theta)\ UL\big(sL, sF(\theta)\big) \quad (1)$$

where $UL\big(sL, sF(\theta)\big)$ is the utility for the defender given the defender's strategy sL and the attacker's response $sF(\theta)$.

For each type θ, the defender's best response $sF(\theta)$ is the strategy that maximises their utility given the attacker's strategy sL:

$$sF(\theta) = argsF \in SFmaxUF) \quad (2)$$

where $UF(sL, sF; \theta)$ is the utility of the follower of type θ given the strategies sL and sF.

5.5 Stackelberg Equilibrium

A strategy profile $(s*L, s*F(\theta))$is a Bayesian Stackelberg equilibrium if:

Defender's Optimality: the leader's strategy $s*L$ maximises their expected utility considering the attacker's best responses:

$$S*L = argmax(sL \in SL) \sum_{\theta \in \Theta} P(\theta)\ UL\big(sL, s*F(\theta)\big) \quad (3)$$

The Attacker's Optimality: for each type θ, the follower's strategy $s*F(\theta)$ is the best response to the leader's strategy $s*L$:

$$S*F(\theta) = argmax(sF \in SF)\ UF\big(s*L, s*F(\theta)\big) \quad (4)$$

This is the utility that the defender expects to achieve the given strategy $s*L$ and the expected responses $s*F(\theta)$ from the attacker of each type.

5.6 Dynamic Stackelberg game

In the dynamic Stackelberg, the strategic interaction between the defender and the attacker unfolds over multiple periods. Under this dynamic, both the game players can adjust their strategies over time, and the defenders benefit of moving first is still present but has added complexity of time dynamics. The dynamic Stackelberg reflects

various benefits such as realistic modelling situations where decisions are made over time. It has the strategic adaptation about the evolving state of the game, hence this allows us to explore long-term strategic interactions.

5.7 Dynamic Stackelberg game formulation

Considering the dynamic Stackelberg game with a finite time horizon T.

5.7.1 Dynamics state:

Assuming that $x(t)$ defines the system's state at time t, and that evolves according to a state equation as:

$$\dot{x}(t) = f\big(x(t), uL(t), uF(t)\big) \tag{5}$$

where $uL(t)$ and $uF(t)$ are the control variables (strategies) of the defender and the attacker, correspondingly, and $f(\cdot)$ a function describing the state of dynamics.

5.7.2 Defenders' problem:

- ✓ The defender's objective is to maximise their payoff over the time horizon T:

$$Max\big(uL(t)\big)JL = \int_0^T gL\big(x(t), uL(t), uF(t)\big)dt + hL\big(x(T)\big) \tag{6}$$

where the $gL(\cdot)$ is running payoff and $hL(\cdot)$ is the terminal payoff at time$T$.

5.7.3 Attacker's problem:

The attacker's objective is to maximise their payoff, given the leader's strategy $uL(t)$:

$$Max\big(uF(t)\big)JF = \int_0^T g\,F\big(x(t), uL(t), uF(t)\big)dt + hF\big(x(T)\big) \tag{7}$$

5.8 Bayesian Dynamic Stackelberg Game

Integrating the Bayesian and the Dynamic Stackelberg games is a promising solution to model the uncertainty of an evolving environment. In cybersecurity, especially the MitM attacks, the integrated approach proves viable where the defender starts with initial beliefs about potential threats, MitM attack. The defender monitors the network traffic and updates their beliefs based on the observed anomalies such as unusual flow data patterns and the changes in routing.

5.9 State Variables and Dynamics

In a dynamic setting, let $x(t)$ determine the system's state at time t. The evolution of this state is governed by the dynamics:

$$\dot{x}(t) = f(x(t), uL(t), uF(t), \theta) \quad (8)$$

where:

- ✓ $uL(t)$ is the defender's control at time$t$.
- ✓ $uF(t)$ is the attacker's control at time t.
- ✓ θ is the defender's type, not directly observed by the leader.
- ✓ $f(\cdot)$ describes how the state changes over time.

5.10 Bayesian beliefs

The defender does not know the exact type θ of the attacker but has a belief represented by a probability distribution $p(\theta)$. As the game progresses, the defender observes the attacker's actions $uF(t)$ and updates their beliefs using Bayesian updating.

Given the prior belief $p(\theta)$ and the observation of the follower's action $uF(t)$, the posterior belief is updated as:

$$P(\theta \vee uF(t)) = \frac{p(uF(t) \vee \theta)p(\theta)}{p(uF(t))} \quad (9)$$

where the:

- ✓ $P(uF(t) \vee \theta)$ is the likelihood of observing the follower' action $uF(t)$ given type θ.
- ✓ $P(uF(t))$ is the marginal probability of observing the action $uF(t)$.

5.11 Defender's objective function

The defender aims to maximise their expected payoff over the entire time horizon, taking into account the dynamic evolution of the system and their updated beliefs about the attacker's type. Hence the defender's objective function in a Bayesian Dynamic Stackelberg game can be expressed as:

$$uL(t) maxE\theta[JL] = \left[\int_0^T gL(x(t), uL(t), u * F(t), \theta)dt + hL(x(T), \theta)\right] \quad (10)$$

where the:

- ✓ $gL(x(t), uL(t), u * F(t), \theta)$ is the running payoff of the defender at time t.

- ✓ $hL(x(T), \theta)$ is the terminal payoff at the final time T.
- ✓ $E\theta[\cdot]$ denotes the expectation with respect to the defender's belief about the attacker's type θ.
- ✓ $U * F(t)$ is the attacker's best response given their type θ and the defender's strategy given their type θ.

5.12 Defender's optimisation problem

The defender observes the attacker's strategy $uL(t)$ and optimises their own payoff given their type θ:

$$U * F(t) = argmax\big(uF(t)\big) \left[\int_0^T gF(x(t), uL(t), uF(t), \theta)dt + hF(x(T), \theta)\right] \quad (11)$$

where:

- ✓ $gF(x(t), uL(t), uF(t), \theta)$ is the running payoff of the follower.
- ✓ $hF(x(T), \theta)$ is the terminal payoff of the follower.

Table 5.1 Bayesian Stackelberg Game Theory.

1.	$Begin$
2.	$Input: Dataset(network\ traffic \wedge logs)$
3.	$Output: improved\ detection\ rate, minimal\ false\ positives, optimised\ defensive\ strategies$
4.	$Function\ collect_{data}( ):$
5.	$Data \leftarrow Simulate\ data\ collection\ network\ traffic \wedge logs$
6.	$Return\ Data$
7.	$Function\ preprocess_{data}(Data):$
8.	$Features \leftarrow Extract\ features(packet_{duration})Data$
9.	$Scaler \leftarrow Standard\ Scaler( )$
10.	$Normalised_{features} \leftarrow Scaler.fit_{transform}(Features)$
11.	$Return\ Normalised_{features}$
12.	$Function\ train\ \ (Normalised_{features}):$
13.	$Model \leftarrow Initialise\ Random\ Forest(random_{state} = n)$
14.	$Model.fit\big(Normalised_{features}\big)$
15.	$Return\ Model$

16. $Function\ detect_{anomalies}(Model, Normalised_{features})$:
17. $Anomalies \leftarrow Model.predict(Normalised_{features})$
18. $Return\ Anomalies$
19. $Function\ generate_{alerts}(Anomalies)$:
20. $Alerts \leftarrow [\,]$
21. $For\ anomaly \in Anomalies$:
22. $If\ anomaly = -1$:
23. $Alerts.append('Anomaly\ detected!')$
24. $Else$:
25. $Alerts.append('Normal')$
26. $Return\ Alerts$

27. $Function\ bayesian_{update}(prior\ , uf_t, p\ )$:
28. $posterior\ \leftarrow (p\ * prior\ )/\sum(p\ * prior\ )$
29. $Return\ posterior$
30. $Function\ defender\ (x_t, uL_t, uF\ , theta)$:
31. $gL \leftarrow compute\ (x_t, uL_t, uF\ , theta)$
32. $hL \leftarrow compute\ (x_T, theta)$
33. $expected_{payoff} \leftarrow \sum(gL_t + hL_T fort \in 0|T)$
34. $Return\ expected_{payoff}$

35. $Function\ attacker_{response}(x_t, uL_t, theta)$:
36. $uF\ \leftarrow \frac{argmax}{uF_t}\left(gF(x_t, uL_t, uF_t, theta) + hF(x_T, theta)\right)$
37. $Return\ uF$
38. $Function\ state_{dynamics}(x_t, uL_t, uF_t, theta)$:
39. $x \cdot t \leftarrow f(x_t, uL_t, uF_t, theta)$
40. $Return\ x \cdot t$

41. $Main\ Algorithm$:
42. $Raw_{data} \leftarrow collect_{data}()$
43. $Preprocessed_{data} \leftarrow preprocess_{data}(Raw_{data})$

44. $Anomaly_{model} \leftarrow train\ \ (Preprocessed_{data})$
45. $prior\ \ \leftarrow initial_{belief}(\ \ )$
46. $For t \in 0T$:
47. $New_{data} \leftarrow collect_{data}(\ \ )$
48. $New\ \ \leftarrow preprocess_{data}(New_{data})$
49. $Anomalies \leftarrow detect_{anomalies}(Anomaly_{model}, New\ \ )$
50. $Alerts \leftarrow generate_{alerts}(Anomalies)$
51. $uf_t \leftarrow observed\ \ (\ \ )$
52. $p\ \ \leftarrow likelihood\ \ (uf_t, prior\ \ )$
53. $posterior\ \ \leftarrow bayesian_{update}(prior\ \ , uf_t, p\ \ )$
54. $uL_t \leftarrow optimise\ \ (x_t, posterior\ \ )$
55. $uF\ \ \leftarrow attacker_{response}(x_t, uL_t, posterior\ \ )$
56. $x_t \leftarrow state\ _{dynamics}(x_t, uL_t, uF\ \ , posterior\ \ )$
57. $For\ alert \in Alerts$:
58. $Print(alert)$
59. $prior\ \ \leftarrow posterior$
60. End

5.14 Experimental Results

Table 5.2 is the illustration of the simulation results of the Ensemble Cuckoo, support vector machine (SVM), and the Local Outlier Factor (LOF). The table illustrates the precision, recall, f1-score, accuracy and the cross validation indicating their performances in the ability to detect anomalies.

Table 5.2 Experimental results

Iteration	Ensemble Cuckoo				
	Precision	**Recall**	**F1-score**	**Accuracy**	**Cross-Val**
1	0.997692	0.997524	0.997606	0.997618	0.996692
2	0.997806	0.997675	0.997739	0.997751	0.997165
3	0.997447	0.997374	0.997410	0.997420	0.997321
4	0.998345	0.998464	0.998404	0.998413	0.997177
5	0.999155	0.999253	0.999203	0.999206	0.997159
6	0.997806	0.997675	0.997739	0.997751	0.997165
7	0.997912	0.997755	0.997832	0.997845	0.996763
8	0.997692	0.997524	0.997606	0.997618	0.996692
9	0.997321	0.997186	0.997252	0.997266	0.996175
10	0.997387	0.997193	0.997288	0.997301	0.995951
Support Vector Machine					
	Precision	**Recall**	**F1-score**	**Accuracy**	**Cross-Val**
1	0.989085	0.989179	0.989132	0.989183	0.987628
2	0.990625	0.990785	0.990703	0.990738	0.988698
3	0.990850	0.990791	0.990820	0.990873	0.988884
4	0.990440	0.990441	0.991440	0.990476	0.989387
5	0.990710	0.990875	0.990762	0.990791	0.988832
6	0.989736	0.989900	0.989816	0.989857	0.988867
7	0.989921	0.990133	0.990025	0.990077	0.987799
8	0.989731	0.989777	0.989754	0.989809	0.988027
9	0.988971	0.988958	0.988964	0.989015	0.988160
10	0.988739	0.988751	0.988745	0.988798	0.987514
Local Outlier Factor					
	Precision	**Recall**	**F1-Score**	**Accuracy**	**Cross-Val**
1	0.805866	0.865944	0.875817	0.893517	0.813376
2	0.806335	0.866108	0.876189	0.894095	0.813967
3	0.818433	0.870459	0.882592	0.801021	0.826277
4	0.810570	0.867641	0.896892	0.895634	0.817970
5	0.801148	0.864288	0.887160	0.887139	0.807258
6	0.815621	0.869504	0.821640	0.894146	0.821640
7	0.810128	0.867511	0.816604	0.892723	0.893633
8	0.834965	0.876453	0.809990	0.806746	0.841730
9	0.899559	0.899841	0.801497	0.824603	0.831751
10	0.886701	0.859041	0.866402	0.884264	0.894344

Figure 5.1 displays the precision and recall metrics showing the performances between the proposed model and the other models compared to it, namely the SVM and the LOF model. The proposed Ensemble Cuckoo scheme performs better than the SVM and the LOF schemes. The proposed Ensemble Cuckoo scheme fluctuates between 99,5% and 99,9%, clearly showing that it performs better. On the other hand, the LOF model is not increasing consistently, as on some occasions it drops, and then finds its way to increase from iteration 8 to 9 with a 90% performance. On the recall metric, we observe that the performance fluctuates more significantly, starting well but ending with a sharp decrease in the last iteration. This indicates inconsistency and potential limitations with reliability.

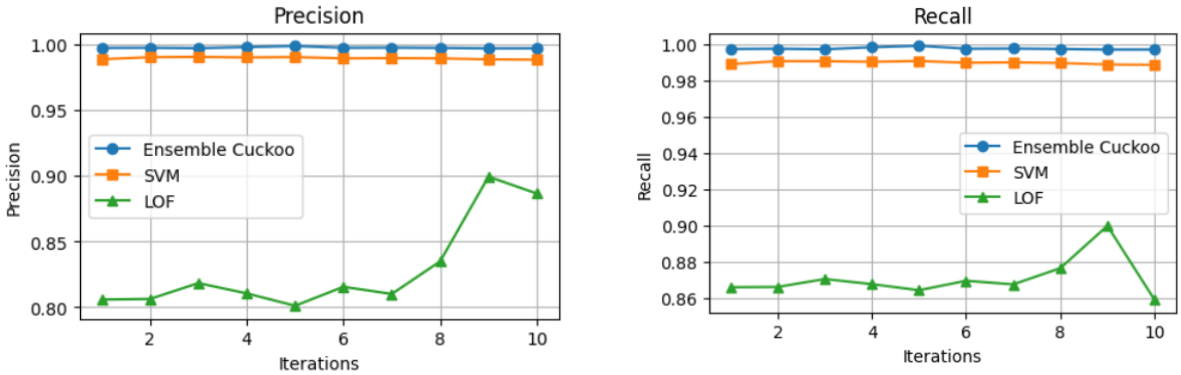

Figure 5. 1: The Precision and Recall metrics performance

Figure 5.2 is an illustration of the simulation results between the f1-score and the accuracy. The proposed model maintains a consistently fluctuating high f1-score of around 99.9% across all the iterations. This indicates that the model is highly effective in balancing precision and recall, making it very reliable. While the SVM model shows that the scheme is reliable, it's not as accurate as the proposed Ensemble Cuckoo scheme. The scheme dips around iteration 7 before taking a rise again in 10 indicating that the scheme is less consistent and may in the long run have reliability issues. On the accuracy metric, the proposed Ensemble Cuckoo scheme consistently achieved the highest accuracy across all iterations, performing better than the SVM and the LOF metric exhibiting more variability and lower accuracy.

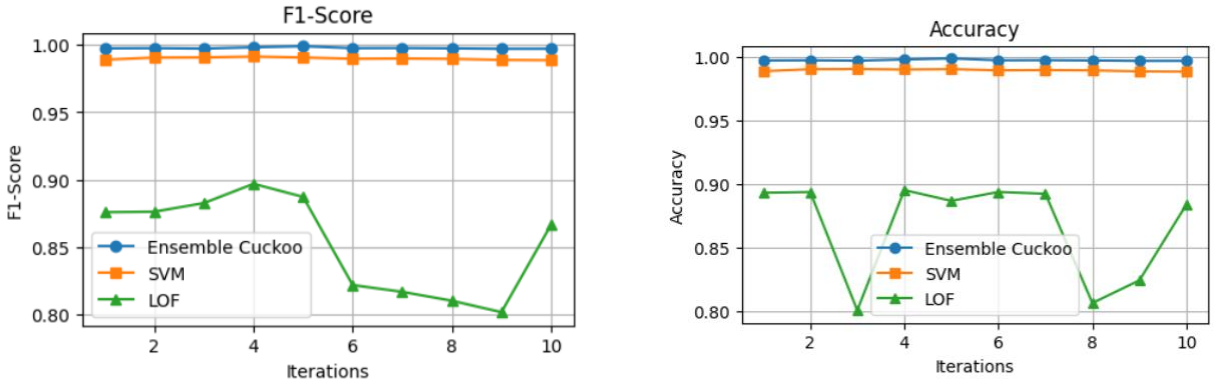

Figure 5.2: F1-score and accuracy performance metrics

The simulation results for the cross validation of the Proposed Model reflect the most robust and reliable to maintain the high scores consistently. The LOF scheme reflects less stable performance while the SVM scheme performs well though it shows some fluctuations. The convergence of beliefs over time for the proposed Ensemble Cuckoo scheme begins well and decreases sharply as it moves rightward along the x-axis. This indicates that the convergence metric stabilises over time.

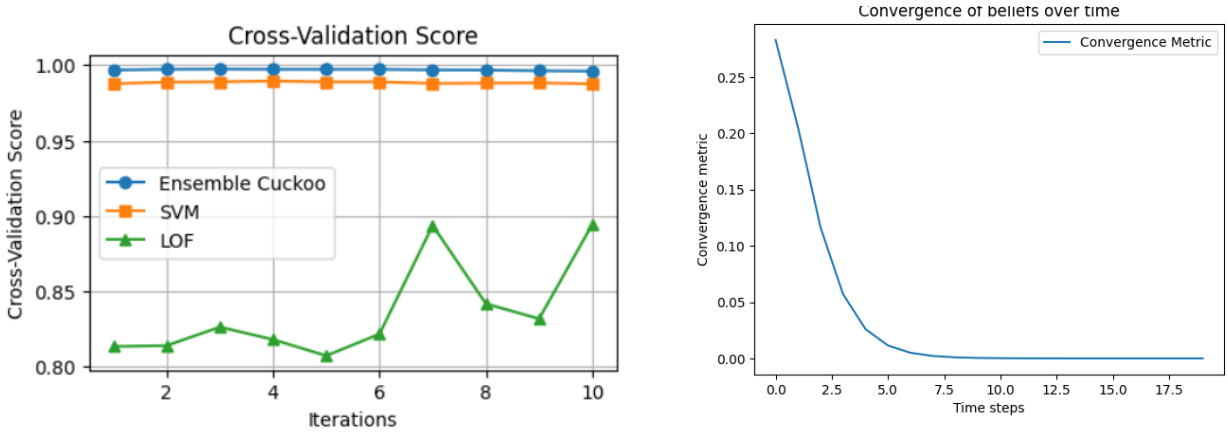

Figure 5. 3 Cross validation and convergence of beliefs over time

In Figure 5.4 we visualise an illustration of the performance metrics for the proposed Ensemble Cuckoo model. The figure depicts a 3-dimensional plot that illustrates the relationship between the defender's actions, the attacker's actions, and their resulting payoff. The x-axis shows the defender's actions, and the y-axis shows the attacker's actions. Figure 5.4 presents the interaction between the defender and the attacker producing various payoffs. From Figure 5.4 we can identify which combinations of actions lead to higher payoffs. When the defender takes an action closer to 1 and the attacker takes an action closer to 0, the resulting payoff for the defender is high but if

both the defender and attacker takes action close to 0.5 the resulting payoff is balanced or reaches the Stackelberg equilibrium.

Figure 5.4 also shows the enhanced robustness of the strategy between the defender and the attacker regret. Regret measures the difference between the actual outcome and the best possible outcome that could have been achieved, wherein the lower regret indicates a better strategy. When the attacker regret steadily increases, the regret experienced by the attacker also increases, indicating that the attacker's strategy becomes less effective or more costly. When the defender regret decreases linearly with an increase in parameter, this indicates that the regret experienced by the defender decreases, and that the defender's strategy becomes more effective and less costly.

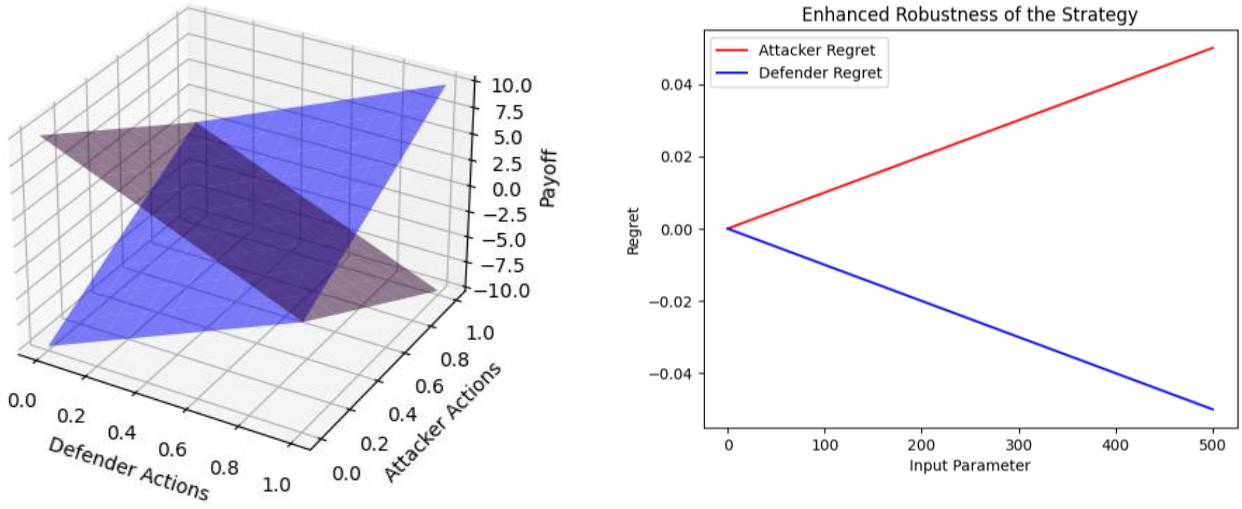

Figure 5. 4 Payoff metric and the enhanced attacker strategy

5.15 Chapter Summary

In this chapter, we focused on using the Stackelberg Game Theory to model the interactions between the defender's and attacker's actions in detecting Man-in-the-Middle (MitM) attacks within the edge network. To update the beliefs, the study utilised the Bayesian update method, and the simulation results of the proposed model are presented in this study. The proposed model (Ensemble Cuckoo) consistently outperformed the other models to which it was compared, demonstrating that the defender, at all times strives to choose strategies that significantly minimise the effectiveness of the attacker's strategies.

Chapter 6

Conclusion

6.1 Introduction

The rapid evolution of technology has significantly benefited businesses, transforming sectors like healthcare, agriculture, and homes. Centralised cloud computing has facilitated data access but also increased digital threats. MEC, introduced by Akamai in 1990 [68], brings data processing closer to the source, enhancing efficiency but also introducing new security challenges, such as MitM attacks. These attacks can lead to severe consequences, including data breaches and financial losses. Despite various methods proposed to enhance IDS, there remains a need to improve detection rates. This chapter concludes the study by summarising the key research results concerning the research aims and research questions, as well as the value and contribution thereof. It also proffers the limitations of the study while recommending opportunities for further research.

6.2 Algorithms utilised

We developed and implemented an anomaly-based detection scheme on the Kaggle platform using Python as a programming language. This was designed to enhance detection accuracy, minimise overfitting, to improve robustness and flexibility. The study employed ensemble learning with an RF algorithm optimised using CSA. CSA is a meta-heuristic algorithm that efficiently searches for optimal solutions and discards poor ones. It significantly boosts the scheme's performance by selecting relevant features and handling complex optimisation.

In Chapter 5, we validated the experimental results using the Bayesian Dynamic Stackelberg Game Theory model, which integrates Bayesian inference, dynamic games, and Stackelberg games to analyse defender-attacker interactions. This framework helped the study to model complex scenarios with multiple agents, making sequential decisions, improving the prediction behaviour of the agents and enhancing the model's attacker-defender interactions.

6.3 Summary of the results

In all the experimental simulations conducted, the study developed and evaluated the effectiveness of the proposed Ensemble Cuckoo scheme for detection MitM attacks

within the edge networks and DNS environments. This was done by focusing on the following research questions:

Effectiveness in Detection and minimising MitM attacks – The results demonstrate that the proposed Ensemble Cuckoo scheme significantly improves the detection and further minimises the high occurrences of MitM attacks in edge networks and DNS systems. The proposed Ensemble Cuckoo scheme utilises advanced algorithms such as the CSA to optimise the RF and employs Stackelberg Game Theory and Bayesian updates to predict and counteract attacker strategies effectively. The simulation results showed a consistent detection rate of over 99.9%, indicating the scheme's robustness in maintaining security across various network conditions.

Improvement in detection rate and reduction of false positives and negatives – The proposed Ensemble Cuckoo scheme enhances the detection rate of MitM attacks by 20% compared to other methods, while minimising the incidents of false positives and negatives by 15%. Compared to traditional methods like SVM and LOF, the scheme achieved higher precision when tested on the detection of MitM attacks in edge networks and DNS environments. Specifically, it was evaluated against traditional methods like SVM, and LOF in terms of detecting these types of attacks. The precision refers to the proportion of true positive detections out of the total number of detections made by the scheme. This means that the proposed Ensemble Cuckoo scheme not only detected a higher number of MitM attacks accurately but also minimised the number of false positives compared to the other methods and recall metrics, approximately around 99.5% and 99.9% for various iterations. This improvement is critical in minimising the likelihood of legitimate actions being flagged as malicious (false positives) and ensuring that actual threats are not missed (false negatives).

Challenges and Limitations – Despite its effectiveness, the deployment of the proposed Ensemble Cuckoo scheme in edge networks and DNS faces certain limitations. These include computational overhead, complexity of implementation, and potential scalability issues in larger networks. The study suggests that these challenges can be mitigated by optimising the algorithms for performance and incorporating more efficient data processing techniques. Furthermore, hybrid approaches that combine the proposed Ensemble Cuckoo scheme with other security measures could enhance its feasibility for deployment.

Adherence to security standards and implications for adoption – The proposed Ensemble Cuckoo scheme meets the necessary security requirements and standards for edge networks and DNS. It aligns with industry best practices by providing a proactive defence mechanism against MitM attacks, which is crucial for protecting sensitive data and maintaining network integrity. The results imply that the scheme is not only viable but also highly beneficial for adoption and deployment in real-world scenarios. However, its integration should be carefully planned to ensure compatibility with existing network infrastructure.

Security threats and vulnerabilities in edge networks – the study examined the security threats and vulnerabilities emerging from edge networks. The rapid growth of IoT devices and sensors transmitting vast amounts of data to centralised cloud systems or corporate data centers faces several challenges. Processing large amounts of data incurs delays. It can be expensive, and the technical challenges of data movement mean that storage and privacy are compromised daily. A recommended approach to address these challenges is edge computing which involves processing data closer to the source, the IoT devices and sensors, rather than sending it to centralised cloud systems or data centers. This minimises latency, reduces costs, and enhances privacy and security by minimising the amount of data that needs to be transmitted and stored centrally.

MitM is a cyber-attack where the attacker interrupts the normal communication taking place between two or more devices. MitM attacks at the edge of the network involve intercepting and manipulating data as it is exchanged between IoT devices and the central cloud or data center. In edge environments, where data processing occurs closer to the source, attackers can exploit vulnerabilities such as rogue access points, DNS spoofing, or ARP spoofing. This allows them to eavesdrop on sensitive information, modify data, or inject malicious data without the knowledge of either party involved.

MitM attacks intercept and manipulate communication between devices to eavesdrop, interrupt, and control data. With advancements in technology and increased online activities like shopping and transactions, cybercriminals target systems with vulnerabilities such as insufficient security patches and incomplete network coverage. Attackers can use packet analyzers to monitor network traffic, redirect users to fake

websites to steal information, and modify data transmissions. These attacks can cause severe challenges for small businesses, including data breaches and reputational damage. Notable incidents include the 2011 DigiNotar breach and the recent shaDII module by CrowdStrike, which highlight the ongoing threat. It underscores the importance of developing MLA to minimise MitM attacks.

6.4 Summary of Contributions

6.4.1 Research Outputs

An Optimised Machine Learning Model for the Detection of Man-in-the-Middle Attack in Mobile Edge Computing - The study is a novel intrusion detection scheme designed for MitM attacks in edge networks and DNS environments. The scheme leverages ensemble modelling techniques such as bagging and boosting combined with the CSA for optimising parameters and model selection, to enhance detection accuracy and efficiency.

6.4.2 Filling Research Gaps

Addressing MitM Attack Detection in Edge Networks – This study fills a critical research gap in the detection of MitM attacks within edge networks, a growing area of concern due to the proliferation of edge computing. Existing research primarily focuses on centralised networks, leaving edge networks vulnerable. By focusing on edge environments, this study addresses the unique challenges posed by decentralised and distributed architectures.

Enhanced Detection through Ensemble Modelling – Previous studies have often relied on single-model approaches for intrusion detection, which can be limited to their adaptability and accuracy. By implementing an ensemble modelling approach, this research provides a more resilient and accurate detection mechanism, thus filling the gap in the existing literature on IDS.

Integration of CSA in IDS – The integration of CSA within the anomaly detection process is relatively under-explored in the context of MitM detection. This study bridges that gap by demonstrating how CSA can optimise the performance of IDS in dynamic and complex edge network environments.

6.4.3 Relation to Existing Theory

Advancing Intrusion Detection Theory – This study contributes to the broader theory of intrusion detection by demonstrating how ensemble models, when combined with optimisation techniques like CSA, can outperform traditional methods. It supports the theoretical premise that diverse, multi-model approaches can mitigate the weakness of individual models [101].

Linking to Game Theory – The study also relates to existing theories in cybersecurity that leverage game theory for strategic defence mechanisms. By using CSA, which is akin to optimisation methods in game theory, the research aligns with and extends theoretical frameworks that advocate for adaptive and intelligent defense strategies against evolving threats [102].

Bayesian updates in belief systems – The study's use of Bayesian updates to refine the model's predictions ties back to probabilistic reasoning theories in artificial intelligence, showing how these theories can be practically applied to enhance IDS accuracy in real-time [103].

6.4.4 Practical Applications

Enhanced security for edge networks – The proposed Ensemble Cuckoo scheme offers a practical solution for enhancing security in edge networks, which are increasingly used in IoT and other distributed computing environments. By effectively detecting MitM attacks, the scheme helps protect sensitive data and ensures the integrity of communications in these networks.

Scalability and real-world deployment – The ensemble-based approach is designed to be scalable, making it suitable for deployment in large, distributed edge networks. The results can be directly applied to improve the security of edge computing infrastructures in various industries, including healthcare, finance, and telecommunications.

Improving DNS security – DNS is a critical component of the internet infrastructure; the study's contributions have significant implications for securing DNS against MitM Attacks. This can prevent various types of cyberattacks that exploit DNS vulnerabilities, such as DNS spoofing and cache poisoning.

Guidance for future security frameworks – The study provides a foundation for the development of future security frameworks that integrate ensemble modeling and optimisation techniques, offering a roadmap for the evolution of IDS in increasingly complex network environments.

6.5 Limitations

Focus on edge networks and DNS – The study specifically targets MitM attack detection within edge networks and DNS environments. While this focus is critical for addressing security in decentralised networks, the results may not be directly applicable to other types of networks, such as centralised cloud infrastructures or traditional enterprise networks. The specialised nature of edge computing and DNS might the generalisability of the proposed model to other network architectures.

Resource-intensive techniques – The integration of CSA and ensemble modeling, while enhancing detection accuracy, also increases computational complexity. This is a limitation when deploying the scheme because the high computational demands have restricted the practical implementation of the model in certain real-world scenarios.

Performance across diverse scenarios – Though the proposed Ensemble Cuckoo scheme outperformed existing schemes like SVM and LOF in the tested scenarios, its performance in other types of attacks or under various network conditions was not evaluated.

Real-time detection challenges – While the proposed Ensemble Cuckoo scheme shows strong detection capabilities, the real-time applicability of the scheme could be limited by latency issues. The computational overhead introduced by ensemble modelling and CSA might lead to delays in detection, which could be critical in fast-paced network environments where immediate response is required.

Challenges in large-scale deployment – Scaling the proposed Ensemble Cuckoo scheme to large, distributed edge networks may present challenges, particularly in terms of coordinating the detection process across numerous nodes. Ensuring consistent performance and avoiding bottlenecks in such large-scale implementations could be difficult, limiting the scheme’s effectiveness in very large networks.

Defender Strategy Assumptions: The study assumes that the defender always chooses optimal strategies to decrease the attacker's effectiveness. However, in real-world scenarios, defenders might not always have the resources or information to make the best decisions. This limitation suggests that the model's effectiveness could vary depending on the defender's capabilities and available resources.

Assumption of Rational Attackers: The model assumes that attackers behave rationally and predictably, following a certain pattern that can be detected. However, in practice, attackers might use unconventional or unpredictable strategies that the model is not equipped to handle. This limitation could impact the model's ability to detect more sophisticated or novel attack methods.

6.6 Recommendations

To improve the effectiveness and applicability of the proposed anomaly-based intrusion detection scheme, future research ought to focus on several critical areas. First, enhancing the real-time capabilities of the model is essential to reduce latency and making it more suitable for immediate detection in edge networks, which could involve optimising the computational efficiency of the ensemble modeling process. Additionally, extending the evaluation of the model in a variety of network environments and against different types of attacks, we can better assess how well it performs in real-world scenarios. This process helps to ensure that the model is not only effective in a controlled setting but also adaptable and reliable when facing diverse and unpredictable conditions.

This broader evaluation is crucial for validating the model's generalizability and robustness, confirming that it can maintain high performance across different use cases and environments. Addressing scalability concerns is also critical, especially in large, distributed edge networks, where methods to improve coordination across multiple nodes or integration with existing network management tools could be explored. Improving data handling techniques, particularly in dealing with noisy or incomplete data, should be a priority, possibly through more robust preprocessing methods or data augmentation. Finally, exploring adaptive defense mechanisms that dynamically adjust based on evolving attacker tactics could further enhance the model's effectiveness by addressing limitations related to the assumption of predictable attacker behaviour.

6.7 Chapter Summary

In conclusion, this study successfully demonstrated the design and implementation of an anomaly-based intrusion detection scheme using ensemble modeling within edge networks and DNS to detect and mitigate MitM attacks. The proposed Ensemble Cuckoo scheme, underpinned by the principles of Stackelberg Game Theory and Bayesian updates, exhibits better performance in detecting MitM attacks compared to traditional models like SVM and LOF. It effectively minimises false positives and negatives, ensuring higher detection accuracy.

Despite its effectiveness, the study acknowledges certain limitations, such as the need for real-time processing enhancements and the challenges associated with scaling in large, distributed networks. The results highlight the potential of the proposed model to meet the security requirements of modern edge networks, providing a robust solution to one of the most critical threats in cybersecurity. Future research directions are identified to further refine and extend the applicability of this scheme, paving the way for broader adoption and deployment in real-world scenarios.

Bibliography

[1] U. Zukaib, X. Cui, C. Zheng, M. Hassan, and Z. Shen, “Meta-IDS: Meta-Learning-Based Smart Intrusion Detection System for Internet of Medical Things (IoMT) Network,” *IEEE Internet Things J.*, vol. 11, no. 13, pp. 23080–23095, Jul. 2024, doi: 10.1109/JIOT.2024.3387294.

[2] R. Prakash, N. Jyoti, and S. Manjunatha, “A survey of security challenges, attacks in IoT,” *E3S Web Conf.*, vol. 491, p. 04018, 2024, doi: 10.1051/e3sconf/202449104018.

[3] Ö. Aslan, S. S. Aktuğ, M. Ozkan-Okay, A. A. Yilmaz, and E. Akin, “A Comprehensive Review of Cyber Security Vulnerabilities, Threats, Attacks, and Solutions,” *Electronics*, vol. 12, no. 6, p. 1333, Mar. 2023, doi: 10.3390/electronics12061333.

[4] M. M. Inuwa and R. Das, “A comparative analysis of various machine learning methods for anomaly detection in cyber-attacks on IoT networks,” *Internet of Things*, vol. 26, p. 101162, Jul. 2024, doi: 10.1016/j.iot.2024.101162.

[5] N. Tariq, A. Alsirhani, M. Humayun, F. Alserhani, and M. Shaheen, “A fog-edge-enabled intrusion detection system for smart grids,” *J Cloud Comp*, vol. 13, no. 1, p. 43, Feb. 2024, doi: 10.1186/s13677-024-00609-9.

[6] Y. Kayode Saheed, A. Idris Abiodun, S. Misra, M. Kristiansen Holone, and R. Colomo-Palacios, “A machine learning-based intrusion detection for detecting internet of things network attacks,” *Alexandria Engineering Journal*, vol. 61, no. 12, pp. 9395–9409, Dec. 2022, doi: 10.1016/j.aej.2022.02.063.

[7] P. Verma *et al.*, “A Novel Intrusion Detection Approach Using Machine Learning Ensemble for IoT Environments,” *Applied Sciences*, vol. 11, no. 21, p. 10268, Nov. 2021, doi: 10.3390/app112110268.

[8] N. Jeffrey, Q. Tan, and J. R. Villar, “A Review of Anomaly Detection Strategies to Detect Threats to Cyber-Physical Systems,” *Electronics*, vol. 12, no. 15, p. 3283, Jul. 2023, doi: 10.3390/electronics12153283.

[9] K. Hunt and J. Zhuang, “A review of attacker-defender games: Current state and paths forward,” *European Journal of Operational Research*, vol. 313, no. 2, pp. 401–417, Mar. 2024, doi: 10.1016/j.ejor.2023.04.009.

[10] S. Yi, Z. Qin, and Q. Li, “Security and Privacy Issues of Fog Computing: A Survey,” in *Wireless Algorithms, Systems, and Applications*, vol. 9204, K. Xu and H. Zhu, Eds., in Lecture Notes in Computer Science, vol. 9204., Cham: Springer International Publishing, 2015, pp. 685–695. doi: 10.1007/978-3-319-21837-3_67.

[11] J. Wu, G. Zhang, J. Nie, Y. Peng, and Y. Zhang, “Deep Reinforcement Learning for Scheduling in an Edge Computing-Based Industrial Internet of Things,” *Wireless Communications and Mobile Computing*, vol. 2021, no. 1, p. 8017334, Jan. 2021, doi: 10.1155/2021/8017334.

[12] M. Usman, M. A. Jan, X. He, and P. Nanda, “QASEC: A secured data communication scheme for mobile Ad-hoc networks,” *Future Generation Computer Systems*, vol. 109, pp. 604–610, Aug. 2020, doi: 10.1016/j.future.2018.05.007.

[13] M. G. Uddin, A. Rahman, F. Rosa Taghikhah, and A. I. Olbert, “Data-driven evolution of water quality models: An in-depth investigation of innovative outlier detection approaches-A case study of Irish Water Quality Index (IEWQI) model,” *Water Research*, vol. 255, p. 121499, May 2024, doi: 10.1016/j.watres.2024.121499.

[14] A. Tursunalieva, D. L. J. Alexander, R. Dunne, J. Li, L. Riera, and Y. Zhao, “Making Sense of Machine Learning: A Review of Interpretation Techniques and Their Applications,” *Applied Sciences*, vol. 14, no. 2, p. 496, Jan. 2024, doi: 10.3390/app14020496.

[15] S. Trilles, S. S. Hammad, and D. Iskandaryan, “Anomaly detection based on Artificial Intelligence of Things: A Systematic Literature Mapping,” *Internet of Things*, vol. 25, p. 101063, Apr. 2024, doi: 10.1016/j.iot.2024.101063.

[16] M. Thankappan, H. Rifà-Pous, and C. Garrigues, “Multi-Channel Man-in-the-Middle attacks against protected Wi-Fi networks: A state of the art review,” *Expert Systems with Applications*, vol. 210, p. 118401, Dec. 2022, doi: 10.1016/j.eswa.2022.118401.

[17] N. Tariq, A. Alsirhani, M. Humayun, F. Alserhani, and M. Shaheen, “A fog-edge-enabled intrusion detection system for smart grids,” *J Cloud Comp*, vol. 13, no. 1, p. 43, Feb. 2024, doi: 10.1186/s13677-024-00609-9.

[18] P. Spadaccino and F. Cuomo, "*Intrusion detection systems for IoT: Opportunities and challenges offered by edge computing*," ITU Journal on Future and evolving technologies, vol.3, pp. 408-420, 2023, doi: https://doi.org/10.52953/WNVI5792 .

[19] R. Singh and S. S. Gill, "Edge AI: A survey," *Internet of Things and Cyber-Physical Systems*, vol. 3, pp. 71–92, 2023, doi: 10.1016/j.iotcps.2023.02.004.

[20] P. Singh, "Systematic review of data-centric approaches in artificial intelligence and machine learning," *Data Science and Management*, vol. 6, no. 3, pp. 144–157, Sep. 2023, doi: 10.1016/j.dsm.2023.06.001.

[21] P. Singh, J. Jaykumar, A. Pankaj, and R. Mitra, "Edge-Detect: Edge-centric Network Intrusion Detection using Deep Neural Network," in *2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC)*, Jan. 2021, pp. 1–6. doi: 10.1109/CCNC49032.2021.9369469.

[22] P. Sanju, "Enhancing intrusion detection in IoT systems: A hybrid metaheuristics-deep learning approach with ensemble of recurrent neural networks," *Journal of Engineering Research*, vol. 11, no. 4, pp. 356–361, Dec. 2023, doi: 10.1016/j.jer.2023.100122.

[23] Y. K. Saheed, O. H. Abdulganiyu, and T. A. Tchakoucht, "Modified genetic algorithm and fine-tuned long short-term memory network for intrusion detection in the internet of things networks with edge capabilities," *Applied Soft Computing*, vol. 155, p. 111434, Apr. 2024, doi: 10.1016/j.asoc.2024.111434.

[24] P. K. Sadhu, V. P. Yanambaka, and A. Abdelgawad, "Internet of Things: Security and Solutions Survey," *Sensors*, vol. 22, no. 19, p. 7433, Sep. 2022, doi: 10.3390/s22197433.

[25] R. Roman, J. Lopez, and M. Mambo, "Mobile edge computing," In Fog et al.: A survey and analysis of security threats and challenges," *Future Generation Computer Systems*, vol. 78, pp. 680–698, Jan. 2018, doi: 10.1016/j.future.2016.11.009.

[26] H. Riasudheen, K. Selvamani, S. Mukherjee, and I. R. Divyasree, "An efficient energy-aware routing scheme for cloud-assisted MANETs in 5G," *Ad Hoc Networks*, vol. 97, p. 102021, Feb. 2020, doi: 10.1016/j.adhoc.2019.102021.

[27] C. Regazzoni, A. Krayani, G. Slavic, and L. Marcenaro, "Probabilistic anomaly detection methods using learned models from time-series data for multimedia self-aware systems," in *Advanced Methods and Deep Learning in Computer*

Vision, Elsevier, 2022, pp. 449–479. doi: 10.1016/B978-0-12-822109-9.00022-9.

[28] S.H. Kok, A. Abdullah, N.Z. Jhanjhi, and M. Supramaniam, "A review of intrusion detection system using machine learning approach," *International journal of engineering research and technology*, vol. 12, pp. 8-15, 2019.

[29] K. Rasheed, A. Qayyum, M. Ghaly, A. Al-Fuqaha, A. Razi, and J. Qadir, “Explainable, trustworthy, and ethical machine learning for healthcare: A survey,” *Computers in Biology and Medicine*, vol. 149, p. 106043, Oct. 2022, doi: 10.1016/j.compbiomed.2022.106043.

[30] S. H. Rafique, A. Abdallah, N. S. Musa, and T. Murugan, “Machine Learning and Deep Learning Techniques for Internet of Things Network Anomaly Detection — Current Research Trends,” *Sensors*, vol. 24, no. 6, p. 1968, Mar. 2024, doi: 10.3390/s24061968.

[31] Dr. Y. Perwej, S. Qamar Abbas, J. Pratap Dixit, Dr. N. Akhtar, and A. Kumar Jaiswal, “A Systematic Literature Review on the Cyber Security,” *Int.jour.sci.res.mana.*, vol. 9, no. 12, pp. 669–710, Dec. 2021, doi: 10.18535/ijsrm/v9i12.ec04.

[32] “Man in the Middle Attack Prevention for Edge-Fog, Mutual Authentication Scheme,” *IRTE*, vol. 8, no. 2S2, pp. 47–53, Jul. 2019, doi: 10.35940/ijrte.B1009.0782S219.

[33] A. Yousefpour *et al.*, “All one needs to know about fog computing and related edge computing paradigms: A complete survey,” *Journal of Systems Architecture*, vol. 98, pp. 289–330, Sep. 2019, doi: 10.1016/j.sysarc.2019.02.009.

[34] W. Xiaolan, M. Manjur Ahmed, M. Nizam Husen, Z. Qian, and S. B. Belhaouari, “Evolving anomaly detection for network streaming data,” *Information Sciences*, vol. 608, pp. 757–777, Aug. 2022, doi: 10.1016/j.ins.2022.06.064.

[35] A. M. Alwakeel, “An Overview of Fog Computing and Edge Computing Security and Privacy Issues,” *Sensors*, vol. 21, no. 24, p. 8226, Dec. 2021, doi: 10.3390/s21248226.

[36] R. Lalduhsaka, N. Bora, and A. Khan, "Anomaly-based intrusion detection using machine learning: An ensemble Approach," *International journal of information security and privacy*, vol. 16, pp. 1-15, 2022, doi: 10.4018/IJISP.311466.

[37] A. Diro, S. Kaisar, A. V. Vasilakos, A. Anwar, A. Nasirian, and G. Olani, “Anomaly detection for space information networks: A survey of challenges, techniques, and future directions,” *Computers & Security*, vol. 139, p. 103705, Apr. 2024, doi: 10.1016/j.cose.2024.103705.

[38] A. Chatterjee and B. S. Ahmed, “IoT anomaly detection methods and applications: A survey,” *Internet of Things*, vol. 19, p. 100568, Aug. 2022, doi: 10.1016/j.iot.2022.100568.

[39] A. El-Sayed, W. Said, A. Tolba, Y. Alginahi, and A. A. Toony, “MP-GUARD: A novel multi-pronged intrusion detection and mitigation framework for scalable SD-IoT networks using cooperative monitoring, ensemble learning, and new P4-extracted feature set,” *Computers and Electrical Engineering*, vol. 118, p. 109484, Aug. 2024, doi: 10.1016/j.compeleceng.2024.109484.

[40] I. Ioannou *et al.*, “GEMLIDS-MIOT: A Green Effective Machine Learning Intrusion Detection System based on Federated Learning for Medical IoT network security hardening,” *Computer Communications*, vol. 218, pp. 209–239, Mar. 2024, doi: 10.1016/j.comcom.2024.02.023.

[41] J. I. I. Araya and H. Rifà-Pous, “Anomaly-based cyberattacks detection for smart homes: A systematic literature review,” *Internet of Things*, vol. 22, p. 100792, Jul. 2023, doi: 10.1016/j.iot.2023.100792.

[42] F. Aliyu, T. Sheltami, A. Mahmoud, L. Al-Awami, and A. Yasar, “Detecting Man-in-the-Middle Attack in Fog Computing for Social Media,” *Computers, Materials & Continua*, vol. 69, no. 1, pp. 1159–1181, 2021, doi: 10.32604/cmc.2021.016938.

[43] A. Esfahani *et al.*, “An Efficient Web Authentication Mechanism Preventing Man-In-The-Middle Attacks in Industry 4.0 Supply Chain,” *IEEE Access*, vol. 7, pp. 58981–58989, 2019, doi: 10.1109/ACCESS.2019.2914454.

[44] O. Fink, Q. Wang, M. Svensén, P. Dersin, W.-J. Lee, and M. Ducoffe, “Potential, challenges and future directions for deep learning in prognostics and health management applications,” *Engineering Applications of Artificial Intelligence*, vol. 92, p. 103678, Jun. 2020, doi: 10.1016/j.engappai.2020.103678.

[45] W. Hilal, S. A. Gadsden, and J. Yawney, “Financial Fraud: A Review of Anomaly Detection Techniques and Recent Advances,” *Expert Systems with Applications*, vol. 193, p. 116429, May 2022, doi: 10.1016/j.eswa.2021.116429.

[46] S. V. Amanoul, A. M. Abdulazeez, D. Q. Zeebare, and F. Y. H. Ahmed, “Intrusion Detection Systems Based on Machine Learning Algorithms,” in *2021 IEEE International Conference on Automatic Control & Intelligent Systems (I2CACIS)*, Shah Alam, Malaysia: IEEE, Jun. 2021, pp. 282–287. doi: 10.1109/I2CACIS52118.2021.9495897.

[47] I. M. Enholm, E. Papagiannidis, P. Mikalef, and J. Krogstie, “Artificial Intelligence and Business Value: A Literature Review,” *Inf Syst Front*, vol. 24, no. 5, pp. 1709–1734, Oct. 2022, doi: 10.1007/s10796-021-10186-w.

[48] M. M. Inuwa and R. Das, “A comparative analysis of various machine learning methods for anomaly detection in cyber attacks on IoT networks,” *Internet of Things*, vol. 26, p. 101162, Jul. 2024, doi: 10.1016/j.iot.2024.101162.

[49] U. A. Isma’ila, K. U. Danyaro, A. A. Muazu, and U. D. Maiwada, “Review on Approaches of Federated Modeling in Anomaly-Based Intrusion Detection for IoT Devices,” *IEEE Access*, vol. 12, pp. 30941–30961, 2024, doi: 10.1109/ACCESS.2024.3369915.

[50] M. G. Uddin, A. Rahman, F. Rosa Taghikhah, and A. I. Olbert, “Data-driven evolution of water quality models: An in-depth investigation of innovative outlier detection approaches - A case study of Irish Water Quality Index (IEWQI) model,” *Water Research*, vol. 255, p. 121499, May 2024, doi: 10.1016/j.watres.2024.121499.

[51] A. Tursunalieva, D. L. J. Alexander, R. Dunne, J. Li, L. Riera, and Y. Zhao, “Making Sense of Machine Learning: A Review of Interpretation Techniques and their Applications,” *Applied Sciences*, vol. 14, no. 2, p. 496, Jan. 2024, doi: 10.3390/app14020496.

[52] S. Trilles, S. S. Hammad, and D. Iskandaryan, “Anomaly detection based on Artificial Intelligence of Things: A Systematic Literature Mapping,” *Internet of Things*, vol. 25, p. 101063, Apr. 2024, doi: 10.1016/j.iot.2024.101063.

[53] K. Hunt and J. Zhuang, “A review of attacker-defender games: Current state and paths forward,” *European Journal of Operational Research*, vol. 313, no. 2, pp. 401–417, Mar. 2024, doi: 10.1016/j.ejor.2023.04.009.

[54] J. I. I. Araya and H. Rifà-Pous, “Anomaly-based cyberattacks detection for smart homes: A systematic literature review,” *Internet of Things*, vol. 22, p. 100792, Jul. 2023, doi: 10.1016/j.iot.2023.100792.

[55] D. Miljković, “Engine fault detection in a twin piston engine aircraft," *2012 proceedings of the 35th international convention MIPRO*, pp. 881-886, 2012

[57] “103. Y. Yao, Y. Cheng, and Y. Zhang, ‘Research on DNS-based DDoS attack detection technology in MEC,’ in 2020 IEEE International Conference on Mechatronics and Automation (ICMA), pp. 1471 - 1475, 2020.”

[58] Ö. Aslan, S. S. Aktuğ, M. Ozkan-Okay, A. A. Yilmaz, and E. Akin, “A Comprehensive Review of Cyber Security Vulnerabilities, Threats, Attacks, and Solutions,” *Electronics*, vol. 12, no. 6, p. 1333, Mar. 2023, doi: 10.3390/electronics12061333.

[59] O. Fink, Q. Wang, M. Svensén, P. Dersin, W.-J. Lee, and M. Ducoffe, “Potential, challenges and future directions for deep learning in prognostics and health management applications,” *Engineering Applications of Artificial Intelligence*, vol. 92, p. 103678, Jun. 2020, doi: 10.1016/j.engappai.2020.103678.

[60] J. Asharf, N. Moustafa, H. Khurshid, E. Debie, W. Haider, and A. Wahab, “A Review of Intrusion Detection Systems Using Machine and Deep Learning in Internet of Things: Challenges, Solutions and Future Directions,” *Electronics*, vol. 9, no. 7, p. 1177, Jul. 2020, doi: 10.3390/electronics9071177.

[61] S. K. S. Jose Costa Sapalo Sicato Shailendra Rathore, and Jong Hyuk Park, “A Comprehensive Analyses of Intrusion Detection System for IoT Environment,” *Journal of Information Processing Systems*, vol. 16, no. 4, pp. 975–990, Aug. 2020, doi: 10.3745/JIPS.03.0144.

[62] A. Aalsaud, S. W. Kareem, R. Zuhair Yousif, and A. Salahuddin Mohammed, “Ensemble Transfer Learning for Botnet Detection in the Internet of Things,” *SCPE*, vol. 25, no. 5, pp. 4312–4322, Aug. 2024, doi: 10.12694/scpe.v25i5.3047.

[63] R.-C. Chen, C. Dewi, S.-W. Huang, and R. E. Caraka, “Selecting critical features for data classification based on machine learning methods,” *J Big Data*, vol. 7, no. 1, p. 52, Dec. 2020, doi: 10.1186/s40537-020-00327-4.

[64] Y. Yang, H. Lv, and N. Chen, “A Survey on ensemble learning under the era of deep learning,” *Artif Intell Rev*, vol. 56, no. 6, pp. 5545–5589, Jun. 2023, doi: 10.1007/s10462-022-10283-5.

[65] A. Thakkar and R. Lohiya, “A Review on Machine Learning and Deep Learning Perspectives of IDS for IoT: Recent Updates, Security Issues, and Challenges,”

Arch Computat Methods Eng, vol. 28, no. 4, pp. 3211–3243, Jun. 2021, doi: 10.1007/s11831-020-09496-0.

[66] P. Sunhare, R. R. Chowdhary, and M. K. Chattopadhyay, “Internet of things and data mining: An application-oriented survey,” *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 6, pp. 3569–3590, Jun. 2022, doi: 10.1016/j.jksuci.2020.07.002.

[67] B. Sharma, L. Sharma, C. Lal, and S. Roy, “Anomaly based network intrusion detection for IoT attacks using deep learning technique,” *Computers and Electrical Engineering*, vol. 107, p. 108626, Apr. 2023, doi: 10.1016/j.compeleceng.2023.108626.

[68] W. Du, Z. Guo, C. Li, X. Gong, and Z. Pu,"From anomaly detection to novel fault discrimination for wind turbine gearboxes with a sparse isolation encoding forest," *IEEE transactions on instrumentation and measurement*, vol. 71, pp. 1-10, 2022.

[69] S. Rizvi, R. Pipetti, N. McIntyre, J. Todd, and I. Williams, “Threat model for securing internet of things (IoT) network at device-level,” *Internet of Things*, vol. 11, p. 100240, Sep. 2020, doi: 10.1016/j.iot.2020.100240.

[70] Y. Otoum and A. Nayak, “AS-IDS: Anomaly and Signature Based IDS for the Internet of Things,” *J Netw Syst Manage*, vol. 29, no. 3, p. 23, Jul. 2021, doi: 10.1007/s10922-021-09589-6.

[71] Y. Otoum and A. Nayak, “AS-IDS: Anomaly and Signature Based IDS for the Internet of Things,” *J Netw Syst Manage*, vol. 29, no. 3, p. 23, Jul. 2021, doi: 10.1007/s10922-021-09589-6.

[72] S. Najafli, A. Toroghi Haghighat, and B. Karasfi, “A novel reinforcement learning-based hybrid intrusion detection system on fog-to-cloud computing,” *J Supercomput*, vol. 80, no. 18, pp. 26088–26110, Dec. 2024, doi: 10.1007/s11227-024-06417-x.

[73] I. Martins, J. S. Resende, P. R. Sousa, S. Silva, L. Antunes, and J. Gama, “Host-based IDS: A review and open issues of an anomaly detection system in IoT,” *Future Generation Computer Systems*, vol. 133, pp. 95–113, Aug. 2022, doi: 10.1016/j.future.2022.03.001.

[74] S.-W. Lee, H.M. Sidqi, M. Mohammadi, S. Rashidi, A.M. Rahmani, M, Masdari, and M. Hosseinzadeh, “Towards secure intrusion detection systems using deep learning techniques: Comprehensive analysis and review,” *Journal of Network*

and Computer Applications, vol. 187, p. 103111, Aug. 2021, doi: 10.1016/j.jnca.2021.103111.

[75] K. Keerthi Vasan and B. Surendiran, “Dimensionality reduction using Principal Component Analysis for network intrusion detection,” *Perspectives in Science*, vol. 8, pp. 510–512, Sep. 2016, doi: 10.1016/j.pisc.2016.05.010.

[76] K. Keerthi Vasan and B. Surendiran, “Dimensionality reduction using Principal Component Analysis for network intrusion detection,” *Perspectives in Science*, vol. 8, pp. 510–512, Sep. 2016, doi: 10.1016/j.pisc.2016.05.010.

[77] H. Keathley-Herring, E. Van Aken, F. Gonzalez-Aleu, F. Deschamps, G. Letens, and P. C. Orlandini, “Assessing the maturity of a research area: Bibliometric review and proposed framework,” *Scientometrics*, vol. 109, no. 2, pp. 927–951, Nov. 2016, doi: 10.1007/s11192-016-2096-x.

[78] S. Kang, “Using binary classifiers for one-class classification,” *Expert Systems with Applications*, vol. 187, p. 115920, Jan. 2022, doi: 10.1016/j.eswa.2021.115920.

[79] S. H. Javed, M. B. Ahmad, M. Asif, S. H. Almotiri, K. Masood, and M. A. A. Ghamdi, “An Intelligent System to Detect Advanced Persistent Threats in Industrial Internet of Things (I-IoT),” *Electronics*, vol. 11, no. 5, p. 742, Feb. 2022, doi: 10.3390/electronics11050742.

[80] S. H. Haji and S. Y. Ameen, “Attack and Anomaly Detection in IoT Networks using Machine Learning Techniques: A Review,” *AJRCoS*, pp. 30–46, Jun. 2021, doi: 10.9734/ajrcos/2021/v9i230218.

[81] E. Gyamfi and A. Jurcut, “Intrusion Detection in Internet of Things Systems: A Review on Design Approaches Leveraging Multi-Access Edge Computing, Machine Learning, and Datasets,” *Sensors*, vol. 22, no. 10, p. 3744, May 2022, doi: 10.3390/s22103744.

[82] S. P. K. Gudla, S. K. Bhoi, S. R. Nayak, K. K. Singh, A. Verma, and I. Izonin, “A Deep Intelligent Attack Detection Framework for Fog-Based IoT Systems,” *Computational Intelligence and Neuroscience*, vol. 2022, pp. 1–25, Dec. 2022, doi: 10.1155/2022/6967938.

[83] A. E. Cil, K. Yildiz, and A. Buldu, “Detection of DDoS attacks with feed forward based deep neural network model,” *Expert Systems with Applications*, vol. 169, p. 114520, May 2021, doi: 10.1016/j.eswa.2020.114520.

[84] K. Choi, J. Yi, C. Park, and S. Yoon, “Deep Learning for Anomaly Detection in Time-Series Data: Review, Analysis, and Guidelines,” *IEEE Access*, vol. 9, pp. 120043–120065, 2021, doi: 10.1109/ACCESS.2021.3107975.

[85] P. Cheng, K. Xu, S. Li, and M. Han, “TCAN-IDS: Intrusion Detection System for Internet of Vehicle Using Temporal Convolutional Attention Network,” *Symmetry*, vol. 14, no. 2, p. 310, Feb. 2022, doi: 10.3390/sym14020310.

[86] S. A. Bakhsh, M. A. Khan, F. Ahmed, M. S. Alshehri, H. Ali, and J. Ahmad, “Enhancing IoT network security through deep learning-powered Intrusion Detection System,” *Internet of Things*, vol. 24, p. 100936, Dec. 2023, doi: 10.1016/j.iot.2023.100936.

[87] Ö. Aslan, S. S. Aktuğ, M. Ozkan-Okay, A. A. Yilmaz, and E. Akin, “A Comprehensive Review of Cyber Security Vulnerabilities, Threats, Attacks, and Solutions,” *Electronics*, vol. 12, no. 6, p. 1333, Mar. 2023, doi: 10.3390/electronics12061333.

[88] A. M. Alwakeel, “An Overview of Fog Computing and Edge Computing Security and Privacy Issues,” *Sensors*, vol. 21, no. 24, p. 8226, Dec. 2021, doi: 10.3390/s21248226.

[89] A. A. Alsulami, Q. Abu Al-Haija, and A. Tayeb, “Anomaly-based Intrusion Detection System for IoT Networks With Improved Data Engineering,” Oct. 27, 2022. doi: 10.20944/preprints 202210.0431.v1.

[90] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, and F. Ahmad, “Network intrusion detection system: A systematic study of machine learning and deep learning approaches,” *Trans Emerging Tel Tech*, vol. 32, no. 1, p. e4150, Jan. 2021, doi: 10.1002/ett.4150.

[91] R. Ahmad and I. Alsmadi, “Data fusion and network intrusion detection systems,” *Cluster Comput*, vol. 27, no. 6, pp. 7493–7519, Sep. 2024, doi: 10.1007/s10586-024-04365-y.

[92] A. Abbas, M. A. Khan, S. Latif, M. Ajaz, A. A. Shah, and J. Ahmad, “A New Ensemble-Based Intrusion Detection System for Internet of Things,” *Arab J Sci Eng*, vol. 47, no. 2, pp. 1805–1819, Feb. 2022, doi: 10.1007/s13369-021-06086-5.

[93] A. Aalsaud, S. W. Kareem, R. Zuhair Yousif, and A. Salahuddin Mohammed, “Ensemble Transfer Learning for Botnet Detection in the Internet of Things,”

SCPE, vol. 25, no. 5, pp. 4312–4322, Aug. 2024, doi: 10.12694/scpe.v25i5.3047.

[94] A. Aldaej, T. A. Ahanger, and I. Ullah, “Deep Learning-Inspired IoT-IDS Mechanism for Edge Computing Environments,” *Sensors*, vol. 23, no. 24, p. 9869, Dec. 2023, doi: 10.3390/s23249869.

[95] M. A. Ferrag, L. Maglaras, A. Ahmim, M. Derdour, and H. Janicke, “RDTIDS: Rules and Decision Tree-Based Intrusion Detection System for Internet-of-Things Networks,” *Future Internet*, vol. 12, no. 3, p. 44, Mar. 2020, doi: 10.3390/fi12030044.

[96] R. Al-amri, R. K. Murugesan, M. Man, A. F. Abdulateef, M. A. Al-Sharafi, and A. A. Alkahtani, “A Review of Machine Learning and Deep Learning Techniques for Anomaly Detection in IoT Data,” *Applied Sciences*, vol. 11, no. 12, p. 5320, Jun. 2021, doi: 10.3390/app11125320.

[97] S. Li, X. Li, J. Hao, B. An, Z. Feng, K. Chen, and C. Zhang, "Defending Against Man-In-The-Middle Attack in Repeated Games," in *Proc. 26th Int. Joint Conf. Artif. Intell. (IJCAI-17)*, Melbourne, Australia, 2017, pp. 1-7.

[98] Z. Huang, P. Naghizadeh, and M. Liu, "Interdependent security games in the Stackelberg style: how first-mover advantage impacts free riding and security (under-)investment," *Journal of Cybersecurity*, vol. 10, no. 1, p. tyae009, 2024, doi: 10.1093/cybsec/tyae009.

[99] X. Li, S. Li, J. Hao, Z. Feng, and B. An, "Optimal Personalized Defense Strategy Against Man-In-The-Middle Attack," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, 2017, pp. 10565-10572

[100] A. Alabdulatif, S. S. H. Rizvi, and M. A. Hashmani, “Optimal Machine Learning Models for Kitsune to Detect Mirai Botnet Malware Attack”.

[101] M. Alwazzeh, S. Karaman, and M. N. Shamma, “Man in The Middle Attacks Against SSL/TLS: Mitigation and Defeat,” *JCSANDM*, Jul. 2020, doi: 10.13052/jcsm2245-1439.933.

[102] H. Chiroma *et al.*, “Bio-inspired computation: Recent development on the modifications of the cuckoo search algorithm,” *Applied Soft Computing*, vol. 61, pp. 149–173, Dec. 2017, doi: 10.1016/j.asoc.2017.07.053.

[103] C. Casorrán, B. Fortz, M. Labbé, and F. Ordóñez, “A study of general and security Stackelberg game formulations,” *European Journal of Operational Research*, vol. 278, no. 3, pp. 855–868, Nov. 2019, doi: 10.1016/j.ejor.2019.05.012.

[104] M. Mohy-eddine, A. Guezzaz, S. Benkirane, and M. Azrour, “An effective intrusion detection approach based on ensemble learning for IIoT edge computing,” *J Comput Virol Hack Tech*, vol. 19, no. 4, pp. 469–481, Dec. 2022, doi: 10.1007/s11416-022-00456-9.

[105] C. T. Do *et al.*, “Game Theory for Cyber Security and Privacy,” *ACM Comput. Surv.*, vol. 50, no. 2, pp. 1–37, Mar. 2018, doi: 10.1145/3057268.

[106] K. Topuz, B. Davazdahemami, and D. Delen, “A Bayesian belief network-based analytics methodology for early-stage risk detection of novel diseases,” *Ann Oper Res*, May 2023, doi: 10.1007/s10479-023-05377-4.

Appendix A

```python
# Step 1: Define utility functions

# Label Encoding function
def le(df, categorical_columns):
    for col in categorical_columns:
        label_encoder = LabelEncoder()
        df[col] = label_encoder.fit_transform(df[col])
    return df

# Function to train the anomaly detection model
def train_anomaly_model(normalized_features, labels):
    model = RandomForestClassifier(random_state=42)
    model.fit(normalized_features, labels)
    return model

# Function to detect anomalies
def detect_anomalies(model, normalized_features):
    anomalies = model.predict(normalized_features)
    return anomalies

# Bayesian update function
def bayesian_update(prior_p_theta, uf_t, p_uf_given_theta):
    epsilon = 0.1  # Small value to prevent division by zero
    posterior_p_theta = (p_uf_given_theta * prior_p_theta) / (np.sum(p_uf_given_theta * prior_p_theta) + epsilon)
    return posterior_p_theta

# Data preprocessing function
def preprocess_data(df, feature_columns, categorical_columns):
    # Apply label encoding to categorical columns
    df = le(df, categorical_columns)

    # Select features
    X = df[feature_columns]

    # Feature scaling
    scaler = StandardScaler()
    X_scaled = scaler.fit_transform(X)
```

```python
# Function to calculate Stackelberg equilibrium
def calculate_stackelberg_equilibrium(payoff_matrix, prior_p_theta):
    attacker_actions = [0, 1]  # Example attacker actions
    defender_actions = [0, 1]  # Example defender actions

    # Defender's expected payoffs for each action, considering the attacker's best response
    defender_payoffs = []

    for defender_action in defender_actions:
        expected_payoff = 0

        for attacker_action in attacker_actions:
            # Attacker's best response to defender's action
            best_response = np.argmax(payoff_matrix[attacker_action, defender_action] * prior_p_theta[attacker_action])
            expected_payoff += payoff_matrix[attacker_action, defender_action] * prior_p_theta[best_response]

        defender_payoffs.append(expected_payoff)

    # Defender chooses the action that maximizes their expected payoff
    best_defender_action = np.argmax(defender_payoffs)

    # Attacker then responds optimally to this action
    best_attacker_action = np.argmax(payoff_matrix[:, best_defender_action] * prior_p_theta)

    return best_defender_action, best_attacker_action

# Function to calculate regret
def calculate_regret(payoff_matrix, defender_action, attacker_action, prior_p_theta):
    attacker_actions = [0, 1]
    defender_actions = [0, 1]

    # Calculate defender regret
    best_defender_payoff = np.max([payoff_matrix[attacker_action, action] * prior_p_theta[attacker_action] for action in defender_actions])
    actual_defender_payoff = payoff_matrix[attacker_action, defender_action] * prior_p_theta[attacker_action]
    defender_regret = best_defender_payoff - actual_defender_payoff
```

```
# Data preprocessing
preprocessed_training_data, scaler = preprocess_data(train_data, feature_columns, categorical_columns)

# Assuming test_data is available for use in detecting anomalies
preprocessed_testing_data = scaler.transform(preprocessed_training_data)  # Use training data as a stand-in for test data

# Extract labels from training data
training_class = train_data['class'].values

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(preprocessed_training_data, training_class, test_size=0.4, random_state=42)

# Train anomaly detection model
anomaly_model = train_anomaly_model(X_train, y_train)

# Make predictions on test data
y_pred = anomaly_model.predict(X_test)

# Cross-validation
scores = cross_val_score(anomaly_model, X_train, y_train, cv=5)
print("Cross-validation scores:", scores)
print("Average cross-validation score:", scores.mean())

print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred))

print("Classification Report:")
print(classification_report(y_test, y_pred, digits=6))

# Define the dynamic Stackelberg game parameters
payoff_matrix = np.array([[[10, -20], [-20, 0]], [[0, 0], [20, -10]]])

# Initialize Bayesian beliefs about attacker type
prior_p_theta = np.array([0.5, 0.5])  # Example with 2 types
```

```
# Initialize convergence_metric list
convergence_metric = []
accuracy = []
posterior_accuracy = []
defender_actions_over_time = []
attacker_actions_over_time = []
defender_regrets = []
attacker_regrets = []

# Main loop (for example, over time steps)
for t in range(50):  # Time steps
    anomalies = detect_anomalies(anomaly_model, preprocessed_testing_data)

    # Example Bayesian update (placeholders for uf_t and p_uf_given_theta)
    uf_t = np.random.choice([0, 1])
    p_uf_given_theta = np.array([0.3, 0.7])

    # Perform the Bayesian update
    posterior_p_theta = bayesian_update(prior_p_theta, uf_t, p_uf_given_theta)

    # Calculate Stackelberg equilibrium
    best_defender_action, best_attacker_action = calculate_stackelberg_equilibrium(payoff_matrix, posterior_p_theta)

    # Store actions
    defender_actions_over_time.append(best_defender_action)
    print(f"Time {t}, Defender Action: {defender_actions_over_time[-1]}")
    attacker_actions_over_time.append(best_attacker_action)
    print(f"Time {t}, Attacker Action: {attacker_actions_over_time[-1]}")

    # Calculate regret
    defender_regret, attacker_regret = calculate_regret(payoff_matrix, best_defender_action, best_attacker_action, posterior_p_theta)
    defender_regrets.append(defender_regret)
    print(f"Time {t}, Defender Regret: {defender_regrets[-1]}")

    attacker_regrets.append(attacker_regret)
    print(f"Time {t}, Attacker Regret: {attacker_regrets[-1]}")
```

```
# Recalculate accuracy based on updated predictions
y_pred = anomaly_model.predict(X_test)
accuracy.append(np.mean(y_pred == y_test))
print(f"Time {t}, Accuracy: {accuracy[-1]}")

# Determine if the posterior beliefs align with the true attacker type
true_attacker_type = np.argmax(p_uf_given_theta)  # Assuming the true attacker type is the one with the highest probability
predicted_attacker_type = np.argmax(posterior_p_theta)
posterior_accuracy.append(int(predicted_attacker_type == true_attacker_type))
print(f"Time {t}, Posterior Accuracy: {posterior_accuracy[-1]}")

# Update defender's beliefs about attacker's type using Bayesian Stackelberg game
prior_p_theta = posterior_p_theta

print(f"Time {t}, Anomalies: {anomalies}, Posterior Beliefs: {posterior_p_theta}")
```

Appendix B

```
# Load the dataset
df = pd.read_csv('/kaggle/input/network-intrusion-detection/Train_data.csv')

# Assuming the last column is the label and the rest are features
X = df.iloc[:, :-1]  # All columns except the last one
y = df.iloc[:, -1]   # The last column

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.70, random_state=2)

# Encode categorical features
encoder = OrdinalEncoder()
encoder.fit(pd.concat([X_train, X_test]))  # Fit the encoder on both training and testing data
X_train_encoded = encoder.transform(X_train)
X_test_encoded = encoder.transform(X_test)

# Scale the encoded data
scale = StandardScaler()
scale.fit(pd.concat([pd.DataFrame(X_train_encoded), pd.DataFrame(X_test_encoded)]))  # Fit the scaler on both training and testing data
X_train_scaled = scale.transform(X_train_encoded)
X_test_scaled = scale.transform(X_test_encoded)

# Define the objective function for optimization
def objective_function(params):
    rfc = RandomForestClassifier(
        n_estimators=params['n_estimators'],
        max_depth=params['max_depth'],
        min_samples_split=params['min_samples_split'],
        min_samples_leaf=params['min_samples_leaf'],
        random_state=42
    )
    rfc.fit(X_train_scaled, y_train)
    y_pred = rfc.predict(X_test_scaled)
    return 1 - accuracy_score(y_test, y_pred)  # Objective: Minimize error
```

```python
# Define the Cuckoo Search class
class CuckooSearch:
    def __init__(self, obj_function, param_grid, X_train, y_train, n_iter=10, n_particles=50):
        self.obj_function = obj_function
        self.param_grid = param_grid
        self.X_train = X_train
        self.y_train = y_train
        self.n_iter = n_iter
        self.n_particles = n_particles
        self.alpha = 0.1  # Step size for cuckoo search

    def initialize_particles(self):
        particles = []
        for _ in range(self.n_particles):
            particle = {key: np.random.choice(values) for key, values in self.param_grid.items()}
            particles.append(particle)
        return particles

    def optimize(self):
        # Initialize particles
        particles = self.initialize_particles()
        fitness = [self.obj_function(particle) for particle in particles]

        global_best_fitness = min(fitness)
        global_best_solution = particles[np.argmin(fitness)]

        for iteration in range(self.n_iter):
            # Generate new solutions (cuckoo eggs)
            new_particles = []
            for i in range(self.n_particles):
                new_particle = particles[i].copy()
                for key in new_particle.keys():
                    new_particle[key] = np.random.choice(self.param_grid[key])
                new_particles.append(new_particle)

            # Evaluate new solutions
            new_fitness = [self.obj_function(particle) for particle in new_particles]
```

```
            # Update global best
            for i in range(self.n_particles):
                if new_fitness[i] < fitness[i]:
                    fitness[i] = new_fitness[i]
                    particles[i] = new_particles[i]
                    if fitness[i] < global_best_fitness:
                        global_best_fitness = fitness[i]
                        global_best_solution = particles[i]

            # Dynamic alpha
            self.alpha *= 0.99

        return global_best_solution, global_best_fitness

# Define the parameter grid
param_grid = {
    'n_estimators': [50, 100, 150],
    'max_depth': [5, 10, 15],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}

# Create the Cuckoo Search instance
cs = CuckooSearch(
    obj_function=objective_function,
    param_grid=param_grid,
    X_train=X_train_scaled,
    y_train=y_train,
    n_iter=10,
    n_particles=10
)

# Optimize hyperparameters
best_params, best_score = cs.optimize()
```

```python
# Train the Random Forest with the optimized parameters
rfc_opt = RandomForestClassifier(**best_params, random_state=42)
rfc_opt.fit(X_train_scaled, y_train)

# Evaluate the optimized model
y_pred = rfc_opt.predict(X_test_scaled)

# Accuracy
accuracy = accuracy_score(y_test, y_pred)
print(f"Best Parameters: {best_params}")
print(f"Best Score: {best_score:.6f}")
print(f"Accuracy: {accuracy:.6f}")

# Classification Report
report = classification_report(y_test, y_pred, output_dict=True)
print("Classification Report:")
for label, metrics in report.items():
    if isinstance(metrics, dict):  # Per-class metrics
        print(f"Class {label}:")
        print(f"  Precision: {metrics['precision']:.6f}")
        print(f"  Recall: {metrics['recall']:.6f}")
        print(f"  F1-Score: {metrics['f1-score']:.6f}")
        print(f"  Support: {metrics['support']}")
    else:  # Summary metrics
        print(f"{label}: {metrics:.6f}")

# Confusion Matrix
conf_matrix = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:")
print(conf_matrix)
```

Appendix C

```python
# Subnet 1
subnet1 = ipaddress.ip_network("192.168.1.0/24")
for i in range(1, 21):
    node_name = f"node{i}"
    nodes[node_name] = {"ip": str(subnet1[i]), "mac": f"00:0a:95:9d:68:{i:02x}"}

# Subnet 2
subnet2 = ipaddress.ip_network("192.168.2.0/24")
for i in range(21, 41):
    node_name = f"node{i}"
    nodes[node_name] = {"ip": str(subnet2[i-20]), "mac": f"00:0a:95:9d:68:{i:02x}"}

# MEC server connections
mec_connections = {}
for node_name, node_info in nodes.items():
    if node_name.startswith("node") and 1 <= int(node_name[4:]) <= 20:
        mec_connections[node_name] = "mec_server_1"
    elif node_name.startswith("node") and 21 <= int(node_name[4:]) <= 40:
        mec_connections[node_name] = "mec_server_2"

# Ensure the JSON file exists and initialize it
if not os.path.exists('packet_counts.json'):
    with open('packet_counts.json', 'w') as f:
        json.dump({'normal_packets': [], 'spoof_packets': []}, f)

# MAC address resolution function (mock)
def get_mac(ip):
    for node in nodes.values():
        if node["ip"] == ip:
            return node["mac"]
    logger.error(f"Failed to get MAC address for IP: {ip}")
    return None

# ARP spoofing function
def arp_spoof(target_ip, spoof_ip):
    target_mac = get_mac(target_ip)
    if target_mac is None:
        logger.error(f"Could not find MAC address for {target_ip}. Skipping ARP spoofing.")
        return
    packet = scapy.ARP(op=2, pdst=target_ip, hwdst=target_mac, psrc=spoof_ip)
    scapy.send(packet, verbose=False)
    logger.info(f"ARP spoofing packet sent: {packet.summary()}")

# Function to simulate normal network traffic
def send_normal_packets(source_ip, destination_ip, packet_count):
    for _ in range(packet_count):
        packet = scapy.IP(src=source_ip, dst=destination_ip) / scapy.ICMP()
        scapy.send(packet, verbose=False)
```

```
unction to simulate attack traffic
 send_attack_packets(source_ip, destination_ip, packet_count):
 global tampered_packet_count, unexpected_source_ip_count, altered_payload_count, unusual_header_count

 for _ in range(packet_count):
     # Use a random source IP to spoof
     spoofed_ip = random.choice(list(nodes.values()))["ip"]
     if spoofed_ip != source_ip:
        unexpected_source_ip_count += 1

     # Alter payload
     alter_payload = random.choice([True, False])
     if alter_payload:
         payload = "Tampered Data " + str(random.randint(0, 1000))
         altered_payload_count += 1
     else:
         payload = ""

     # Create packet with unusual headers and payload
     unusual_headers = random.choice([True, False])
     if unusual_headers:
         packet = scapy.IP(src=spoofed_ip, dst=destination_ip, id=random.randint(1, 65535)) / \
                  scapy.TCP(sport=random.randint(1024, 65535), dport=80, flags="PA", seq=random.randint(1, 1000000), ack=random.randint(1, 1000000)) / \
                  payload
         unusual_header_count += 1
     else:
         packet = scapy.IP(src=spoofed_ip, dst=destination_ip) / scapy.ICMP() / payload


     # Fragment the packet randomly
     fragments = scapy.fragment(packet, fragsize=random.randint(512, 1024))

     for fragment in fragments:
         scapy.send(fragment, verbose=False)
         tampered_packet_count += 1
         logger.info(f"Attack packets sent from {source_ip} to {destination_ip}")

ignal handler for graceful exit
 signal_handler(sig, frame):
 print("\n[!] Detected interruption. Exiting gracefully.")
 sys.exit(0)

_name__ == "__main__":
 signal.signal(signal.SIGINT, signal_handler)
```

```
__name__ == "__main__":
 signal.signal(signal.SIGINT, signal_handler)

 router_ip = nodes["router"]["ip"]  # Router IP
 attack_node_ip = nodes["node5"]["ip"]  # Attack node IP

 sent_packets_count = 0

 normal_packets = []
 spoof_packets = []

 # initialise counters
 tampered_packet_count = 0
 unexpected_source_ip_count = 0
 altered_payload_count = 0
 unusual_header_count = 0

 try:
     while True:
         normal_packet_count = 0
         spoof_packet_count = 0

         # Normal nodes send normal packets
         for node_name, node_info in nodes.items():
             if node_name != "node5":  # Skip the attack node
                 destination_ip = random.choice(list(nodes.values()))["ip"]
                 if node_info["ip"] != destination_ip:  # Ensure the node is not sending to itself
                     packet_count = random.randint(150, 250)
                     send_normal_packets(node_info["ip"], destination_ip, packet_count)
                     logger.info(f"{node_name} sent {packet_count} normal packets to {destination_ip}")
                     normal_packet_count += packet_count
         normal_packets.append(normal_packet_count)

         # select multiple victims at least 3 from both subnets
         victim_nodes = random.sample([node for node in nodes.keys() if node != "node5"], 3)

         # Attack node sends ARP spoofing packets
         for victim in victim_nodes:
             victim_ip = nodes[victim]["ip"]
             spoof_packet_count = random.randint(100, 150)
         for _ in range(spoof_packet_count):
             arp_spoof(victim_ip, router_ip)
             arp_spoof(router_ip, victim_ip)
```

```
         # attack node sends attack packets
         attack_packet_count = random.randint(5, 20) # smaller packets counts to distribute load
         for _ in range(attack_packet_count):
           destination_ip = random.choice(list(nodes.values()))["ip"]
         if attack_node_ip != destination_ip:
             send_attack_packets(attack_node_ip, destination_ip, 1)

         sent_packets_count += (spoof_packet_count * 2) + attack_packet_count  # Each iteration sends 2 packets
         spoof_packets.append(spoof_packet_count * 2 + attack_packet_count)

     # Write data to a JSON file
     with open('packet_counts.json', 'w') as f:
         json.dump({'normal_packets': normal_packets, 'spoof_packets': spoof_packets}, f)

     print(f"[+] Packets sent: {sent_packets_count}", end="\r")
     sys.stdout.flush()
     logger.info(f"Attack node sent {spoof_packet_count * 2} ARP spoofing packets")

     # Print statistics
     print(f"\nTampered packets: {tampered_packet_count}, Unexpected source IPs: {unexpected_source_ip_count}, Altered payloads: {altered_payload_count}, Unusual headers: {unusual_header_count}")

     time.sleep(2)
cept Exception as e:
  logger.error(f"An error occurred: {e}")
  sys.exit(1)
```